Helmut Krcmar · Ralf Reussner
Bernhard Rumpe *Editors*

Trusted Cloud Computing

Springer

Trusted Cloud Computing

Helmut Krcmar · Ralf Reussner
Bernhard Rumpe
Editors

Trusted Cloud Computing

Springer

Editors
Helmut Krcmar
Technische Universität München
Munich
Germany

and

fortiss - An-Institut Technische Universität München
Munich
Germany

Ralf Reussner
Karlsruhe Institute of Technology
Karlsruhe
Germany

and

FZI - Forschungszentrum Informatik am KIT
Karlsruhe
Germany

Bernhard Rumpe
RWTH Aachen
Aachen
Germany

and

RIT - Rumpe Information Technologies
Aachen
Germany

ISBN 978-3-319-12717-0 ISBN 978-3-319-12718-7 (eBook)
DOI 10.1007/978-3-319-12718-7

Library of Congress Control Number: 2014955316

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Bundesministerium
für Wirtschaft
und Energie

Message of Greeting

by Sigmar Gabriel,

Federal Minister for Economic Affairs and Energy

for the *Trusted Cloud* Technology Programme run by the Federal Ministry for Economic Affairs and Energy

Cloud computing is emerging as a key tool of digitisation: it creates opportunities for companies - for example when it comes to finding tailored software, computing power and storage capacity via the internet. This saves companies the need to invest in large-scale IT infrastructure. It is a crucial advantage, not least for start-ups as it allows companies to access innovative technologies that so far have been reserved for the big players of the industry. In addition: employees can have mobile access to their data. This makes companies flexible and opens up space for new business models.

The Federal Ministry for Economic Affairs and Energy launched the *Trusted Cloud* Technology Programme in order to open up the big innovation and market potential of cloud computing, especially to SMEs. The programme demonstrates the opportunities and diversity of the new applications in different areas, ranging from industry and crafts to the health and public sector. Special emphasis has been placed on security by design: from the outset, IT security and data protection have been part of the technological development.

To successfully introduce cloud computing in Germany and help SMEs make use of it, we need reliable solutions for a number of different tasks. Germany has in-depth scientific and engineering skills and is therefore very well placed to set benchmarks by giving best practice examples and introducing cloud applications in different industries.

This book presents important scientific results from different areas of application of the *Trusted Cloud* Programme. The applicability and usability of research results always play an important role, be it for security aspects, software architecture or business models.

Germany has every chance to establish itself as a leading player on this promising market. The foundation for our success is a strong scientific base. I hope you will find this publication enjoyable and stimulating.

Sincerely yours,

Preface

The research projects of the Trusted Cloud Initiative of the Federal Ministry for Economic Affairs and Energy are developing novel cloud services and making them, as well as already existing services, usable for trustable IT applications. Cloud services have a number of advantages over in-house services, starting with cost efficiency and flexible payment models, where customers only pay for resources and services they actually use. Combined with the elasticity of service provision, that is the dynamic adaptation to seasonal or task-depending resource demands of a service's operation, it makes Cloud Computing an attractive model for service provisioning on many layers (as application platform, programming model, or for domain-specific services).

However, the user gives up some control over the operation of the IT services to the service provider. This particularly concerns non-functional aspects of service provision, including quality attributes that impact the trustworthiness of the service. However, since the platform provider, due to economy of scale, can address these aspects more professionally, the service is expected to operate at a higher level of quality and under more robust security standards. Service-level agreements specify the qualities promised by the service provider, yet this does not solve the general problem that the service provider has to be fundamentally trusted. If the user does not or cannot grant this trust, Cloud Computing is not an option and its benefits not available.

The projects of the Trusted Cloud Technology Programme address this issue. The research and development conducted in groups of research institutes and enterprises aim at making Cloud Computing practical for trustworthy IT applications. This book documents the progress and results of these programmes and their critical discussion during a workshop at the Forschungszentrum Informatik (FZI) on the 10th of July 2013 in Karlsruhe which was organized by researches from the Karlsruhe Institute for Technology (KIT), the RWTH Aachen University, and the Technische Universität München. This workshop and the book at hand are a platform for presenting the scientific results of these projects and focus less on the business-related aspects. Nevertheless, the discussions have shown that business models, legislative circumstances, technical possibilities, and realizable security are tightly entangled and thus

have to be addressed jointly. This way, legislative requirements may be reflected in the Cloud's physical properties, for instance by a "private cloud" distribution model, or by ensuring that the service provider is operating in an appropriate legal sphere.

A holistic consideration of technological, societal and legal aspects is necessary to ensure the security of cloud services and the data they process, and to gain the trust of society and science in these services, especially in the light of the recent public discussions concerning the security of online data. Nevertheless, it is clear that the issue of security cannot be the responsibility of private enterprises alone. Small and medium enterprises, which are prevalent and of central importance in Germany, cannot stand up to foreign powers and intelligence agencies alone. The security of the Internet and the service that are based on has to be a national concern, if not a European one, in order to compete, and in some areas to catch up, in the marketplaces of the Internet. The Internet, including the operating systems of its nodes, its communication middleware and the application software, no longer is an infrastructure that is just relevant to IT enterprises. Instead, it impacts all areas of economy in Germany. The Trusted Cloud Programme is a highly relevant building block for innovation in the German economy.

Additional building blocks are the programmes Big Data of the BMWi and the Smart Data of the BMBF, both of which deal with the gathering of data over the Internet. We refer in this context to the Smart Data Manifesto, that originated as a key result of the discussions during the Trusted Cloud workshop.

Cloud components are moreover key parts of the cyber-physical infrastructure that will arise in the manufacturing, transportation, and service industry, as well as in the automation of mechanical activities through intelligent, cooperating robots, machines, and transporters. This change in our manufacturing business called "Industry 4.0" will be based on a strong cloud-based IT-infrastructure. It will need to be trustworthy to prevent massive economic damage beyond the IT industry. Reluctance in embracing this change will not be an option, since otherwise new business models will emerge outside of Germany instead of inside.

The research topics discussed in this book are diverse. They are in part fitted to the specific application domain of their projects, nevertheless, as it is typical in computer science, their fundamental technological solutions and platforms are portable to other domains.

The contributions in this book are categorized into the following topics:

- Security and Privacy
- Software Engineering and Quality
- Platforms and Middleware
- Social Aspects
- Business Models
- Standards

We thank all the authors and reviewers for their help to bring this book into existence. Herbert Weber was of great help with his advice for the projects and the field and Jennifer Welp helped organizing the projects from their start up to their current status. The projects also profited from a science and technology visit

to the Silicon Valley with Nadine Schön, MdB and Jennifer Welp. We especially thank Matthias Huber, Antonio Navarro Perez, and Jan Wollersheim that helped to organize the workshop and put the book together.

We hope that the book will inspire you to further scientific research or to use the results for the implementation of trustworthy cloud services. This would achieve one goal of the workshop and this book, to distribute these results to a broader audience and to applications beyond the scope of the Trusted Cloud projects.

Karlsruhe, Aachen, and Munich, March 2014
Helmut Krcmar (Fortiss and TUM)
Ralf Reussner (FZI and KIT)
Bernhard Rumpe (RIT and RWTH)

Contents

Part I
Security and Privacy

Chapter 1
GeneCloud: Secure Cloud Computing for Biomedical Research

Martin Beck, V. Joachim Haupt, Janine Roy, Jan Moennich, René Jäkel, Michael Schroeder, and Zerrin Isik

Abstract Applications in the biomedical sector have a rising demand for computational power due to the growing amount of biological data. However, the setup and maintenance of a sufficient computational infrastructure is costly. Cloud computing allows distributed computing over a network and maximizes the effectiveness of the shared resources. Nevertheless, security concerns rise when valuable research data is transferred to a public Cloud. Herein, we describe – from the application and security perspective – three biomedical case studies from different domains: Patent annotation, cancer outcome prediction, and drug target prediction. We elaborate on different strategies to secure the data and results in the Cloud as well as on the infrastructure needed. The proposed Cloud solutions could help to adapt other similar algorithms from different domains to benefit from Cloud computing.

1.1 Introduction

Data in life sciences – such as sequence, structural, pharmacological/drug data and biomedical literature – are ever growing and demand appropriate computational resources. Cloud services offer such resources but require adjusted algorithms and data management.

Martin Beck
TU Dresden, Institute of Systems Architecture, Germany
e-mail: martin.beck1@tu-dresden.de

Joachim Haupt · Jan Moennich · Michael Schroeder · Zerrin Isik
TU Dresden, BIOTEC, Germany
e-mail: michael.schroeder@biotec.tu-dresden.de

René Jäkel
TU Dresden, Center for Information Services and High Performance Computing (ZIH), Germany
e-mail: rene.jaekel@tu-dresden.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_1

The use of High-performance computing (HPC) systems for biomedical research has a long history [1] and is in the transition to Cloud computing (e.g. Cloud storage systems for DNA sequencing [2]) – going away from in-house solutions. However, this brings new problems such as vendor lock-in and a potentially untrusted environment. Contrary to the increased economical efficiency, flexibility and all the other advantages of the Cloud computing, there are still a few major drawbacks related to privacy issues and possible loss, which prevent broad adoption. There are many proposals on how to enhance data privacy in the public and hybrid Cloud computing scenario [3]. This includes the use of trusted platform modules (TPM) for trusted computing [4], effective separation of data depending on the secrecy level [5], the complete use of fully homomorphic encryption [6] or multi-party computation between several non-colluding parties [7]. However, these techniques possess advantages and disadvantages, that need to be considered before application. Some methods like homomorphic encryption provide a high security level, but data transformation is extremely time consuming, thus increasing execution time dramatically. In contrast, techniques – like anonymization – have only a small impact on the computational complexity, but do not provide a high level of security. The balance between different security mechanisms and the efficiency of data transformation is shown in Figure 1.1.

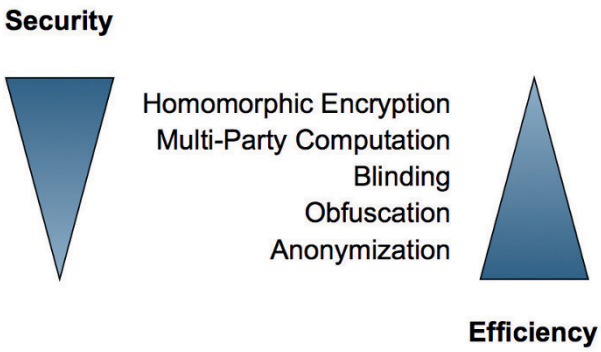


Fig. 1.1: Balance between security and efficiency

In the context of the project GeneCloud we develop strategies to overcome issues in data security on identical complexity levels [8] for services in a potentially untrusted environment (public Cloud). The presented strategies – involving (homomorphic) encryption, minimization and anonymization – are tailored to three case studies from different domains: Patent annotation (text mining), cancer outcome prediction (translational bioinformatics), and drug target prediction (structural bioinformatics). We present a fully data-centric security approach, utilizing customized algorithms to achieve a desired privacy – performance tradeoff. In this approach, sensible data is preprocessed and secured locally in full control of the

user and subsequently transferred to the Cloud platform. Ideally, the Cloud services run on encrypted data, evading an access by an intruder and ultimately decreasing privacy related usage restrictions of Cloud infrastructure. To perform the presented analysis, we use a private Cloud infrastructure and a high performance computing (HPC) cluster.

1.2 Cloud Infrastructure

The following section describes the architectural model as well as the infrastructure. We will elaborate on the different solutions available and on the approaches necessary to run biomedical research on a public Cloud.

1.2.1 Architecture Model

Within the GeneCloud project, we realized a Cloud infrastructure based on two widely used Cloud middlewares, namely OpenNebula and OpenStack. These open source middlewares provide a broad range of management functionalities. Both can operate on top of different virtualization solutions (hypervisors), such as Kernel Based Virtual Machine (KVM) or Xen. In our solution, we are using KVM as default, since it has a long and stable history as open source project and a broad distribution. In the case of OpenNebula, the middleware supports interfaces to address other Cloud infrastructures via servers. This allows for an elastic scaling of needed computing demands by using either the proprietary EC2 interface (e.g. to submit VMs to the Amazon Cloud) or the open quasi-standard OCCI [9].

Based on this basic model, we can realize not only a private GeneCloud infrastructure, but also extend our computing needs to other infrastructures – such as the local HPC cluster – and realize a hybrid Cloud solution for our services. This is a big advantage in terms of interoperability, especially for smaller companies to avoid a dependency to a particular Cloud provider. With such a flexible approach even a federated Cloud [10,11] could be realized in order to build a fully functional community Cloud, maintained by several providers.

1.2.2 Service Infrastructure

There are versatile ways to develop and provide services in the Cloud, either by using vendor-specific platforms to create services or by using available open source APIs like libcloud or Deltacloud, as abstraction layer to a broad range of infrastructure providers. Nowadays, the Cloud middlewares provide APIs to realize services

based on the middleware's functionalities. Due to the interoperability features to other infrastructures, the services remain not restricted to a particular middleware.

Based on our infrastructure, we can realize our services in different ways and stay very flexible, since the aimed GeneCloud services are technically highly heterogeneous. For service operation, we have also examined a community solution based on OpenNebula. This SVMSched project [12] abstracts the service registry in an own server instance and replaces the OpenNebula scheduler by its own one. In this way, the project hides the configuration details from the user, who only needs to specify the hardware options for the VMs or the path to the data by invoking a client. At the end, only the operator of the Cloud infrastructure can clearly define the way the service can operate on the available resources.

1.2.3 Data-centric Security Approach

We described an architecture that does not rely on specific precautions, protocols, standards or components on the provider side to ensure privacy of the supplied data. Therefore, we implement application specific privacy solutions. These are tailored to the algorithms used and security required. The following application sections carry a customized security analysis together with privacyenhancing proposals.

1.3 Use Cases

In the following section, we briefly describe three case studies – from the application and security perspective – for different domains: Patent annotation [13], cancer outcome prediction [14,15], and drug target prediction[16,17]. Please refer to the original studies for detailed information.

1.3.1 Text Mining

Text mining is used to automatically extract knowledge from text for various purposes such as improving document search or discovering new connections between entities. This section describes two different approaches that we are applying to solve these challenges.

Patent analysis. While many different platforms have been developed to offer advanced functionality for literature search (e.g., GoPubMed [18], GeneView [19]), they usually only offer the option to search scientific publications such as the abstracts from the Medline database. Patents might be another valuable application for text mining. Because patented information must not be revealed before application and also putative information can be patented, patents may contain information

earlier than publications. Usually, pharmaceutical companies are concentrating on creating patents rather than publishing papers.

We are developing GoPatents, a new patent search platform that will incorporate advanced search functionality to facilitate the search in 2 million full text patents and classify them into MeSH and IPC-Ontology [13]. The processing of all patents requires approximately 6,000 CPU h, which is nearly 20 times slower than analyzing abstracts for GoPubMed. Computational complexity is linear in the count of sentences of a document, but O(n3) on the sentence level (n is the word count in a sentence), which are longer in patents [20].

Sentence analysis. In the biomedical domain, information about the relations between different entities such as proteins, drugs and diseases is extremely important. However, this information is usually not easily accessible from a database, it has to be collected from many different journal publications. We have developed a system that retrieves a list of statements about the relation between two different entities of the given drugs, proteins and diseases. After a search for two entities, all documents are automatically fetched and all sentences with both entities are extracted. These sentences are ranked according to their importance using a maximum entropy classifier. While such analysis for combinations of drugs, proteins or diseases is already available, we are developing an online application that identifies all counterparts for one known entity. As these calculations are computationally intensive (appr. 3 CPU months for 22 million Medline abstracts), access to a Cloud environment will improve the execution time considerably.

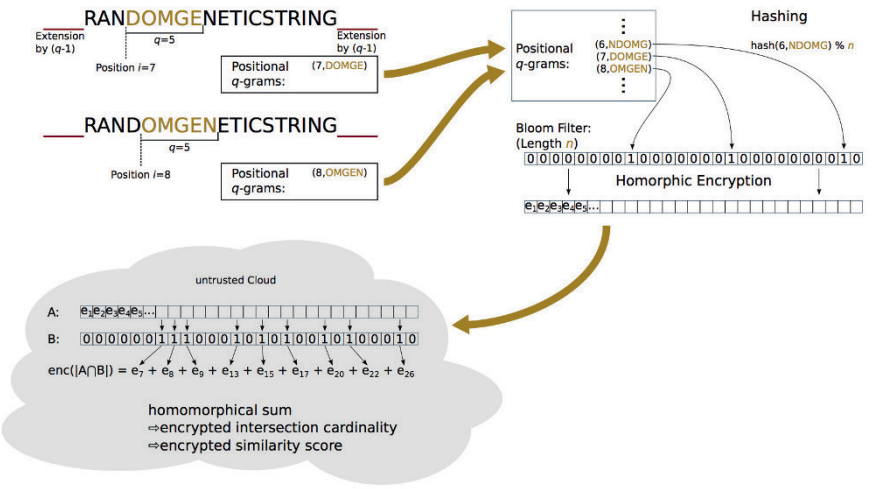


Fig. 1.2: Text matching with Bloom filters

Security Analysis. In such literature and patent search platforms, the query of a user is assumed to be confidential, since the user might validate some experimental

or original results by searching available resources. Hence, such search platforms should guarantee the content of input and output data in the Cloud environment. The algorithms used for extracting information out of literature are rather involved and complex in terms of converting them into privacy-respecting versions. Some useful parts were already discussed within the security community, like homomorphically encrypted machine learning [21].

A simpler primitive that is always needed is secure string comparison as known from private record linkage [22] and DNA matching [23]. All these proposals do not scale well with increased string lengths, so we built a privacypreserving scheme that approximates the Levenshtein Distance and has linear complexity in the longest string [8]. Figure 2 shows an overview of the secure matching algorithm. The results will only be accessible to one of the two parties, which fits nicely into the Cloud Computing scenario, where we want to borrow computational capacity from a remote machine that we do not trust.

In cases where approximate search is not applicable and exact matches are necessary, we apply two solutions to perform secure searches. One version uses the results from Blass et al. [24], who transform huge documents into hashed tables and apply techniques from single-server private information retrieval for deciding if a searched term can be found within such a document. The other solution is a trivial one, which takes documents and search terms as input. Both are split into appropriate tokens: words, word based n-grams or character based n-grams. All tokens are processed using a keyed hash, and sent to the Cloud for executing the actual search – a one to one comparison between all documents and query tokens. The matches are returned to the client. This solution is potentially faster compared to Blass et al. [24]. Building upon these exact and approximate matching blocks, privacy-preserving versions of simple text mining algorithms can be constructed to securely extract information out of confidential literature like full text corpora of patents or scientific publications.

1.3.2 Cancer Outcome Prediction

Cancer outcome prediction aims to address the problem of learning how to forecast disease progression from gene expression and thus allows refined treatment options. The gene signatures obtained in such analyzes could be addressed as biomarkers for cancer progression. We developed an algorithm – called NetRank – that employs protein-protein interaction networks and ranks genes by using the random surfer model of Google's PageRank algorithm. It has been shown, that NetRank improves the outcome prediction accuracy by 7% for pancreas cancer patients [14] and provides an average 6% accuracy improvement on a larger benchmark dataset, consisting of 13 different cancers [15]. The NetRank algorithm assigns a rank to a gene that is based on a combination of its score (gene expression) and the scores of genes linked to it. Thus, the rank r_j^n of gene j in the n^{th} iteration is defined as:

$$r_j^n = (1-d)s_j + d\sum_{i=1}^{N} \frac{m_{ij} r_j^{n-1}}{deg_i} 1 \leq j \leq N \tag{1.1}$$

where $d \in (0,1)$ is a parameter describing the influence of the network on ranking of genes, s is the gene expression value, $m \in \mathbb{R}^{N \times N}$ is a symmetric adjacency matrix for the gene network and deg is the degree of genes in the network.

NetRank performs a supervised learning process combined with a Monte Carlo cross-validation scheme to predict biomarker genes. The main component of the algorithm is a matrix (network) – vector (patient data) multiplication, thus having a computational complexity of approximately $O(N^2)$. Depending on the amount of patients and network size, the running time of one biomarker prediction can easily reach up to 20,000 CPU h. In order to reduce this running time, each cross-validation step is submitted to the Cloud as one worker process. The gene expression data obtained from patients and some type of interaction networks are highly confidential. Therefore, the security of the input data and the predicted biomarkers should be guaranteed in the Cloud environment. Extra computation time is needed depending on applied security approach (e.g. blinding, homomorphic encryption, randomization) on the data. For an efficient cross-validation the network and patient data is loaded into memory, which introduces a memory-related overhead. In the current version,

running the matrix-vector multiplication with a network of approximately 100,000 edges consumes at least 1GB of memory. For multiple networks this might be potentially growing up to TBs, making the memory the major bottleneck of the algorithm. Since these factors cause the algorithm to be very memory intensive, an optimization is necessary before efficiently running it on a public Cloud. Security Analysis. At the core of the NetRank algorithm are two important primitives, matrix-vector multiplications and machine learning algorithms working on confidential data. Privacy-preserving solutions for matrix-vector multiplications are available based on different security assumptions like semi-honest multi-party computation [25] or blinding [26,27]. We implement solutions upon blinding, which achieves a high efficiency, increased privacy, but still leak some information. For data randomization we transform the vector-like patient data in a trusted environment by addition with a random vector. Afterwards we efficiently calculate the matrix-vector multiplication in the untrusted Cloud environment. The blinded result is transferred back to the trusted systems and decrypted via subtraction by a prepared un-blinding vector (Figure 1.3).

Another implemented solution is based on additive homomorphic encryption [28]. The encrypted approach is perfectly preserving privacy, but is less efficient than the blinding solution. It comes in two possible flavours, using an additive encryption scheme like [28] to protect one side of the matrix multiplication, or a scheme from pairing-based cryptography [29] that protects all input data with less efficiency. The machine learning part of NetRank was also targeted by specific privacypreserving solutions. Graepel et al. [21] propose a new class of machine learning algorithms based on leveled homomorphic encryption. New binary classifiers are build with a multiplicative circuit depth of low polynomial degree. This ensures rather efficient evaluation through somewhat homomorphic encryption

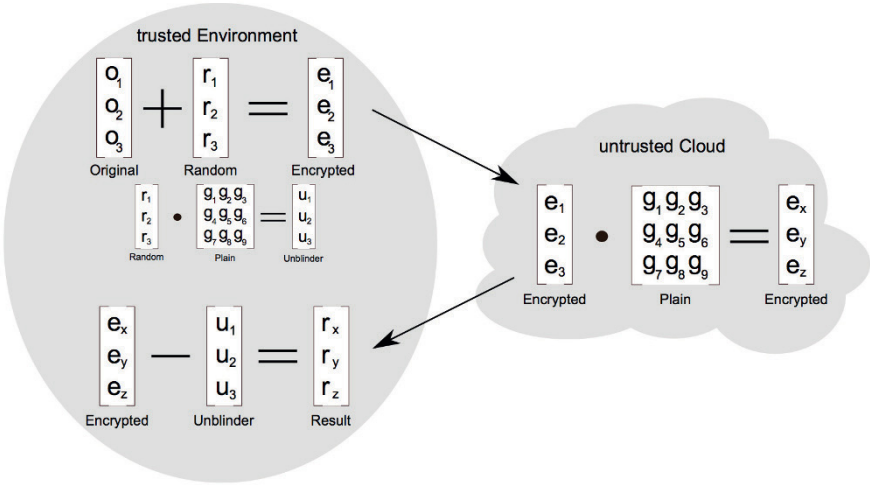


Fig. 1.3: Randomization of vectors

schemes. Another efficient and less privacy-preserving solution based on blinding is to randomize the data, while preserving important properties like the order and approximate relative distance. The blinding based approach also yields comparable classification results on the original data with nearly no performance impact. The achievable privacy margin is on the opposite rather low.

1.3.3 Drug Target Prediction from Binding Site Similarity

Drug development is a very costly and failure-prone endeavour with about 81% of the drugs failing [30]. Major causes for failure are lack of drug efficacy and off-target binding. Thus, drug target prediction is a valuable tool in support of drug discovery helping to predict disadvantageous off-targets and finding new uses for already known drugs. The latter strategy is termed drug repositioning (saving up to 40% of drug development costs [31]) and yielding treatment options for tropical and rare diseases [16]. We implemented a structural bioinformatics drug target prediction and assessment pipeline. It uncovers local similarities between pairs of protein structures and scores the alignments according to the spatial proximity of the bound drug ligands [17].

Protein structures and bound drugs are represented in atomic coordinate files from the Protein Data Bank (PDB). Atoms are points in space and sparsely (implicitly or explicitly) connected by atomic bonds. The spatial atom arrangement is abstracted to a graph, with groups of atoms as nodes and distances between the atoms as edge labels [16,32]. The local alignment of a pair of protein structures is

then given by the maximum weight common subgraph of their graph representation abstracted from the molecular structure.

Here, we faced two problems: First, this task is computationally expensive since the subgraph search is NP-hard (100 CPU h per 1000 alignments on average) and second, n = 2,284 PDB structures had to be aligned pair-wise (giving $n_a = n(n-1)2^{-1} = 2,607,186$ alignments) and read from disk. For performance reasons, operations on files were performed on a RAM disk, which can optionally be encrypted for good security. The n_a alignments $\{a_1, ..., a_{n_a}\}$ were split into j jobs such that the number of accessed coordinate files was minimized for each job, thus reducing concurrent read operations to a minimum. This was achieved by splitting the upper triangular matrix of the n_a alignments into j equally sized submatrices. The submatrices were optimized in size due to the triangular form of the full matrix. Suppose such a submatrix has $lm \approx \frac{n_a}{j} = n_a^{'}$ elements with $l \geq m$ and $l - m \leq 2$. Thus, $l + m = n'$ coordinate files have to be loaded to the RAM disk to compute $n_a^{'}$ alignments. In total, the number of disk read accesses for j jobs is bound by $jn' = j(l+m) \leq 2jl = 2j\frac{n}{j} = 2n \in O(n)$, which is optimal. Thus, the disk read overhead is constant with a factor of 2 and independent of the number of jobs j. In contrast, a naive approach without the RAM disk would need $n_a \geq O(n^2)$ disk reads. However, j has to be set such that n' coordinate files can be stored on the RAM disk.

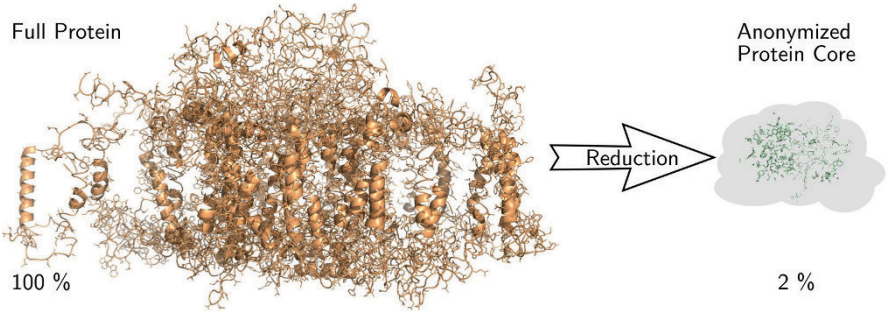


Fig. 1.4: Full protein structures are anonymized locally for processing in the Cloud.

After having computed the local structure alignments, successful alignments were scored by the quality of the spacial superposition of their ligands. This measure is termed LigandRMSD [17] and computes the difference in root mean square deviation (RMSD) in comparison to an optimal alignment of the ligands.

Security Analysis. We first tried to build a privacy-enhanced version of the binding site-alignment step discussed in the previous section. However, the used tools and algorithms are not available for modification and can only be used in a black-box setting, preventing the development of a privacy-preserving version. As it is still essential to enhance the security while working with black-box algorithms, we analyzed the input and output data to reduce it to a minimum. Protein structure de-

termination is costly and time consuming. Thus, unpublished structures have to be secured appropriately. Besides that, predicted drug targets are potentially valuable and should be secured. In case of the binding site alignment tool, structural information about the compared proteins are supplied as input in form of PDB files. From these files, we remove all but the essential information as follows: First, remove all lines which are no atom or ligand entries and all entries with coordinates further than $d\mathring{A}$ away from any ligand's center of mass, followed by a removal of all entries which are further than $d' < d\mathring{A}$ away from any ligand entry. Second, permute atoms randomly per amino acid and amino acids per chain. Third, rotate and translate all entries randomly and set all columns except for the coordinates, chain, atom and residue number.

We applied this strategy to a set of proteins and removed on average 91% of the structural information, while the binding site alignment was still calculating correct results. Figure 4 shows an example of what remains after the anonymization step. A test set of 2,171 protein structures was anonymized this way. To test, how easy these structures can be recovered, we ran a BLAST search over all PDB entries. This search returned the anonymized structure on average at position 25, while 182 were not found. Thus, this approach offers no full security, but instead hides the results making it tedious for an attacker to recover the actual hits.

1.4 Results and Conclusions

We described three different biomedical applications (text/patent mining, cancer outcome prediction and drug target prediction) in a Cloud environment. The underlying infrastructure was set up particularly with these applications in mind and the employed Cloud middleware is purely based on open standards and capable of easy deployment of virtual machines e.g. to the local HPC cluster or other Cloud providers. Particular focus was put on how to secure these approaches to work privacy-preserving. This is important when computing on valuable research data or patient data. We applied different approaches for securing the computation as well as the input/output data: Homomorphic encryption, hiding, blinding and anonymization.

We secured the text mining algorithms using approximate matching and homomorphic encryption such that information can be securely retrieved from confidential texts without increasing the computational complexity. The cancer outcome prediction – running on confidential medical data – was secured by blinding in one approach and by additive homomorphic encryption in another. Although the latter offers the highest security while still being in $O(n^2)$, we preferred the first for production use. This is due to a constant factor in the order of 106 introduced by the homomorphic encryption, making it slow in practice. The drug target prediction approach involves black-box like libraries, which could not be secured. Instead, we concentrated on reducing the input data to a minimal amount leading to an anonymization of the input coordinate files. This makes the mapping of the results

back to the original data tedious for an attacker without increasing computational complexity.

As the proposed security solutions are independent of the actual Cloud infrastructure, the services can be run on virtually any Cloud platform. This allows more flexibility, interoperability and efficiency compared to platform dependent solutions. The implemented solutions are tailored to the applications, but the ideas can easily be adopted to build tailored security solutions for similar tasks. As a result, the security risk in using public Cloud computing upon private data decreases and new applications might arise.

1.5 Acknowledgements and Funding

Funding by the German Federal Ministry of Economics and Technology is kindly acknowledged as GeneCloud is part of the Trusted Cloud Initiative.

References

1. Krishnan, A.: Gridblast: a globus-based high-throughput implementation of blastin a grid computing framework. Concurrency and Computation: Practice and Experience 17(13) (2005) 1607–1623
2. Morgan, J.C., Chapman, R.W., Anderson, P.E.: A next generation sequence processing and analysis platform with integrated cloud-storage and high performance computing resources. In: Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine. BCB '12, New York, NY, USA, ACM (2012) 594–594
3. Ren, K., Wang, C., Wang, Q.: Security Challenges for the Public Cloud. IEEEInternet Computing 16(1) (January 2012) 69–73
4. Santos, N., Gummadi, K.P., Rodrigues, R.: Towards trusted cloud computing. In: Proceedings of the 2009 conference on Hot topics in cloud computing. HotCloud'09, Berkeley, CA, USA, USENIX Association (2009) 3
5. Bugiel, S., Nürnberger, S., Sadeghi, A.R., Schneider, T.: Twin Clouds: Secure Cloud Computing with Low Latency. In: 12th Communications and Multimedia Security Conference (CMS'11). Volume 7025 of LNCS., Springer (2011)
6. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings ofthe 41st annual ACM symposium on Theory of computing. STOC '09, New York, NY, USA, ACM (2009) 169–178
7. Yao, A.C.C.: How to generate and exchange secrets. In: Foundations of ComputerScience, 1986., 27th Annual Symposium on. (1986) 162–167
8. Beck, M., Kerschbaum, F.: Approximate two-party privacy-preserving string matching with linear complexity. (2013)
9. Open Grid Forum Working Group: OCCI – Open Cloud Computing Interface (2009)
10. Buyya, R., Ranjan, R., Calheiros, R.: InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services. In Hsu, C.H., Yang, L., Park, J., Yeo, S.S., eds.: Algorithms and Architectures for Parallel Processing. Volume 6081 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2010) 13–31
11. Rochwerger, B., Breitgand, D., Epstein, A., Hadas, D., Loy, I., Nagin, K., Tordsson,J., Ragusa, C., Villari, M., Clayman, S., Levy, E., Maraschini, A., Massonet, P., Mu andoz, H.,

Tofetti, G.: Reservoir - When One Cloud Is Not Enough. Computer 44(3) (March 2011) 44 –51
12. Chakode, R., Yenke, B.O., Mehaut, J.F.: Resource Management of Virtual Infrastructure for On-demand SaaS Services. In: CLOSER2011: Proceedings of the 1st International conference on Cloud Computing and Service Science. (May 2011) 352–361
13. Eisinger, D., Tsatsaronis, G., Bundschus, M., Wieneke, U., Schroeder, M.: Automated patent categorization and guided patent search using ipc as inspired by mesh and pubmed. Journal of Biomedical Semantics 4(Suppl 1) (2013) S3
14. Winter, C., Kristiansen, G., Kersting, S., Roy, J., Aust, D., Knösel, T., Rümmele, P., Jahnke, B., Hentrich, V., Rückert, F., Niedergethmann, M., Weichert, W., Bahra, M., Schlitt, H.J., Settmacher, U., Friess, H., Buühler, M., Saeger, H.D., Schroeder, M., Pilarsky, C., Grützmann, R.: Google goes cancer: improving outcome prediction for cancer patients by network-based ranking of marker genes. PLoS Comput Biol 8(5) (2012) e1002511
15. Roy, J., Winter, C., Isik, Z., Schroeder, M.: Network information improves canceroutcome prediction. Brief Bioinform (Dec 2012)
16. Haupt, V.J., Schroeder, M.: Old friends in new guise: Repositioning of knowndrugs with structural bioinformatics. Brief Bioinform 12(4) (Jul 2011) 312–326
17. Haupt, V.J., Daminelli, S., Schroeder, M.: Drug promiscuity in PDB: Protein binding site similarity is key. PLoS One (2013)
18. Doms, A., Schroeder, M.: Gopubmed: exploring pubmed with the gene ontology.Nucleic Acids Res 33(Web Server issue) (Jul 2005) W783–W786
19. Thomas, P., Starlinger, J., Vowinkel, A., Arzt, S., Leser, U.: Geneview: a comprehensive semantic search engine for pubmed. Nucleic Acids Res 40(Web Server issue) (Jul 2012) W585–W591
20. Verberne, S., D'hondt, E., Oostdijk, N.: Quantifying the Challenges in ParsingPatent Claims. 1st International Workshop on Advances in Patent Information Retrieval (2003)
21. Kwon, T., Lee, M.K., Kwon, D.: ML Confidential: Machine Learning on EncryptedData. Information Security and Cryptology ICISC 2012 Lecture Notes in Computer Science 7839 (2013) 1–21
22. Bonomi, L., Xiong, L., Chen, R., Fung, B.: Privacy Preserving Record Linkage viagrams Projections. arXiv preprint arXiv:1208.2773 (2012)
23. Katz, J.: Secure text processing with applications to private DNA matching. Proceedings of the 17th ACM conference on 1 (October 2010) 485–492
24. Blass, E., Pietro, R.D., Molva, R., Onen, M.: PRISMPrivacy-Preserving Search in MapReduce. Privacy Enhancing Technologies (2012)
25. Atallah, M.J., Frikken, K.B.: Securely outsourcing linear algebra computations. In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security - ASIACCS '10, New York, New York, USA, ACM Press (April 2010) 48
26. Atallah, M., Pantazopoulos, K., Spafford, E.: Secure outsourcing of some computations. Computer (1996) 1–24
27. Atallah, M.J., Pantazopoulos, K.N., Rice, J.R., Spafford, E.H.: Secure outsourcingof scientific computations. ADVANCES IN COMPUTERS (1998) 215 – 272
28. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree ResiduosityClasses. Advances in Cryptography - Eurocrypt '99 1592 (1999) 223–238
29. 29. Boneh, D., Goh, E.J., Nissim, K.: Evaluating 2-DNF formulas on ciphertexts. Theory of Cryptography (TCC) '05 3378 (2005) 325–341
30. DiMasi, J.A., Feldman, L., Seckler, A., Wilson, A.: Trends in risks associated withnew drug development: success rates for investigational drugs. Clin Pharmacol Ther 87(3) (Mar 2010) 272–277
31. Chong, C.R., Sullivan, D.J.: New uses for old drugs. Nature 448(7154) (Aug 2007) 645–646
32. Xie, L., Bourne, P.E.: Detecting evolutionary relationships across existing foldspace, using sequence order-independent profile-profile alignments. Proc Natl Acad Sci U S A 105(14) (Apr 2008) 5441–5446

Chapter 2
Sealed Cloud – A Novel Approach to Safeguard against Insider Attacks

Hubert A. Jäger, Arnold Monitzer, Ralf Rieken, Edmund Ernst, Khiem Dau Nguyen

Abstract Security and privacy have turned out to be major challenges of the further Internet evolution in general and cloud computing, in particular. This paper proposes a novel approach as to how to to safeguard against previously unimpeded insider attacks, referred to as Sealed Cloud. A canonical set of technical measures is described, which, in conjunction, sufficiently complicate and thus economically prevent insider access to unencrypted data. This paper shows the advantages versus end-to-end encryption relative to communication services. Another application of the Sealed Cloud, referred to as Sealed Freeze, provides a seminal solution to privacy issues pertaining to data retention.

2.1 Introduction

For a long time, IT security concerns have focused on perimeter security, assuming the providers of software as a service (SaaS), clouds, and cloud-based services to be trustworthy. However, data theft and privacy violation statistics [12], [9] reveal that at least every fourth harmful attack originates from within providing organizations. This data only confirms what many potential customers of SaaS and cloud based offers already sense regarding data security. Therefore, mission critical applications are not outsourced to cloud resources, and privacy preserving services have not been established on a significant scale, to date [4]. In other words, integrated security is absolutely essential, as recently postulated by many IT security experts, e.g. [6]. Is data created outside the cloud, then client encryption of this data provides basic security. However, is data to be generated within the cloud, then the demand for a technical breakthrough protecting user data processed by providers is imperative.

Hubert A. Jäger · Arnold Monitzer · Ralf Rieken · Edmund Ernst · Khiem Dau Nguyen
Uniscon universal identity control GmbH, Agnes Pockels-Bogen 1, 80992 Munich, Germany
e-mail: {hubert.jaeger,arnold.monitzer,ralf.rieken,edmund.ernst,Khiem}@uniscon.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_2

The present proposal was elaborated within the development framework of a Web privacy service [11], where, in an SaaS architecture, the data security exigence was extended to also consistently embrace the server components. Once this condition precedent was fulfilled, the resultant technical measures proved to equally solve the issue in general computing infrastructure.

Outline

The remainder of this article is subdivided as follows. Section 2.2 gives account of previous work. The Sealed Cloud proposal is presented in Section 2.3. The advantages of the novel concept for communications and web privacy services, as well as data retention technologies, is elaborated in Section 2.6. Section 2.7 refers to some legal aspects. Finally, Section 2.8 presents the conclusion.

2.2 Previous Work

In literature, there are several approaches as to how to secure computing infrastructure by employing Trusted Platform Modules (TPM), e.g. [5] or [18] for improved software integrity. In [7], a closed-box execution environment is used, to protect the virtual machines against unauthorized access by an administrator. According to [2], this method has not been implemented, yet.

These approaches secure the software's integrity and thus substantially restrict administrators' liberty to abuse infrastructure and data but do not fundamentally prevent access to unencrypted user data during processing. E.g., if the operation kernel of a processor fails or is provoked to fail, unencrypted data is written to core dumps.

Similar ideas to clean up data as the ones presented in this paper, when perimeter security is surpassed, may be found in literature on tamper-proof hardware, e.g. [3].

The only somewhat comparable alternative to Sealed Cloud known to the authors to date is (fully) homomorphic encryption [17], [19] and [8]. However, this enabling technology (still in stage of research) discloses all meta-data or connection data (i.e., who communicates with whom, how much and when) to the operators of these services. This is also valid for all end-to-end client encrypting services. Even if "mixing networks" (e.g. [20]) are used to access an infrastructure, which computes on encrypted data, the operator can see which operations are dependent on each other. Thus, these alternatives do not meet secure cloud computing requirements to a sufficient degree.

Hence, in practice, controls as per ISO/IEC 27002, for example, which are integrated into elaborated information security management systems pursuant, e.g., to ISO/IEC 27001, are implemented on an organizational (yet not primarily technical) level.

The following proposal to technically protect against insider attacks is a set of innovative technical measures yet employs off-the-shelf physical components only. It has been implemented for a concrete Web privacy service, and currently enjoys prototype development for generic use.

2.3 Approach

A processing infrastructure is assumed, hosting applications that process sensitive, critical or personal data.

Sensitive, critical or personal data is considered any data related to users or subject matter the users deal with using such applications and deemed worthy of protection against unauthorized access.

Unauthorized access is specified as any access of a party having no business directly related to the business logic of an application nor legally justified access rights.

Unauthorized parties are external attackers but may also be internal service or infrastructure operator staff, provided that no human interaction is needed for an application's business logic. Often, the user of an application and a legal entity are the only persons to have authorized access within the narrow scope of applicable law.

The following proposes a set of technical measures aimed at protecting sensitive, critical or personal data from unauthorized access. It is essential that said protection of sensitive and mission critical application data be sufficiently effective by technical means only. I.e., it is paramount that potential impact of human deviance be minimized.

Basic Idea

Due to the fact that current computing is normally only secured via state-of-the-art perimeter protection and, in crucial cases, additionally protected by a comprehensive set of measures insuring software integrity, infrastructure administrators and the administrators of the hosted applications still have access to unencrypted sensitive, critical or personal data.

Of course, operators of such infrastructure and respectively implemented services are well aware of this weakness and tend to complement protection of unencrypted processing data via organizational, non-technical means, i.e., respectively defined processes and staffing with upright personnel they deem trustworthy.

A good example of named full set of procedures is described in [16] or in the ISO/IEC 2700X standards. The aforementioned elaborates the best combination of technical, formal and informal measures, to maximize security.

In contrast, our proposal replaces this non-technical makeshift by commensurate key distribution and tailored data clean-up procedures.The latter measures, when

combined with perimeter security and software integrity, can close contemplated gaps. Thus, with Sealed Cloud, no unencrypted processing data is easily accessible to unauthorized parties.

Key Distribution

Let's assume that all data stored on persistent memory is encrypted. In order to avoid that this encrypted data is accessed by the operator of the infrastructure or the operator of the services in unencrypted form, it is necessary to either (a) use an encryption method, in which the operator (once the data is encrypted) is not able, in turn, to decrypt the information, e.g., asymmetric encryption, or (b) delete the encryption key from the processor's memory, as soon as encryption is completed. The latter method is appropriate if the encrypted information is to be again used at a later point in time in unencrypted form.

These methods allow distribution of power among the various parties involved in an application's business logic.

The most straightforward use case consists of user data encryption in the database of the service deployed in Sealed Cloud, with a key derived from credetials, provided by the client of the application. If the data is again to be used in the cloud at a later point of time, no asymmetric key is used, and, consequently, the application has to delete the key, once the session or another unit representing the interaction with named data is completed. In subsection 2.4.2 details of a preferred implementation are given.

A further use case comprises an application, which needs to provide access to specific data for a third party, e.g., when access of a business partner of the client is intentional, to ensure data access needed for partnership with the client organization. Such data can be encrypted in the Sealed Cloud with a business partner's public key, exported in encrypted form to the partner, and, once there, safely decrypted with the partner's private key.

2.4 Architecture and Implementation Example

2.4.1 System Overview

Figure 2.1 illustrates an implementation example of the described set of measures. The cloud user's personal computers or other electronic devices are connected to Sealed Cloud, which is run by the cloud operator. The application software, i.e., IDGARD [10], executed in Sealed Cloud, was developed and produced by the application operator, Uniscon GmbH, and was examined and certified by one or multiple external auditors (ongoing process with TÜV Informationstechnik GmbH), before

it was deployed in Sealed Cloud. All players' domains of control are indicated in Figure 2.1 with dashed lines, respectively.

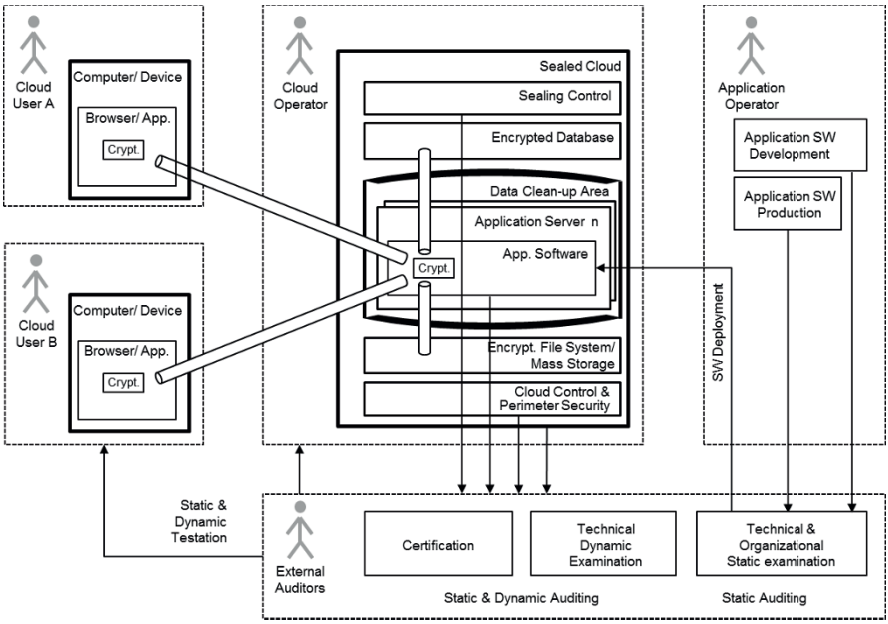


Fig. 2.1: A sample implementation of the canonical set of measures for a Sealed Cloud infrastructure

The structure of Sealed Cloud in this implementation is depicted in Figure 2.1 within the domain of the cloud operator. It consists of a so-called data clean-up area in the center (emphasized by two boldly printed "sealing" bows at the bottom and the top of the area) and the database and encrypted file system, as well as peripheral sealing and cloud control.

When the user connects to Sealed Cloud, an encrypted communication channel from the browser or any other application running on the user's personal computer or device is established to one of the application servers in the data clean-up area, pursuant to well-known standard procedures, e.g., secure socket layer protocol. The selection of the actual application server is performed by load distributing mechanisms, implemented within the routers and servers of the cloud control unit, which also hosts the state-of-the-art mechanisms for perimeter security, such as firewall and intrusion detection and prevention. A fundamental feature consists in the necessary shared secret or certificate for this encrypted connection (for the purposes of Sealed Cloud) being unknown to the cloud operator but under the control of an external auditor, who deployed a non-reverse-engineerable software agent on each application server. For practical purposes, this can be approximated by code ob-

fuscation [1]. Furthermore, each of these agents is individually produced for each respective application server, so that its execution is possible only on the individual server with the server's specific fingerprint.

The sensitive, critical or personal data is then processed in unencrypted form in the application server. For persistent storage, the data is encrypted with a key derived from the user's login credentials at the beginning of the session. The application software deletes these login credentials the instant the storage key is generated. External auditors focus on this deletion procedure, in particular. The data is then stored in the database in encrypted form. In the next session, the application software generates the key from the login credentials anew, so the data may be read back from the database. At the end of each session, this derived key is also deleted. This procedure is also a main focus of examination through external auditors. The data encryption keys in the mass storage may be stored in the encrypted data, which, in turn, is stored in the database.

Access to the unencrypted data during processing within the data clean-up area is prevented by the data clean-up method. The following illustrates this method as per implementation example in Figure 2.1: The sealing control unit monitors a comprehensive set of sensors and agents running on all system components, to detect an access attempt to the Sealed Cloud infrastructure. In the event that the data clean-up area is accessed without authorization, the affected application servers immediately stop operation and delete any unencrypted data. For the purpose of simplification, the data clean-up area of this implementation example contains volatile memory only. The deletion procedure is, e.g., brute forced by power-down of the affected application servers. This applies to both logical and physical attempts to access the data clean-up area. Such reversal of priorities, that privacy is ranked even higher than high availability, leads to such system behavior. In the event of authorized access, e.g., for maintenance purposes, the same data clean-up mechanism is triggered only once access rights (authorization, system availability requirements, et al.) are confirmed by a state-of-the-art access control system.

When starting or restarting the application servers or other components of the Sealed Cloud system, integrity of the system must be verified. A chain of trust is established that embraces the entire stack (from server hardware to highest application software layers), employing, e.g., as shown in the herein mentioned implementation, the individual server's fingerprint as root of said chain. Moreover, if previous unauthorized access is detected or external power supply is used, the application servers can not be started.

2.4.2 Key Management

2.4.2.1 Client-IDGARD Connection and User Management

When the user connects to Sealed Cloud, a connection according to secure socket layer protocol (SSL) is established. IDGARD is continuously adjusted to the de-

velopment of new cipher codes and web servers and deals with these, to achieve best ratings and use cipher codes, preferably with Forward Secrecy (FS). Currently (September 2013), strong cipher codes with good key length (e.g. ECDHE_RSA_WITH_AES_256_CBC_SHA256, ECDH 571 bits, which is equivalent to 15360 bits RSA) based on Diffie-Hellmann key exchange are employed, to ensure the FS property for the SSL connection. In addition, a browser add-on is provided, which checks the IDGARD / Sealed Cloud certificate and finger print, to ensure that no man-in-the-middle attack remains undiscovered. I.e., if such an attack is detected, an alarm is triggered for the user.

At the beginning of the session, the user enters a user name (UN) and password (PW). In the application servers, which are situated in the data clean-up area, the following is triggered using these two user credentials: Both UN and PW are each hashed (#) with SHA-256 and then concatenated, to form the user identification string (IS) and search for an appropriate entry for the user in the data base. If this user identification string exists in the data base, the user is authenticated. Figure 2.2 illustrates these operations.

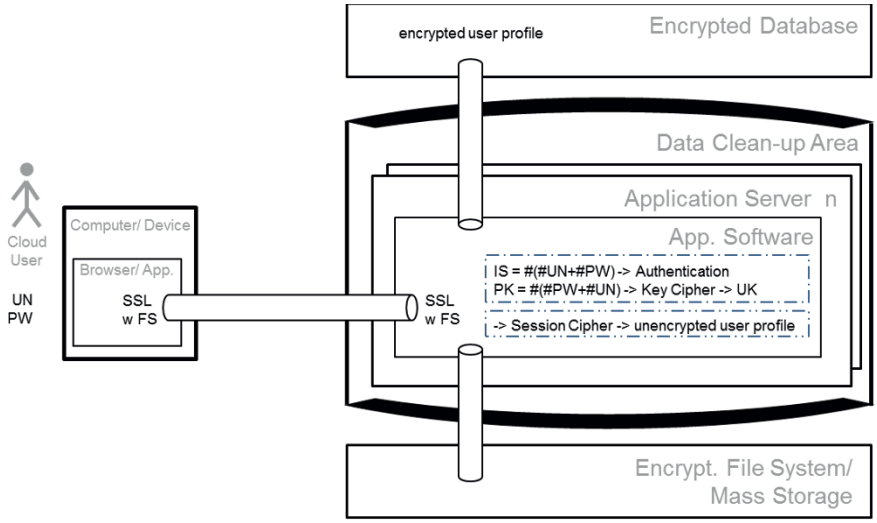


Fig. 2.2: Authentication of IDGARD users and the creation of the session cipher

Subsequently, UN and PW are again used to generate (also via hashing with SHA-256 and concatenation yet with a different rule) the profile cipher, to decrypt the user key (UK). This user key is then used to generate a session cipher code to decrypt the user profile, and, after modification, also encrypts the user profile with AES-256. It is a java development kit (JDK) property that these ciphers can not be stored persistently and are therefore not recoverable. At the end of this step, all content in the single dot dashed box (– - –), i.e. the UN, the PW, the user IS, and

the profile cipher, are deleted from the volatile memory of the application server in Sealed Cloud. The session cipher and the unencrypted user profile in the double dotted-dashed box (– - - –) are also deleted from the volatile memory of the application server at the end of each session. Both deletion processes are a focal point of the technical audit of Sealed Cloud. Please note that each user profile is encrypted with separate keys via AES-256. There is no such thing as a system key in IDGARD / Sealed Cloud.

2.4.2.2 Privacy Box Keys

Once the session is established, the user is granted access to his private data domain, i.e., the so-called Privacy Boxes and their content, using the aforementioned session cipher, to read the respective content.

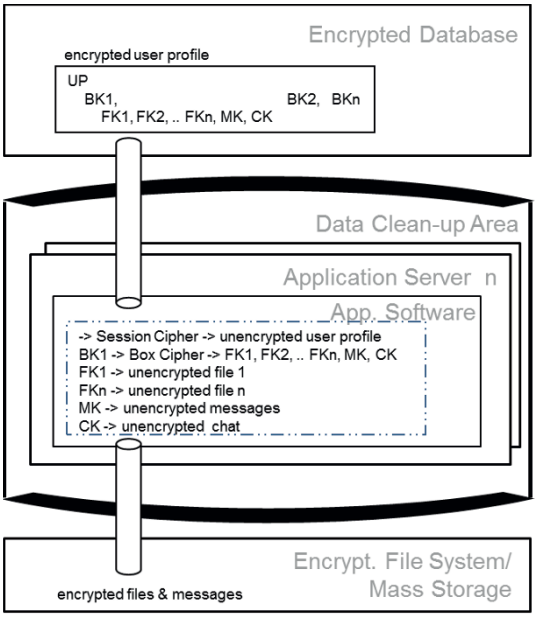


Fig. 2.3: Box encryption scheme in IDGARD

The aforementioned session cipher is used for this to read the user profile (UP) and the box keys (BKi) in the UP, in which a different key is used for each box. Consequently, a box cipher from the BKi is created which is then used to decrypt the various file keys (FKi) and the box specific message key (MK), as well as the box specific chat key (CK). Figure 2.3 illustrates these operations. In other words,

each file has its own key, and each box its own key for messaging and chats; these are all for AES-256 encryption.

2.4.2.3 Substituting the Vulnerable Channel by a Vulnerable Phase

Normally, in order to share data by inviting others to a shared Privacy Box, the owner of a box would need to transmit a secret to new box members via channels of uncertain security, i.e., often vulnerable channels. When the secret is copied by an eavesdropper along the vulnerable channel, the eavesdropper can also (assuming he has knowledge of the corresponding data base entry) decipher the shared content. IDGARD, on the other hand, uses a more sophisticated approach. It uses, in contrary, a vulnerable phase, allowing the box to be opened for joining the box and closed (or sealed) immediately thereafter, as described in Subsection 2.4.2.4. To open a Privacy Box, a so-called box link is generated, which is a URL, containing the BK. It is created from a URL trunk and the unencrypted BK, as depicted in Figure 2.4.

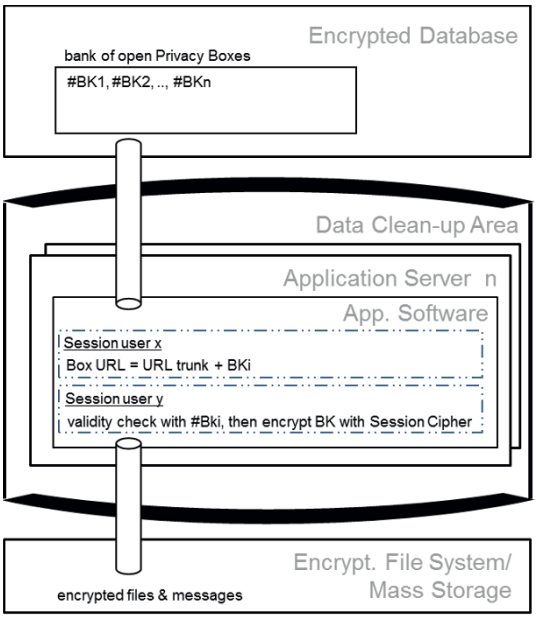


Fig. 2.4: Sharing Privacy Box access with a box link

Once this box link is transmitted via channels of uncertain security from user x to user y, user y can enter this box link in IDGARD. The application servers then check whether the hashed (SHA-256) value of the box link #BKi can be found in the bank

of hashed box links and is therefore valid. This setting is depicted in Figure 2.5. If the box link is valid, user y becomes a new member in said Privacy Box, and carries the BKi in his (user y) profile from that moment onwards.

2.4.2.4 Closing or Sealing of a Privacy Box

The closing or sealing of a Privacy Box is illustrated in Figure 2.5.

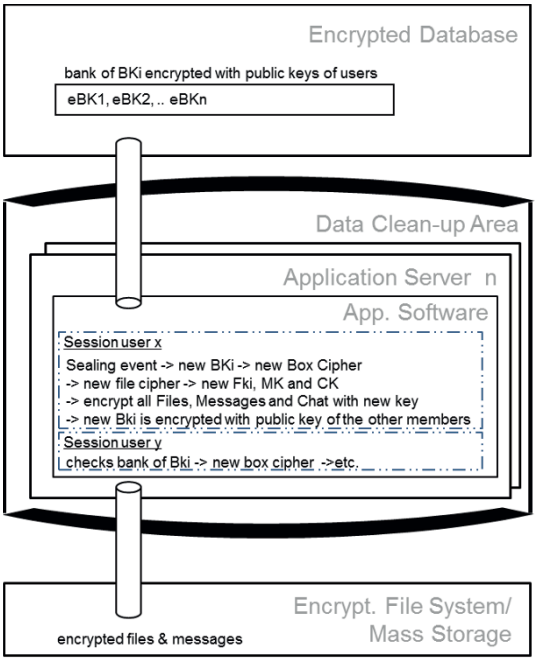


Fig. 2.5: Sealing of a Privacy Box (blue to green)

The click of a user on the seal button in IDGARD's GUI is a sealing event. Exhaustion of the access quota specified by the owner of the open Privacy Box (marked blue in GUI), i.e., access expiration turning to "0", also constitutes a sealing event. When such sealing occurs, a new BK and a new box cipher are generated. Then, a new file cipher and new keys are created for all files, as well as the respective messages and the chats. With these, all files, messages and chats are encrypted anew. The old encrypted files, messages and chats are deleted in the file system. Finally, the new BK is encrypted with the public key of the IDGARD / Sealed Cloud internal public key infrastructure (PKI) and added to the bank of encrypted BKi in the data base. The color of the Privacy Box in the GUI changes in IDGARD from blue to green.

When another member of the Privacy Box (e.g., user y) finds the box in the sealed state, IDGARD checks the bank of BKi for the encrypted BKi (eBKi), decrypts it with the private key contained in user y's UP, creates the new box cipher, and can thus once again read the content.

Such sealed (or green) boxes are then only accessible by the members of the box. No box key is available to any administrator, operator or user who is not a member of the Privacy Box.

2.4.2.5 Sharing Content within a Managed Account (Business Account)

Each Privacy Box constitutes a communication circle of users; they can also be viewed as secured data rooms. A simplified mechanism is available for users who are members of a managed account, i.e., a so-called Business Account, to join a Privacy Box. Within a Business Account, members are administered by the owner of the account. The method for sharing Privacy Boxes in Business Accounts is depicted in Figure 2.6.

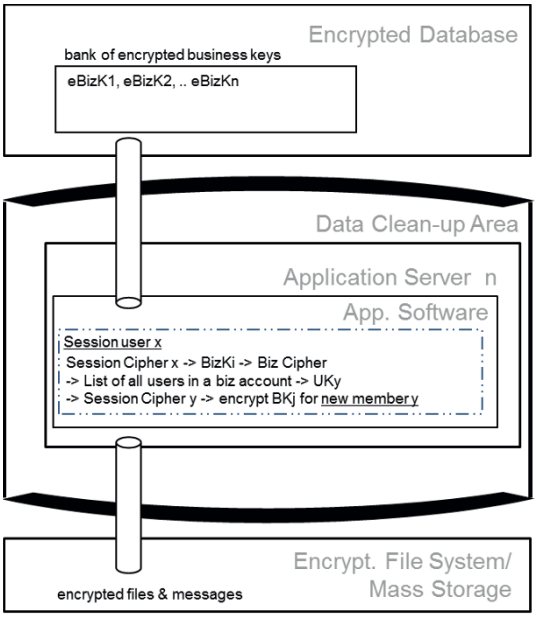


Fig. 2.6: Sharing Privacy Box access within a Business Account

In the session of user x, the session cipher is able to read a business key (BizKi) from the bank of encrypted business keys (eBizKi). A business cipher (biz cipher) is derived from this key, which allows decryption of the UK of all users of the business

account i. Given user y should be added to the list of members of a particular privacy box j, then session cipher y is generated, and the respective BKj is stored in the profile of user y in encrypted form.

With this method, the communication circles in the Privacy Boxes can be established in a very fast and efficient way: The owner or creator of a Privacy Box simply clicks on the names of the users that should be members of the box, and all users are instantly included in the Privacy Box without having to first open and then close or seal the box. The Privacy Box remains sealed (indicated green in GUI) at the highest security level at all times.

2.4.3 Data Clean-Up

The Sealed Cloud database does not contain unencrypted data. Pursuant to business logic, the key to said data is only available for the party owning it. However, unencrypted data is inevitably found in the volatile memory of the processing infrastructure. The application servers in the data clean-up areas of IDGARD / Sealed Cloud contain no persistent writable memory. The servers are operated without disks. Authorized access to the servers (e.g., for maintenance purposes) must be considered. Unplanned access cannot be excluded, either, since perimeter security can, in most cases, set off an alarm when intrusion is detected but not always prevent it effectively.

Data clean-up, as implemented in IDGARD and illustrated in Figure 2.7, ensures that planned or unplanned access to the volatile memory is not possible until sensitive, critical or personal data has been deleted.

The sealing control unit generates trigger signals for planned access upon access request by the cloud operator. For maintenance work on a specific segment of IDGARD / Sealed Cloud, a ticket is issued by a 3rd party Trust Center and forwarded in a protected form by means of access code to the maintenance engineer's mobile device. The access code is sent to the system's sealing control unit via Bluetooth, in order to initiate the access to said segments. Then, the sealing control unit issues the commands to the cloud control unit depicted in Figure 2.7, to clean said segment from active user sessions and erase all data. To guarantee complete deletion of all user data, the application servers' power supply is turned off automatically and stays off for 15 seconds, before the servers are rebooted. After reboot, all application servers are in clean default state.

In case of unplanned, i.e., unauthorized access, trigger signals are generated by perimeter security, which covers both logical and physical intrusion attempts. Again, the sealing control unit issues appropriate data clean-up commands to the cloud control and application servers and prevents the start-up of servers, where applicable. The sealing control provides the opening signals to the electro-mechanical doors of the respective segment only once the entire data clean-up procedure has been completed.

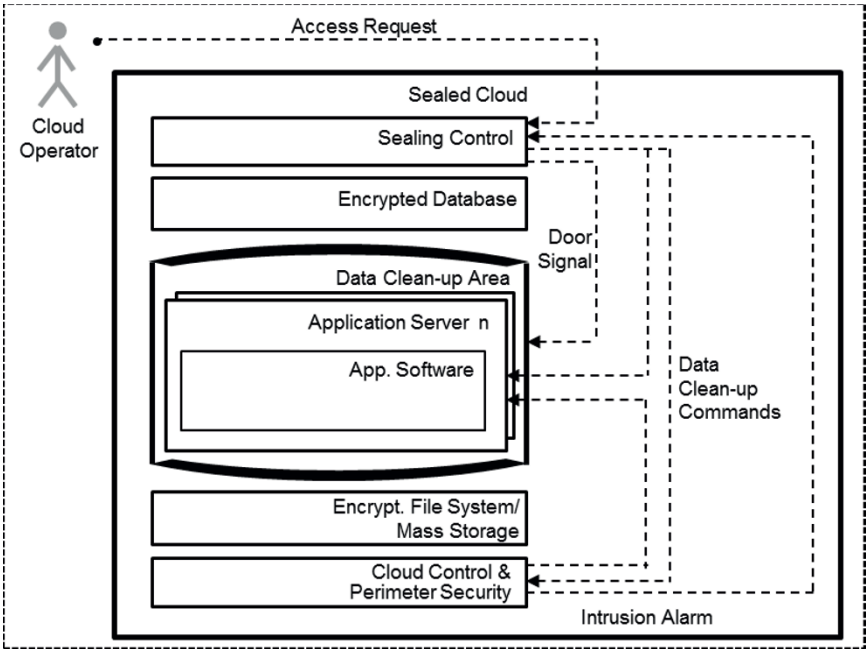


Fig. 2.7: The data clean-up procedure

Such a data clean-up can be implemented using components of the shelf (COTS) as far as hard-and firmware is concerned. That means ordinay electo-magnetically controlled locks, housings, racks, shelters, and cages etc, can be employed.

2.5 Organizational Measures and Audit

The user must be able to trust the Sealed Cloud operator and the application provider, i.e., that the system behaves as claimed and that both hardware and software in the system are trustworthy and execute the specified functions only. Implementation complexity of IDGARD is limited by hierarchical structuring and encapsulation of the system modules, so external auditors, in particular, are able to understand and examine all components and the critical focal points of an audit. Only then can external auditors issue certificates, thus providing the user an expert opinion, to justify trust in the operating parties. Uniscon GmbH works according to BSI IT-Grundschutz rules and uses the BSI Grundschutz-Tool for structure analysis and documentation, both for the organization within Uniscon and for the IDGARD / Sealed Cloud infrastructure. In addition, Uniscon works with TÜV Information-

stechnik GmbH, to dynamically observe system behavior and certify dynamic system integrity by dynamic attestation methodology.

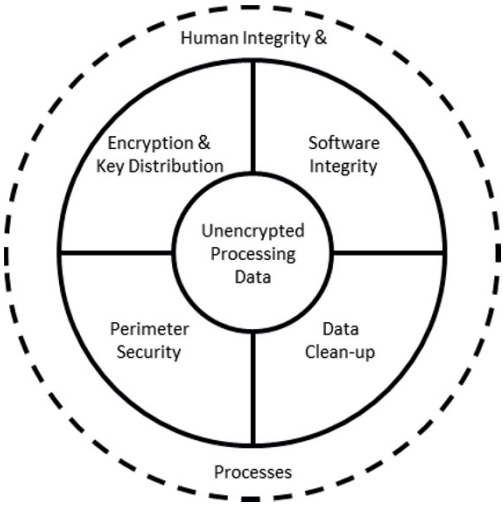


Fig. 2.8: Organizational measures are shifted to the second line of defence

To further improve the coverage of examination by external auditors, they employ software agents, to dynamically observe system behavior and issue dynamic system integrity attestation for the user. Hence, despite the fact that the technical measures 'key distribution' and 'data clean-up' sufficiently complicate insider access to unencrypted processing data and therefore protect against insider attacks, organizational measures are needed, to secure a proper auditing and certification process by external auditors. That means that human integrity and processes are still important for the operation of the overall Sealed Cloud. However, this set of measures is shifted to the second line of defense, as illustrated in Figure 2.8.

Core Principle

The core principle underlying present proposal, is to implement a set of appropriate technical measures, to enforce the distribution of power between various parties. Such distribution of power (concerning the access to data), of course, only works, as long as no malicious coalitions are built between the various parties. The probability of such coalitions decreases, the less the external auditors depend on the operators and the more they depend on the users. This stipulates that no monopoly, neither for the operator nor for the auditor, is acceptable.

Canonical Set of Measures

The presented set of measures is classified as canonical, because the entirety of technical measures, serving the purpose of protecting the unencrypted processing data, can be classified into the presented four categories ”perimeter security”, ”software integrity”, ”key distribution” and ”data-clean-up”. Despite the various measures’ dependency, each technical measure can be unambiguously categorized into one of the given groups of measures.

2.6 Applications

As mentioned in Section 8.1, the Sealed Cloud concept was elaborated to develop a Web service designed to protect user privacy. The properties and a fundamental privacy advantage of such a service, in particular, compared to end-to-end-encryption, is described as a first application example in this section. The second application example was also developed in this connection. Cases with an obligation to court-ordered disclosure of data, e.g. connection data in telecommunications systems, constituted the design of Sealed Freeze.

Web Privacy Services

Web privacy services empower the user to enjoy the opportunities of modern networking technology, while pro-actively maintaining user privacy alike. Sealed Cloud is an enabling technology, generating trust in web privacy services.

The Web Privacy Service IDGARD is the first privacy service to offer Sealed Cloud infrastructure. With a proxy function and additional measures as part of the application on the application servers, the source address and other identifying parameters of the user device are disguised, to allow the user pseudonymous visits of websites. A certain number of users of such a service is necessary, for the individual user to be hidden among numerous fellow users. Further, Sealed Cloud can securely host user names and passwords safely, to provide convenient and secure on-line authentication. Finally, shared storage allows a full range of communication services, such as e-mail, chat, file sharing, etc. The latter use case is illustrated in Figure 2.9. On the left, communication is depicted between users A-D via standard communication services. The connection data, i.e., who is connected with whom, when, and how much data is trafficked, is visible to the operator of the standard communication service. In contrast, a Sealed Cloud based communication service, as depicted on the right of Figure 2.9, does not disclose any of this connection data to the service operator.

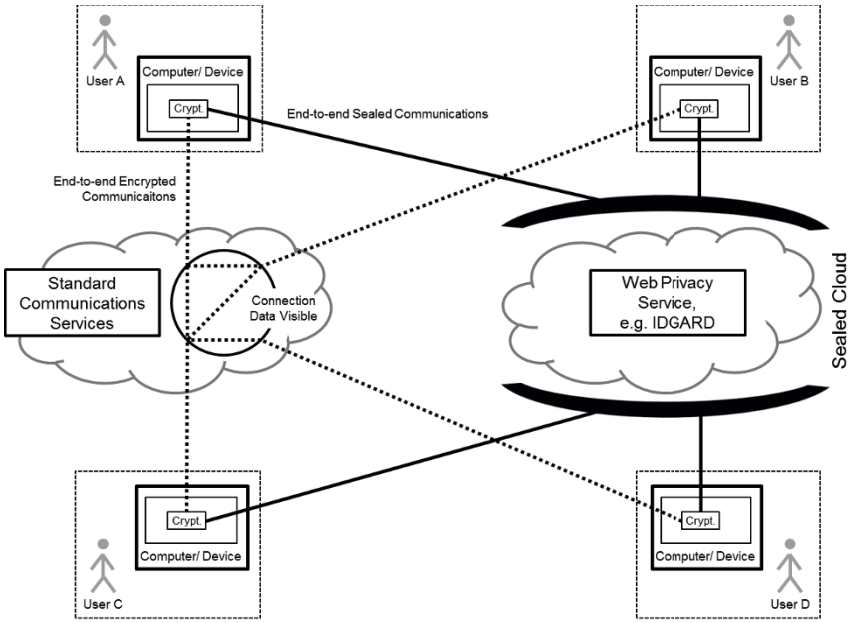


Fig. 2.9: Sealed cloud also ensures connection data privacy

Sealed Freeze

Vis-a-vis legal philosophy, aforementioned web privacy services ultimately ensure free democratic order. However, to prevent these services from degenerating to hiding places for criminals or terrorists, a method for authorized persons to be able to access connection data within a very restricted constitutional framework is imperative. Yet, the property that the operators, technically, have no access to this data, has to be held upright. Moreover, the strict rules of the tight constitutional framework of justified access should be enforced, technically.

Figure 2.10 depicts the basic concept of Sealed Freeze. Any relevant data acquisition and processing system, e.g. telecommunications networks, social networks or video surveillance systems, to name only a few, feature data acquisition devices and a system to transport the data to a storage area. With Sealed Freeze, a key store generates pairs of asymmetric keys, keeps them in volatile memory only[1], and provides the public key to the data acquisition devices. These encrypt the data to be retained block by block, each with a specific public key, respectively, and then forward the encrypted data to the storage area. In case court-ordered or other authorized persons are legally obliged to access the retained data, they can request the matching private

[1] These keys are in fact stored in volatile memory only, in the sense that there is no persistant storage for these keys. This has of course implications; however is the core idea of Sealed Freeze approach.

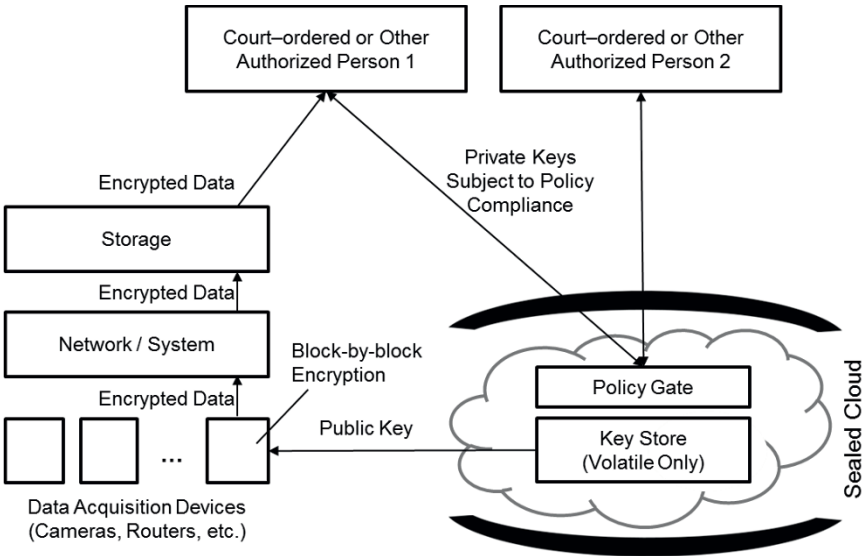


Fig. 2.10: Sealed Freeze based on Sealed Cloud technology: an approach as to how to resolve privacy issues regarding data retention

keys from Sealed Freeze. The policy gate in Sealed Freeze will disclose the matching private keys only if the request fulfills the policy rules, as defined by lawmakers in advance and as programmed into the policy gate. The policy cannot be changed with retroactive effect, since all keys are deleted during deployment of a new policy. The policy can contain rules regarding a four-eyes principle, maximum storage duration, volume of disclosure, flows of disclosure within a given period of time, et al. The rule set within the policy can be chosen in a way that no dragnet investigation is possible, because the number of private keys to be disclosed is limited. Through the rule defining that private keys be deleted after a specific amount of time, deadlines can be enforced, technically. Here, too, Sealed Cloud is the enabling technology that resolves privacy issues.

2.7 Legal Aspects

The concept of Sealed Cloud has a significant impact on the placement of public cloud computing from a legal point of view. This applies at least for German law, yet is similar in other legal systems. Without the elaborated operator-proof approach, public cloud computing cannot be used without violation of professional secrecy, because (1) the mere technical feasibility to read professional secrets is sufficient to constitute disclosure and (2) IT staff of the public cloud operator im-

plementing organizational protection measures cannot be regarded as abettors under German criminal law (StGB) [14]. With Sealed Cloud, the placement of public cloud computing is different, because technical feasibility to access professional secrets is sufficiently complicated. Thus, using Sealed Cloud does not imply disclosure of professional secrets and therefore complies with Germany law even for persons carrying professional secrets [13]. Table 2.1 shows a summary of legal applicability for public cloud types.

Cloud Type	**Business Use**	**Business with Professional Secrets**
Public Cloud (depending partially on organizational measure to secure access to user data)	Only compliant if German Bundesdatenschutzgesetz (BDSG) is obeyed (i.e. Auftragsdatenverarbeitung gem. §11)	Not compliant; exception proposals in preparation [15]
Sealed Cloud	BDSG compliant	StGB (§203), WP (§43), et.al., compliant

Table 2.1: Legal Applicability for Cloud Types

2.8 Conclusion

The present proposal is a good example of an integrated security approach in information technology. Unauthorized access of any kind is effectually complicated by technical means and thus prevented efficiently. Unauthorized parties include the service and infrastructure operators. The resultant Sealed Cloud therefore constitutes an unrivaled, trustworthy processing infrastructure for clients of hosted applications, as opposed to the user having to rely on the mere trustworthiness of the provider.

The present paper is a proposal, opening a field of research regarding the suggested measures' implementation options. Fields of interest are, in particular, software integrity in environments with virtual engines and approaches to reliable data clean-up in standard cloud application interfaces.

The Sealed Cloud prototype infrastructure is pursued by Uniscon GmbH, Fraunhofer Institute of Applied and Integrated Security, and SecureNet GmbH, and is co-funded by the German Ministry of Economics and Technology within the framework of the so-called Trusted Cloud initiative [21].

References

1. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S., Yang, K.: On the (Im)possibility of Obfuscating Programs. In: J. Kilian (ed.) Advances in Cryptology – CRYPTO '01, volume 2139 of Lecture Notes in Computer Science, pp. 1–18. Springer (2001)
2. Brunette, G., Mogull, R., editors: Security Guidance for Critical Areas of focus in Cloud Computing v2.1. Cloud Security Alliance (2009)
3. Bryant, E.D., Atallah, M.J., Stytz, M.R.: A Survey of Anti-Tamper Technologies (2004). URL `https://www.cerias.purdue.edu/assets/pdf/bibtex_archive/2004-55.pdf`
4. Catteddu, D., Hogben, G., Perilli, A., Manieri, A., Algom, A., Rhoton, J., Rohr, M., Biran, O., Samani, R.: Cloud Computing: Benefits, Risks and Recommendations for Information Security. European Network and Information Security Agency (ENISA) (2009)
5. Dawoud, W., Takouna, I., Meinel, C.: Infrastructure as a Service Security: Challenges and Solutions. In Informatics and Systems (infos). In Informatics and Systems (INFOS), 2010 The 7th International Conference on Informatics and Systems (INFOS) p. 1 to 8 (2010)
6. Eckert, C.: Itk-Kompendium 2010. in: Marlene Neudörffer (Hrsg.), IT-Sicherheit der nächsten Generation – Herausforderungen und Entwicklungen, FAZ-Institut (2009)
7. Garfinkel, T., Pfaff, B., Chow, J., Rosenblum, M., Boneh, D.: Terra: a Virtual Machinebased Platform for Trusted Computing. In Proceedings of the nineteenth ACM symposium on Operating systems principles, SOSP'03 p. 193 to 206 (2003)
8. Gentry, C.: Computing Arbitrary Functions of Encrypted Data (2008). URL `http://crypto.stanford.edu/craig/easy-fhe.pdf`
9. Holmlund, L., Mucisko, D., Kimberland, K., Freyre, J.: 2010 Cybersecurity Watch Survey: Cybercrime Increasing Faster than some Company Defenses. Carnegie Mellon University, Software Engineering Institute, CERT Program (2010)
10. www.idgard.de (2013)
11. Jaeger, H.A., Monitzer, A.: Device for Generating a Virtual Network User. Patent application WO 2010/084017 (January 22nd 2009)
12. Keeney, M., Kowalski, E., Cappelli, D., Moore, A., Shimeall, T., Rogers, S.: Insider Threat Study: Computer System Sabotage in Critical Infrastructure Sectors. Carnegie Mellon University, Software Engineering Institute, CERT Program (2005)
13. Kroschwald, S.: Anwendung des Daten- und Geheimnisschutzrechts auf "betreibersichere" Clouds am Beispiel der "Sealed Cloud". In: J. Taeger (ed.) Law as a Service (LaaS): Recht im Internet- und Cloud- Zeitalter, vol. 1, pp. 289–308. Oldenburger Verlag, Edewecht (2013)
14. Kroschwald, S., Wicker, M.: Kanzleien und Praxen in der Cloud - Strafbarkeit nach § 203 StGB. Computer und Recht **11**, 758–764 (2012)
15. Leutheusser-Schnarrenberger, S.: Regelungsbedarf bei Cloud Computing in Kanzleien. AnwBl 2012 **6**, 477 (2012)
16. Mishra, S., Dhillon, G.: Defining Internal Control Objectives for Information Systems Security: A Value Focused Assessment. In: W. Golden, T. Acton, K. Conboy, H. van der Heijden, V.K. Tuunainen (eds.) 16th European Conference on Information Systems, pp. 1334–1345. Galway, Ireland (2008)
17. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. in Advances in Cryptology. EUROCRYPT'99, LNCS, Volume 1592 p. 223 to 238 (1999)
18. Santos, N., Gummadi, K.P., Rodrigues, R.: Infrastructure as a Service Security: Challenges and Solutions. In Informatics and Systems (infos). In Proceedings of the 2009 Conference on Hot Topics in Cloud Computing, HotCloud'09, Berkeley, CA, USA (2009)
19. Smart, N.P., Vercauteren, F.: Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes. In Proceedings of the Conference on Practice and Theory in Public Key Cryptography (2010)
20. Syverson, P., Reed, M., Goldschlag, D.: Anonymous Connections and Onion Routing. Proceedings of IEEE Symposium on Security and Privacy, Oakland, CA pp. 44–54 (1997)

21. Bundesministerium für Wirtschaft und Technologie (BMWi) Referat Entwicklung konvergenter IKT, D.Z.f.L.u.R.e.P.i.D.: Sichere Internet-Dienste – Sicheres Cloud Computing für Mittelstand und öffentlichen Sektor (Trusted Cloud). Ein Technologiewettbewerb des Bundesministeriums für Wirtschaft und Technologie, http://www.bmwi.de (2010)

Chapter 3
Side Channels in Secure Database Outsourcing on the Example of the MimoSecco Scheme

Matthias Huber and Gunnar Hartung

Abstract Cloud Computing has many advantages like flexibility and reduced costs of IT infrastructure. Privacy issues, however, remain a major drawback. A client outsourcing its data loses control over it. In this paper, we present MimoSecco, a novel technique for secure database out- sourcing. We provide a security notion, a formal security proof. Furthermore, we identify side channels that apply to many secure database outsourcing schemes.

3.1 Introduction

Cloud Computing offers many opportunities such as lower cost of IT-infrastructure, more flexibility, and minimized risk of data loss due to specialized providers. Security concerns, however, are a huge obstacle for the adoption of cloud computing. A client that uploads data to the cloud cannot control if it is copied or misused. A malicious system administrator, for example, may copy sensitive data and sell it to competitors. There may even be legal requirements forcing providers to disclose data of their clients to government agencies.

Currently, most security measures focus on external attacks. Secure channels prevent external adversaries from eavesdropping and authorization and authentication mechanism prevent access of unauthorized users. Measures against internal adversaries mostly try to restrict physical access to servers to authorized staff. Examples like PRISM or Wikileaks, however, show that legal insiders also have to be considered.

In this paper, we present the MimoSecco [1] scheme, a novel approach to secure database outsourcing. We provide a formal definition of the transformation MimoSecco applies to databases and a security notion for secure database outsourcing:

Matthias Huber · Gunnar Hartung
Institut für Kryptographie und Sicherheit Karlsruher Institut für Technologie
e-mail: matthias.huber@kit.edu, gunnar.hartung@student.kit.edu

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_3

Indistinguishability under independent column permutations (Ind-ICP). Informally, this notion hides relations in databases. After a formal proof that MimoSecco fulfills this notion, we provide a security analysis. We demonstrate that even if a scheme provides provable security, there are still issues to be considered. We show that these so called *side-chanels* are not specific to MimoSecco, but as well apply to other secure database outsourcing schemes.

The structure of this paper is as follows: In Section 3.2, we discuss related work and provide the formalisations used throughout this paper. We provide a formal definition of the MimoSecco transformation as well as the definition of the security notion Ind-ICP, and the security proof in Section 3.3. In Section 3.4, we provide a security analysis of MimoSecco and show side channels that also apply to other schemes. Section 3.5 concludes.

3.2 Preliminaries and Related Work

The problem of *secure database outsourcing* was defined by Hacigümüş et. al [8] in 2002. Here, a client wants to outsource a database to an untrusted server. The server shoud learn as little as possible about the database, while queries to the outsourced database should still be executed efficiently. Since then, different schemes for secure database outsourcing have been proposed.

In this section, we provide an overview over the related work. In the last part of this section, we will present notations used throughout this paper.

3.2.1 Related Work

Secure database outsourcing could also be implemented by well-known cryptographic schemes, namely fully homomorphic encryption and secure multi party computations [5, 6]. While providing provable security, however, the huge overhead of these schemes would cancel out the benefits of outsourcing.

Practical approaches find a trade-off between security and efficient support a large number of queries. An efficient and practical database outsourcing scheme should provide support for sub-linear query execution time in the size of the database [11].

To the best of our knowlege, one of the first practical solutions to the problem of secure database outsourcing is from Hacigümüş et. al [7]. It involves row-wise encryption of the original database, an adapter that handels en- and decryption and holds local indices in order to support efficient query execution. Based on this approach, additional solutions have been suggested [4, 9]. The basic idea is to create coarse-grained indices in order to support efficient query processing. They contain keywords, ranges, or other identifiers and row pointers to the encrypted database. Depending on the indices and the queries this can lead to a large overhead. Fur-

thermore, it is unclear what level of security these approaches provide for realistic scenarios.

Another idea found in literature is to achieve privacy by data partitioning. The approaches in [2, 17, 16] introduce privacy constraints that can be fulfilled by fragmenting or encrypting the data [2, 17] or fragmentation and addition of dummy data [16]. In [14], Nergiz et. al. use the Anatomy approach [18] to achieve privacy in a database outsourcing scenario. The level of privacy achieved by these partition-based approaches depends on the actual data and user defined privacy constraints. They do not provide any security notion or analysis beyond that.

Different to the approaches described above is CryptDB [15]. It does not rely on explicit indices or fragmentation but uses the concept of onions of encryptions. Different encryption schemes with different properties are nested with the clear text in the middle. If a query cannot be executed because of the current encryption level of an attribute, the the outermost layer of all entries of this attribute is removed. Therefore, the security of this approach depends on the type of queries issued to the database.

3.2.2 Notation

We define the notation we use througout this paper in this section. First, we formally define the term database (Definition 1). Then, we define two sets that we need for the construction of the MimoSecco scheme and for the security proof: The set of all values of a given row (Definition 2) and the set of all indices of all rows containing a given value in a given column (Definition 3). For the definition of our security notion, we use the term database transformation that we define in Definition 4.

Definition 1. A database d is a finite ordered multiset of tuples from the same set $M_1 \times M_2 \times \cdots \times M_c$. Let $j \in \{1,\ldots,c\}$ and $i \in \{1,\ldots,|d|\}$: We call a tuple $l \in d$ a row. We denote row i of d with $d(i,\cdot)$. With $d(i,j)$, we denote Element j of row $d(i,\cdot)$. Such an element is also called a value of d. We call tuples of the form $d(\cdot,j) := \begin{pmatrix} d(1,j) \\ \vdots \\ d(r,j) \end{pmatrix}$ column j of d. We call *DB* the set of all databases.

We use the terms table and database interchangeably. If we depict databases, we add the tuple $M_1, M_2, \ldots, M_c$ as column names. Throughout this paper, d is a database according to Definition 1, $|d| = r$, $j \in \{1,\ldots,c\}$, and $i \in \{1,\ldots,r\}$.

Definition 2. The set of all values of a column is defined as $\mathscr{V}(d,j) := \{d(i,j) | i \in \{1,\ldots,r\}\}$. Let this set be ordered.

Note, that $\mathscr{V}$ is a set in contrast to a database which is a multiset.

Definition 3. For a value $v \in M_j$, $\mathscr{R}(d,j,v) := \{i \,|\, d(i,j) = v\}$ denotes the set of all indices of all rows containing value v in cloumn j.

Definition 4. A transformation $f : DB \to DB$ is called *database transformation*.

3.3 The MimoSecco Database Outsourcing Scheme

Similar to other database outsourcing schemes [7, 2, 15], MimoSecco relies on an adapter that handels query translation as well as en- and decryption of data. Since the adapter needs the encryption keys, we need to partially trust the server the adapter is deployed to. Figure 3.1 shows an example architecture for MimoSecco with different zones of trust. In secure database outsourcing scenarios, we do not trust database server. The client is trusted, while we need to trust the adapter not to leak data in its volatile memory. If a client issues a query, the adapter translates it to queries for

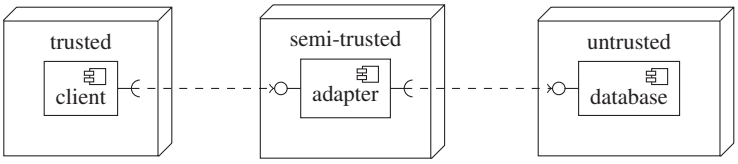


Fig. 3.1: An example deployment scenario for secure database outsourcing. The adapter translates queries issued by the client and decrypts results. Therefore, the adapter needs to be trusted with the encryption keys.

the database. In this section, we will describe the database scheme of MimoSecco as well as the transformation that has to be applied before storing a database on the untrusted database server. In the last two parts of this section, we provide a formal security notion as well as a security proof for the MimoSecco transformation.

3.3.1 Database Scheme

In this section, we define the MimoSecco transformation $\mathscr{M}$ formally. In [1, 10] we already described this transformation informally. In order to support efficient execution of queries, $\mathscr{M}$ stores the values of a database d with c columns and r rows partially redundantly, partially encrypted, and partially in plain text in $c+1$ tables. Therefore $\mathscr{M}$ uses an symmetric encryption scheme $\Sigma = (\text{KeyGen}, \text{Enc}, \text{Dec})$. In the following, let κ be the encryption key.

The first table $\mathscr{M}$ generates is the data table d_{data} which contains a row wise encryption of d as well as an index column.

Definition 5. The data table of $\mathscr{M}(d)$ is defined as:

name	surname
Alice	Smith
Bob	Smith
Alice	Jones

Fig. 3.2: A simple example for a database d. Figure 3.3 depicts $\mathscr{M}(d)$; the result of the MimoSecco transformation of d.

row	data
1	ENC(Alice, Smith)
2	ENC(Bob, Smith)
3	ENC(Alice, Jones)

(a) d_{data}

col_1	rows
Alice	ENC(1, 3)
Bob	ENC(2)

(b) $d_{index,1}$

col_2	rows
Smith	ENC(1, 2)
Jones	ENC(3)

(c) $d_{index,2}$

Fig. 3.3: The database d of Figure 3.2 after the MimoSecco transformation $\mathscr{M}$: The data table d_{data} (a) and the index tables $d_{index,1}$ (b) and $d_{index,2}$ (c). The index tables contain encrypted row indices of the data table.

$$d_{data} := \{(i, \mathrm{Enc}(d(i,\cdot), \kappa)) \mid i \in \{1, \ldots, r\}\}$$

This table is ordered according to the first column and the order of $\mathbb{N}$.

In order to support efficient execution of queries, besides the data table, $\mathscr{M}$ generates c index tables; one for each column. An index table $d_{index,j}$ describes in which rows a value $v \in \mathscr{V}(d,j)$ occurs in column j of d. It contains each $v \in \mathscr{V}(d,j)$ and lists of row indices for each v. The lists of row indices are encrypted.

Definition 6. The index table for column j of a database d is defined as:

$$d_{index,j} := \{(v, \mathrm{Enc}(\mathscr{R}(d,j,v), \kappa)) \mid v \in \mathscr{V}(d,j)\}$$

This table is ordered according to first column and the order of $\mathscr{V}(d,j)$.

With these two definitions, we can define the MimoSecco transformation formally. For a given database d, the MimoSecco transformation generates the data table and the index tables:

Definition 7. Let d be a database . The MimoSecco transformation $\mathscr{M}$ of d is defined as $\mathscr{M}(d) = \{d_{data}, d_{index,1}, \ldots, d_{index,c}\}$.

For an example consider the tables in Figures 3.2 and 3.3 (taken from [1]) Figures 3.2 shows a simple example for a database d that gets transformed into the tables shown in Figures 3.3. The table d has two columns. Therefore, $\mathscr{M}$ generates three tables. Figure 3.3a depicts the data table while Figures 3.3b and 3.3c depict the index tables. As a simple example for a query to such a database consider the query q:

$$q = \sigma_{\text{name = Alice} \wedge \text{surname = Smith}}(d)$$

where σ is the selection operation from relational algebra. The adapter transforms such a query into the subqueries $q_{index,1}$ and $q_{index,2}$:

$$q_{index,1} = \pi_{\text{row}}(\sigma_{\text{col1 = Alice}}(d_{index,1}))$$

$$q_{index,2} = \pi_{\text{row}}(\sigma_{\text{col2 = Smith}}(d_{index,2}))$$

where here, π is the projection operation from relational algebra. As a result, the adapter gets the results $\text{ENC}(1,3)$ from table $d_{index,1}$ and $\text{ENC}(1,2)$ from table $d_{index,2}$. After decryption of these results, the adapter issues the query q_{data}:

$$q_{data} = \pi_{\text{data}}(\sigma_{\text{rows} \in \{1,3\} \cap \{1,2\}}(d_{data}))$$

As a result the adapter gets the tuple ENC(Alice, Smith). The adapter decrypts this tuple and returns it to the client. Note, that all subqueries can be answered in sublinear time regarding the size of d. For some queries, however, (e.g. aggregations) post-processing on the adapter is necessary.

3.3.2 The Security Notion Ind-ICP

In this section, we define the security notion Ind-ICP. Informally a transformation that fulfills Ind-ICP hides relations in a database. In order to formally define our notion, we define an experiment where an adversary should have no advantage compared to random guessing. Game-based security notions are common in cryprography (cf. e.g. the notion Ind-CCA [12]). As defined in Section 3.2.2, let $d \in DB$ be a database with c columns and r rows.

Definition 8. A tuple $\pi = \{\pi_1, \ldots, \pi_c\}$ of c permutations of the set $\{1, \ldots, r\}$ is called *independent column permutation* of a database d. This tuple is a database transformation that maps d to $d_\pi := \pi(d)$ with

$$d_\pi(i, j) = d(\pi_j(i), j)$$

An independent column permutation permutes the values within each column independently. With this definition, we can define the experiment, we need in order to define our security notion. In this experiment an adversary chooses two databases d_0 and $d_1 = \pi(d_0)$. Then, the adversary has to distinguish the tranformations $f(d_0)$ and $f(d_1)$:

Definition 9. Let f be a database transformation, π an independent column permutation, $\mathscr{A}$ an adversary and s a security parameter. The experiment $\text{IND-ICP}_{f,\mathscr{A}}(s)$ is defined as follows:

$$\text{IND-ICP}_{f,\mathscr{A}}(s) := \{$$

$$
\begin{array}{lll}
(d_0, \pi, q) & \leftarrow & \mathscr{A}(1^s) \\
d_1 & \leftarrow & \pi(d_0) \\
b & \leftarrow & \text{random}(\{0,1\}) \\
\overline{b} & \leftarrow & \mathscr{A}(f(d_b), q) \\
\text{return 1 if } b = \overline{b} & & \text{else 0}
\end{array}
$$

}

Whis this experiment, we can define the security notion IND-ICP where an adversary should only have negligible advantage over random guessing in the experiment defined in Definition 9:

Definition 10. A database transformation f is called IND-ICP-secure, if the function

$$\text{Adv}_{\text{IND-ICP}_{f,\mathscr{A}}}(s) := \left| \Pr\left[\text{IND-ICP}_{f,\mathscr{A}}(s) = 1\right] - \frac{1}{2} \right|$$

is negligible for all PPT (probabilistic polynomial-time) adversaries $\mathscr{A}$.

Note, that Definition 10 only considers polynomial bounded adversaries and since in Definition 9, the adversary chooses a database, its size also is polynomially bounded by the security parameter s.

3.3.3 MimoSecco is Ind-ICP

In this section, we prove that the MimoSecco transformation fulfills IND-ICP.

Lemma 1. *The MimoSecco transformation $\mathscr{M}$ is* IND-ICP*-secure under the following conditions:*

1. *The symmetric encryption scheme $\Sigma = (KeyGen, Enc, Dec)$ used by $\mathscr{M}$ is* IND-CPA*-secure.*
2. *For databases d_1, d_2, with $|d_1| = |d_2|$ and $|\mathscr{R}(d_1, j_1, v_1)| = |\mathscr{R}(d_2, j_2, v_2)|$, the representations of $\mathscr{R}(d_1, j_1, v_1)$ and $\mathscr{R}(d_2, j_2, v_2)$ as bitstrings have the same size.*
3. *For a database d, all $d(i, \cdot)$ have length l. l is the maximal possible lenght of all $\pi(d)(i, \cdot)$ for all independent column permutations π.*

We discuss the conditions of Lemma 1 at the end of this section. For the proof, we assume that $\mathscr{M}$ is not IND-ICP-secure and create a contradiction. We construct an adversary that has non negligible advantage in the experiment IND-m-CPA of encryption scheme Σ (m is determined during the proof). Therefore, we define two transformations: One that transforms a database d in a tuple of plain texts, and one that transforms this tuple of ciphertexts into $\mathscr{M}(d)$:

Proof. Let $\mathscr{M}$ be not IND-ICP-secure under the assumptions in Lemma 1. According to Definition 10, there is a PPT adversary $\mathscr{A}$, that has non negligible advantage $\mathrm{Adv}_{\text{IND-ICP}_{\mathscr{M},\mathscr{A}}}(s)$

Let d_0 and d_1 be the databases determined by $\mathscr{A}$ in the experiment IND-ICP with c columns and r rows.

With d_0 and d_1, we now construct the tuples t_0 and t_1 that are similar to the databases returned by $\mathscr{M}$ but without encrypted values. For $i \in \{0,1\}$, let t_i be defined as in Figure 3.4. The first r entries in t_i represent the rows of d_i. The remaining

$$t_i = \begin{pmatrix} d_i(1,\cdot) \\ \vdots \\ d_i(r,\cdot) \\ \vdots \\ \mathscr{R}(d_i,1,v_{1,1}) \\ \vdots \\ \mathscr{R}(d_i,1,v_{1,V_1}) \\ \vdots \\ \mathscr{R}(d_i,c,v_{c,1}) \\ \vdots \\ \mathscr{R}(d_i,c,v_{c,V_c}) \end{pmatrix} \qquad t_b = \begin{pmatrix} \mathrm{Enc}(d_i(1,\cdot),\kappa) \\ \vdots \\ \mathrm{Enc}(d_i(r,\cdot),\kappa) \\ \vdots \\ \mathrm{Enc}(\mathscr{R}(d_i,1,v_{1,1}),\kappa) \\ \vdots \\ \mathrm{Enc}(\mathscr{R}(d_i,1,v_{1,V_1}),\kappa) \\ \vdots \\ \mathrm{Enc}(\mathscr{R}(d_i,c,v_{c,1}),\kappa) \\ \vdots \\ \mathrm{Enc}(\mathscr{R}(d_i,c,v_{c,V_c}),\kappa) \end{pmatrix}$$

Fig. 3.4: The Definition of the tuples t_i and t_b needed for the proof.

entries are sets of row indices for each value in each row. In Figure 3.4, $j \in \{1,\ldots,c\}$ is a column index, the set $V_j := |\mathscr{V}(d_0,j)|$ is the number of different values in column j and $v_{j,p}$ is the p-th value of $\mathscr{V}(d_0,j)$ (recall that $V(d_0,j)$ is ordered). Note that $|\mathscr{V}(d_0,j)| = |\mathscr{V}(d_1,j)|$.

According to Conditions 2 and 3, the binary representation of t_0 and t_1 has equal size element-wise. With this knowledge, we can construct an adversary $\mathscr{A}'$ that has non negligible advantage in the IND-m-CPA of the encryption Σ used by $\mathscr{M}$, with $m = |t_1|$:

First, $\mathscr{A}'$ simulates $\mathscr{A}$. When $\mathscr{A}$ terminates, $\mathscr{A}'$ transforms the tables d_0 and d_1 determined by $\mathscr{A}$ into the tuples t_0 and t_1, returns these tuples and terminates. In the second step of the experiment, $\mathscr{A}'$ recieves a tuple of the form as the tuple t_b defined in Figure 3.4. With this tuple, d_0, and d_1, $\mathscr{A}'$ can construct $\mathscr{M}(d_b)$. Now, $\mathscr{A}'$ simulates $\mathscr{A}$ with the input $\mathscr{M}(d_b)$. $\mathscr{A}$ returns a $\overline{b}$ which $\mathscr{A}'$ also returns. According to our assumptions, the advantage of $\mathscr{A}$ is non negligible, therefore the advantage of $\mathscr{A}'$ also is non negligible, which is in conflict with Condition 1. Therefore, $\mathscr{M}$ is IND-ICP-secure under Conditions 1-3.

This leaves the discussion of the reasonability of Conditions 2 and 3 as an open point. In most cases, integers are represented as bitstings of fixed length (e.g. 16 bit).

Therefore, Condition 2 is automatically fulfilled for most implementations. Fulfilling Condition 3 is more difficult: In general two rows of a table do not have the same length. This can, however, be achieved with padding. For example padding the bistring of each value of a database to a fixed length fulfills Condition 3.

3.4 Side Channels in Secure Database Outsourcing

In the last section, we formally defined a database outsourcing scheme and provided a security notion and a proof, that the scheme fulfills this notion. There are, however, so-called *side-channels* that allow an adversary to break the security of the scheme by exloiting an aspect of reality, that is not in the security model. In this section, we will discuss security problems of database outsourcing schemes on the example of the MimoSecco scheme. The side-channels discussed in this section, however, are not specific to MimoSecco but can be found in many database outsourcing schemes. This raises the question of how to deal with them.

3.4.1 Exclusion of Possible Database Contents

The model in that we have proven the security of the MimoSecco scheme does not consider background knowledge of adversaries. In the experiment $\text{IND-ICP}_{f,\mathscr{A}}(s)$, the adversary has to choose two databases (one directly and one by choosing an independent column permutation). In reality, an adversary would not choose a database, but is presented with the output of $\mathscr{M}$ of an unknown database. If the adversary, however, has background knowledge, she can exclude possible original databases. For an example consider the Tables in Figure 3.5b. The possible original

name	pregnant
Alice	yes
Bob	no

name	pregnant
Alice	no
Bob	yes

(a) d_0 and d_1

col_1	rows
Alice	ENC(1)
Bob	ENC(2)

col_2	rows
yes	ENC(1)
no	ENC(2)

id	data
1	ENC(Alice, yes)
2	ENC(Bob, no)

(b) $\mathscr{M}(d_0) = \mathscr{M}(d_1)$

Fig. 3.5: Possible original databases (a) and an output of the MimoSecco scheme (b). With background knowledge, it is possible to determine the original database.

database for these tables are depicted in Figure 3.5a. An adversary with the background knowledge, that men can not be pregnant, however, can rule out the second database.

This problem is not specific to the MimoSecco scheme. According to the *No-Free-Lunch Theorem* [13] this problem is always present in schemes that allow ad-

versaries to learn something about the database, even if the model captures background knowledge.

3.4.2 Access Statistics

In our model, the Adversary does not know to access statistics. An adversary that has access to the database server can log accesses to the database. Rows in the index and data tables that are queried in short intervals are more probable to be correlated than other rows. Observing access patterns over time, an adversary can learn the parts of the database that are queried often. This of course only works if the queries involve index tables. Attacks involving access statistics, however, are not specific to the MimoSecco scheme. Even if a database is completely encrypted, an adversary with access to access statistics can gain information about the database [11]. This lead to the development of PIR [3] mechanisms and the notion of query indistinguishability [3, 11].

As mentioned above, countermeasures to this attack or involve PIR mechanisms that prevent an adversary from learning the information a client queries from the server. Such mechanisms, however, do have a very high overhead and are not applicable under the performance constraints of most realistic scenarios.

Another method to counter this attack is to rely on architectural assumptions (as in e.g. [2]). For example we can distribute each table to a different server. Assuming they do not collaborate maliciously a single server does not have access to the access statics of the other servers and therefore can not conduct the attack described above.

3.4.3 Order of Values on Physical Storage

Databases are modelled as sets or multisets. We assume databases to be ordered (cf. Definition 1). Real databases, however, do have an order potentially different to the order in the model if stored on physical device. An adversary can exploit this order and potentially break the security notion. This problem, also, is not specific to MimoSecco. Any scheme, that distributes data to different tables (e.g. [14]) is potentially vulnerable to this kind of attack.

In realistic scenarios, application data is stored in the database over time by using the application. So the order of data on the physical device may, therefore, even reflect the order it has been stored in the database. In such a scenario, an adversary may learn correlations of a database transformed by an IND-ICP-secure by simply reconstructing them from the physical order of the entries of the index table on the storage device.

Consider for example Figure 3.5b. If an adversary can infer by the order of the data on the physical storage, that the tuples (Bob, ENC(2)), (no, ENC(2)), and (2,

ENC(Bob, no)) were added after the other tuples, she can infer, that the original database has to be database d_0 depicted in Figure 3.5a.

This attack can be prevented by either enforcing the order of the database and the model to be the same or by enforcing a random order on the physical storage device. This, however, can induce a substantial performance overhead. Depending on the database and the filesystem used, for every insert the adapter has to download the whole database, permute the rows and write them back to the database server. A more feasible solution could be caching changes on the adapter until there are k cached and independent queries. Then, the adapter can execute them in random order. The overall system would then provide Ind-ICP only for blocks of size k in the database.

Another method that prevents this attack is encryption of the index tables. This, however, either restricts the number of queries that can be executed efficiently, or introduces an additional overhead. For example in order to efficiently support substring queries, searchable encryption has to be used for the attribute values. This introduces additional index structures and the need for additional queries as well additional decryption steps on the adapter.

3.4.4 Database Updates

Databases can be altered. For example queries can add or change tuples. Consider for example Figure 3.6. This Figure depicts the states of a database before (Figure 3.6a) and after an update (Figure 3.6a). If an adversary observers these two

col_1	rows
Alice	ENC(1)
Bob	ENC(2)
Eve	ENC(3)

col_2	rows
flu	ENC(1, 3)
cancer	ENC(2)

id	data
1	ENC(Alice, flu)
2	ENC(Bob, cancer)
3	ENC(Eve, flu)

(a) example database before update

col_1	rows
Alice	ENC(1)
Bob	ENC(2)
Eve	ENC(3)
Carol	ENC(4)

col_2	rows
flu	ENC(1, 3)
cancer	ENC(2)
lupus	ENC(4)

id	data
1	ENC(Alice, flu)
2	ENC(Bob, cancer)
3	ENC(Eve, flu)
4	ENC(Carol, lupus)

(b) example database after update

Fig. 3.6: Example database before (a) and after (b) an update. An adversary can determine the new tuple.

states, sche can deterime the new tuple and therefore get information she should not

get according to the security notion. This is possible since the security model does not consider database updates.

This attack can be prevented by already presented methods like distributing the tables to different severs (cf. Section 3.4.2), caching updates or encryption of the index tables (cf. Section 3.4.3).

This side channel should be considered in future security models for database outsourcing. In contrast to usage scenarios for encryption, updates are an important aspect of databases. In order to model this side channel you can introduce *update indistiguishability* to the model. This means that an adversary should not be able to distiguish two updates by seeing their effects on the outsourced database. This however could be to strong. Another possibility is to model that an adversary should not gain additional information through updates that help breaking the security notion.

3.4.5 Active Adversaries

Since we do only consider passive adversaries, MimoSecco does not improve the security of a database system against active adversaries. A malicious database server could for example manipulate results before sending them to the adapter. The server also can attack the availability of the system by supressing answers. Since these attacks can be discovered easily by the client or the adapter, we discuss them only marginally.

3.4.5.1 Manipulation of Results

Instead of returning the correct answer for a query issued to the database, a malicous server could exchange encrypted parts of a result. Consider the example $q_{index,1}$ from Section 3.3.1. The correct answer to this query is $\textsc{Enc}(1,3)$. If the database returns the second row $\textsc{Enc}(3)$ from table $d_{index,1}$ instead of the correct row, the adapter returns a result inconsistent with the query to the client. Although such behaviour can be detected easily, it can compromise the functionality of the client application.

3.4.5.2 Attacks on Availability

A malicious server can delay or withhold answers to queries. As with manipulation of results such behavoiur is detected easily, but also can compromise the client application. The interest of database service providers of denying their service to clients, however, is questionable and depends on the business model. Clients detecting such a behaviour can simply change the provider.

3.5 Conclusion

In this paper, we presented the MimoSecco scheme, a provably secure database outsourcing scheme. We provided a formal security notion and a proof that MimoSecco fulfills this notion. Furthermore, we demonstrated that even if a database outsourcing scheme has provable security, there are still security issues to be considered. Many of issues are not specific to MimoSecco and also apply to other secure database outsourcing schemes. Such so called side-channels in the implementation of a provably secure scheme are present due to the fact that formal models provide an abstraction and cannot capture all aspects of reality as well as all usage scenarios. Furthermore, we discussed possible countermeasures for these side-channels as well as methods to account for them in the security model. In the future, we want to examine models and schemes that account for the security problems discussed in this paper.

References

1. Achenbach, D., Gabel, M., Huber, M.: Mimosecco: A middleware for secure cloud storage. In: D.D. Frey, S. Fukuda, G. Rock (eds.) Improving Complex Systems Today, Advanced Concurrent Engineering, pp. 175–181. Springer London (2011). DOI 10.1007/978-0-85729-799-0_20. URL http://dx.doi.org/10.1007/978-0-85729-799-0_20
2. Aggarwal, G., Bawa, M., Ganesan, P., Garcia-Molina, H., Kenthapadi, K., Motwani, R., Srivastava, U., Thomas, D., Xu, Y.: Two can keep a secret: A distributed architecture for secure database services. In: The Second Biennial Conference on Innovative Data Systems Research (CIDR 2005) (2005). URL http://ilpubs.stanford.edu:8090/659/
3. Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M.: Private information retrieval. J. ACM **45**(6), 965–981 (1998). DOI 10.1145/293347.293350. URL http://doi.acm.org/10.1145/293347.293350
4. Damiani, E., Vimercati, S.D.C., Jajodia, S., Paraboschi, S., Samarati, P.: Balancing confidentiality and efficiency in untrusted relational DBMSs (2003). DOI http://doi.acm.org/10.1145/948109.948124
5. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st annual ACM symposium on Theory of computing, STOC '09, pp. 169–178. ACM, New York, NY, USA (2009). DOI 10.1145/1536414.1536440. URL http://doi.acm.org/10.1145/1536414.1536440
6. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: Proceedings of the nineteenth annual ACM symposium on Theory of computing, pp. 218–229. ACM, New York, NY, USA (1987)
7. Hacigümüs, H., Iyer, B., Li, C., Mehrotra, S.: Executing SQL over encrypted data in the database-service-provider model. In: Proceedings of the 2002 ACM SIGMOD international conference on Management of data, pp. 216–227. ACM (2002)
8. Hacigümüs, H., Iyer, B., Mehrotra, S.: Providing database as a service. In: ICDE '02: Proceedings of the 18th International Conference on Data Engineering, p. 29. IEEE Computer Society, Washington, DC, USA (2002)
9. Hore, B., Mehrotra, S., Tsudik, G.: A privacy-preserving index for range queries. In: VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases, pp. 720–731. VLDB Endowment (2004)
10. Huber, M., Gabel, M., Schulze, M., Bieber, A.: Cumulus4j: A provably secure database abstraction layer. In: A. Cuzzocrea, C. Kittl, D.E. Simos, E. Weippl, L. Xu, A. Cuzzocrea,

C. Kittl, D.E. Simos, E. Weippl, L. Xu (eds.) CD-ARES Workshops, *Lecture Notes in Computer Science*, vol. 8128, pp. 180–193. Springer (2013)
11. Kantarcioglu, M., Clifton, C.: Security issues in querying encrypted data. Tech. rep. (2004)
12. Katz, J., Lindell, Y.: Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series). Chapman & Hall/CRC (2007)
13. Kifer, D., Machanavajjhala, A.: No free lunch in data privacy. In: Proceedings of the 2011 ACM SIGMOD International Conference on Management of data, SIGMOD '11, pp. 193–204. ACM, New York, NY, USA (2011). DOI 10.1145/1989323.1989345. URL http://doi.acm.org/10.1145/1989323.1989345
14. Nergiz, A.E., Clifton, C.: Query processing in private data outsourcing using anonymization. In: Proceedings of the 25th annual IFIP WG 11.3 conference on Data and applications security and privacy, DBSec'11, pp. 138–153. Springer-Verlag, Berlin, Heidelberg (2011). URL http://dl.acm.org/citation.cfm?id=2029896.2029914
15. Popa, R.A., Redfield, C., Zeldovich, N., Balakrishnan, H.: CryptDB: Protecting Confidentiality with Encrypted Query Processing. In: Symposium on Operating Systems Principles (SOSP). Cascais, Portugal (2011)
16. Soodejani, A.T., Hadavi, M.A., Jalili, R.: k-anonymity-based horizontal fragmentation to preserve privacy in data outsourcing. In: DBSec, *Lecture Notes in Computer Science*, vol. 7371, pp. 263–273. Springer (2012). URL http://dblp.uni-trier.de/db/conf/dbsec/dbsec2012.html#SoodejaniHJ12
17. De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Fragments and loose associations: respecting privacy in data publishing. Proc. VLDB Endow. **3**(1-2), 1370–1381 (2010). URL http://dl.acm.org/citation.cfm?id=1920841.1921009
18. Xiao, X., Tao, Y.: Anatomy: simple and effective privacy preservation. In: Proceedings of the 32nd international conference on Very large data bases, VLDB '06, pp. 139–150. VLDB Endowment (2006). URL http://dl.acm.org/citation.cfm?id=1182635.1164141

Chapter 4
ReDS: A System for Revision-Secure Data Storage

Tobias Pöppke and Dirk Achenbach

Abstract The CLOUDwerker project seeks to develop a tool that allows for the collaborative creation of documents in a cloud environment. This neccessitates a secure, non-repudiable document storage layer. We introduce ReDS, a software backend that stores encrypted documents in the cloud. The system also guarantees the non-repudiability of changes, makes older revisions of files accessible and has access control. Our architecture makes use of a trusted master server to store encryption keys and perform authentication and authorization. We implemented ReDS using Python and several open-source components. ReDS is open source and available for download.

4.1 Introduction

Cloud computing has developed from a hype to a widely used technology and business model. Seemingly infinite resources and the possibility to adapt the use of resources to a changing demand make cloud technology attractive even to small and medium enterprises (SMEs). The general fear of industrial espionage and a growing uncertainty regarding the privacy of outsourced data [1] make the technology difficult to adopt, however. Since SMEs are usually not capable of operating their own data centers, public cloud solutions seem practical. Therefore, there is a demand for technological solutions to increase data security and privacy in cloud scenarios.

Builders and trade contractors often work out joint contracts to provide a combined offer which includes the work of several contractors. Cloud storage systems provides a facility to share documents for cooperation. For an efficient workflow perfectly interacting systems are necessary. The CLOUDwerker project aims at providing a solution for this problem. The solution will employ cloud technology in

Tobias Pöppke · Dirk Achenbach
Karlsruhe Institute of Technology
e-mail: {dirk.achenbach,tobias.poeppke}@student.kit.edu

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_4

order to guarantee a high availability and scalability. The partners in the CLOUDwerker project are CAS Software AG, 1&1 Internet AG, Haufe-Lexware GmbH & Co.

KG, Fraunhofer IOA and the Karlsruhe Institute of Technology (KIT). A secure document storage system is an important part of the tool. The documents that are jointly drafted by the users of the system need to be stored in a reliable data store. Public cloud storage solutions usually offer a high level of availability. The CLOUDwerker document storage system must guarantee confidentiality against the storage provider, however. Also, the storage backend must offer versioning, non-repudiation of changes and access control for the users of the system.

Our Contribution We introduce the ReDS system, a document storage system for public cloud stores. It offers the recovery of earlier versions of documents, maintains an append-only revision history, and ensures that modifications to documents are non-repudiable. Further, it ensures the confidentiality, integrity and authenticity of the stored documents. It also offers an access control mechanism to simplify user management. We implemented ReDS using open-source components like Mercurial, Cement and S3QL. We released the sourcecode for ReDS itself under the Apache License (see Section 3).

4.1.1 Related Work

Feldman et al. proposed the SPORC [3] system for collaborative work with untrusted cloud storage. It uses an untrusted cloud server for computation. SPORC uses digital signatures for non-repudiation, but does not support the recovery of earlier revisions of a document. Depot by Mahajan et al. [6] focuses more on the aspect of availability than SPORC and our system. As in SPORC there are no revisions of documents available. Wingu [2] by Chasseur et al. is a system to coordinate concurrent access to cloud storage. Our systems overall design is similar to the one introduced in Wingu, but with the focus on information security. Cryptonite [5] by Kumbhare et al., like our system, uses well-known security techniques to implement a secure data storage on a public cloud. Cryptonite also does not support revision control for documents, however.

4.2 ReDS

In this section we give an overview of the design of ReDS. ReDS employs a master server, which coordinates the access to the documents. The master itself is structured using a layered approach with the connection to the cloud storage as the first layer. The second layer consists of the revision-control system and in the third layer, access control mechanisms are implemented. These layers are also dicussed in this section.

4.2.1 Overview

To meet the requirements for the secure document storage service, we use a master component, similar to the approach by Chasseur and Terrell [2]. The master coordinates the access to the cloud storage and the overall system is therefore designed as a client-server system. This architecture makes it easy to coordinate concurrent access to the documents, which is the primary concern of the contribution by Chasseur and Terrell.

To ensure the confidentiality of the documents, we propose the use of symmetric-key encryption on all data that is stored in the cloud storage. For the purpose of encoding and decoding, the master chooses and stores the used symmetric key locally. This neccessitates the premise that the master is to be trusted regarding the confidentiality of the documents. The clients authenticate themselves to the master by using public-key cryptography. Therefore the master has to have access to the public keys of all authorized clients. Public-key cryptography is also used to digitally sign all documents. The master verifies the signature by using either a public-key infrastructure or by storing the used public-keys of the clients. Refer to 14.1 for a graphical overview of the system.

Fig.1. Overview of the system structure. Clients connect to the cloud store through the master server. The master server stores the symmetric key for data encryption and the public keys of the clients.

The master component is designed using a layered approach with three layers. Therefore the implementation of any layer can be replaced without affecting other layers. The lowest layer has the task to make the cloud storage available to the upper layers. This layer is also responsible for the encryption of the data that is transmitted to the cloud storage provider.

The second layer implements the revision-control system as well as the verification mechanisms for the used digital signatures. By verifying the digital signatures, the integrity and authenticity of the documents is also verified. Furthermore, the version control system also has to coordinate concurrent access to the documents. Figure 2 shows the layers and the security goals they achieve.

Figure 14.2 Overview of the layers in the master and the security goals they achieve. The layered architecture ensures the modularity of our approach.

The clients store their respective private keys for authentication and digital signatures. This implies that the clients are responsible for signing documents they wish to store in the cloud store. The master will refuse to store documents that have no valid signature. The master communicates with the clients over secure communication channels to mitigate the possibility of attacks on the communication. With the proposed design it is possible to combine well-known security mechanisms and concepts and a cloud infrastructure. We now describe the particular layers in detail.

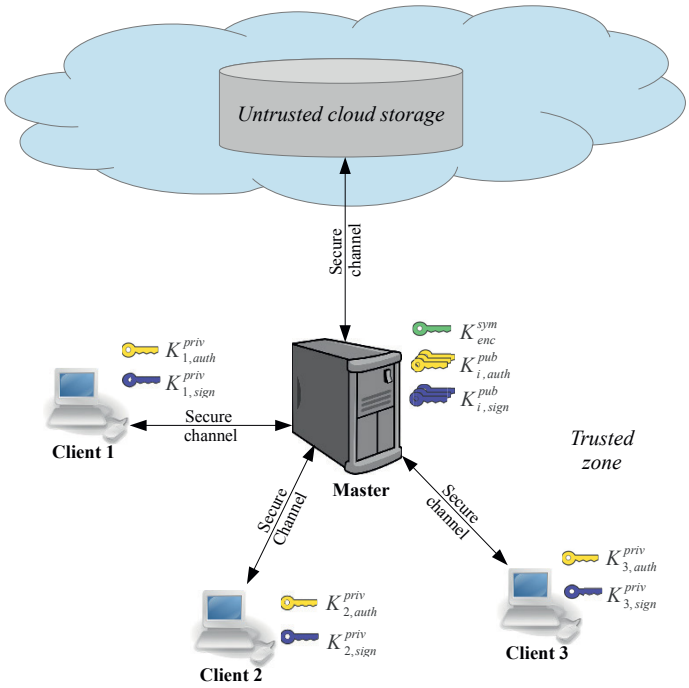


Fig. 4.1: Overview of the system structure. Clients connect to the cloud store through the master server. The master server stores the symmetric key for data encryption and the public keys of the clients.

4.2.2 Storage Connection

To connect the cloud storage with the system, we propose the use of well-known and tested security mechanisms to combine them into a cryptographic filesystem. We use symmetric-key encryption to protect the confidentiality of the data. The symmetric key is stored on the master. With a centrally stored key, it is not necessary to distribute the symmetric key to the clients like in other systems. Therefore it is not necessary to use concepts like broadcast encryption [FN94], which has to be used in systems like SPORC or Cryptonite.

In our system, the master knows the symmetric key of the filesystem encryption and therefore has access to the unencrypted data. Therefore, the master has to be trusted to protect the confidentiality of the data. However, the master has not to be trusted with regard to the integrity of the data, because of the used digital signatures. To check the integrity and authenticity of the data stored in the cloud storage, the cryptographic filesystem uses message authentication codes (MACs). They are used

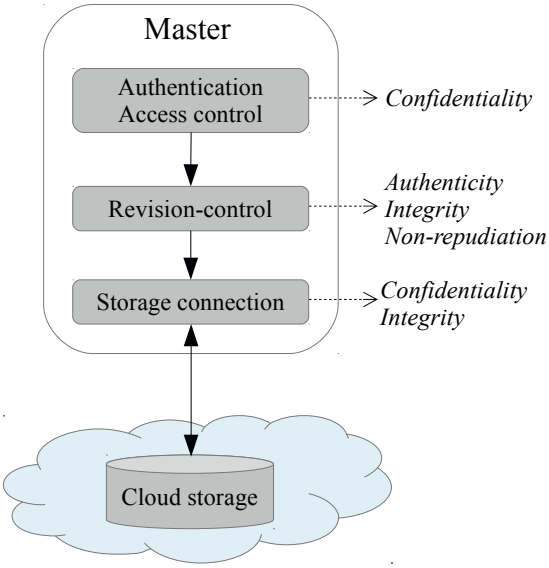


Fig. 4.2: Overview of the layers in the master and the security goals they achieve. The layered architecture ensures the modularity of our approach.

to check that the data in the cloud storage system was not altered. After checking the integrity of the data received from the cloud storage, the master mounts the cryptographic filesystem locally to make it available to the revision-control system.

4.2.3 Revision Control

The revision control system is responsible for tracking every modification of the status of stored documents and making sure that the modification is not repudiatable afterwards. To this end, we combine digital signatures with the concept of hash-trees, which were introduced by Merkle [7]. Hash-trees are used in popular distributed version control systems like git or Mercurial to store the history of a repository and to check its integrity.

Distributed version control systems are also capable of efficiently handling concurrent access on the data of a repository. Therefore we propose the use of a distributed version control system to fulfill the tasks of the revision-control layer in the master. This decision makes it also possible for the clients to work offline.

Additionally to this, the revision-control layer demands a digital signature from the client on any revision that should be uploaded to the storage. With the digital sig-

nature on the head-revision, the client also signs the entire history of the repository. This is a property of the hash-trees that are used in the distributed version-control system—the hash of a node recursively includes the hashes of its children. Therefore every revision prior to the signed revision can be traced to the user who digitally signed the head-revision.

By using digital signatures on the revisions, it is also possible to verify the integrity and authenticity of the revision and the included documents.

Additionally it is not possible for someone with access to the master to forge documents or modify them. An attacker could not forge the documents, because they would have to be signed to go unnoticed by the clients. Therefore the attacker would have to sign the respective revisions and would be tracable as the author.

The master could dispute the reception of a revision, because using a digital signature only provides a proof of origin. In a fair non-repudiation protocol, there should also be a proof of receipt from the master [9].

4.2.4 Access Control and Authentication

One important aspect of data confidentiality is the restriction of access to the data. Because we use a central master in our system, it is possible to use well-established methods to control access to the data. We propose the use of role-based access control (RBAC [4]). With RBAC it is possible to restrict the access of a client so that it is only possible to access the documents and repositories for which it is authorized to do so. We propose RBAC because of its flexibility and wide acceptance. Before authorization can occur, the client must be authenticated. To authenticate a client, we propose the use of an asymmetric key exchange protocol. This not only authenticates the client to the master but can also be used to create a secure communication channel between the two. To this purpose, the master stores the public-keys of the clients.

4.3 Implementation

In this section we discuss the implementation of the design given in Chapter 2. The ReDS master was implemented in Python. Python makes it easy to rapidly develop functionality by providing existing modules for many common tasks. It is also possible to call external programs with little effort. This capability is used a lot in the implementation of the master. Python uses an interpreter for the application to run, which is slower than native code. This could become an issue in systems with many users. The implementation of the master consists of approximately 2000 source lines of code. As a framework for the master we used the command-line application framework Cement. Cement allows the use of extensions and plug-ins and handles the configuration files. We use this functionality by only defining the inter-

faces of the three layers in the core of the application. The actual implementation of the functionality is then done inside an extension which is easily exchangeable.

The implementation of the master component of ReDS is licensed under the Apache License, Version 2.0. It is available through the Python Package Index or Bitbucket. For our implementation we rely on the S3QL file system. S3QL is an cryptographic file system designed for the use of online storage. It supports Google Storage, Amazon S3 and Open Stack, among others. With an existing Python 2.7 installation it can be easily downloaded and installed by executing `pip install redsmaster`.

In the remainder of this section we describe how the connection to the storage, the revision control and the access control components are implemented. The section closes with a description of the clients that can connect to the master.

4.3.1 Storage Connection

The interface of the storage connection component is defined in the `IEncryptedFS` class. In our implementation we used the cryptographic filesystem S3QL as the underlying filesystem. This filesystem has several advantages over other cryptographic filesystems like EncFS . S3QL was specifically designed to access cloud storage and is therefore optimised to produce only minimal network traffic. To accomplish this, it stores the filesystem metadata in a database that is cached locally. Therefore operations such as moving files or renaming them only require updating the database. The database is then periodically written back in encrypted form to the cloud storage or upon unmounting [8].

Another advantage of S3QL is the transparent data de-duplication and data compression to reduce the used storage space. Additionally, the filesystem only creates blocks of data in the cloud storage. No information of the content of those blocks is visible to an adversary with access to the cloud storage. EncFS in contrast mirrors the exact directory structure and only encrypts the files themselves so that an attacker could gain valuable information about the stored data. Moreover, most cryptographic filesystems are not designed to work with cloud storage. One downside with S3QL is that it does not support multiple mounts of the same cloud storage, which makes it impossible to run multiple instances of the master component.

S3QL uses SHA256-HMAC as the message authentication code to ensure the integrity of the data. The computed hash of the data that is to be uploaded is then encrypted (using AES-256) together with the data and uploaded to the cloud storage provider. The used symmetric key is a random master key, which itself is encrypted using a passphrase. That makes it easier to change the passphrase, because only the master key has to be reencrypted. For transport layer security, S3QL uses SSL/TLS, if supported by the storage provider.

We implemented the interface to S3QL as a wrapper around the console interface of S3QL in the class `S3QLHandler`. Because S3QL is implemented as a FUSE

filesystem, it can be mounted without requiring special rights. But this also limits the supported operating systems for the master to the ones which support FUSE.

4.3.2 Revision Control

In the class IVersionControl the interface for the revision control component is defined. Our implementation in the `HgHandler` class wraps the Mercurial distributed version control system to manage the documents. In addition to using Mercurial, the implemented revision control component ensures that all repositories used by ReDS use the Commitsigs extension for Mercurial. The extension makes it possible to embed and verify digital signatures in revisions uploaded to the master. In the case of a non-existing or invalid digital signature, the master refuses to accept the changes. The supported systems to digitally sign revisions with the extension are GnuPG and X.509 certificates. Therefore it is possible to use existing public key infrastructures such as the Web of Trust with ReDS. The interface between the storage connection layer and the revision control layer in our implementation is given through filesystem operations, because S3QL mounts the cloud storage locally. Mercurial can then access the data through the mounted filesystem.

4.3.3 Access Control and Authentication

Access control is performed by our implementation of the core RBAC model [4] which is implemented in the `RedsAccessManager` class. To store the access control policies a SQLite database is used. Because the database consists of a single file, the database can be stored in the cloud storage using the cryptographic filesystem. In this way the access control policies are available independently of the specific master used. Five basic rights are defined to control access to the data:

repo.read The user has read access to a specific repository.
repo.write The user has write access to a specific repository. This also includes read access.
repo.create The user has the right to create new repositories.
admin.repos The user has full access rights to all repositories, including creation and deletion.
admin.usermanagement The user can change the access rights of users.

All admin operations are granted on the root directory, that means the tuple `(admin.repos)` describes the access rights of a repository administrator. For all other operations the path to the repository has to be given, for example `(repo.read, myrepo)`.

In the database used to store the access control policies, the public keys to authenticate the clients are also stored, as well as the hashes of defined pass-

words. The authentication is done by our implementation of an SSH server in the `SSHServerManager` class. This implementation is based on the SSH implementation Paramiko and also provides a secure communication channel to the clients.

4.3.4 Clients

Because the master is implemented using Mercurial to manage the documents, it is possible for the clients to use a Mercurial client to access the documents. The client can use the master like any Mercurial server. Our server enforces access control and does not accept unsigned revisions or revisions with invalid signatures, however. To generate digital signatures which the master accepts, it is necessary for the client to use the Commitsigs extension. But this is only needed if a revision is to be uploaded to the master. A Mercurial client can read the repositories which he has access to, even without this extension.

4.4 Summary and Outlook

We presented the ReDS system, a revision-secure document store. ReDS is intended for the use in conjunction with a cloud data store. The architecture of the system relies on a master server to store encryption keys, authenticate users and ensure consistency. We implemented ReDS in Python using several open source frameworks. Our implementation is open source itself and available for download.

Our document storage system is a backend. Because it behaves like a server of the Mercurial revision control system, any Mercurial client can be used to interact with it. To make the system more user friendly however, there is a need for a web frontend.

The implementation of our design performed well in preliminary tests. We plan to perform usability tests with a larger group of users in the near future.

References

1. U.S., British intelligence mining data from nine U.S. Internet companies in broad secret program. Washington Post (2013). `http://www.washingtonpost.com/investigations/us-intelligence-mining-data-from-nine-us-internet-companies-in-broad-secret-program/2013/06/06/3a0c0da8-cebf-11e2-8845-d970ccb04497_story.html`
2. Chasseur, C., Terrell, A.: Wingu - a synchronization layer for safe concurrent access to cloud storage (2010)
3. Feldman, A.J., Zeller, W.P., Freedman, M.J., Felten, E.W.: Sporc: Group collaboration using untrusted cloud resources. OSDI, Oct (2010)

4. Ferraiolo, D.F., Sandhu, R., Gavrila, S., Kuhn, D.R., Chandramouli, R.: Proposed nist standard for role-based access control. ACM Transactions on Information and System Security (TISSEC) **4**(3), 224–274 (2001)
5. Kumbhare, A., Simmhan, Y., Prasanna, V.: Cryptonite: A secure and performant data repository on public clouds. In: Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on, pp. 510–517 (2012). DOI 10.1109/CLOUD.2012.109
6. Mahajan, P., Setty, S., Lee, S., Clement, A., Alvisi, L., Dahlin, M., Walfish, M.: Depot: Cloud storage with minimal trust. ACM Trans. Comput. Syst. **29**(4), 12:1–12:38 (2011). DOI 10.1145/2063509.2063512. URL http://doi.acm.org/10.1145/2063509.2063512
7. Merkle, R.C.: A certified digital signature. In: G. Brassard (ed.) Advances in Cryptology - CRYPTO 89 Proceedings, *Lecture Notes in Computer Science*, vol. 435, pp. 218–238. Springer New York (1990). DOI 10.1007/0-387-34805-0_21. URL http://dx.doi.org/10.1007/0-387-34805-0_21
8. Rath, N.: http://www.rath.org/s3ql-docs/impl_details.html (access: 07/19/2013)
9. Zhou, J., Gollman, D.: A fair non-repudiation protocol. In: Security and Privacy, 1996. Proceedings., 1996 IEEE Symposium on, pp. 55–61 (May). DOI 10.1109/SECPRI.1996.502669

Chapter 5
Automatic Data Protection Certificates for Cloud-Services based on Secure Logging

Thomas Kunz, Annika Selzer, Ulrich Waldmann

Abstract Cloud services promise numerous advantages for companies over traditional in-house data processing in terms of flexibility and cost efficiency. These cloud services vary considerably with respect to the security mechanisms provided. As a result, many security-aware companies have strong concerns in using such services, e. g., with respect to confidentiality, data protection, availability, and control. Moreover, they complain the lack of transparency concerning the security measures and processes the cloud provider has installed. As a solution for the latter one, auditors may evaluate cloud providers and issue certificates attesting whether or not the cloud provider meets certain security requirements or legal regulations. However, due to the characteristics of cloud computing, on-site inspections in the data centers of a cloud provider do not seem to be realistic. In this paper we present a technical solution of an automatically generated data processing certificate for cloud services. Formal policies containing the security requirements the cloud service must comply with are the basis for this certificate. Our contribution uses secure log files as a trustworthy data base for the evaluation of a cloud service. We introduce a secure logging method which is resistant against internal manipulation attempts and that creates tamper-proof and confidential log data. Thus, the logging method is very well suited for the application in the data center of a potential untrustworthy cloud provider.[1]

Thomas Kunz · Annika Selzer · Ulrich Waldmann
Fraunhofer Institute for Secure Information Technology SIT, Rheinstr. 75, 64295 Darmstadt
e-mail: {thomas.kunz$|$annika.selzer$|$ulrich.waldmann}@sit.fraunhofer.de

[1] This contribution has been created within the project CloudCycle (http://www.cloudcycle.org). CloudCycle is funded by the German Federal Ministry for Economic Affairs and Energy (BMWi) according to a decision of the German Federal Parliament within the funding priority "Trusted Cloud".

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_5

5.1 Introduction

Nowadays, using cloud services is very attractive. Especially for companies cloud services promise numerous advantages with respect to flexibility and cost efficiency. They are hosted in data centers that are professionally managed, physically protected, and equipped with failsafe hardware.

However, in business context, security aspects like confidentiality, privacy, data protection, and protection against unauthorized access, become very important [5]. This is one of the reasons why many companies still have concerns regarding the use of cloud services [10]. On the one hand, companies are still responsible for the protection of their data, even if they shift the storage and the processing of the data in the cloud. But on the other hand, they do not trust the cloud providers in terms of confidentiality and data protection even though they expect the cloud providers to act according to their agreements (SLAs). The companies' doubts also stem from news about security incidents that arise nearly every day. In particular, companies often complain the lack of transparency concerning the security measures and processes the cloud provider has installed.

In this context, in Germany the contract data processing according to Sec. 11 of the Federal Data Protection Act (BDSG) is crucial. Contract data processing takes place if a contract data processor (short: "processor") collects, processes, or uses personal data on behalf of a controller, and if the controller holds the authority to issue instructions concerning these actions ([9] and Sec. 11 I 1, III BDSG). The controller remains responsible for the personal data. He has to carefully choose an appropriate processor. With the beginning of the data processing, he has to evaluate periodically that the processor has installed technical and organizational measures to protect the processed personal data (Sec. 11 II 4 BDSG). However, in the context of cloud computing it is doubtful how a cloud service user ("controller") shall comply with his obligation to evaluate the cloud provider (processor). In particular, a personal on-site control at the cloud provider does not seem to be realistic. Often, users want to use many different cloud services and they want to adjust these services frequently to their individual needs. Furthermore, they do not want to restrict themselves regarding the number of used cloud services and the geographic distance of the cloud providers. Such restrictions do not match the advantages and characteristics of cloud computing (rapid elasticity, resource pooling, and cost-efficiency). On the other hand, a personal on-site control would evolve to a literal "control tourism", which would have to be accomplished periodically [3].

A solution for the above mentioned problem is the evaluation of cloud providers and services through trusted third parties ("auditors"). By means of certificates, auditors may attest that cloud providers meet certain security requirements or legal regulations. In [3] a certificate based on standardized test criteria is proposed as an alternative to the on-site control of cloud providers.

5.1.1 Our Contribution

Due to the specific characteristics of cloud computing, audits within a cloud environment are exceedingly difficult. The scalability of cloud computing leads to dynamic and demand-oriented resource allocation, frequently altering applications and potentially changing processors. The efficient and shared usage of software and hardware resources through many cloud users additionally increases the complexity. Furthermore, different cloud users may have different security requirements to cloud services and cloud providers. For instance, one customer may require that his data have to be processed in Germany only, whereas other customers allow the whole European Economic Area (EEA) as a location for data processing. Therefore, annual audits of cloud providers are insufficient for evaluating the security measures installed by a cloud provider. In [7] a continuous evaluation based on monitoring, auditing, and checks is demanded. Such a continuous and customer-specific evaluation is feasible only by means of automatic checks.

In this paper, we present a comprehensive technical solution of an automatically generated data processing certificate for cloud services taking the before mentioned problems into account. The certificate is based on security requirements expressed in a formal policy as a part of the cloud service itself and it attests whether or not the cloud provider meets these requirements. The requirements may be derived from legal regulations such as the contract data processing according to Sec. 11 BDSG, but customers may also express individual requirements that can be much stronger than the legal ones. Since not all measures that protect personal data may be automatically evaluated, an auditor can optionally add results of checks performed manually on-site in the cloud provider's data center.

For an automated cloud service evaluation, the analysis of a complete and trustworthy data base is essential. Our contribution uses log files as a data base since log data is routinely collected in cloud data centers and it reflects the whole lifecycle of a cloud service. Furthermore, log data not only allow the analysis of a cloud service's current state but also to look back into the past. Unfortunately, commonly used logging methods like syslog [8] are not capable of providing a complete and reliable data base. For this reason, we introduce a secure logging method that creates tamper-proof and confidential log data. This logging method is very well suited for the application in the data center of a potential untrustworthy cloud provider since it is resistant against internal manipulation attempts.

The remainder of the paper is organized as follows. Section 5.2 gives an overview of the legal background. Related work is discussed in Section 5.3. Section 5.4 gives details about the secure logging method, while Section 5.5 illustrates the generation of data processing certificates. The paper ends with some conclusions in Section 5.6.

5.2 Legal Aspects

According to Sec. 11 II 4 BDSG contract data processing requires mandatory processor control by the controller. The following section provides an overview of this mandatory control.

5.2.1 Control According to Sec. 11 II 4 BDSG

Control Content. The control focuses on the question whether or not the processor takes the necessary technical and organizational measures to protect the personal data processed by him. In accordance with the annex to Sec. 9 BDSG he needs to ensure the following:

- Access control: Measures shall be taken to prevent unauthorized persons from gaining access to data processing systems for processing or using personal data.
- Access control: Measures shall be taken to prevent data processing systems from being used without authorization.
- Access control: Measures shall be taken to ensure that persons authorized to use a data processing system have access only to those data they are authorized to access, and that personal data cannot be read, copied, altered or removed without authorization during processing, use and after recording.
- Disclosure control: Measures shall be taken to ensure that personal data cannot be read, copied, altered or removed without authorization during electronic transfer or transport or while being recorded onto data storage media, and that it is possible to ascertain and check which bodies are to be transferred personal data using data transmission facilities.
- Input control: Measures shall be taken to ensure that it is possible after the fact to check and ascertain whether personal data have been entered into, altered or removed from data processing systems and if so, by whom.
- Job control: Measures shall be taken to ensure that personal data processed on behalf of others are processed strictly in compliance with the controller's instructions.
- Availability control: Measures shall be taken to ensure that personal data are protected against accidental destruction or loss. Control of separate processing: Measures shall be taken to ensure that data collected for different purposes can be processed separately.

Control Cycle and Documentation. The controller needs to ensure that the processor takes the above mentioned technical and organizational measures before he actually starts processing personal data for the controller. In addition, the controller needs to verify periodically that the processor still takes all the necessary technical and organizational measures while processing the personal data (Sec. 11 II 4 BDSG). How often such a control needs to take place mainly depends on the protection requirements of the processed personal data. In general, however,

an annual to biennial control cycle will be appropriate. The control accomplishment result needs to be documented (Sec. 11 II 5 BDSG).

Contractual Regulated Control. The controller should contractually seek the right of an on-site control in the processor's data center [6]. The processor should be required to participate and issue truthful and complete statements as well (Sec. 11 II 2 No. 7 BDSG). If the controller allows the processor to subcontract the processing, the controller should seek the right contractually to control the subcontract-processors as well [20].

5.2.2 Current Variants of Control

Currently there are different variants of how to control a processor: He can be controlled personally and on-site by the controller, or the controller can trust in an audit undertaken by a third party or the processor himself. However, each of the variants has disadvantages [20]:

- The controller controls the processor personally on-site: In times of cloud computing, where controller and processor are not necessarily geographical close to one another and where the controller might choose different processors for different services, the processor would need to personally travel to all relevant data centers to control the processor(s). Besides the fact that he might not have the necessary technical and legal knowledge to control the processor it is questionable if the on-site control is realistic when it comes to time, effort and money spend by the controller
- The processor audits himself: It is questionable if this audit meets the requirements of Sec. 11 II 4 BDSG [3] [25]. Some German data protection authorities stated that a self-audit in fact does not meet the requirements of Sec. 11 II 4 BDSG and that the controller at least needs to verify the results of the self-audit [3] [25]. On the other hand, other German data protection authorities confirmed the sufficiency of a self-audit, so there is a legal uncertainty whether or not a self-audit meets the requirement of Sec. 11 II 4 BDSG
- The processor is audited by a third party: The controller needs to evaluate the audits and auditors to be sure they meet the requirements of Sec. 11 II 4 BDSG due to the quality-variation of auditors and audit-criteria. It is questionable whether or not the controller has enough technical and legal knowledge to judge if an audit and auditor meet the requirements of Sec. 11 II 4 BDSG.

5.2.3 New Ways of Control

To solve the above mentioned problem the legal working group of the technology-program "Trusted Cloud" suggested a standardized audit [3]. The following sub-chapter introduces this suggestion to the reader.

Standardized Audit. According to the legal working group of "Trusted Cloud" the above mentioned problems can be solved by enabling the controller to contract an independent third party to audit the processor by using standardized criteria for both – the audit and the auditor [3]. The controller option to contract an independent third party for the audit needs to be regulated by written law within the EEA to achieve legal certainty. The same applies to the criteria for the audit and the auditor [3].

A standardized audit as suggested by the legal working group of "Trusted Cloud" could therefore result in legal certainty in regard to the question whether or not a third party may audit the processor on behalf of the controller, and if so which criteria must be met by the audit and the auditor.

The proposal of the data processing certificate made by the legal working group of "Trusted Cloud" [3] has the potential to provide a logically consistent solution. By including an automated part into the data processing certificate it is possible to control the implementation of the technical and organizational measures at the processing party not only every one to two years but also at any time during regular operation and also with regard to past processing.

5.3 Related Work

Today there exist many cloud-specific certifications and certification authorities [18]. *EuroCloud Star Audit*[2] (EuroCloud), *EuroPriSe*[3] (ULD[4] et al.), and *ISO 27001* [13], to name a few, are generally based on document reviews and on-site controls. The certification-related requirements do not take into account the specific requirements of different application sectors. The very number and variety of certifications and certification authorities reveal also their downside: Cloud users are usually not able to distinguish the qualities and underlying evaluation criteria of the available certifications. Furthermore, the certificates do not assure the continuous adherence to any certification criteria, but only an indication of adherence at certification time. Besides the lack of common criteria for certifications and corresponding authorities, the evaluation processes lack automated controls during the service operating phase. This would allow to identify incidents of data protection and security breaches immediately and to decide quickly on countermeasures. A constant service quality level could become verifiable only by means of continuous automated controls. Ini-

[2] `http://www.eurocloud.de`

[3] `https://www.european-privacy-seal.eu`

[4] Unabhängiges Landeszentrum für Datenschutz Schleswig-Holstein

tial approaches for continuous monitoring have been defined by NIST [16] and are considered in the context of the US American cloud certification FedRAMP[5].

The commonly used logging methods - like the de facto standard syslog [8] - are by no means capable of providing a complete and reliable data base as it is required for an automatically generated certificate. This is mainly due to the fact that logging methods were usually designed for internal servicing or troubleshooting rather than for auditing or forensic investigations lead by a third party. Conventional logging methods lack especially in multi-vendor interoperability, revision capability, and integrity [4]. Furthermore, an insufficient resilience to internal attacks is a common condition of use. Several researches have been done to provide log data that are resilient against subsequent tampering. The method of Schneier and Kelsey [19] enables "forward security" defined by Bellare and Yee [2] particularly for relatively untrusted components. Data, which are currently integrity-protected, shall remain so without becoming vulnerable to undetected deletion or manipulation in future attacks. An attacker, who for instance has brought the current MAC keys under his control, will not be able to derive any previously used keys, which he could have used to tamper previously secured data with. Further research attempted to improve the revision capability via public key methods [12] and resilience against internal attacks via personal cross-checks and manual signatures [21], digital black box with TPM [1] and log data storage at a trustee [24]. Though these ideas do not appear convincing for a highly heterogeneous cloud context, some of the suggestions are taken up in Section 5.4 to define a secure cloud-friendly logging method.

5.4 Secure Logging

A secure logging method is essential as secure data basis to automatically control the implementation of the technical and organizational measures. Automated checks to generate a certificate can be based on the information given in log data as these data routinely accrue in data processing centers and seem indispensable to achieve transparency and traceability. The whole life-cycle of cloud services, i. e. instantiation, configuration, usage, and termination of the corresponding service instances, have to be made comprehensive by available log data. Our secure logging method ensures the authenticity, forward integrity and confidentiality of log data and protects the log data in a presumably untrusted environment of the cloud provider. Furthermore our method takes the different access rights of heterogeneous cloud log data into account.

[5] http://cloud.cio.gov/fedramp

5.4.1 Secure Logging Method

In this section we present a secure logging method that extends the method of Schneier and Kelsey [19] to include publicly verifiable signatures and a trusted third party. On the side of the cloud provider a semi-trusted logging component ensures that each new log entry that is to be written into the current log file is connected to the previous entry by means of encryption, hash values and message authentication codes (MAC). In addition to the logging component, a trusted key storage that is independent from the cloud provider is required. For each new log file, the logging component creates two symmetric keys A_1 and B_1 that become the trust anchors and first base keys of the log file, are securely transfered to the key storage, and afterwards deleted in the logging component. Only the key storage device keeps these anchor keys that will never be disclosed.

Figure 5.1 shows how the logging component creates the current log entry (shown as middle bar of the three main bars) of a given log file. In the previous step (not shown) the component has derived the two base keys A_j and B_j (shown as small bars on the right). The operational keys K_j and L_j for the current log entry are in turn derived from these base keys concatenated with the individual *Read Permissions* (e. g. an identifier of the customer or VM) of the log data and then used for encrypting the original log data and computing the MAC to make up the current log entry. Then, the new base keys A_{j+1} and B_{j+1} (shown on the right of the current log entry) are derived by hashing the used base keys and temporarily stored for the next log entry (not shown).

All base keys are deleted after use. Only the new unused base keys are transiently stored for the log entry to come. The meta-data *LogID* and *Time* at the beginning of each log entry are used to quickly identify log entries, their origin and overall position. Only the key storage can re-compute the individual operational keys from the anchor keys and specified read permissions. The operational keys are disclosed to external authorized entities that have requested the keys to verify, decrypt and evaluate individual log data.

By means of the two key derivation chains (base keys and operational keys), the base keys never need to be disclosed and no entity can derive new operational keys from available keys of an existing log file. However, without any further preventive measures operational keys could be reused by external entities, if they wrongfully intend to alter and re-encrypt received log data. Therefore, to prevent any such later manipulation, a hash tree [15] that includes the hashes of all log entries is cryptographically timestamped by the trusted key storage, both inserted as last fields of the last log entry; cf. the main bottom bar in Figure 5.1. The hash tree enables to quickly verify a whole log file and focus on a subset of interesting log entries.

Thus, the following measures are essential for log security: registry of each log file and related timestamp with the key storage, and the reliable interaction of all components involved in logging, key storage and external evaluation. Exclusively the logging component is allowed to create secured log files and request any associated timestamp from the key storage. Every evaluating entity has to verify the rightful origin of the log file by verifying the hash tree and the timestamp first.

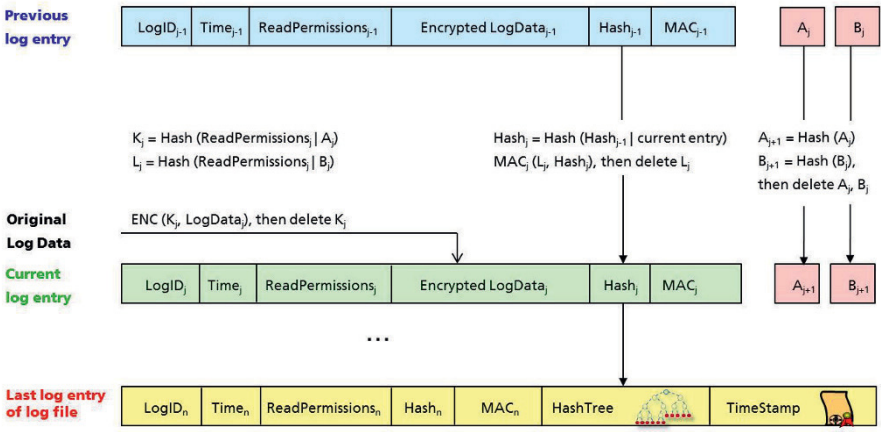


Fig. 5.1: Secure logging method

5.4.2 Secure Log System

It seems unrealistic that all technical components of the cloud provider will be adapted, in order to support a new logging method. Instead appropriate log adapters may be introduced, in order to collect the log entries from different sources and to forward them to a central logging component, where the above described logging method transfers the original into the specified secured format, cf. Figure 5.2.

In an optimal situation, the central logging component assorts the received log data according to the individual access conditions that have to be applied later (e. g. log data of customer X) and accordingly creates secure log files, each with log data belonging to the access-authorized entity. The log data of the virtual service instances serve as a good starting point for this since they can normally be attributed to an authorized owner or customer. The hypervisor may serve as a further important log data source. A log adapter of the logging component may read the logs of the hypervisor and underlying hardware components by means of the hypervisor API, assort them regarding the individual services instances and hand them over to the actual logging method.

An essential property of the log system is the resilience of its logging component against internal manipulation attempts as the cloud provider cannot per se be considered as trustworthy, because he may have an interest to revise the log data of past unfavorable events. Therefore, the key storage device providing security-related functions should be realized as a hardware security module (HSM) that the cloud provider has no direct access to. Furthermore, a separate log-supporting HSM would probably be better positioned in a security certification process than pure software components of similar functions.

Figure 5.2 shows a possible configuration with a separate HSM run by an independent third party as a trustee. In this configuration the trustee is not supposed to

manage the real log data, but only their trust anchors and meta-data (cryptographic keys and certificates, timestamps, authentication data). For this reason it seems realistic that cloud providers and customers might agree on this procedure without raising concerns over data protection issues. Furthermore, the cloud provider simply has to set up the logging component and provide space for storing the log files. Cloud providers, customers, auditors and trustees should regulate their relationships by contracts.

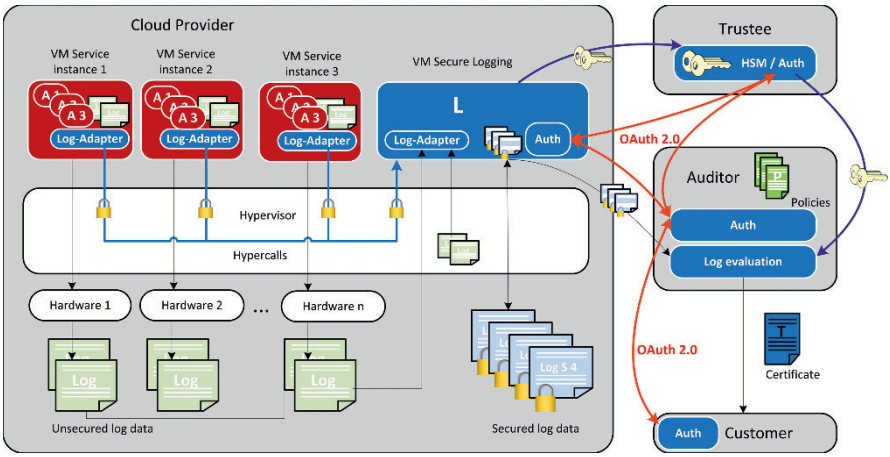


Fig. 5.2: Log system with trustee

Interestingly, large providers have started to offer HSM services to their customers,[6] who then keep the HSM cryptographic keys under their exclusive control. Under this condition a configuration seems possible with a trustee who in turn outsources his logging key services to the cloud.

5.4.3 Access to Secured Log Data

External entities, such as an auditor, can verify and evaluate log data as follows: The auditor authenticates towards the trustee by means of an established authentication protocol, preferably OAuth 2.0 [11], presenting an "Authentication Code" (in OAuth terminology), thus providing evidence that he has been authorized by the owner of the log data (generally the customer of the cloud service) to access the log data.

In OAuth terms the customer is called "Resource Owner", who entrusts the auditor ("Client") with the evaluation of his log data ("Resource"). The logging component is the "Resource Server", the HSM of the trustee serves as the "Authoriza-

[6] e. g. the CloudHSM of Amazon Web Services, see `http://aws.amazon.com/cloudhsm`

tion Server". The auditor requests from the trustee an authorization object ("Access Ticket") and then exchanges this ticket for the customer's log data from the logging component. The logging component of the cloud provider, the customer, auditor and trustee communicate with each other on the basis of a mutual authentication over secured TLS connections.

The following steps are again rooted in the mechanisms of the secure logging method. The auditor verifies the timestamp of the log file on the basis of a public available signature certificate of the trustee's HSM. Then, he sends the identifiers and read permissions of the log entries that he wants to decrypt and evaluate to the HSM, and this way requests the corresponding operational keys. The trustee's HSM re-computes the desired operational keys from the anchor keys and the submitted access rights, and then sends the operation keys to the auditor. Now, the auditor can use the keys, in order to verify the MACs, encrypt and evaluate the original log data. Finally, based on the evaluation of log data the auditor may confirm or refute the compliance with the agreed audit criteria (policies) in form of a data processing certificate.

5.4.4 Security Assumptions

The new log adapters of the VM service instances and logging component are assumed to be the feasible starting points for securing log data, since the heterogeneous hardware and system interfaces (e. g. log formats) could hardly all be adjusted to a new logging method. The logging method can not prevent that original log data might be altered or deleted right at their sources, in order to e. g. obscure SLA breaches. However, the log adapters should pull the original log data from their original sources at short time intervals and forward them to the secure logging VM. Furthermore, it depends on the quality of log data evaluation that unusual empty time intervals or inconsistent log contents will be noticed. At least, the parallel manipulation of few or even a single source of log data should become apparent.

Both, log adapters and the logging VM are assumed to be secured against internal attacks, be largely independent from the cloud provider and thus semi-trusted. The secure logging VM routinely deletes all symmetric encryption keys after use and only temporarily holds the specific keys for the upcoming log entry. Afterwards, any key must be requested from and recalculated on the side of the trustee by authorized entities (e. g. auditor). Thus, an attacker would have to submit to the same security conditions as the legitimate participants of the logging method (cloud provider, customers, auditors, trustee), in order to gain advantage in secured log data. How the legitimate participants get to trust each other and exchange communication certificates in preparation of the secure logging operations is out of scope. Therefore, organizational and technical measures at a higher level of the log system are required. The participants have to securely register with the trustee.

The characteristics of the trusted server (secure key store) are essential for the confidentiality and integrity of the secured log data. For this purpose, Schneier and

Kelsey [19] have proposed special servers in secure environments (e. g. tamper-proof tokens, ATMs) or a network of semi-trusted devices with distributed partial keys and redundant log data storage. The latter would require a close cooperation of cloud providers. For our solution, an environment, in which the cloud provider has only restricted access to, e. g. a pre-configured and controllable HSM of a third party, seems more practical. Such a HSM securely stores the anchor keys of the log files as well as the keys for signatures and timestamps and enforces secure communication channels to the participants. The use of the HSM shall be exclusively restricted to registered participants.

How secure would be half-completed log files temporarily stored outside of the semi-trusted logging environment? These files could neither be altered nor completed by internal attackers. However, the confidentiality and integrity of log files fully apply only to completed log files that include a hash tree and timestamp. Otherwise, it would be conceivable that a verifying entity that have received the encryption and MAC keys will be able to subsequently alter a file and "secure" it with the old keys. In fact, this altered file could not have valid a timestamp, since no entity (other than the logging component) can successfully request such a timestamp. Thus, the verification of the hash tree and timestamp is essential for proving the authenticity of log files.

5.5 Data Processing Certificate

Generally, certificates express the result of the inspection of certain processes or products and are issued by a trusted third party ("auditor"). In most cases, the auditor verifies the current state of the process or product.

In the cloud context, an auditor – by means of a "data processing certificate" – attests that a cloud provider complies with certain security requirements (e. g. concerning his data processing center) and that he implements certain technical or organizational measures. In this case, not only the current state is of interest, but also information about prior implementations of these measures in order to be able to evaluate if the measures have been installed in time and without interruptions. For this, the auditor checks which security measures the cloud provider has installed, and whether these measures suffice to meet the given security requirements.

Because of the special characteristics of cloud services (rapid elasticity, resource pooling etc.), an efficient inspection of a cloud service and the issuing of a certificate are only feasible in an automated way. Due to the automation, the checks may be performed at any time. This allows a continuous monitoring without an auditor operating on-site at the cloud provider. As an example, the location of processing the customer data of a specific cloud service could be monitored continuously. A precondition for such automated checks is the availability of secure and trustworthy data as a basis that can be analyzed and interpreted automatically. These data must be complete and authentic, must have integrity and must reflect the whole life cycle of a cloud service. Secure log files (cf. Section 5.4) seem to be very well suited for

this purpose since they allow looking far back in the past. Thus, an automatically generated data processing certificate bases on two pillars: Security requirements and a secure and trustworthy data basis (cf. Figure 5.3).

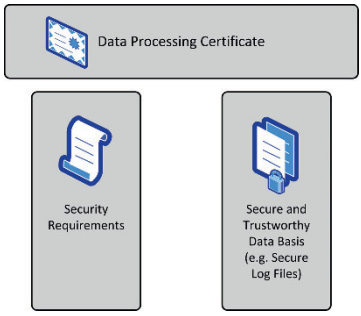


Fig. 5.3: Two pillars of the data processing certificate

One of these two pillars, secure log data as a data basis, has already been introduced in Section 5.4. Therefore the following sections deal with the derivation of security requirements and the generation of a certificate.

However, you may assume that in general not all checks can be automatically performed. In particular, this is true for monitoring organizational measures and evaluating the security measures of buildings (lightning protection etc.). The auditor is still responsible for these checks and he must operate on-site at the cloud provider. Consequently, the data processing certificate will consist of two parts: an automatically generated part and the results of the auditor's on-site checks. These on-site checks are out of scope of this paper.

5.5.1 Requirements to Cloud Services

Security requirements may have different origins. First of all, they can be derived from legal regulations (e. g. contract data processing). In addition, the service creator may define requirements on the environment in which his cloud service should run. Plus, the cloud consumer may express individual requirements on the cloud provider or the cloud services he wants to use. Examples for such requirements are the location of processing personal data (Germany, EEA, Switzerland, etc.), and the level of separation of data (e. g. dedicated server or shared server). When auditing a cloud service, these requirements serve as test criteria.

In order to guarantee an automated control of the cloud service, the requirements must be described in a policy using a formal language. Besides, additional use cases for such policies exist: Customers may use such policies to find adequate cloud providers or cloud services which comply with the customer's requirements ("automated match making"). Also, such policies may be used by cloud management

environments to enable an automated enforcement of the security requirements. A prerequisite for these use cases are standardized policies to enable a widespread support through many cloud management environments. In this context, the Topology and Orchestration Specification for Cloud Applications (TOSCA) [14], a recently initiated standardization effort from OASIS, provides an interesting approach. Using TOSCA, cloud creators are able to describe the topology, the deployment, and the management of portable cloud services. TOSCA allows an automatic deployment and management of cloud services at any cloud environment that supports TOSCA. In addition, TOSCA allows the definition of policies that describe certain requirements. Current research deals with the definition of non-functional requirements (e. g. security requirements) in TOSCA as policies and how these policies can automatically be processed by a cloud environment [22] [23].

Figure 5.4 shows at which stages in the cloud service life-cycle the policies become relevant. [17] describes a cloud service life-cycle consisting of the stages "modeling a cloud service", "distribution", "installation", "instantiation", "usage", and "termination". During the modeling stage, the cloud service creator describes his cloud service. He may specify policies with some security related requirements concerning the cloud provider or the cloud environment. After the cloud service has been installed and is offered by a cloud provider, the customer can select a cloud service that complies with his own requirements. When he has closed a contract with the cloud provider concerning the usage of this service, the customer has the option to add further requirements or to choose requirements from a set of alternatives. As an example, the customer could choose between different data processing locations or he could select a certain security level regarding the encryption of his data. When the requirements are set, the cloud provider creates a new service instance for the customer with a customer-specific policy. During the usage stage, the customer may authorize an auditor to monitor the cloud provider and the customer's service instance. The customer-specific policy is the basis for all checks to be done, i. e. it will be verified if the requirements contained in the policy are met.

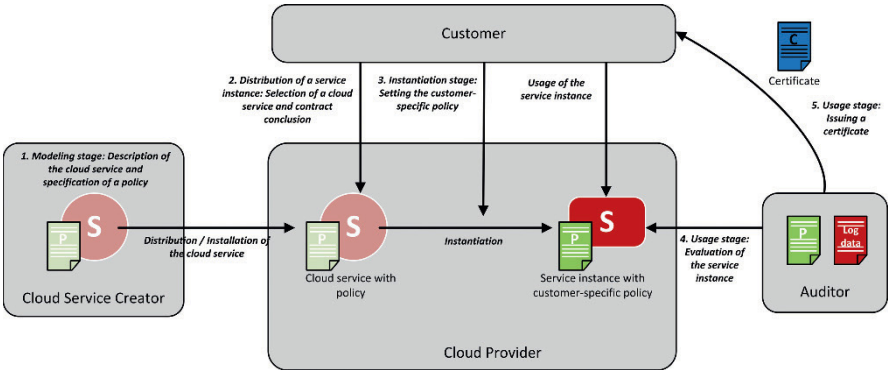


Fig. 5.4: Customer-specific test criteria

5.5.2 Certificate Generation

Generating a certificate could be as follows: The auditor initializes the processing of the automated checks. For this purpose, the auditor needs a secure access for requesting the log data. When processing the checks, the evaluation period, the service instance to be checked and with it the security requirements (policy) have to be considered. The analysis of the log data should detect whether the cloud provider has met all security requirements during the whole evaluation period without any interruption. After the completion of the checks, the test results will be merged in the automated certificate and the device used for generating the certificate will digitally sign this part of the certificate. The auditor verifies the automated certificate with regard to completeness, plausibility, and correctness. Besides the test results, the certificate should contain additional information like the name of the cloud provider, an identifier of the evaluated cloud service, a reference to the underlying policy, the evaluation period, and the time of issuing the certificate.

Optionally, the auditor may carry out additional checks on-site at the cloud provider. After this, he merges the automatically generated part of the certificate and his own test results, adds the validity period of the certificate, and confirms the overall results with his digital signature.

5.6 Summary and Conclusions

In the era of cloud computing, any personal on-site inspection by the cloud customer (controlling party) at the cloud service provider (processing party) seems anachronistic and inconsistent with the structural conditions as it would demand a sort of inspection tourism that no controller could meet. An automated data processing certificate has the potential to provide a logically consistent solution. Automated certificates may serve as evidence that the processing party faithfully and over time adhered to the technical and organizational measures according to the agreed policies. In case a certificate report is negative the controlling party could react quickly, in order to fulfill its ultimate responsibility concerning processed personal data.

For this long-term goal, the solution of data processing certificates should be stated in European law, such as the future European General Data Protection Regulation, as an acceptable substitution of on-site inspections to control contract data processing. Trusted third parties should be admitted to act as auditors who carry out semi-automated controls of the processing party on behalf of the controlling party.

The described secure logging method is an important element of the evidence chain for auditing. It creates tamper-proof and confidential log data, in order to prevent internal manipulation attempts of a possibly untrusted cloud provider. The solution provides an information base for security controls by third parties and has the potential to meet the requirements of different sectors with sensitive personal data, such as health care, law and finance.

As regards the resulting performance and storage overhead of the proposed solution the cloud provider should provide more resources, since the log data has to be centrally secured and stored. A detailed performance analysis of the solution in a realistic cloud environment has to be done. Furthermore, the evaluation of log data is a big challenge in regard to coping with a variety of different log formats as well as to logically linking log data that result from different sources but belong to the same transaction. For instance, the log entries of authentication servers have to be correlated to log entries of administrative configuration activities, in order to evaluate the rightfulness of an authentication process and subsequent data access. Furthermore, future work should also define appropriate metrics to quantize the degree of compliance with evaluation criteria according to the effective policy of a cloud service, in order to make such evaluation results transparent and comparable across different providers.

The evaluation of secured log data provides information about the actual processes that have taken place at the cloud provider. The automated and continuous controls enable auditors to control the current implementation state of the technical and organizational measures at the processing party and how these measures were effectively implemented in the past as well. Furthermore, the automated verifiable policies allow these controls being targeted at customer-specific service instances. The customer himself could carry out controls on a random or continuous basis.

A secure log base and appropriate evaluation could serve as information base for several purposes in addition to the generation of data processing certificates, such as forensic analysis and monitoring. Therefore, one mid-term goal should be the development of protection profiles according to the Common Criteria[7]. A cloud provider who has installed an accordingly certified logging component could advertise with automatically auditable and monitorable cloud services. Thus, it is conceivable that on-line marketplaces offer such enhanced cloud services.

Further research should also address the definition of application- and sector-specific policies or policies of a certain jurisdiction that could be standardized and made available e. g. by a supervisor authority. A set of obligatory policies could serve as guidance for service providers to implement sector-specific legal requirements as well as for auditors to define corresponding certification practice statements.

References

1. Accorsi, R.: Log Data as Digital Evidence: What Secure Logging Protocols Have to Offer? In: 33rd Annual IEEE International Computer Software and Applications Conference (COMPSAC '09), pp. 398–403, (2009)
2. Bellare, M., Yee, B. S.: Forward Integrity for Secure Audit Logs. University of California at San Diego (1997)
3. Borges, G. et al.: Datenschutzrechtliche Lösungen für Cloud Computing. Kompetenzzentrum Trusted Cloud, http://www.trusted-cloud.de, (2012)

[7] Common Criteria: http://www.commoncriteriaportal.org

4. BSI: Studie über die Nutzung von Log- und Monitoringdaten im Rahmen der IT-Frühwarnung und für einen sicheren IT-Betrieb (2007)
5. Contu, R., Pingree, L., Ahlm, E.: Predicts 2013: Security Solutions. Technical Report, Gartner (2012)
6. Eurocloud Deutschland_eco e.V.: Leitfaden Cloud Computing, (2010)
7. European Network and Information Security Agency (ENISA): Critical Cloud Computing - A CIIP perspective on cloud computing services, Version 1.0, (2012)
8. Gerhards, R.: The Syslog Protocol, Request for Comments: 5424, Internet Engineering Task Force IETF (2009)
9. Gola, P., Schomerus, R.: BDSG Kommentar (2012)
10. Gonzales, N., Miers, C., Redigolo, F., Simplicio, M., Carvalho, T., Näslund, M., Pourzandi, M.: A quantitative analysis of current security concerns and solutions for cloud computing. In: Journal of Cloud Computing: Advances, Systems and Applications, 1(1) (2012)
11. Hardt, D. (Ed.): The OAuth 2.0 Authorization Framework, Request for Comments: 6749, Internet Engineering Task Force IETF (2012)
12. Holt, J. E.: Logcrypt: forward security and public verification for secure audit logs. In: Proceedings of the 2006 Australasian workshops on Grid computing and e-research, Volume 54, ACSW Frontiers '06, pp. 203–211, Australian Computer Society (2006)
13. ISO/IEC 27001: Information technology - Security techniques - Information security management systems - Requirements (2013)
14. Lipton, P., Moser, S., Palma, D., Spatzier, T.: Topology and Orchestration Specification for Cloud Applications (TOSCA),Version 1.0, `http://www.oasis-open.org/committees/tc\_home.php?wg\_abbrev=tosca` OASIS specification (2013)
15. Merkle, R.: Protocols for Public Key Cryptosystems. In: Proceedings of the 1980 IEEE Symposium on Security and Privacy, Oakland, USA (1980)
16. National Institute of Standards and Technology (NIST): Guide for Applying the Risk Management Framework to Federal Information Systems - A Security Life Cycle Approach, NIST Special Publication 800-37, (2010)
17. Niehues, P., Kunz, T., Posiadlo, L.: Das CloudCycle-Ökosystem, Technical report, `http://www.cloudcycle.org`, CloudCycle (2013)
18. Schneider, S., Lansing, J., Sunyaev, A.: Empfehlungen zur Gestaltung von Cloud-Service-Zertifizierungen, In: Industrie Management - Zeitschrift für industrielle Geschäftsprozesse, pp. 13–17 (2013)
19. Schneier, B., Kelsey, J.: Secure audit logs to support computer forensics, In: ACM Transactions on Information and System Security (TISSEC) 1999, Volume 2, pp. 159–176, ACM New York, USA (1999)
20. Selzer, A.: Die Kontrollpflicht nach § 11 Abs. 2 Satz 4 BDSG im Zeitalter des Cloud Computing, In: DuD 04/2013
21. Stathopoulos, V., Kotzanikolaou, P., Magkos, E.: A Framework for Secure and Verifiable Logging in Public Communication Networks, In: Critical Information Infrastructures Security, Lecture Notes in Computer Science, Springer Berlin Heidelberg, pp. 273–284 (2006)
22. Waizenegger, T., Wieland, M., Binz, T., Breitenbücher, U.: Towards a Policy-Framework for Provisioning and Management of Cloud Services, In: SECURWARE 2013, The Seventh International Conference on Emerging Security Information, Systems and Technologies (2013)
23. Waizenegger, T., Wieland, M., Binz, T., Breitenbücher, U., Haupt, F., Kopp, O., Leymann, F., Mitschang, B., Nowak, A., Wagner, S.: Policy4TOSCA: A Policy-Aware Cloud Service Provisioning Approach to Enable Secure Cloud Computing, In: DOA-Trusted Cloud'13 International Conference on Secure Virtual Infrastructures (2013)
24. Waters, B. R., Balfanz, D., Durfee, G., Smetters, D. K.: Building an Encrypted and Searchable Audit Log, Princeton University and Palo Alto Research Center (2004)
25. Weichert, T.: Cloud Computing und Datenschutz, In: DuD 10/2010

Chapter 6
A Trust Point-based Security Architecture for Sensor Data in the Cloud

Martin Henze, René Hummen, Roman Matzutt, and Klaus Wehrle

Abstract The SensorCloud project aims at enabling the use of elastic, on-demand resources of today's Cloud offers for the storage and processing of sensed information about the physical world. Recent privacy concerns regarding the Cloud computing paradigm, however, constitute an adoption barrier that must be overcome to leverage the full potential of the envisioned scenario. To this end, a key goal of the SensorCloud project is to develop a security architecture that offers full access control to the data owner when outsourcing her sensed information to the Cloud. The central idea of this security architecture is the introduction of the trust point, a security-enhanced gateway at the border of the information sensing network. Based on a security analysis of the SensorCloud scenario, this chapter presents the design and implementation of the main components of our proposed security architecture. Our evaluation results confirm the feasibility of our proposed architecture with respect to the elastic, on-demand resources of today's commodity Cloud offers.

6.1 Introduction

One of the key goals of the SensorCloud project is the *secure* interconnection of diverse types of sensor devices with today's Cloud computing environments in order to leverage their elastic, on-demand computation and storage resources. Sensor devices may thereby be located in a wide variety of network deployments ranging from personal homes to offices, cars, industrial facilities, or public areas [13, 38]. However, while the integration of sensor networks with Cloud computing environments is a striking proposition, the desired interconnection is far from trivial as sensed data often contains sensitive information that third parties may strive to exploit [22, 20, 52]. For example, sensor readings from an industrial deployment can

Martin Henze, René Hummen, Roman Matzutt, Klaus Wehrle
Communication and Distributed Systems, RWTH Aachen University, Germany
e-mail: {henze,hummen,matzutt,wehrle}@comsys.rwth-aachen.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_6

provide competitors with valuable information about the employed equipment and its degree of capacity utilization, thus providing them with a competitive advantage. Moreover, sensitive information may not only be contained in the sensed data but could also be derived from the corresponding meta information, e.g., time or location. As a result, owners of sensor data typically refrain from unconditionally revealing their sensor data to others. Furthermore, recent security incidents in well-known Cloud offers nurture adoption barriers that stem from the inherent multi-tenancy characteristics of Cloud environments [43]. Educated data owners with sensitive sensor data therefore often do not utilize Cloud offers in order to prevent loss of control over their data [8, 21, 40].

To overcome these adoption barriers, additional security mechanisms that enable the data owner to prevent unauthorized access to her sensitive information in the Cloud have to be put in place. Current state-of-the-art approaches in research thereby typically aim at providing hard, cryptographic security guarantees. To this end, they employ novel cryptographic primitives such as fully homomorphic encryption [15] or base their approach on specific hardware, e.g., trusted platform modules [44, 48]. However, these approaches either involve an impractical cryptographic overhead if applied to comparably small-sized sensor readings [11], or do not offer a satisfying fine-grained level of control for the data owner over her data in the Cloud.

Hence, a core research challenge when outsourcing processing and storage of potentially sensitive sensor data to the Cloud is the design and implementation of a practically viable security architecture. The goal of this security architecture must be to enable the control of the data owner over her data once it has been outsourced to the Cloud. To attain this goal, the security architecture has to offer measures to: i) protect potentially sensitive sensor data already in the sensor network where it is still under control of the data owner, ii) guarantee confidentiality and integrity of the sensor data after it has left the sensor network, and iii) offer data owner-centric access control of sensor data for trustworthy services.

Our contributions presented in this chapter are as follows. First, we describe the abstract scenario of the SensorCloud project and discuss its security considerations. We then derive design goals for the SensorCloud security architecture. Based on these goals, we present our trust point-based security architecture and describe its core components: i) a pairing procedure that binds the trust point to the data owner as her digital representative in our architecture, ii) transport security between the sensor network and the Cloud entry point, iii) object security between the sensor network and services, and iv) fine-grained, user-centric data access control. We then discuss our evaluation results and show that the performance and the storage overheads of our proposed security architecture indeed are practically viable in today's commodity Cloud offers. Finally, we outline how our approach differentiates from related work and conclude this chapter with a discussion of our next steps and further research directions.

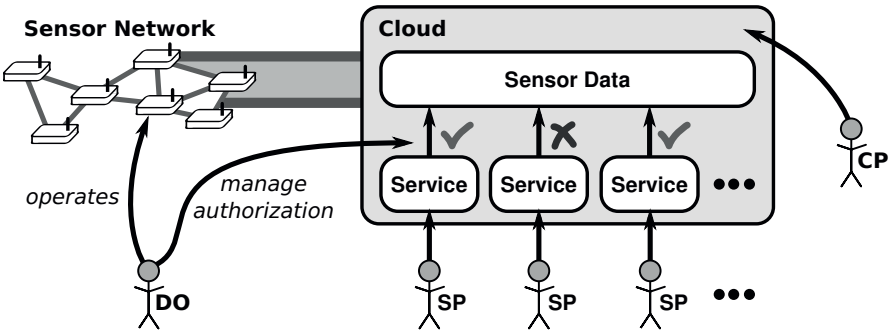


Fig. 6.1: High-level overview of the SensorCloud scenario. The data owner (*DO*) uploads her sensor data to the Cloud, which is operated by the Cloud provider (*CP*). Service providers (*SP*) deploy their services to the Cloud, which allows the data owner to authorize specific services to access her data.

6.2 Scenario and Security Considerations

We start our discussion with a description of the entities involved in the SensorCloud scenario and their interactions. Furthermore, we discuss security-relevant implications of the outlined scenario.

Figure 6.1 depicts the assumed scenario of the SensorCloud project. The *data owner (DO)* maintains a network consisting of sensor nodes that frequently produce sensor data. She is in possession of any sensor data that is produced within her sensor network domain. The data owner outsources the (long-term) storage and processing of her sensor data to a *Cloud* computing environment via the Internet.

The Cloud computing environment is operated by the *Cloud provider (CP)*. We assume this environment to be *public*. Thus, the Cloud provider offers its infrastructure to anyone who is willing to pay for it. As its operator, the Cloud provider is able to monitor any component of its infrastructure at any given point in time for maintenance purposes. In addition to the mere infrastructure, the cloud provider offers a storage service for sensor data as well as execution environments for Cloud-hosted third-party services that process outsourced sensor data. The individual *Cloud services* are offered by *service providers (SP)*. We assume that the Cloud provider has appropriate measures in place to separate the execution of the SensorCloud platform and the Cloud services running within the context of the SensorCloud platform from other third-party Cloud services. To this end, the Cloud provider may, for example, employ special virtual machine placement policies that do not map other Cloud services to the same physical machines that are used by the SensorCloud platform [22]. Likewise, we assume that the Cloud provider enforces strict separation of execution environments for individual services running on the same physical machine, e.g., as described in [22]. This is to prevent Cloud services from interfering with each other.

6.2.1 Security Considerations

In the scope of the SensorCloud project, we assume that the sensor data is adequately protected against attacks in the local sensor network. For example, existing ZigBee security mechanisms [53] could be employed for ZigBee-based sensor network. For IP-based sensor networks, as currently advocated by standardization bodies [34, 54], traditional IP security solutions could be deployed within the sensor network domain. These solutions then require modifications of the corresponding protocols concerning the typical device [26, 25] and network constraints [23, 24].

Once sensor data leaves the sensor network domain and is transported via the Internet, we also have to consider external attackers, who may try to manipulate or eavesdrop on the communication with the Cloud computing environment. Hence, the *confidentiality and the integrity* of sensor data and of management-related communication *between the sensor network and the Cloud* must be protected. Furthermore, considering the multitenancy characteristics of a Cloud computing environment, the data owner requires her sensor data not to be revealed to an entity that she did not *explicitly authorize* to access her data. Specifically, a data owner's sensor data must not be accessible by other data owners sharing the storage facilities of the Cloud. Likewise, neither the Cloud provider nor unauthorized services must be able to access stored sensor data. Any *modification* of stored sensor data must be perceivable by either the data owner or a service processing this data. Still, we observe that the Cloud provider has full control over its hardware and, therefore, can inspect the memory of its physical machines for sensitive information as long as data processing is performed by services on plaintext sensor data. However, contractual obligations and liabilities can render such inspections unattractive for the Cloud provider as shown by the US government's use of Amazon's AWS GovCloud offer [2]. Likewise, by strictly restricting access to its monitoring capabilities to a small number of trusted administrators who must be located within the data center providing the Cloud infrastructure, the Cloud provider can mitigate the risk of exposure due to attacks against its monitoring facilities. This is especially true when taking the Sealed Cloud project into account.

6.3 Trust Point-based Security Architecture

To address the above stated security requirements and to mitigate the anticipated loss of control over data once it is stored and processed in the Cloud, we previously presented a trust point-based security architecture for the user-controlled processing and storage of sensor data in the Cloud [22, 20]. In this section, we outline and discuss the design and implementation of this security architecture. First, we present and discuss high-level design goals for a security architecture that aims at protecting sensor data in the Cloud. We then have closer look at the sensor data flow in the SensorCloud scenario. An analysis of the sensor data flow leads us to the introduction of the trust point as a logical entity for protecting sensor data already in the

sensor network and thus still within the control sphere of the data owner. Based on the design goals and the insights from our scenario analysis, we then present and discuss the four core components of our trust point-based security architecture for sensor data in the Cloud: i) establishing a binding between the data owner and the trust point in the Cloud, ii) securing the communication channel between the sensor network and the Cloud entry point, iii) securing the sensor data between the sensor network and authorized services, and iv) granting access to sensor data for authorized services and managing the keys required to implement our proposed access control mechanisms for sensor data in the Cloud.

6.3.1 Design Goals

We now briefly discuss the essential design goals of a security architecture that protects sensor data in the Cloud against unauthorized access. These design goals either directly stem from the security requirements presented in Section 6.2.1 or originate from the vision of SensorCloud to interconnect off-the-shelf sensor devices with commodity public Cloud offers.

Secure transmission and storage

Data that a data owner transfers to and stores in the Cloud must not be unintentionally exposed to any entity but the data owner and her authorized Cloud services. In particular, the Cloud provider and any other third-party Cloud service must not be able to access sensitive data during transit as well as data at rest. The design of the security architecture must offer means for the data owner to verify that her sensor data was not altered. Likewise, the Cloud provider must be able to authenticate data owners for proper billing and to protect their data from unauthorized modification.

Selective access authorization

Cloud services that are trusted by the data owner should eventually be able to process the data owner's sensor data. To this end, the security architecture must enable the data owner to select Cloud services and to grant these service access to her sensor data. This access control should be as fine-grained as possible. We call such trusted Cloud services with access to the data owner's sensor data *authorized services*.

Feasibility

All security measures put in place to provide for data protection and selective access authorization have to be feasible given the available resources of today's commodity Cloud offers. Hence, these measure have to consider the special characteristics of protection large amounts of individually small data items. Consequently, on the one hand, the architecture has to scale with the amount of data items. On the other

hand, the resulting overhead, e.g., for the storage of sensor data has to be reasonable compared to the rather small size of the individual data items. With respect to the computation overhead, the architecture should sufficiently efficient to process sensor data in a timely manner, e.g., to allow the data owner to monitor a service-processed view on her currently generated sensor data.

Applicability

Our security architecture should be applicable with respect to off-the-shelf sensor devices and commodity public Cloud offers. Thus, the architecture should not prevent a Cloud provider from executing the SensorCloud platform (including its security architecture) in parallel to other Cloud-based services. Concerning the data owner, the security architecture should minimize the necessary changes to the sensor network. In particular, the security architecture must not require the data owner to replace existing sensor devices or to alter their deployed firmware.

6.3.2 Security Architecture

To mitigate the security issues identified in Section 6.2 and at the same time meet the design goals formulated in Section 6.3.1, we now present our trust point-based security architecture for sensor data in the Cloud. To this end, we first have a closer look at the flow of sensor data in the SensorCloud scenario. Then, we introduce the trust point, a logical entity for protecting potentially sensitive sensor data already within the sensor network. Thus, the protection still happens under the control of the data owner. We then describe how a data owner can ensure the Cloud that she is the operator of a specific trust point. Afterwards, we discuss measures for securing the communication channel and protecting sensor data until it reaches an authorized services. Finally, we show how a data owner can authorize services to access data.

6.3.2.1 Sensor Data Flow

We now describe a typical sensor data flow in the SensorCloud scenario and give a high-level description of the involved processing steps (see Figure 6.2). A data flow commonly starts at the sensor network and consists of periodically generated *data items*. In other words, a data item is an atomic fragment of a sensor data stream and represents the reading of one sensor node at a specific point in time. The payload of a data item consists of one or more *data fields* which contain the measured values from individual sensors deployed on a specific sensor node. Additionally, meta information (e.g., time or location) is stored within each data item.

Data items are forwarded inside the sensor network towards a *gateway* device that is responsible for uploading sensor data to the Cloud for further storage and processing. Hence, *all* data items generated inside the sensor network traverse the

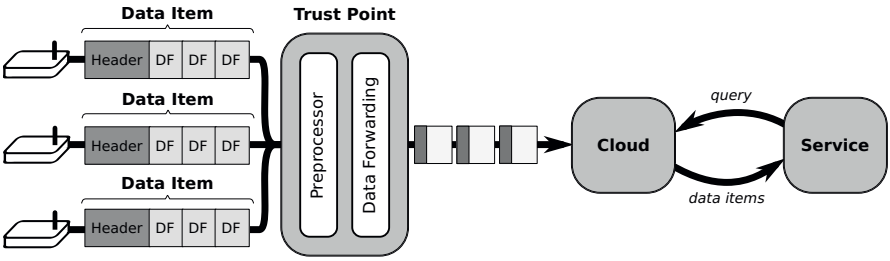


Fig. 6.2: Sensor data flow in the SensorCloud scenario. Data items contain meta information (*header*) and data fields (*DF*) with the sensed values. The trust point performs preprocessing steps on each data item and uploads it to the cloud. Authorized services can then access the data items.

gateway as the last entity situated inside the sensor network. As a dedicated network element, the gateway may be limited with respect to storage and processing resources. Hence, it may neither be able to store large amounts of sensor data nor to perform excessive computational tasks.

After a data item is received at the entry point to the Cloud computing environment, it is stored persistently in the storage back-end of the Cloud. In addition to this storage service, the Cloud also offers a platform for services. This Cloud platform hosts third-party services that process data items. To start this processing for (a fraction of) her sensor data, the data owner authorizes individual services to perform the processing of her relevant data items. The authorized services then request the corresponding data items from the Cloud and processes these data items. Finally, the data owner can review the results of the processing via an external service interface such as a website or have the results stored in the Cloud.

6.3.2.2 The Trust Point

We identify three main trust domains in the SensorCloud scenario (see Figure 6.3): the *producer domain*, the *storage domain*, and the *consumer domain*. The producer domain consists of the sensor nodes and the gateway device in the sensor network. All devices in this domain are under control of the data owner. However, once sensor data leaves the producer domain, the data owner loses control over her data. The Cloud and its provider therefore constitute a second trust domain. As the Cloud is primarily responsible for persistently storing outsourced sensor data, we call this domain the storage domain. Finally, the consumer domain consists of the service providers and their provided services. These services require access to plaintext sensor data in order perform their processing tasks.

In this trust model, the gateway is located at the border of the data owner's control domain to the storage domain. A central idea of our security architecture is to enhance this gateway with additional mechanisms that afford the secure outsourcing

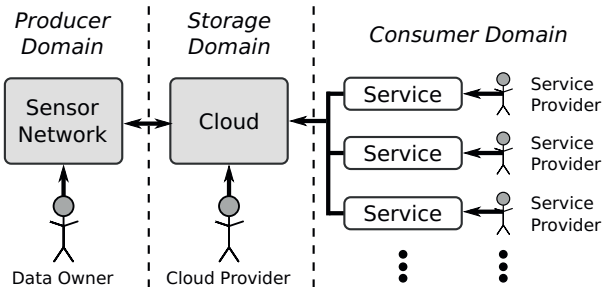


Fig. 6.3: Trust domains in the SensorCloud scenario. The producer domain is fully trusted. For the storage domain, an honest-but-curious adversary model is assumed. Finally, the consumer domain may contain entities that are not trusted.

of sensor data to the Cloud. More precisely, our enhanced gateway preprocesses the generated sensor data on the behalf of the data owner and applies confidentiality as well as integrity protection measures before uploading the protected sensor data to the Cloud. As this enhanced gateway device is owned and thus trusted by the data owner, we call it the *trust point*.

As the trust point constitutes a representative of the data owner towards the Cloud, a binding between the independent entities data owner and trust point are required at the Cloud provider. We propose a *pairing scheme* that assures the Cloud provider of the legitimate binding between the data owner and her trust point such that no spoofing of the sensor network identity is possible (Section 6.3.3).

Intuitively, the trust point manages the upload of sensor data to the Cloud as it is the gateway device of the sensor network. As the communication between the trust point and the Cloud traverses the Internet, the *communication channel between the trust point and the Cloud has to be secured*. This is not only limited to providing confidentiality of the communicated information, but also includes the mutual authentication of the two communication peers. Thus, the trust point can be sure that it is in fact communicating with the Cloud. Similarly, the Cloud can verify the identity of the trust point and, due to the pairing, a linkage to the data owner (Section 6.3.4).

However, mere transport protection does not suffice to protect the transmission and storage of sensor data in the Cloud as the transport protection is terminated at the Cloud entry point. Hence, sensor data would be unprotected within the Cloud environment. Therefore, additional *object-level protection* of sensor data is required. To this end, the trust point also encrypts individual data fields and signs each data item before sending it towards the Cloud (Section 6.3.5).

The trust point is also responsible for the resulting *management of encryption keys*. We show how the trust point can manage the encryption keys despite its restricted storage capacities. Since the trust point takes the role of the data owner's representative in our architecture, the trust point is also in charge of maintaining *service authorizations* on behalf of the data owner (Section 6.3.6).

Thus, we identified four major processes that are needed for securely outsourcing sensor data to the Cloud: i) pairing between trust point and data owner, ii) securing the communication channel between sensor network and Cloud, iii) protecting the sensor data, and iv) service access granting and key management. In the remainder of this section, we discuss these processes of our SensorCloud architecture in more detail.

6.3.3 Pairing between Trust Point and Data Owner

The Cloud provider must be able to associate incoming sensor data with the data owner whose sensor network generated the data. Otherwise, the Cloud provider could neither bill its customers properly, nor could it provide means to protect sensor data against unauthorized modification. However, this task is per se not trivial for the Cloud provider as the data owner does not upload her sensor data in person but lets the trust point upload the data on her behalf. Thus, whenever sensor data is uploaded, the Cloud cannot authenticate the data owner directly to obtain a correlation between the sensor data and its owner.

As a trivial solution, the Cloud provider could require the data owner to share her secret credentials, which she uses to authenticate herself against the Cloud, with her trust point. To minimize the influence of potential hacking attacks against the trust point, we want to avoid this solution and rather comply with the principle of least privilege [14]. Hence, we aim at establishing the desired association between the data owner and her sensor data by means that do not require the data owner to reveal her secret credentials to any other entity except the Cloud. We propose the use of the *OAuth 2.0* protocol [17], which can be used by the data owner to enable the trust point to access the protected sensor data upload interface in the Cloud on her behalf. OAuth thereby allows the trust point to prove that the data owner granted authorization for accessing (a subset of) protected resources by providing special tokens to the Cloud instead of using the data owner's secret credentials. In the following, we refer to this process as *pairing* of a data owner and a trust point.

Figure 6.4 illustrates the five steps in our pairing procedure:

1. **Initiation**. To initiate the pairing procedure, the data owner interacts with the user interface offered by the trust point, e.g., by using a web browser and chooses to pair herself and the trust point.
2. **Authorization request**. The trust point sends an authorization request to the Cloud by redirecting the data owner's web browser to the Cloud and thereby adding the necessary request parameters as `GET` parameters.
3. **Data owner authentication**. The data owner has to authenticate herself with the Cloud over a secure channel (e.g., HTTPS) using her secret credentials. Commonly, she provides a combination of user name and password. This is secure since the data owner and the Cloud established a protected communication channel and thus neither the trust point nor any other third party is able to obtain the secret credentials.

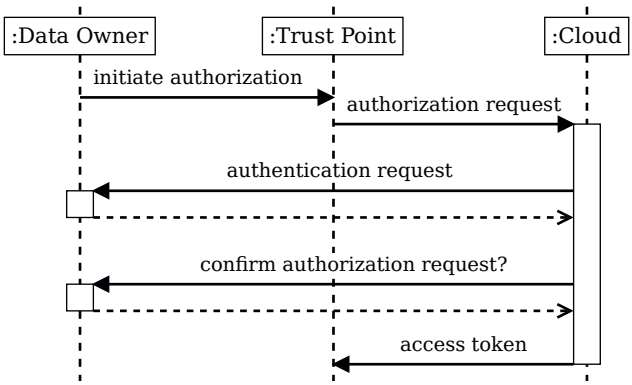


Fig. 6.4: The pairing of a data owner and a trust point is triggered by the data owner and results in an access token for the trust point.

4. **Trust Point Authorization**. The Cloud displays a summary of the trust point's authorization request for the successfully authenticated data owner, thereby revealing the trust point's identity and authorization scope to the data owner. Now, the data owner can check whether the presented identifier indeed corresponds to the identifier of her trust point. This identifier can, for instance, be chosen by the data owner during the initial setup of the trust point or pre-configured by the manufacturer and labeled on the physical device.
5. **Access token transfer**. If the data owner decides to grant authorization, the Cloud issues an authorization token to the trust point that the trust point can later on use to receive an access token. In order to deliver the authorization token, the Cloud redirects the web browser back to the user interface of the trust point. Additionally, the Cloud adds the authorization token, in form of an authorization code, as a `GET` parameter.

After this pairing procedure has been executed successfully, the trust point is in possession of an access token in order to store sensor data on behalf of the data owner. The cloud provider can safely assume that each entity that is able to present a valid access token has indeed been authorized by the data owner. Hence, access tokens suffice to signal the Cloud that the trust point acts as the representative of the data owner.

For this pairing procedure, a secure communication channel between the trust point and the Cloud has to be established in order to guarantee confidentiality of the communication. We discuss in the next section how this secure communication channel can be realized.

6.3.4 Securing the Communication Channel

Once the trust point and the Cloud have successfully been paired, the actual sensor data transmission can take place. However, the communication channel between the trust point and the Cloud has to be secured in order to guarantee confidentiality, authentication, and integrity. We thus identify three core requirements regarding the protection of this transmission: i) authentication of the Cloud against the trust point before potentially sensitive data is transferred to the Cloud, ii) authentication of the trust point for the Cloud before data is received, and iii) secure transport of information between trust point and the Cloud.

To realize these main high-level requirements for a transport security protocol when bridging sensor networks with the Cloud, we present practical requirements for transport security in this scenario. We then analytically evaluate three different standardized transport security protocols with respect to these requirements and thus retrieve the best choice for a transport security protocol in the SensorCloud scenario.

6.3.4.1 Practical Requirements

To meet the above discussed high-level core requirements, a transport security protocol for bridging sensor networks and the cloud has to fulfill a number of practical requirements. These practical requirements either directly originate from our design goals (Section 6.3.1), the assumed network scenario in the SensorCloud context, or from experience in the design and implementation of security architectures [18]. In the following, we discuss these practical requirements in more detail. This will allow us to then make a substantiated decision for a transport security protocol.

Authentication

The transport security protocol should provide support for mutual certificate-based authentication of the communication peers. In contrast to authentication using shared symmetric keys or plain asymmetric keys, certificate-based authentication allows to store additional, authenticated meta information in combination with the public key. Thus, certificates, e.g., include information about the issuer and expiration date. Additionally, compromised entities (i.e., trust points) can be invalidated before the expiration date of their certificates using pre-defined revocation lists. These aspects increase the ease of deployment and maintenance of the key management functionality as part of the SensorCloud security architecture significantly and at the same time reduce its susceptibility to errors.

Cryptographic Agility

The transport security protocol has to support the negotiation of the used cryptographic primitives. In the SensorCloud scenario, such a negotiation is required for two reasons. On the hand, gateways with potentially strongly varying computing

resources have to communicate with the Cloud. The negotiation of cryptographic primitives allows to adapt authentication and encryption to the available resources of the communication partners. On the other hand, we want our security architecture to evolve with the latest findings in the security community. Utilizing the negotiation of cryptographic primitives allows us to replace primitives classified as insecure without the need of switching to a completely new transport security protocol.

Implementation

The transport security protocol should not be developed in-house. Instead, we aim at leveraging a well-approved and standardized solution. This requirement originates from the recognition that often not the specification of a protocol itself but rather the actual implementation is vulnerable to attacks. Through avoiding in-house development, we can utilize widely used implementations of standardized protocols.

Connectivity

The security protocol must not limit the general connectivity of the gateway. On-path network elements such as NATs and firewalls [7] strongly limit the choice of applicable protocols in numerous network environments. In home networks and networks of smaller companies, Network Address and Port Translators (NAPT) impose special requirements for transport protocols in order to apply their address translation. Contrary, in larger company networks, firewalls restrict the choice of protocols to web and email protocols in order to prevent unwanted exchange of data.

6.3.4.2 Protocol Selection and Choice of Cryptographic Primitives

We base our selection of a transport security protocol for our trust point-based security architecture on our design goals (Section 6.3.1) as well as the discussed practical requirements. In order to select the best transport security protocol for the SensorCloud scenario, we analytically evaluated and compared the following three standard IP security protocols: i) Host Identity Protocol (HIP) [35], ii) Internet Key Exchange (IKE) [30], and iii) Transport Layer Security (TLS) [12]. Additionally, after selecting a transport security protocol, we also have to decide on the cryptographic primitives which should be used. Table 6.1 summarizes our evaluation and comparison of these security protocols which we discuss in the following.

The theoretical inspection of all three IP security protocols revealed that all candidates are suited for securing the communication between trust point and Cloud with respect to the identified security requirements. Additionally, all considered candidates offer support for certificate-based authentication and allow to dynamically negotiate the supported cryptographic primitives (cryptographic agility). Open source implementations for each of the evaluated protocol candidates exist, e.g., HIPL [45] for HIP, strongSwan [47] for IKE, and OpenSSL [46] for TLS.

	HIP [35]	IKE [30]	TLS [12]
Primary security properties	+	+	+
Authentication	+	+	+
Cryptographic agility	+	+	+
Implementation	o	+	++
Connectivity	o	o	+

Table 6.1: Results of the evaluation of transport security protocol for the SensorCloud scenario.

However, HIP and IKE cannot be used in all network scenarios without further measures. For example, they have to be explicitly whitelisted when using restrictive firewalls and require UDP encapsulation in order to make these port agnostic protocols translatable for NAPTs. In contrast, TLS over port 443 (HTTPS) is allowed in virtually all network scenarios and can thus be used without further configuration.

Based on these considerations, we chose TLS as secure transport protocol for our trust-point based security architecture. As cryptographic primitives we chose ECDSA with the elliptic curve NIST P-256 for the authentication of the communication peers, AES with a key length of 128 bit in CBC-mode for encryption, and SHA256 for data integrity protection. Based on current recommendations of the German Federal Office for Information Security (BSI) this combination of security procedures and levels is expected to be reasonable secure for protection data transmission beyond the year 2015. Even if these selected cryptographic primitives should become insecure one day, the cryptographic agility of TLS allows us to easily replace them with more secure primitives.

6.3.5 Protecting Sensor Data

To protect sensor data after the transport security has been stripped at the Cloud entry point, we additionally employ object security. The goal here is to secure the sensor data until it is requested and processed by an authorized service. Especially, the cloud provider should not be able to access or tamper with the stored sensor data. To this end, the trust point has to employ object security mechanisms for confidentiality and integrity. Only an authorized service can then access the sensor data. In particular, the protected information remains confidential during transport and storage and no attacker can tamper with the sensor data.

To realize confidentiality and integrity protection for sensor data during transport and storage, we first derive requirements for securing sensor data which arise from the SensorCloud scenario. Based on these requirements, we choose appropriate cryptographic primitives and a representation of protected sensor data.

6.3.5.1 Requirements for Protecting Sensor Data

We now identify requirements for object security that are especially important in the SensorCloud scenario. These requirements, which originate directly from our scenario description and security considerations (cf. Section 6.2 and Figure 6.1), have to be considered when choosing cryptographic primitives for object security mechanisms and serializing sensor data.

End-to-end Security

To deny unauthorized third parties access to sensor data in the Cloud, data items have to protected end-to-end from the trust point to an authorized service. This includes data in transfer as well as data stored in the Cloud. By protection, we mean confidentiality protection for potentially sensitive information and integrity protection for all information.

Flexible Encryption for Multiple Parties

As discussed in Section 6.3.2.1, one sensor data item usually consists of several data fields which either contain raw measurement values (e.g., temperature or humidity) or meta information (e.g., time or location). Typically, not every service needs access to all of these data fields. Thus, multiple parties (i.e., services) should each only have access to the specific subset of data fields of a sensor data item that they need in order to provision their service.

Integrity Protection and Proof of Origin

To prevent manipulation of sensor data in the Cloud, sensor data items have to be protected against manipulation during storage as well as during transport from the trust point to an authorized service. Additionally, sensor data has to be unambiguously associated with its data owner even after transport security has been terminated. This allows the Cloud provider on the one hand to employ additional access control mechanisms, e.g., for updating a stored data item, and on the other hand to realize proper billing of customers, e.g., based on the consumed storage space.

6.3.5.2 Choice of Cryptographic Primitives and Sensor Data Representation

We followed two goals while selecting our object security mechanism. First, the cryptographic primitives should guarantee the security of data outsourced to the Cloud for the longest possible period of time. To guarantee data security up to the year 2030, the US-american National Institute of Standards and Technology (NIST) recommends a security level of at least 112 bit [3]. Second, the computational costs on the trust point should be minimized. As we have discussed in Section 6.3.2.1, the trust point usually has very restricted processing resources. Whenever possible, we

hence try to leverage the computational asymmetry found in many cryptographic primitives in order to shift as many computational burden as possible to the Cloud instead of the trust point.

As discussed before, a data item can consist of multiple data fields. Our object security solution encrypts each of these data fields (i.e., raw data and meta information) with a separate symmetric key. This separation regarding the encryption of data fields enables the data owner to grant a service access to a specific subset of data fields. To additionally enable the data owner to restrict data access also to a certain period of time, we furthermore introduce the *key change interval*. The key change interval specifies the amount of time after which we periodically change each encryption key used for encrypting a specific data field. Thus, we offer the data owner the possibility to restrict access to her sensor data to a time interval at the granularity of the key change interval. When changing, e.g., the encryption keys every hour, the data owner can specify time constraints for accessing her sensor data at a granularity of one hour.

We chose the widely used AES cryptosystem [36] for data field encryption as well as integrity protection and use a key length of 128 bit with Galois/Counter Mode (GCM). Furthermore, we employ the asymmetric crypto system ECDSA [37] for guaranteeing integrity protection. This choice was made for two reasons. First, ECDSA allows for shorter key as well as signature length at the same security level when compared to factoring-based approaches such as RSA. This reduces the amount of data that has to be transferred to and stored in the Cloud. Second, ECDSA requires less time for signing a data item at the same security level when compared to factoring-based approaches. Thus, we can use the scarce and limited resources of the trust point more efficiently. In order to reach the determined security level, we use the NIST-defined elliptic curve P-224.

In order to serialize sensor data, the SensorCloud project decided to use JavaScript Object Notation (JSON) [10]. The representation of sensed information thereby is based on the Sensor Markup Language (SenML) [28] which is currently being standardized at the IETF. Thus, the solutions for object security and integrity protection must provide the possibility to apply them to objects serialized with JSON. To increase the interoperability of security measures between JSON-based protocols, the Javascript Object Signing and Encryption (JOSE) Working Group of the IETF is currently in the process of standardizing the representation of keying material as well as signatures and encryption of JSON objects. We leverage these efforts and use JOSE in order to represent the (encrypted) keys as JSON objects and to include both signature and encryption information in the JSON-represented sensor data item. Figure 6.5 exemplary shows a JSON-serialized sensor data item before (Figure 6.5a) and after applying our object security mechanisms (Figure 6.5b).

While designing and implementing our object security architecture, we take special care that the cryptographic primitives can be replaced without conceptional changes to the architecture. This allows to leverage new cryptographic primitives if future research findings should make this necessary or beneficial.

```
{
 "typ":"sensordata",
 "gw":"cb2a[...]319b0",
 "bn":"SensorNode 5",
 "bt":1378750285,
 "e":[{
  "n":"TemperatureSensor",
  "sv":"42"









 }],
 "sig":{}









}
```

(a) Plaintext data item

```
{
 "typ":"sensordata",
 "gw":"cb2a[...]19b0",
 "bn":"SensorNode 5",
 "bt":1378750285,
 "e":[{
  "n":"TemperatureSensor",
  "ev":[
   "unprotected":{
    "alg":"dir",
    "enc":"AESGCM128",
    "kid":"c730[...]597f",
    "typ":"sv"
   },
   "iv":"AQEB[...]AA==",
   "ciphertext":"38xx[...]UA=="
  ]
 }],
 "sig":{
  payload:"426a[...]a8bd",
  "signatures":[{
   "header":{"alg":"ES224"},
   "signature":"VeVr[...]Uj4="
  }]
 }
}
```

(b) Encrypted and signed data item

Fig. 6.5: A plaintext data item in its JOSE representation (Fig. 6.5a) and its preprocessed, secured version (Fig. 6.5b)

6.3.6 Service Access Granting and Key Management

To grant user-selected services access to specific data items of a sensor node, the trust point provides these services with the necessary keys for decrypting the protected sensor data. More precisely, the data owner instructs its trust point to reveal the data protection keys used for encrypting the data fields that a service is authorized to access. Thus, we realize a fine-grained access control on a per data-field basis for specified sensor nodes and selected services.

As the authorization of data access should be performed locally on the trust point for security as well as psychological and emotional reasons, the trust point potentially becomes a single point of failure. Thus, it has to be ensured that access to data stored in the Cloud is possible at any time, even if the trust point responsible for the data is not reachable. To ensure that after a service has been authorized, no third party can spoof the identity of this authorized service, the data owner has to be able to (cryptographically) identify a service. Similarly, it has to be ensured that only the authorized service itself is able to access the data fields for which it has been granted access. This requires that neither the Cloud provider nor its employees have direct access to the key material necessary for decryption. Additionally, a service must not be able to decrypt a data item or data field for which it has not been authorized.

Thus, a service must not have access to key material necessary for decryption of data items or data fields it is not authorized for.

To meet these requirements and to realize service access granting and key management for sensor data in the Cloud, we propose the use of *access policies* and a *key depot in the Cloud* and discuss these components in the following sections.

6.3.6.1 Access Policies

We introduce *access policies* on the trust point as means of instructing the trust point to release data protection keys to authorized services. These access policies are instructions for the trust point on which services should have access to specific data fields during a certain period of time. Our main motivation for realizing these policies on the trust point instead of in the Cloud is to maximize the control of the data owner over her data by performing this task locally. As the trust point belongs to the trust domain controlled by the data owner (cf. Section 6.3.2.2), the data owner also is the only person in control of managing access to her sensor data.

For the process of selecting services that should be authorized, we envision a service marketplace (similar to app stores for smart phones) in which all available services are listed. The market place has to provide a description of each service. At a minimum, descriptions state type and amount of usage of accessed sensor data. This statement has to be analyzed and verified by the Cloud provider, e.g., using static code analysis as it has previously been proposed for web applications [16]. All services available in the marketplace are advertised to the data owner on her local trust point.

Each service is identified by a public/private key-pair. The data owner can retrieve the public key of a service using the service marketplace. Should the data owner decide to authorize a certain service, she invokes the creation of a new access policy on her trust point. This access policy instructs the trust point to allow the authorized service to access the necessary (symmetric) keys for decrypting the specified sensor data which was generated during the specified period of time. To this end, the trust point encrypts the corresponding data protection keys with the public key of the service. Moreover, this procedure is repeated by the trust point whenever a data protection key is changed after one key change interval.

We selected RSA-2048 for encrypting these keys, as it provides sufficient security after the year 2013 [3]. Additionally, RSA as a factoring-based approach yields better encryption performance on the resource-constrained trust point than elliptic curve approaches. We encode the key using JSON and represent the encryption using JOSE, similarly to the encryption of data items as described in Section 6.3.5.2. The encrypted data protection keys can then be made available to the service in the Cloud without giving third parties the possibility to access them. Managing these encrypted keys reliably is the task of the second component of our access granting and key management scheme, the key depot in the Cloud.

6.3.6.2 Key Depot in the Cloud

In our scenario, the storage of all protected data encryption keys at the trust point is infeasible for two reasons. First, the limited storage resources of the trust point do not allow to store all encryption keys that were ever generated by this trust point, especially for fine-grained key change intervals. Second, if services had to fetch the keys from the trust point at the time of data access, they could only access data when the trust point is online and reachable. By outsourcing the storage of keys to the Cloud, we do not rely on the trust point to be online and reachable all the time. Even if the trust point is offline, the sensor data stored in the Cloud can be accessed by an authorized service at any time, because the keys needed for decryption as well as the actual data are securely stored in the Cloud.

Thus, after each key change interval, the trust point checks which services should at this point in time have access to which data field of a specific sensor. It then encrypts the affected keys with the public keys of all authorized services as discussed in Section 6.3.6.1. Then, instead of storing the keys locally, it sends them to the key depot in the Cloud, where the keys are stored persistently and can be accessed only by the authorized services.

The central task of the key depot in the Cloud is the reliable storage of protected data encryption keys, with which authorized services can decrypt the sensor data they have been granted access to. To this end, each (symmetric) key is identified by a trust point-unique key identifier. For each data item and data field, the corresponding key identifier of the key needed for decryption is stored. Thus, the Cloud is able to deliver the necessary keys for the decryption of a data item on request of the service. Only the service authorized to access the specific data can then decrypt the encrypted keys using its private key. Then, using the decrypted (symmetric) keys it is able to decrypt exactly those data fields for which it has explicitly been authorized by the data owner.

One core security requirement we identified before stated that third-parties should not be able to access keying material (and thus also the sensor data) in an unauthorized manner. As we encrypt the data encryption keys using asymmetric cryptography, neither the Cloud provider nor its employees are able to access these keys. Only the authorized service is in possession of the private key which is necessary for decrypting the encrypted keys. Thus, our security architecture ensures that only the authorized services are able to access the necessary keys while realizing high availability and redundancy by storing the keys in the Cloud.

As we will see in more detail in our evaluation of the storage overhead, the amount of storage resources needed for storing the encrypted data protection keys of one data field depends on the number of authorized services and the key change interval. The more services are authorized to access a specific data field and the shorter the key change interval, the more encrypted data protection keys have to be stored in the key depot in the Cloud.

6.4 Evaluation

To evaluate the performance of our trust point-based security architecture and prove its feasibility, we implemented OpenSSL-based prototypes for trust point, Cloud platform, and a service. We chose the Raspberry Pi as an exemplary embedded device for the trust point. More specifically, we use the Raspberry Pi Model B equipped with an 700 MHz ARM processor, 512 MiB of RAM, and Raspian Linux as operating system. For our performance evaluation of Cloud-hosted services, we use Amazon Web Services first generation EC2 64-bit instances of type large (M1.large) [1] with Ubuntu 12.04 LTS as operating system. The performance trade-offs of our proposed security architecture involve computing and storage resources on the trust point and in the cloud. We focus our overhead discussion on the object security and key management components of our security architecture as the performance overhead of our TLS-based transport security mechanism has already been studied extensively [9]. The following discussion is based on previous work [22, 20].

6.4.1 Performance Overhead

Regarding the trust point, performance overhead originates from the encryption of individual data *fields*, the integrity protection of each data *item*, and the encryption of symmetric data encryption keys for authorized services. Similarly, authorized services have to decrypt data encryption keys, verify the integrity of data *items*, and finally decrypt data *fields* before they can operate on the sensor data. As discussed in Section 6.3.6.1, the trust point periodically exchanges data encryption keys in order to allow fine-grained access control to the sensor data. Each key change results in one key encryption operation on the trust point and one key decryption operation for each service that is granted access to the corresponding sensor data. In the following, we evaluate and quantify the resulting overhead using our prototype implementation. We evaluate the mean time for processing one data item on the trust point as well as in a service. For each measurement point, we conducted a total of 50 measurements. We used a data item with exactly one data field. In our prototypical implementation, we used as cryptographic primitives AES with 128 bit keys in CBC mode for encrypting the data field, RSA with 2048-bit keys for encrypting the resulting AES keys, and ECDSA with NIST curve P-224 for signing the SHA-256 hash value of a data item as its integrity protection checksum.

Figure 6.6 shows the average processing time for one data item on the trust point with one authorized service for an increasing key change interval. In our evaluation, we performed one key change per run and one new data item was created every second. Even in the extreme case of changing the key every single second, the trust point can process 59 data items per second despite its restricted resources. If the key change interval is increased to 10 seconds, the trust point can process 167 data items per second. For key change intervals larger than 10 seconds, more than 80 percent of the processing time per data item result from the cryptographic operations that must

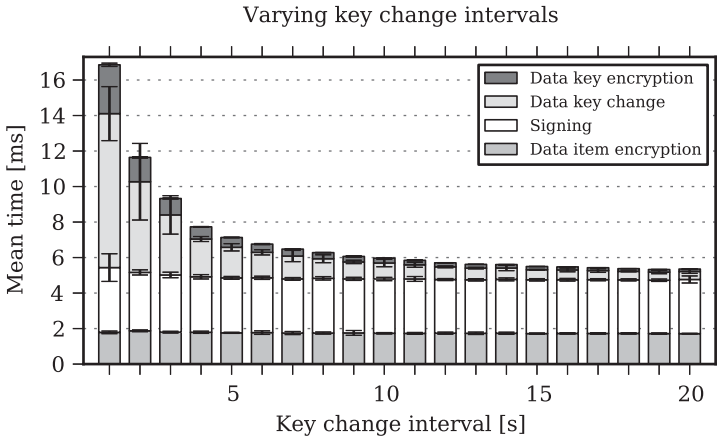


Fig. 6.6: The time for encrypting and signing one data item for one authorized service on the trust point depends on the key change interval.

be performed for each data item. Increasing the key change interval further thus has only a marginal impact on the processing time of one data item on the trust point.

Figure 6.7 depicts the average processing time for one data item on the trust point with a key change interval of 10 seconds and an increasing number of authorized services. We chose this key change interval based on our previous considerations regarding the key change performance overhead per data item. When adding an additional service, the number of key encryption operations that the trust point has to perform on each key change increases by one. Our results show that for ten authorized services and our considered key change interval of 10 seconds, the trust point can process 140 data items per second. The figure also shows that the data key encryption overhead for an increasing number of authorized services does not exceed the fixed per-data-item overhead for a realistic number of authorized services.

To put these results into perspective, we consider a scenario with a sampling rate of one measurement per second and sensor, in which the data of each sensor should be processed by ten authorized services. Even in this extreme case, our resource restricted trust point is able to process and thus protect the sensor measurements of more than 140 sensor nodes when it is configured to use a key change interval of 10 seconds. We can easily scale up the hardware on which the trust point runs or utilize a cryptographic co-processor in order to realize higher per-sensor sampling rates and larger amounts of sensor nodes. With these measures, we can achieve throughputs that are one to two orders of magnitude higher than those we can achieve with the Raspberry Pi. We identify the asymmetric cryptography operations that are used for data integrity protection as main limiting factor for protecting sensor data on the trust point. Fortunately, modern hardware allows to speed up and parallelize these computations.

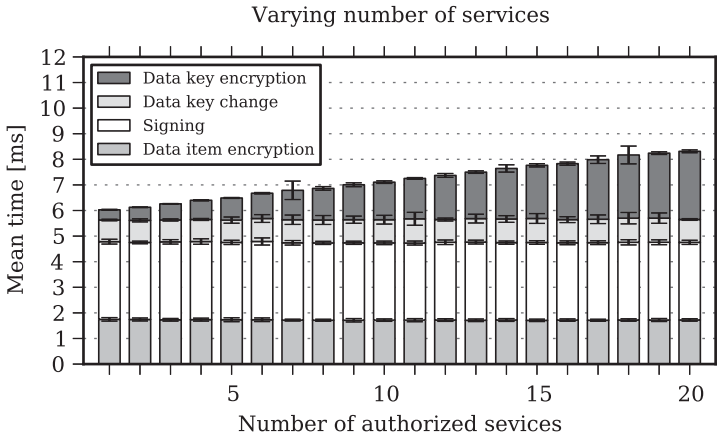


Fig. 6.7: The time for encrypting and signing one data item on the trust point with a fixed key change interval depends on the number of authorized services.

Finally, our results indicate that the number of data fields per data item is only a minimal influential factor for the maximum throughput we can achieve on the trust point. As we use symmetric encryption for encrypting data fields, the Raspberry Pi can perform the necessary operations efficiently in 0.004 ms. Thus, our architecture is able to easily scale with an increasing amount of data fields per data item.

With respect to the performance overhead of services, Figure 6.8 evaluates the average time for decrypting and verifying one data item in relation to the key change interval. Even in the extreme case of changing the data encryption key every second, the service can verify and decrypt 220 data items per second. Note that, due to a creation frequency of 1 data item per second, this value holds for the case of using each data encryption key only for one data item. As the key change interval reaches 20 seconds, the throughput increases by a factor of 4.5 such that over 1 000 data items can be processed per second.

Similar to the trust point, the measures for integrity protection constitute for a fixed performance overhead. To mitigate this performance overhead, we consider two different approaches: probabilistic verification and verification-as-a-service. Probabilistic verification exploits the assumption that the Cloud provider commonly is at least partly trustworthy. Thus, the service can probabilistically verify the integrity of only a random subset of the sensor data. This reduces the amount of expensive cryptographic operations and thus enormously increases the throughput of the service. Our preliminary results show that we can increase the throughput up to 3 600 data items per second when randomly verifying 10 percent of the data items and even up to 4 800 data items per second when randomly verifying only 1 percent of the data items. However, if the Cloud provider is completely untrusted, deterministic verification has to be used. Here, we propose to use verification-as-a-service,

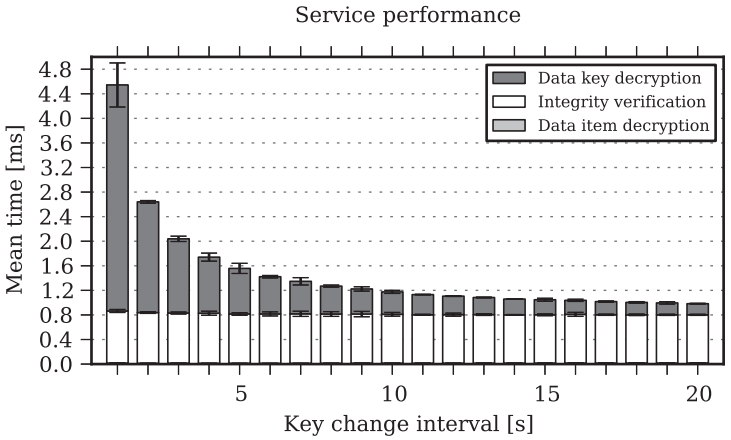


Fig. 6.8: The time for decrypting one data item and verifying its integrity protection in a service depends on the key change interval.

where dedicated, trusted services continuously verify the integrity of all data stored in the cloud [49].

To conclude, our performance evaluation shows that our trust-point based security architecture offers a sufficient scalability for the scenario faced by the SensorCloud project.

6.4.2 Storage Overhead

With respect to the storage overhead in the Cloud, we evaluate the size of additional information that has to be stored in the Cloud. Our trust point-based security architecture essentially requires to store two different types of information in the Cloud. First, the information needed for our object security mechanisms, i.e., information regarding the encryption and integrity protection of data, has to be stored in the Cloud. Second, the encrypted keys have to be stored in the Cloud for each authorized service. In the following, we quantify and discuss the overhead caused by storing this additional information in the Cloud.

Figure 6.9 shows the storage overhead of our object security mechanisms, i.e., encryption and integrity protection. For this, we take a simple data item with four meta data fields (data item type, trust point identifier, sensor identifier, and timestamp) which remain unencrypted and are JSON-encoded as discussed in Section 6.3.5.2. We then increase the number of data fields that contain measurement values, where each of them is to be encrypted. The base size of the JSON-encoded sensor data item with only the four meta data fields and thus without any measurement value consists of 109 bytes. Before encryption, each additional measured value adds ad-

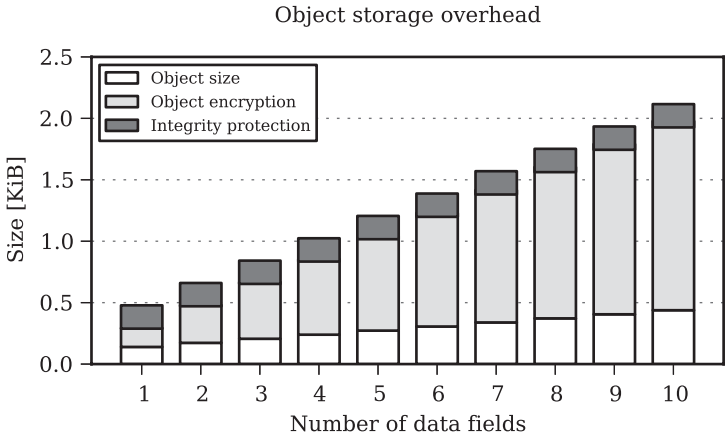


Fig. 6.9: The overhead of storing the encrypted data protection keys in the Cloud depends on the number of data fields.

ditional 34 bytes (under the assumption that each measured value fits into a 16 byte block). Thus, the base line for our storage overhead evaluation consists of 109 bytes plus 34 bytes for each measured value.

For our evaluation, we distinguish the overhead introduced per data *field* when considering encryption of data *items* and the overhead per data *item* when considering our integrity protection mechanisms. As the encrypted measured value has the same size as the unencrypted value, the overhead for encryption consists for each data field of 6 bytes for encoding the encrypted value, 22 bytes for the initialization vector and 131 bytes for the JSON-encoded JWE information. The constant per data item overhead for the integrity protection consists of 40 bytes for the hash over the data item, 86 bytes for the actual checksum, and 69 bytes for encoding it in JSON using JWS. Thus, we observe that the overhead for our object security mechanisms grows linearly with the number of data fields per data item. However, this linearly growing overhead can sufficiently be handled by the elastic storage provided by current Cloud offers for a sensible number of data fields per data item.

Table 6.2 shows the overhead for storing the RSA-2048 encrypted keys of one month of sensor data in the Cloud. As discussed in Section 6.3.6, the Cloud has to store an encrypted key for accessing protected sensor data for each authorized service in the Cloud's dedicated key depot. Hence, the overhead for storing the encrypted keys for one data field of a sensor strongly depends on the key change interval as well as the number of services which are authorized to access this specific data field. Additionally, we have a constant overhead for representing the key as well as its encryption in JSON using JOSE (cf. Section 6.3.6.1).

For this evaluation, we assume a per sensor sampling rate of one data item per second. Thus, for one month, we consider the keys needed to access 2 592 000 data items. Our calculations show that for reasonable key change intervals, the overhead

		Key Change Interval					
		1 second	1 minute	1 hour	1 day	1 week	1 month
Data Fields	1	1.27GiB	0.02GiB	0.36MiB	0.02MiB	2.21KiB	0.52KiB
	5	6.37GiB	0.11GiB	1.81MiB	0.08MiB	11.05KiB	2.58KiB
	10	12.75GiB	0.21GiB	3.63MiB	0.15MiB	22.10KiB	5.16KiB

Table 6.2: The overhead for storing the keys for one authorized service depends on the key change interval and the number of data fields.

for storing encrypted keys approaches more than feasible sizes. When changing the encryption key every hour, only 0.36 MiB of encrypted keys have to be stored per authorized service and data field. To further decrease this overhead, the data owner could aggregate or delete older data and, thus, render further storage of the corresponding encryption key unnecessary. To conclude, the elastic storage capabilities of nowadays Cloud solutions are capable of storing the encrypted keying material imposed by a reasonable key change interval.

6.5 Related Work

We split our discussion of related work into three categories. First, we discuss approaches for operating on encrypted data. We then consider Cloud architectures involving a trusted third-party for enabling secure processing and storage in the Cloud. Finally, we present other related approaches for securing data in the Cloud.

6.5.1 Secure Operations on Encrypted Data

The field of secure operations on encrypted data deals with the challenge how certain operations can be performed directly on encrypted data without the need to first decrypt it. We further classify this field into approaches for *secure computations* and for *secure data indexing*.

In the field of secure computations on encrypted data, proposed approaches aim at performing (mathematical) computations directly on the cipher text without the need for prior decryption. Thus, neither the input nor the output of a computation are revealed to the entity performing the computation. The most prominent example for such systems is *(fully) homomorphic encryption* [15, 39]. The core idea here is to exploit known mathematical properties of certain crypto systems to perform operations on encrypted values and thus yielding homomorphic cryptography. As

an example for homomorphic encryption, the multiplication of two ciphertexts generated by the Paillier cryptosystem [39] will result in the ciphertext of the sum of the corresponding plaintexts. Thus, an untrusted entity can calculate the sum of two values without learning the plaintext of the two input values nor the resulting sum. We see such properties as an promising approach for saving the overhead of decrypting data items, e.g., when performing simple statistical analysis over large amounts of data in the SensorCloud scenario. Our flexible design of the object security mechanisms allows us to incorporate a homomorphic encryption scheme when needs arise. While homomorphic encryption supports only one operation (addition or multiplication), fully homomorphic encryption [15] allows to perform arbitrary (mathematical) operations on ciphertext without revealing plaintext. Unfortunately, fully homomorphic encryption in its current state is considered highly inefficient for practical purposes [11]. However, we can extend our architecture and object security mechanisms to leverage fully homomorphic encryption once it becomes practically feasible.

Another approach for secure computations on encrypted data are *garbled circuits* [50, 32]. The goal of garbled circuits is to not only protect the confidentiality of the input and output of a computation but to also keep the computation instructions private. However, each computation requires the interaction with the data owner, who is responsible for protecting the computation instructions represented by a switching circuit. As this operation has to be performed by the resource-constrained trust point that may not be reachable when a service wants to perform a calculation, we consider this approach inappropriate for the SensorCloud scenario.

The idea behind secure data indexing is to enable the efficient lookup of encrypted data. When we consider the SensorCloud scenario, typical queries of a service will ask for the sensor data of a specific set of sensors within a certain interval of time. If we encrypt this meta data, the service would be required to first decrypt all data it is allowed to access to identify the relevant subset of data items. Contrary, if the meta data was not encrypted, sensitive information such as a person's location may be revealed. The CryptDB approach of Popa et al. [42] aims at resolving this dilemma. To support SQL queries over encrypted data, they support, among others, range and order queries as well as key word searches for specifically encrypted data [4, 42]. Additionally, they leverage the above mentioned concept of homomorphic encryption to enable the calculation of sum and average. Complementary to our work, we can integrate their results into our prototype to provide protection for meta information without limiting its usability for indexing and data retrieval.

6.5.2 Trusted Third-parties in Cloud Computing

The concept of trusted third-parties for securing storage and processing of the sensor data in the Cloud, similar to our proposed trust point, has previously been proposed for similar security architectures. Pearson et al. [41] present a security concept for data in the Cloud that puts a special focus on fine-grained access control of the data

owner over her outsourced data. The core idea of their approach is to attach a so-called *sticky policy* to each data item, that states how and under which circumstances the data can be used. When a service wants to access a certain data item it has to acknowledge the data item's policy by asserting the trusted third-party to not violate that policy. If the trusted authority is sufficiently convinced of the service's assertion, it releases the key needed for decrypting the data item. In contrast to our work, their focus lies on the enforcement of data usage policies. Additionally, our trust point is controlled by the user and thus automatically trusted by her while their trusted entity requires the establishment of a trust metric for external entities.

Kamara and Lauter [29] also propose a Cloud security architecture which introduces a dedicated trusted entity. Similarly to our approach, their dedicated machine is responsible for encrypting outgoing data and managing access policies. However, their approach focuses on the storage of data in the Cloud. Hence, they do not consider the secure processing of data by services running in the Cloud. Additionally, requesting a data item requires the retrieval of a specific token from the dedicated machine. Thus, the data stored in the Cloud cannot be accessed if the dedicated machine, running in the network of the data provider, is not reachable.

The Twin Clouds architecture presented by Bugiel et al. [6] utilizes two Clouds, a trusted private Cloud and an untrusted public Cloud. They utilize the concept of garbled circuits to protect data as well as computation instructions in the public Cloud. The authors propose to utilize the resources of a trusted private Cloud to encrypt data as well as computation instructions because this initial setup phase requires demanding computations. Afterwards, the protected computation instructions can securely be performed on demand in the public Cloud. The major draw backs of their approach are that the computation instructions only perform simple operations and that they have to be re-encrypted by the private Cloud after each execution.

6.5.3 Other Approaches to Secure Cloud Computing

In addition to the approaches discussed above, a number of other approaches for secure storage and processing in the Cloud have previously been proposed. We focus our discussion on those approaches that either facilitate the concept of trusted computing [33] or perform the data processing in the trust domain of the data owner.

The core idea of those approaches which leverage trusted computing for Cloud computing [27, 44, 48] is to use trusted hardware components to ensure (to a certain degree) that hardware and software components behave as specified. Typical use cases of trusted computing are memory curtaining, sealed storage, and remote attestation [33]. However, approaches utilizing trusted hardware components depend on hardware support at the IaaS layer, while our solution can be purely realized at the PaaS layer and thus utilize commodity IaaS offers. Additionally, to establish a chain of trust, trusted components are bound to specific hardware. This implies enormous challenges when migrating virtual instances between hardware platforms [48].

Approaches to process data locally [5, 29, 11] only utilize the storage capacities of the Cloud and thus cannot benefit from the elastic computation resources provided by the Cloud. These approaches commonly protect information before storing them in the Cloud. Thus, by processing the data locally within the trust domain of the data owner, these approaches can guarantee that neither the Cloud provider nor any other unauthorized third-parties can gain access to potentially sensitive information. While our security architecture is tailored towards storage *and* processing in the Cloud, it could be extended to support the usage of external services, i.e., services running outside the Cloud. Hence, our security architecture can also provide local processing for extremely sensitive sensor data if required by the data owner.

6.6 Conclusion

In this chapter, we described a trust point-based security architecture that aims at overcoming the adoption barriers of outsourcing storage and processing of sensor data to the Cloud. We first presented the abstract scenario of the SensorCloud project and its involved entities. We then discussed security considerations and derived design goals for a practically viable security architecture for the SensorCloud scenario. Based on these goals, we designed our trust point-based security architecture for sensor data in the Cloud that: i) provides a pairing in order to bind the identity of the data owner to her trust point, ii) secures the communication channel between the sensor network and the Cloud entry point, iii) additionally adds object security between the sensor network and services, and iv) realizes fine-grained, user-centric control over data access and key management. Our extensive evaluation of our prototypical implementation confirmed that our architecture provides sufficient performance and storage properties of today's commodity Cloud offers. In fact, we are convinced that the level of security introduced by our security architecture outweighs the measured performance and storage overhead by far. We now conclude this chapter with an outlook and discussion of future work.

6.6.1 Outlook and Future Work

As shown in our performance evaluation (cf. Section 6.4.1), the creation and verification of public-key signatures for integrity protection purposes constitute the major performance bottleneck of our current security architecture. Thus, in the future, we plan to investigate more efficient signature schemes that relieve both trust point and services from the high computational burdens implied by public-key cryptography. Our initial idea here is to use hash chains [31] in order to amortize the high computation cost of public-key signatures across multiple data items. Our aim is to leverage specifically tailored hash structures to significantly reduce the overhead for the creation as well as for the verification of per data-item signatures.

Currently, our prototype implementation does not support the revocation of data access rights. Of course, the data owner can terminate an access policy in order to stop a service from having access to any future sensor data. She can likewise instruct the Cloud to block access to historic data. However, to cryptographically revoke a service's access to data items already stored in the Cloud, these data items have to be re-encrypted and the resulting keys distributed to the remaining authorized services. Especially for large amounts of data, performing this re-encryption of data and re-distribution of keys on the trust point becomes infeasible. Thus, we aim at utilizing proxy re-encryption to perform these steps in the Cloud [51]. This concept allows offloading of the necessary, expensive computations to an untrusted Cloud environment without revealing any information about the underlying sensor data.

In a first step, we focused our work mainly on security issues that arise when outsourcing processing and storage of sensor data to the Cloud. However, in order to overcome the previously identified adoption barriers, we also have to consider privacy and data protection concerns [40, 52, 21]. One important subset of these concerns are those regarding the handling of data, e.g., in which jurisdiction and period of time data has to be stored [19, 21]. Especially in the EU, companies by law have to follow those data handling requirements when dealing with sensitive data such as sensed information. In the future, we will address these requirements by enriching the sensor data with corresponding data handling annotations [21, 19].

References

1. Amazon Web Services, Inc.: Amazon EC2 Instances. URL `http://aws.amazon.com/en/ec2/instance-types/`. Retrieved: 09/10/2013
2. Amazon Web Services, Inc.: AWS GovCloud (US) Region – Government Cloud Computing. URL `http://aws.amazon.com/en/govcloud-us/`. Retrieved: 09/10/2013
3. Barker, E., Barker, W., Burr, W., Polk, W., Smid, M.: Recommendation for Key Management – Part 1: General (Revision 3). Tech. rep., National Institute of Standards and Technology (2012)
4. Boneh, D., Waters, B.: Conjunctive, Subset, and Range Queries on Encrypted Data. In: S.P. Vadhan (ed.) Theory of Cryptography, *Lecture Notes in Computer Science*, vol. 4392. Springer (2007)
5. Bowers, K.D., Juels, A., Oprea, A.: HAIL: A High-Availability and Integrity Layer for Cloud Storage. In: Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS) (2009)
6. Bugiel, S., Nürnberger, S., Sadeghi, A.R., Schneider, T.: Twin Clouds: Secure Cloud Computing with Low Latency. In: B. Decker, J. Lapon, V. Naessens, A. Uhl (eds.) Communications and Multimedia Security, *Lecture Notes in Computer Science*, vol. 7025. Springer (2011)
7. Carpenter, B., Brim, S.: Middleboxes: Taxonomy and Issues. IETF RFC 3234 (Informational) (2002)
8. Chow, R., Golle, P., Jakobsson, M., Shi, E., Staddon, J., Masuoka, R., Molina, J.: Controlling Data in the Cloud: Outsourcing Computation without Outsourcing Control. In: Proceedings of the 2009 ACM Workshop on Cloud Computing Security (CCSW) (2009)
9. Coarfa, C., Druschel, P., Wallach, D.S.: Performance Analysis of TLS Web servers. ACM Trans. Comput. Syst. **24**(1) (2006)
10. Crockford, D.: The application/json Media Type for JavaScript Object Notation (JSON). RFC 4627 (2006)

11. Danezis, G., Livshits, B.: Towards Ensuring Client-Side Computational Integrity. In: Proceedings of the 3rd ACM Workshop on Cloud Computing Security (CCSW) (2011)
12. Dierks, T., Rescorla, E.: The Transport Layer Security (TLS) Protocol Version 1.2. IETF RFC 5246 (Proposed Standard) (2008)
13. Eggert, M., Häußling, R., Henze, M., Hermerschmidt, L., Hummen, R., Kerpen, D., Navarro Pérez, A., Rumpe, B., Thißen, D., Wehrle, K.: SensorCloud: Towards the Interdisciplinary Development of a Trustworthy Platform for Globally Interconnected Sensors and Actuators. Tech. rep., RWTH Aachen University (2013)
14. Ferraiolo, D., Kuhn, R.: Role-Based Access Control. In: 15th NIST-NCSC National Computer Security Conference (1992)
15. Gentry, C.: Computing Arbitrary Functions of Encrypted Data. Commun. ACM **53**(3) (2010)
16. Guarnieri, S., Livshits, B.: GATEKEEPER: Mostly Static Enforcement of Security and Reliability Policies for JavaScript Code. In: 18th USENIX Security Symposium (USENIX Security) (2009)
17. Hardt, D.: The OAuth 2.0 Authorization Framework. RFC 6749 (Proposed Standard) (2012)
18. Heer, T., Götz, S., Weingärtner, E., Wehrle, K.: Secure Wi-Fi Sharing at Global Scales. In: International Conference on Telecommunications (ICT) (2008)
19. Henze, M., Großfengels, M., Koprowski, M., Wehrle, K.: Towards Data Handling Requirements-aware Cloud Computing. In: 2013 IEEE International Conference on Cloud Computing Technology and Science (CloudCom) (2013)
20. Henze, M., Hummen, R., Matzutt, R., Catrein, D., Wehrle, K.: Maintaining User Control While Storing and Processing Sensor Data in the Cloud. International Journal of Grid and High Performance Computing (IJGHPC) **5**(4) (2013)
21. Henze, M., Hummen, R., Wehrle, K.: The Cloud Needs Cross-Layer Data Handling Annotations. In: 2013 IEEE Security and Privacy Workshops (2013)
22. Hummen, R., Henze, M., Catrein, D., Wehrle, K.: A Cloud Design for User-controlled Storage and Processing of Sensor Data. In: 2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom) (2012)
23. Hummen, R., Hiller, J., Henze, M., Wehrle, K.: Slimfit - A HIP DEX Compression Layer for the IP-based Internet of Things. In: 1st International Workshop on Internet of Things Communications and Technologies (IoT) (2013)
24. Hummen, R., Hiller, J., Wirtz, H., Henze, M., Shafagh, H., Wehrle, K.: 6LoWPAN Fragmentation Attacks and Mitigation Mechanisms. In: Proceedings of the sixth ACM Conference on Security and privacy in Wireless and Mobile Networks (WiSec) (2013)
25. Hummen, R., Shafagh, H., Raza, S., Voigt, T., Wehrle, K.: Delegation-based authentication and authorization for the ip-based internet of things. In: 2014 IEEE International Conference on Sensing, Communications and Networking (SECON) (2014)
26. Hummen, R., Wirtz, H., Ziegeldorf, J.H., Hiller, J., Wehrle, K.: Tailoring End-to-End IP Security Protocols to the Internet of Things. In: 21st IEEE International Conference on Network Protocols (ICNP) (2013)
27. Itani, W., Kayssi, A., Chehab, A.: Privacy as a Service: Privacy-Aware Data Storage and Processing in Cloud Computing Architectures. In: Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC) (2009)
28. Jennings, C., Shelby, Z., Arkko, J.: Media Types for Sensor Markup Language (SENML). IETF Internet-Draft draft-jennings-senml-10 (2013). Work in progress
29. Kamara, S., Lauter, K.: Cryptographic Cloud Storage. In: R. Sion, R. Curtmola, S. Dietrich, A. Kiayias, J. Miret, K. Sako, F. Sebé (eds.) Financial Cryptography and Data Security, *Lecture Notes in Computer Science*, vol. 6054. Springer (2010)
30. Kaufman, C., Hoffman, P., Nir, Y., Eronen, P.: Internet Key Exchange Protocol Version 2 (IKEv2). IETF RFC 5996 (Proposed Standard) (2010)
31. Lamport, L.: Password Authentication with Insecure Communication. Commun. ACM **24**(11) (1981)
32. Lindell, Y., Pinkas, B.: A Proof of Security of Yao's Protocol for Two-Party Computation. Journal of Cryptology **22**(2) (2009)

33. Mitchell, C.J. (ed.): Trusted Computing. IEE (2005)
34. Montenegro, G., Kushalnagar, N., Hui, J., Culler, D.: Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944 (2007)
35. Moskowitz, R., Nikander, P., Jokela, P., Henderson, T.: Host Identity Protocol. IETF RFC 5201 (Experimental) (2008)
36. National Institute of Standards and Technology: FIPS PUB 197: Advanced Encryption Standard (AES) (2001)
37. National Institute of Standards and Technology: FIPS PUB 186-4: Digital Signature Standard (DSS) (2013)
38. Navarro Pérez, A., Rumpe, B.: Modeling Cloud Architectures as Interactive Systems. In: 2nd International Workshop on Model-Driven Engineering for High Performance and Cloud Computing (MDHPCL) (2013)
39. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In: J. Stern (ed.) Advances in Cryptology — EUROCRYPT '99, *Lecture Notes in Computer Science*, vol. 1592. Springer (1999)
40. Pearson, S., Benameur, A.: Privacy, Security and Trust Issues Arising from Cloud Computing. In: 2010 IEEE 2nd International Conference on Cloud Computing Technology and Science (CloudCom) (2010)
41. Pearson, S., Mont, M.C., Chen, L., Reed, A.: End-to-End Policy-Based Encryption and Management of Data in the Cloud. In: 2011 IEEE 3rd International Conference on Cloud Computing Technology and Science (CloudCom) (2011)
42. Popa, R.A., Redfield, C.M.S., Zeldovich, N., Balakrishnan, H.: CryptDB: Protecting Confidentiality with Encrypted Query Processing. In: Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles (SOSP '11) (2011)
43. Robertson, J.: How Private Data Became Public on Amazon's Cloud. URL `http://www.bloomberg.com/news/2013-03-26/how-private-data-became-public-on-amazon-s-cloud.html`. Retrieved: 09/10/2013
44. Santos, N., Gummadi, K.P., Rodrigues, R.: Towards Trusted Cloud Computing. In: USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '09) (2009)
45. The HIPL Project: Host Identity Protocol for Linux. online @ https://launchpad.net/hipl (2013)
46. The OpenSSL Project: OpenSSL. online @ http://www.openssl.org/ (2013)
47. The strongSwan Project: strongSwan - IPsec for Linux. online @ http://strongswan.org (2013)
48. Wallom, D., Turilli, M., Taylor, G., Hargreaves, N., Martin, A., Raun, A., McMoran, A.: myTrustedCloud: Trusted Cloud Infrastructure for Security-critical Computation and Data Managment. In: 2011 IEEE 3rd International Conference on Cloud Computing Technology and Science (CloudCom) (2011)
49. Wang, Q., Wang, C., Ren, K., Lou, W., Li, J.: Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing. IEEE Trans. Parallel Distrib. Syst. **22**(5), 847–859 (2011)
50. Yao, A.C.C.: How to Generate and Exchange Secrets. In: 27th Annual Symposium on Foundations of Computer Science (1986)
51. Yu, S., Wang, C., Ren, K., Lou, W.: Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing. In: 2010 Proceedings IEEE INFOCOM (2010)
52. Ziegeldorf, J.H., Garcia Morchon, O., Wehrle, K.: Privacy in the Internet of Things: Threats and Challenges. Security and Communication Networks (2013)
53. ZigBee Alliance: ZigBee 2012 Specification (2012)
54. ZigBee Alliance: ZigBee Smart Energy Profile 2 (2013)

Part II
Software Engineering and Software Quality

Chapter 7
Quality Analysis Approaches for Cloud Services – Towards a Framework along the Customer's Activity Cycle

Jan Wollersheim and Helmut Krcmar

Abstract This literature review synthesizes the existing research on approaches to the quality analysis of cloud services by investigating 27 relevant sources in detail. Furthermore, the sources are discussed alongside the three dimensions of service quality and the customer's activity cycle: the pre-purchase, purchase and post-purchase phases. This work reveals multiple quality analysis approaches that focus either on spe-cific quality aspects or on a specific step in the activity cycle. Only few approach-es could be identified that actively support customers throughout all phases of their quality analysis work and fully cover the dimensions of service quality. However, because approaches for specific dimensions or phases are far more de-tailed and offer a greater ability to be tailored to challenges in the cloud service ecosystem, we call for further research towards integrating them along the cus-tomer's perspective. Furthermore, empirical studies investigating the challenges faced by organizations in analyzing the quality of cloud services are required to enable the mapping of the long list of literature-based findings presented in this paper and most gnawing questions of organizations in the field.

7.1 Introduction

Cloud computing services are increasingly being accepted by private, governmental and organizational users in Germany. However, organizations lack experience and struggle to analyze and assess cloud service offerings with regard to their individual advantages or disadvantages for the organization [1]. As observed in the liter-

Jan Wollersheim
Fortiss GmbH, München, Germany
e-mail: wollersheim@fortiss.org

Helmut Krcmar
Technische Universität München, München, Germany
e-mail: krcmar@in.tum.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_7

ature on organizational procurement [2,3], organizations need to review and adapt their procurement processes, including activities concerning quality determination as well as quality analysis and assessment to mitigate the risks associated with the procurement of products or services that are new to the organization.

The delivery model of cloud services is defined as "enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction," according to the definition of the National Institute of Standards and Technology [NIST] [4,5], which has gained popularity and seems to achieve agreement from practitioners and academics alike [6]. Following NIST, cloud services are distinguished by five essential characteristics:

- on-demand self-service (provisioning of computing capabilities does not require human interaction with service providers),
- broad network access (service capabilities are available and accessed over the network),
- resource pooling (the provider's computing resources are pooled to serve multiple customers along a multi-tenant model),
- rapid elasticity (service capabilities appear unlimited to the customer and can be elastically provisioned and released to commensurate with demand) and
- measured service (cloud services use an appropriate metering capability to monitor control and report service usage) [5].

Cloud services can be associated with one of three service models, each describing a different service functionality, i.e., Software as a Service (SaaS, where applications are accessible over the network), Platform as a Service (PaaS, where customers can deploy applications on platforms available over the network) and Infrastructure as a Service (IaaS, where customers can access fundamental computing resources such as storage or processing capabilities via a network to, e.g., install platform software and applications). Traditionally, providers of Information Technology (IT) services seek to engage in long-term business relationships, providing their clients with tailored, individual services. To specify this individual demand, client firms rely on a multi-step process starting with information that is easily obtainable from service providers. Based on this rather superficial primary information, the demanded functionality is outlined and sent to potential service providers. Typically, tools such as Requests for Information (RFIs) and Requests for Proposal (RFPs) are used to detail a firm's requirements stepwise and in coordination with providers and their expertise. A provider's replies to those requests, outlines the possible resolutions and associated prices, upon which the procuring organization can easily establish its sourcing decision.

In contrast to tailored IT services, cloud computing services are by definition standardized, industrialized and created for an anonymous market rather than being customer-specific. The iterative development process relying on providers' expertise and problem-solving capabilities is therefore not applicable to cloud services [7,8]. The mapping of individual quality requirements to the solutions on offer needs to be conducted by the procuring organizations themselves. Furthermore, procurement

processes designed to acquire standard software such as office suites or IT hardware typically rely on large sets of evaluation criteria that measure a product's quality but neglect measures concerning the ongoing usage phase of services after the initial buying situation. Because the consumption of services is an ongoing exchange process, suitable quality metrics and performance indicators to measure a service's quality are needed for both the assessment as part of the purchasing process as well as an ongoing monitoring throughout the service-usage phase. Consequently, organizations face the challenge of reviewing and adapting their procurement processes to cloud services, combining quality characteristics of individual IT services as well as standardized IT products [9].

Following Grönroos [10], the quality of services consists of three dimensions: the technical quality of the outcome, the functional quality of the process and the image of the service provider. The technical quality dimension is concerned with what customers receive through their interactions with a firm, as the customer will be influenced by the way in which the outcome or end result of the process (technical quality) is transferred to him. Furthermore, other customers simultaneously consuming the same or different services may influence the way in which a given customer will perceive a service. This is another quality dimension called the functional quality of the process [10]. Moreover, the image of a service provider is of utmost importance because it can affect the perception of quality in various ways, e.g., minor mistakes will most likely be forgiven if an organization holds a positive image or vice versa. Therefore, the dimension "image" can be viewed as a filter for the other two dimensions [10]. Figure 1 as follows subsumes the associations explained.

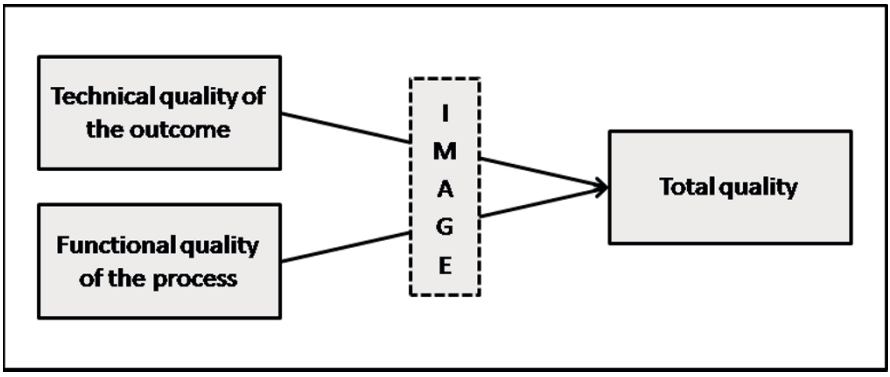


Fig. 7.1: Dimensions of service quality [Following 11,12,10]

Because the requirements and perceptions of a service's performance as well as specific service characteristics vary, quality can be defined as "whatever the customer perceives it to be" [10]. One organization might require 24/7 support, while another might require a flexible pricing schema and specific interfaces to core back-office systems. Because cloud services are only becoming more and more accepted,

organizations lack the experience and blueprints of suitable procurement process designs and quality metrics to address this type of service delivery model. Procurement processes need to be reviewed and, if necessary, adapted or established anew if products or services that are new to the organization are to be procured [13]. Quality criteria that map an organization's requirements need to be outlined individually; however, organizations require guidance in tailoring the existing frameworks and norms to the challenges and characteristics that are typical of cloud services.

To guide organizations in adapting their quality determination and quality analysis approaches to cloud services, we start with reviewing the approaches currently described in related academic as well as practitioner-oriented literature. Procuring cloud services requires skills and knowledge from domains such as IT, business activities to be supported, purchasing, legal or finance. Those cross-functional quality requirements are nothing entirely new in the domain of IT procurement. To structure cloud-sourcing activities, organizations can build upon existing knowledge such as the IT procurement process framework developed by the SIM Working Group on IT Procurement [14], the procurement frameworks of McQueen/Teh [15] and Verville/Halingten [16] or guidelines on packaged software acquisition [17]. However, those frameworks are not yet tailored to the domain of cloud services. It is necessary to concretize and complement these process blueprints concerning the quality determination and quality analysis approaches for the domain of cloud services.

Aiming to conduct the required tailoring, the remainder of this paper will first give an overview of our research process, building on the dimensions of quality outlined by Grönroos [10] and the existing literature in related domains. The dimensions described by Grönroos serve to structure the findings in literature. Subsequently, we discuss the results of the literature review and structure them from a customer's perspective on service quality analysis. The final section of this paper discusses the review's limitations and outlines the identified demands for further research on quality analysis approaches for cloud services.

7.2 Research design

To identify suitable quality analysis approaches for cloud services, we base our work on a recent literature review performed by Hoberg et al. [6]. Their scope of the reviewed literature includes the governance of cloud services and thus includes aspects of quality analysis. According to the taxonomy of literature reviews by Cooper [18], the coverage of the review by Hoberg et al. [6] can be classified as representative because it is limited to samples of articles that also represent other articles but does not explicitly consider the entirety of the literature. To identify the major articles, Hoberg et al. reviewed the major research journals in the area of information systems, selected based on the top 25 research journals according to the ranking developed by Lowry et al. (2004). In addition, the IBM Systems Journal, which is listed as top global practitioner journal (Lowry et al., 2004), was included. Ad-

ditionally, IS conferences were considered in order to cover more recent research not yet published in journals. Thus, the results published by Hoberg et al. [6] provide a profound foundation for this paper, especially because their work follows the framework outlined by vom Brocke et al. [19], which highlights the need for the comprehensible documentation of the process of literature search in a review article. The framework describes five phases to structure a literature review and is also used to guide this research endeavor, described as follows.

Phase 1 (Review Scope): Hoberg et al. [6] outline that research on structures and processes to govern cloud services is still at an early stage and a subsequent backward/forward search could provide further insights. Building upon seven articles identified by Hoberg et al. concerned with the quality analysis of cloud services from a customer's perspective, it is our goal to integrate the findings with respect to the three dimensions of quality outlined by Grönroos [10]. Following the taxonomy of literature reviews by Cooper [18], the focus of our literature review subsequently presented is the research outcomes as well as applications of quality analysis approaches in the domain of cloud computing. The review organization is conceptual, aiming at a neutral representation targeting an audience of specialized scholars as well as practitioners. The coverage of the review is representative because it includes a sample of articles that might typify larger groups of articles.

Phase 2 (Conceptualization): This step addresses the need for a broad conception of what is known about the topic [20]. We use the fundamental work of Grönroos [10] regarding the dimensions of service quality to structure and conceptualize this review.

Phase 3 (Sources): The literature search considers the sources presented in Table 1a/b. Starting from the initial seven papers identified by the review by Hoberg et al., a backward and forward search was performed. The literature used within the identified articles was analyzed and a forward search using the web of science was performed. The decision about whether a retrieved article will be analyzed in detail in this literature review was made based on the titles and abstracts. If the title sounded relevant to the focus of this review, the abstract was screened to make a final decision. Furthermore, complementary to the literature review, practitioner-oriented publications and seminal books in the related discipline of procuring IT, especially standard software, were investigated and added to obtain a full picture. The two following phases of the framework for the literature review, literature analysis and synthesis (4) as well as the development of a research agenda (5) are described in detail in the following sections, as these present the main contribution of the paper.

7.3 Findings

The three dimensions of quality outlined by Grönroos [10] are used to structure the identified sources and findings of this literature review. If an article contributes to two dimensions, it is mentioned in both sections. However, because some articles

contribute to all three dimensions of quality, a dedicated section for those contributions precedes the dimension-specific entries.

7.3.1 Findings: Quality analysis approaches focusing on all dimensions of quality

This section presents the identified quality analysis approaches in the research, addressing all three dimensions of service quality. Overall, nine of the 27 identified articles and books aim to address all three dimensions of quality, with a varying degree

Author Quality	**Quality - Technical**	**Quality - Functional**	**Quality - Image**
Beimborn et al. 2011 [21]	x		
Clemons/Chen 2011 [22]		x	x
Hay et al. 2011 [23]	x		
Jansen 2011 [24]	x	x	
Koehler et al. 2010 [25]		x	
Martens/Teuteberg 2011 [26]		x	
Zainuddin/Gonzales 2011 [27]		x	
Sources identified by backward and forward search			
Benlian et al. 2011 [28]	x		
Cloud Security Alliance (CSA) 2009 [29]	x	x	
ENISA 2009 [30]	x		
ENISA 2012 [31]	x	x	x
ISO/IEC 25010 2011 [32]	x	x	x
Martens et al. 2011 [33]	x		
Prodan/Ostermann 2009 [34]	x		
Repschläger et al. 2011 [35]	x	x	
Repschläger et al. 2011 [36]	x	x	
Repschläger et al. 2012 [37]	x	x	
Repschläger et al. 2013 [38]	x	x	x
Rimal et al. 2009 [39]	x		
Complimentary practitioner-oriented publications and seminal books			
BITKOM 2010 [40]	x	x	x
BSI 2012 [41]	x	x	x
Cloud Security Alliance (CSA) 2011 [42]	x	x	
Eichler 2010 [43]	(x)	(x)	(x)
EuroCloud 2010 [44]	x	x	x
Gronau 2012 [45]	x	x	x
Siegel et al. 2012 [46]	x	x	x
SMI 2011 [47]	x	x	x

Table 7.1: Results of the literature review. Legend: research contributes to the quality-dimension: x, literature review covering articles addressing this quality-dimension: (x)

of detail. The first article analyzed, which was published by the German Association for Information Technology, Telecommunications and New Media (BITKOM) [40], takes a rather practitioner-oriented approach towards quality analysis. Commencing from the general management decisions needed for cloud computing, the paper aims at guiding managers through potential upcoming decision processes to position their organization and prepare for upcoming cloud procurement decisions. The paper highlights criteria such as interfaces to legacy systems, service-level agreements (SLA) in general or support skills to make organizations aware of specifying their requirements concerning these aspects and the level of performance demanded per criteria [40]. The list of quality criteria by BITKOM strives for all three dimensions of quality; however, the dimensions of "image" and "functional quality" are covered only sparsely. The same holds true for the criteria lists provided by the Federal Office for Information Security (BSI) [41], EuroCloud [44], Repschläger et al. [38], the European network and Information Security Agency (ENISA) [31] and the Cloud Security Alliance (CSA) [42]. These lists contain detailed technical criteria that are mostly related to security aspects while only briefly covering other quality dimensions.

Approaches without such a strong technical focus are described by Gronau [45] and the Service Management Institute (SMI) [47,46]. While aiming at the support of practitioners struggling with the analysis and assessment to be conducted when procuring a specific software solution such as an enterprise resource-planning software, Gronau [45] structures his work along the course of action when going through the procurement process. The steps taken to define one's demand and analyze the quality of products or services on offer are therefore split over multiple sections of his book. Compared to the technically oriented papers mentioned above, the criteria guidelines are less detailed, and due to the focus on procuring ERP software, some of them might not be suitable for cloud services. However, all quality dimensions described by Grönroos [10] are covered and examples to guide practitioners are given. Without following the procurement process step by step, the Service Measurement Index (SMI) developed under the lead of Carnegie Mellon University addresses the need for industry-wide accepted measures to calculate the benefits and risks of cloud computing services [46]. SMI is designed to enable the analysis and assessment of cloud services before using a service as well as while using the service. The domains of quality covered by the SMI are accountability, agility, assurance, financials, performance, security and privacy, and usability. The SMI covers a total of 51 attributes within these seven domains and provides examples of measures to guide (potential) customers of cloud services when determining their quality requirements as well as when assessing and comparing the quality of cloud services on offer. Figure 2 as follows outlines the design of the SMI.

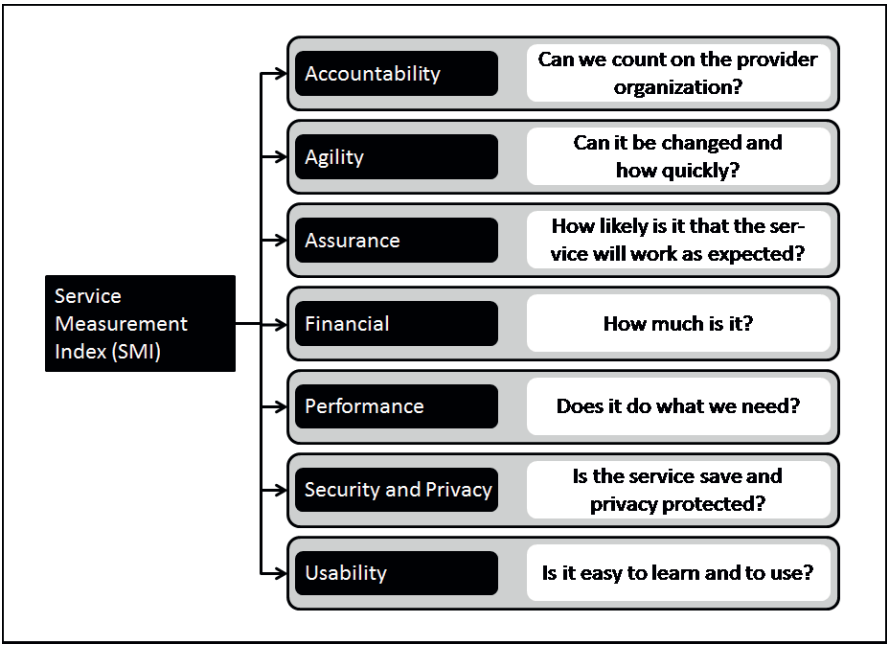


Fig. 7.2: The Service Measurement Index [Following 46]

7.3.2 Findings: Quality analysis approaches focusing on technical aspects of quality

Quality analysis approaches focusing on technical aspects are quite popular, as this literature review reveals. While some authors focus on security aspects of cloud services [30,29,23,24], others address additional aspects such as technical performance indicators, including the interoperability between services and programming frameworks [39,32]. Beimborn et al. investigate technical quality aspects for different cloud service models (IaaS, PaaS, SaaS) and complementary services for the cloud service ecosystem. Business-related criteria as well as criteria related to the technology underlying all types of cloud services are analyzed by Prodan et al. [34], who propose criteria taxonomies that, e.g., address the programming interface or pricing options for services. In addition to reviewing these criteria, Martens et al. 2011 [33] design a community platform to ease an organization's search process for cloud services. The input provided by the community platform designed by the authors covers multiple aspects related to the quality perceived during past service encounters. The feedback of service-using organizations beyond single-criteria assessments extends the focus beyond single-criteria weightings towards a comprehensive service assessment.

Repschläger et al. 2011 [35,37,36] build on some of the sources described previously when they designed their set of criteria, aiming to support organizations

throughout the process of cloud service selection. The authors design a core set of criteria that aims to guide organizations in structuring their requirements first as well as support them throughout the subsequent assessment of qualities on offer versus their individual demand. Because the work of Repschläger et al. [37] provides a broad and comprehensive overview on technical quality criteria as well as on a few functional criteria highlighted in the next section of this paper, figure 3 as follows outlines an overview of their results.

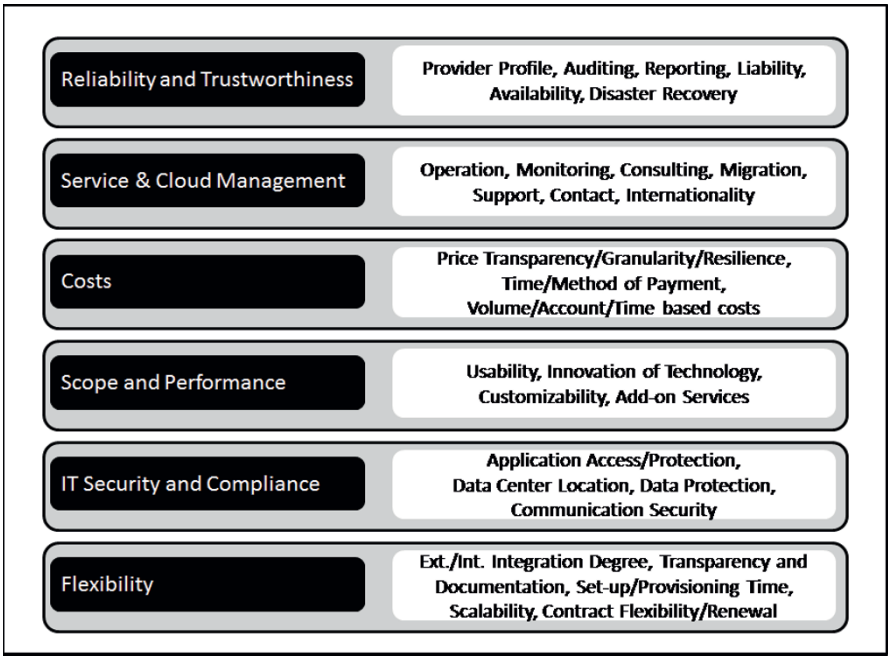


Fig. 7.3: Selection Criteria for Software as a Service [Following 37]

Focusing on quality assessment after the initial buying decision throughout or after the usage of services, Benlian et al. 2011 [28] developed SaaS-Qual, a measure that captures service quality evaluations for SaaS services. Based on previous work on SERVQUAL [48,49], the SaaS literature and empirical research, the authors develop a zones-of-tolerance-based service quality measurement instrument particularly for SaaS services, called SaaS-Qual. Throughout their research, Benlian et al. [28] validated the established service quality dimensions derived from previous SERVQUAL research such as rapport, responsiveness, reliability and features. Furthermore, they could identify two new dimensions, security and flexibility, which are essential for the evaluation of the service quality of SaaS solutions. The resulting measurement instrument is intended to be used by providers as well as by users to spot the strengths and weaknesses in the delivery of SaaS services [28]. Covering technical quality characteristics of both phases in the procurement and

usage cycles of cloud service customers before and after the actual procurement decision, the norm ISO/IEC 25010 [32] provides a comprehensive list of quality characteristics that are important when determining quality needs as well as when measuring the quality of the service received. However, because the ISO/IEC norm is designed for IT services in general, some of the more cloud-specific aspects outlined in the research by Repschläger et al. [37,38] or Benlian et al. [28] are not addressed in such detail.

7.3.3 Findings: Quality analysis approaches focusing on functional quality or image

Eleven of the 27 articles and books identified contain quality analysis approaches that focus on aspects of functional quality or image. The norm ISO/IEC 25010 [32] provides a structured list of quality characteristics that cover the quality of IT services and IT products as such as well as quality characteristics that gain importance when products or services are used. Two of the articles identified specifically address quality attributes concerned with the processes of service delivery and consumption [25,26]. Koehler et al. [25] focus on aspects such as the training required to use a service, the data formats required to exchange and thus use the service, and the way in which support is provided. The audit and subsequently the management of risks and compliance aspects is the focus of Martens and Teuteberg [26]. Further aspects such as the configurability, maturity and value co-creation in Software as a Service settings are covered by Zainuddin and Gonzales [27]. Driving attention to the possibilities of shirking, which include deliberate underperformance or the theft of data by the service provider, Clemons and Chen [22] add aspects to the quality dimension of "image."

The remaining papers and books mainly extend the quality criteria previously outlined in the technical dimension towards the functional dimension, including the service delivery and consumption process. The aspects added are, e.g., concerned with managing and supervising the service delivery process, such as reporting and monitoring options, the reliability and availability of a cloud service, data protection measures or the capability of the provider to cope with attacks [24,35-37]. Additional and more detailed criteria are added by the cloud security alliance [29,42], which highlights the importance of business continuity and disaster recovery measures, the need for monitoring functionality concerning data center operations, functionality to report bugs and other incidents, the encryption and key management or identity and access management functionality needed throughout the operation phase of cloud services.

7.4 Discussion

The identified literature on quality analysis approaches for cloud services addresses all three dimensions of quality outlined by Grönroos [10] in varying detail. Furthermore, some of the approaches identified aim to support organizations throughout the procurement process; others focus on quality analysis and evaluation throughout the phase of service usage or in retrospect after using it.

To structure the results from a customer's point of view, we use the customer's activity cycle as a reference framework to structure the papers identified. The customer's activity cycle structures the means or mechanisms as well as the design, delivery and support of IT solutions through the service lifetime from a customer's point of view, also known as the pre-purchase, purchase and post-purchase sequence [50]. Following Piccolio et al. [51] and Ives et al. [52], this cycle can be broken down into two major steps. The first step covers all aspects and activities until the organization actually starts using the service. It concentrates on the determination of quality requirements and the acquisition process. Here, the customer first realizes the need for a specific service and begins to focus on the required attributes and characteristics. The second step is concerned with activities following the procurement process. It covers activities such as the integration of the service into the existing IT landscape, adjusting configurations, monitoring the service's performance and ensuring an effective as well as efficient service use [51]. Additionally, the concluding step in every customer's activity cycle is retirement, according to Piccolio et al. [51]. Here, the customer is about to finish using a service and may begin to think of buying the old service again or evaluating a new one. A similar pattern of customer activities in evaluating a service is outlined by Grönroos [53], changing from an initial stage in which the customer evaluates the promises given by services on offer in relation to what he is looking for towards the evaluation of the service actually received (perception). If the customer still considers the value a service offered to be good enough and the customer is satisfied with the perceived quality, a repeated cycle might commence. Classifying the literature reviewed according to the customer's activity cycle, the following table 2a/b is developed.

The quality analysis approaches reviewed aim to support customers in different phases of their activity cycle. Because most of the approaches outlined are not integrated or combined along the customer's activity cycle, customers have to bridge gaps and remove obstacles when transferring, e.g., the quality criteria specified using one approach in an early phase towards using another approach in a later phase of the overall activity cycle.

Few approaches support more than one stage of the customer's activity cycle, namely, the work by the ISO/IEC [32], Cloud Security Alliance, ENISA, EuroCloud as well as the SMI outlined in the study by Siegel et al. [47,46]. All approaches provide a set of possible quality criteria in different quality dimensions, and all except one leave the actual development of individual measures to the applying organization without providing specific guidance. Only one of the outlined approaches, the SMI, provides not only lists of possible quality criteria to be measured but further provides assistance regarding how to develop one's own customized measures

Author	Pre-Purchase Focus	Post-Purchase Focus
Beimborn et al. 2011 [21]	x	
Benlian et al. 2011 [28]		x
BITKOM 2010 [40]	x	
BSI 2012 [41]	x	
Clemons/Chen 2011 [22]	x	
Cloud Security Alliance (CSA) 2009 [29]	x	x
Cloud Security Alliance (CSA) 2011 [42]	x	x
ENISA 2009 [30]	x	x
ENISA 2012 [31]	x	x
EuroCloud 2010 [44]	x	
Gronau 2012 [45]	x	
Hay et al. 2011 [23]	x	
ISO/IEC 25010 [32]	x	x
Jansen 2011 [24]	x	
Koehler et al. 2010 [25]		x
Martens et al. 2011 [33]		x
Martens/Teuteberg 2011 [26]		x
Prodan/Ostermann 2009 [34]	x	
Repschläger et al. 2011 [35]	x	
Repschläger et al. 2011 [36]	x	
Repschläger et al. 2012 [37]	x	
Repschläger et al. 2013 [38]	x	
Rimal et al. 2009 [39]	x	
Siegel et al. 2012 [46]	x	x
SMI 2011 [47]	x	x
Zainuddin/Gonzales 2011 [27]	x	x

Table 7.2: IT and cloud services literature covering aspects of quality analysis along the customer's activity cycle

that fit the individual requirements, starting from the list of criteria and continuing towards the individual process of determining the quality requirements and subsequently assessing them. The SMI not only provides a comprehensive list of characteristics covering all dimensions of quality (see the previous section) but also assists organizations in trying to assess the quality of cloud services before and after the service purchase.

However, other criteria lists such as the ones developed by Repschläger et al. [35-38] or the SaaS-Qual measure provided by Benlian et al. [28] for the post-purchase quality analysis of Software Services are more profound regarding specific quality aspects such as cloud service security criteria or perceptions of security and flexibility by service users within organizations. Despite being more specific, these approaches either fail to support all dimensions of service quality or focus on only one step in a customer's activity cycle. It is surprising that only one of the approaches reviewed, the SMI, actively guides and supports organizations in analyzing and assessing the quality of cloud services along the customer's activity cycle.

7.5 Contribution and Outlook

This literature review synthesized the existing research on approaches to quality analysis for cloud services by investigating 27 sources in detail and integrated their results to offer an overview of the existing body of knowledge. This overview of the literature reveals multiple quality analysis approaches focusing either on specific aspects of quality or on a specific step in the customer's activity cycle, pre-purchase and post-purchase of the service. The comprehensive list of quality characteristics combined, e.g., in the ISO/IEC 25010 norm [32], cover quality before and after the purchase, but ignore criteria covering the "image" dimension of service quality. In turn, Repschläger et al. [37,38] cover aspects of all three service quality dimensions, but focus merely on the pre-purchase phase. Only one approach, the SMI [47,46], could be identified that actively supports customers throughout all phases of the activity cycle and covers the three dimensions of service quality outlined by Grönroos [10]. However, because other approaches are much more specific concerning the characteristics of cloud services, the integration of those specialized approaches into the barrier-free, continuous analysis and assessment method developed by the SMI [47] seems valuable. Furthermore, empirical research is sparse in cloud computing research and literature [6]. Studies focusing on the requirements voiced by an organization and its employees concerning quality analysis could not be identified in the literature at all. To note and integrate the specific challenges organizations address when procuring cloud services, an empirical analysis of challenges ranging from factors of influence such as legal regulations and process design challenges such as whom to involve to challenges in determining the relevant quality requirements is required.

This review on quality analysis methods faces the usual limitations of a conceptual literature review aiming at a representative coverage of articles. However, we believe that the identified articles, the detailed and transparent documentation of the literature search process, the proposed approaches to the framework of analysis as well as the research agenda offer useful insights into this emerging field of research.

References

1. KPMG (2013) Cloud Monitor 2013. KPMG & BITKOM, Düsseldorf, Germany
2. Johnston WJ, Bonoma TV (1981) The Buying Center: Structure and Interaction Patterns. Journal of Marketing 45 (3):143-156
3. Osmonbekov T, Bello DC, Gilliland DI (2002) Adoption of electronic commerce tools in business procurement: enhanced buying center structure and processes. Journal of Business & Industrial Marketing 17 (2/3):151-166
4. Mell P, Grance T (2010) The NIST Definition of Cloud Computing. Communications of the ACM 53 (6):50-50
5. NIST (2011) The NIST Definition of Cloud Computing - Recommendations of the National Institute of Standards and Technology - Special Publication 800-145

6. Hoberg P, Wollersheim J, Krcmar H (2012) The Business Perspective on Cloud Computing - A Literature Review of Research on Cloud Computing. Paper presented at the 18th Americas Conference on Information Systems (AMCIS), Seattle, USA
7. Wollersheim J, Hoberg P, Krcmar H (2013) PROCUREMENT OF CLOUD SERVICES: SEVEN PRINCIPLES TO SUCCESS. Paper presented at the 13th International Research Symposium on Service Excellence in Management (QUIS), Karlstad, Sweden
8. Wollersheim J, Krcmar H (2013) Purchasing processes for cloud services - An exploratory study of process influencing factors. Paper presented at the 22nd Annual IPSERA Conference, Nantes, France
9. Wollersheim J, Konstantinidis C, Krcmar H (2012) Sicherheitskriterien bei der Auswahl von ERP-Systemen. ERP-Management 8 (3):50-52
10. Grönroos C (2007) Service Management and Marketing: Customer Management in Service Competition. 3 edn. John Wiley & Sons, Chichester, Great Britain
11. Grönroos C (2011) In The Marketplace There Is Only Service - Facilitating Customers' Value Creation. Paper presented at the 19th European Conference on Information Systems (ECIS), Helsinki, Sweden
12. Grönroos C (1984) A service quality model and its marketing implications. European journal of marketing 18 (4):36-44
13. Van Weele AJ (2005) Purchasing and Supply Chain Management: Analysis - Strategy - Planning and Practice. Thomson Learning, London, Great Britain
14. Heckman RL (1999) Managing the IT Procurement Process. Information Systems Management 16 (1):61-71
15. McQueen R, Teh R (2000) Insight into the Acquisition Process for Enterprise Resource Planning Software Derived from Four Case Studies. Paper presented at the Pacific Asia Conference on Information Systems (PACIS), Hong Kong, China
16. Verville J, Halingten A (2003) A six-stage model of the buying process for ERP software. Industrial Marketing Management 32 (7):585-594
17. Damsgaard J, Karlsbjerg J (2010) Seven principles for selecting software packages. Communications of the ACM 53 (8):63-71
18. Cooper HM (1988) Organizing knowledge syntheses: A taxonomy of literature reviews. Knowledge in Society 1 (1):104-126
19. vom Brocke J, Simons A, Niehaves B, Riemer K (2009) Reconstructing the Giant: On the Importance of Rigour in Documenting the Literature Search Process. Paper presented at the 17th European conference on Information Systems (ECIS), Verona, Italy
20. Torraco RJ (2005) Writing Integrative Literature Reviews: Guidelines and Examples. Human Resource Development Review 4 (3):356–367
21. Beimborn D, Miletzki T, Wenzel S (2011) Platform as a Service (PaaS). Wirtschaftsinformatik 53 (6):371-375
22. Clemons EK, Chen Y (2011) Making the Decision to Contract for Cloud Services: Managing the Risk of an Extreme Form of IT Outsourcing. Paper presented at the 44th Hawaii International Conference on System Sciences (HICSS), Waikoloa, USA
23. Hay B, Nance K, Bishop M (2011) Storm Clouds Rising: Security Challenges for IaaS Cloud Computing. Paper presented at the 44th Hawaii International Conference on System Sciences (HICSS), Waikoloa, USA
24. Jansen WA (2011) Cloud Hooks: Security and Privacy Issues in Cloud Computing. Paper presented at the 44th Hawaii International Conference on System Sciences (HICSS), Waikoloa, USA
25. Koehler P, Anandasivam A, Dan M (2010) Cloud Services from a Consumer Perspective. Paper presented at the 16th Americas Conference on Information Systems (AMCIS), Lima, Peru
26. Martens B, Teuteberg F (2011) Risk and compliance management for cloud computing services: Designing a reference model. Paper presented at the 17th Americas Conference on Information Systems (AMCIS), Detroit, USA

27. Zainuddin E, Gonzalez P (2011) Configurability, Maturity, and Value Co-creation in SaaS: An Exploratory Case Study. Paper presented at the International Conference on Information Systems (ICIS), Shanghai, China
28. Benlian A, Koufaris M, Hess T (2011) Service Quality in Software-as-a-Service: Developing the SaaS-QUAL Measure and Examining Its Role in Usage Continuance. Journal of Management Information Systems (JMIS) 28 (3):85–126
29. CSA (2009) Security Guidance for Critical Areas of Focus in Cloud Computing V2.1. Cloud Security Alliance - CSA, Seattle, USA
30. ENISA (2009) Cloud Computing - Benefits, risks and recommendations for information security. European Network and Information Security Agency
31. ENISA (2012) Procure Secure: A guide to monitoring of security service levels in cloud contracts. European Network and Information Security Agency
32. ISO/IEC 25010 (2011) Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE)
33. Martens B, Teuteberg F, Gräuler M (2011) Design and Implementation of a Community Platform for the Evaluation and Selection of Cloud Computing Services: A Market Analysis. Paper presented at the 19th European Conference on Information Systems (ECIS), Helsinki, Finland
34. Prodan R, Ostermann S (2009) A survey and taxonomy of infrastructure as a service and web hosting cloud providers. Paper presented at the 10th IEEE/ACM International Conference on Grid Computing, Banff, Canada
35. Repschlaeger J, Wind S, Zarnekow R, Turowski K (2011) Developing a cloud provider selection model. Paper presented at the Enterprise modelling and information systems architectures (EMISA), Hamburg, Germany
36. Repschlaeger J, Wind S, Zarnekow R Klassifikationsrahmen für die Anbieterauswahl in der Cloud. In: Jahrestagung der Gesellschaft für Informatik / Workshop: Neue Aspekte der zwischenbetrieblichen Integration durch Enterprise 2.0 (IOS 2.0), Berlin, Germany, 2011
37. Repschlaeger J, Wind S, Zarnekow R, Turowski K (2012) Selection Criteria for Software as a Service: An Explorative Analysis of Provider Requirements. Paper presented at the 18th Americas Conference on Information Systems (AMCIS), Seattle, USA
38. Repschlaeger J, Wind S, Zarnekow R, Turowski K (2013) Decision Model for Selecting a Cloud Provider: A Study of Service Model Decision Priorities. Paper presented at the 19th Americas Conference on Information Systems (AMCIS), Chicago, USA
39. Rimal BP, Choi E, Lumb I (2009) A Taxonomy and Survey of Cloud Computing Systems. Paper presented at the 5th International Joint Conference on INC, IMS and IDC, Seoul, South Korea
40. BITKOM (2010) Cloud Computing – Was Entscheider wissen müssen. BITKOM, Berlin, Germany
41. BSI (2012) Eckpunktepapier - Sicherheitsempfehlungen für Cloud Computing Anbieter - Mindestanforderungen in der Informationssicherheit. Bundesamtes für Sicherheit in der Informationstechnik - BSI, Bonn, Germany
42. CSA (2011) Cloud Controls Matrix. Cloud Security Alliance - CSA. https://cloudsecurityalliance.org/research/ccm/. 17. Sept. 2012
43. Eichler F (2010) Entwicklung eines Kriterienkatalogs zur Analyse von Qualitätsbewertungsmethoden für IT-Services. Otto-von-Guericke-Universität Magdeburg, Magdeburg, Germany
44. EuroCloud (2010) Leitfaden Cloud Computing - Recht, Datenschutz & Compliance. EuroCloud Deutschland_eco e.V., Köln, Germany
45. Gronau N (2012) Handbuch der ERP Auswahl. Gito, Berlin, Germany
46. Siegel J, Perdue J (2012) Cloud Services Measures for Global Use: The Service Measurement Index (SMI). Paper presented at the Service Research and Innovation Institute (SRII) Global Conference, San Jose, USA
47. SMI (2011) Service Measurement Index 1.0. vol 1.0. Carnegie Mellon University Silicon Valley, Moffet Field, USA

48. Parasuraman A, Zeithaml VA, Berry LL (1988) SERVQUAL: A Multiple-Item Scale for Measuring Consumer Perceptions of Service Quality. Journal of Retailing 64 (1):12-40
49. Kettinger WJ, Lee CC (2005) Zones Of Tolerance: Alternative Scales For Measuring Information Systems Service Quality. MIS Quarterly 29 (4):607-623
50. Vandermerwe S (1993) Jumping into the customer's activity cycle: A new role for customer services in the 1990s. The Columbia Journal of World Business 28 (2):46-65
51. Piccoli G, Spalding BR, Ives B (2001) A framework for improving customer service through information technology. Cornell Hotel and Restaurant Administration Quarterly 42 (3):38-45
52. Ives B, Willinger T (1999) To Dell or be Delled: A Leading Edge View of Electronic Commerce. ISRC Notes - September 1999. University of Houston, Houston, USA
53. Grönroos C (1983) Strategic management and marketing in the service sector. Marketing Science Institute, Cambridge, USA

Chapter 8
A Model-based Software Development Kit for the SensorCloud Platform

Lars Hermerschmidt, Antonio Navarro Perez, and Bernhard Rumpe

Abstract The development of software for the cloud is complicated by a tight entanglement of business logic and complex non-functional requirements. In search of a solution, we argue for the application of model-based software engineering to a particular class of cloud software, namely *interactive cloud software systems*. In particular, we outline an architecture-driven, model-based method that facilitates an agile, top-down development approach. At its core it employs a software architecture model that is augmented with additional aspect-specific models. These models serve as concise, formal, first-class specification documents and, thus, as foundation for tool-supported analysis and synthesis, in particular, code generation. We hypothesize that many crucial cloud-specific non-functional requirements can be satisfactory addressed on the architecture level such that their realization in the system can be synthesized by tools with small impact on the business logic.

8.1 Introduction

The number of software-driven embedded devices has been growing for decades. Such devices perceive and affect the real world, serve a multitude of diverse purposes, and are found in many public, industrial, and domestic places. They control vehicles, traffic infrastructure, factory machines, power plants, and energy grids. They measure electricity, temperatures, fill levels, global positioning, velocities, and operating conditions, or capture visual and audio information.

Enabled by the ongoing pervasiveness and sinking costs of internet connectivity, more and more embedded devices are no longer locally isolated but globally accessible and globally interconnected. Information and control shift from the individual device to the internet. Integrated, large-scale distributed systems emerge

Lars Hermerschmidt · Antonio Navarro Perez · Bernhard Rumpe
Department of Software Engineering, RWTH Aachen University, Aachen, Germany
e-mail: \{hermerschmidt,perez,rumpe\}@se-rwth.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_8

as a consequence. Their intimate relation to the real world and their network- and software-driven logic has led to the common label of *cyber-physical systems*. [17]

The software that drives those systems is of a *interactive/reactive* character: it reacts to impulses from its environment (e.g. sensor input or user interaction) with commands that control physical parts of the system (e.g. actors or monitoring systems). The software is also time-sensitive: its reactions have to happen within a given time window.

New use cases explore the potentials of cyber-physical systems, for instance, smart grids, smart traffic and smart homes. [13, 12, 11] The development of systems that implement these use cases, however, is complicated by their inherent complexity and subsequent development costs. Engineers of such systems must integrate system components and aspects on a technological level (e.g. devices, networking, protocols, infrastructure resources, software architecture), on an economic level (e.g. supported business models, integration with legacy systems and processes, customer and installation support), and from the perspective of legislators (e.g. privacy and customer autonomy). As a consequence of this tight integration of concerns, only big companies with big budgets can successfully implement such systems and bring them to product-ready matureness.

The *SensorCloud* project develops a large scale, cloud-based platform for services based around internet-connected sensor and actor devices. [1] The platform's technological purpose is to provide a generalized and easy-to-use infrastructure for sensor- and actor driven cloud services. Its economic purpose is to modularize and industrialize the development and provision of such services. Multiple different stakeholders can contribute different parts of the service stack, for instance, sensor/actor hardware, domain-specific data sets, domain-specific services, and client software.

From a software engineering perspective, the success of such a platform is largely influenced by the efficiency of development of software for the platform. In the context of the SensorCloud, third-parties develop *cloud services* that leverage sensors, sensor data, and actors in order to provide functionality for end customers. However, the development of cloud services is complicated by a tight entanglement of business logic and complex non-functional requirements.

In this paper, we describe the concepts and modeling languages at the core of a model-based SDK for developing cloud-based software in general and SensorCloud services in particular. This SDK is based on the *clArc toolkit*. It is particularly aligned to the domain of cloud-based cyber-physical systems and understands them more generally as *interactive cloud software systems*. At its core, this toolkit core employs a software architecture model as well as accompanying aspect-specific secondary models. These models serve as concise, formal, first-class specification documents and, thus, as foundation for tool-supported analysis and synthesis, in particular, code generation.

Chapter 8.2 describes the clArc toolkit underlying the SDK. Chapters 3 and 4 describe the architecture style and modeling language at the core of the toolkit. Chapters 5 and 6 describe additional modeling languages for deployment and testing.

Chapter 7 briefly outlines the tooling for processing and executing these modeling languages.

8.2 Model-based Engineering of Cloud Software

clArc (for *cl*oud *ar*chitecture) is a model-based, architecture-centric toolkit for developing *interactive cloud software* systems. It is based on the language workbench MontiCore [14, 15, 6] and the architecture description language MontiArc [10, 9]. The toolkit

1. defines a *software architecture style* for distributed, concurrent, scalable and robust cloud software that *permanently interacts* with a heterogeneous environment of physical things, services and users,
2. provides textual, executable *domain-specific modeling languages* for specifying and implementing software systems according to that architecture style
3. provides a *framework* for implementing individual code generators that synthesize an executable runtime framework for cloud software based on models written with these DSLs and targeted towards individual cloud infrastructures and platforms.

These three components are part of a methodology that seeks to make the development of interactive cloud software systems more reliable and efficient. It does so by (a) providing an appropriate level of abstraction for the specification of the software system and (b) by providing means to map the software's specification to an executable implementation without methodological discontinuities between design and implementation activities.

8.2.1 Interactive Cloud Software Systems

The software architecture style [25] of clArc describes architectures as cloud-based [3], interactive [24, 5, 18] software systems. Such systems are characterized by permanent interaction (a) on a system level between the software and its environment and (b) on the software level between the internal parts from which the software is composed. Interacting parts (software components) and their interactions (messages passed between components) are both first-level elements of the architecture style. The emphasis on interaction is a result of essential requirements imposed on those systems:

- *Reactiveness*: the software responds to events and requests from its environment within a given time window.
- *Statefulness*: the software continuously and indefinitely tracks and updates a conversational state between itself and its environment.

- *Concurrency*: the software interacts with many interaction partners in parallel and must, hence, be inherently *concurrent*.
- *Distribution*: the software is composed from distributed parts that are deployed on runtime nodes, communicate asynchronously, may replicate dynamically, support failure awareness and can be recovered from failures.
- *Scalability*: the software can adapt to changing work loads by increasing or decreasing the number of its parts and by consuming or releasing necessary computing resources.

These requirements are usually tightly entangled with each other and with the software's actual business functions. As a result, they hinder the software engineer in focusing on the software's business functionality that he actually wants to analyze, realize, test, extend, or modify.

The central idea behind the clArc toolkit is to address these requirements on an appropriate level of abstraction given by a description of the system's *software architecture* according to the architecture style of interactive systems.

8.2.2 Executable Modeling Languages

Modeling [20, 21, 4, 8] can be employed in the engineering of software systems to describe system aspects explicitly and formally. Domain-specific modeling languages are a tool to efficiently write concise models.

Appropriate high-level structure, distribution and interaction are the key factors when reasoning about requirements and overall system design. The explicit description of system aspects through models makes them visible, substantiated and documented in a coherent notation. Conversely, they remain implicit and invisible on the lower levels of system implementation.

However, software developers too often respond to models with skepticism. [7] In many projects, models are still created only at the beginning of the development process and thereafter abandoned. The software is subsequently developed detached from its models. Changes in requirements or design are only incorporated in the software's source code, but not in its models. As a consequence, the models play no essential part in the software's construction. Developers, thus, perceive models as causing additional work without essential contribution to the final product. Models are seen as documentation, at best.

The disconnect between models and product is grounded in process discontinuities caused by a lack of automated transitions between descriptions in models and the final product: models have to be manually transformed into an implementation and manually synchronized with a changing implementation.

The clArc toolkit incorporates the expressive power of models into the development process of cloud software and focuses on the elimination of process discontinuities. Models in clArc are automatically linked to a respective implementation and, thus, *executable*. A set of valid clArc models, therefore, can be compiled into an implementation through a *code generator*.

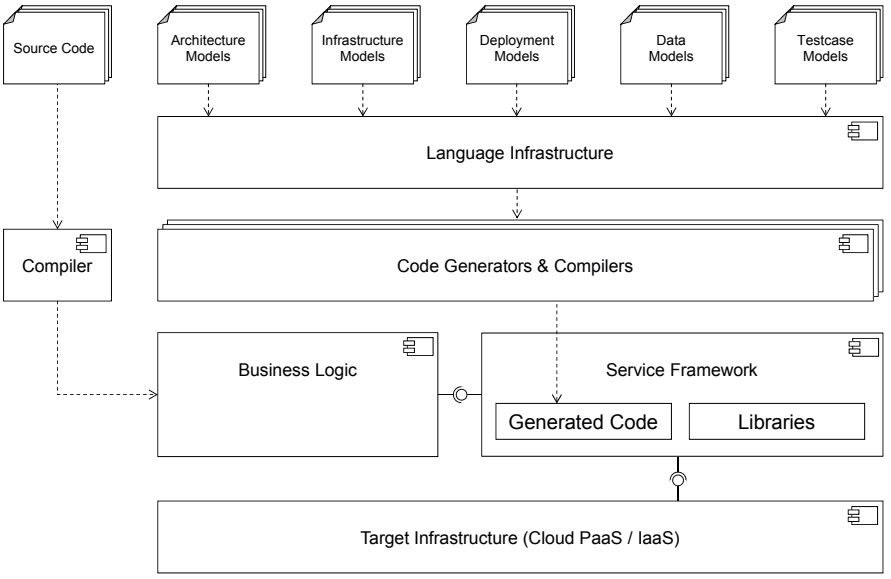


Fig. 8.1: Code Generation

More precisely, the system aspects described by models are realized in the form of a *generated framework*. This framework provides interfaces for *handwritten code* that implements the systems business logic. In this way, the business logic is cleanly separated from system aspects described by models. Models and code are both first-class artifacts that are integrated via well-defined interfaces. Contrasting other model-based methods, the generated code is treated the same way compiled machine code is treated in traditional programming: it is neither modified manually nor inspected.

Code generators are highly specific to the technological infrastructure the cloud software targets. Consequently, the clArc toolkit does not provide concrete code generators, but (a) a framework to efficiently develop individual code generators for specific cloud infrastructures and (b) a library of pre-developed standard libraries to be reused in concrete code generators.

8.3 Modeling Languages

The clArc toolkit employs several textual modeling languages.

The *Cloud Architecture Description Language* (clADL) is the central modeling language of clArc. Its models define *logical software architectures* that implement an *architecture style* targeting the domain of distributed, concurrent, scalable, and

robust cloud software. All other languages integrate with each other by referencing clADL models.

The *Target Description Language* (TDL) describes *physical infrastructure architectures* on which software is executed. The *Mapping Description Language* (MDL) relates clADL and TDL models to each other by defining deployments of software architectures onto infrastructure architectures. In combination, these models describe the overall *system architecture* that comprises the software architecture and infrastructure architecture.

In combination, the clADL, TDL and MDL can be used to configure a code generator to generate code according to a specific system architecture given by models of those languages. The generated part of the software architecture's implementation is then custom generated according to the the deployment given by TDL and MDL models.

The *Architecture Scenario Description Language* defines exemplary interaction patterns in a particular software architecture. Models of this language can be used to specify test cases. The *Test Suite Definition Language* configures test setups of such scenarios. Both languages are used for model-based testing of software architecture implementations.

8.3.1 Architecture Style

The architecture style of clArc uses *components* as the elemental building block of software architectures. Components in clArc are modules with well-defined import and export interfaces. They are executed in their own discrete thread of control and *interact* with each other through *asynchronous message passing* (with FIFO buffers at the receiver's end) over statically defined *message channels*. They encapsulate their state and do not share it with their environment by any other means except explicit message passing.

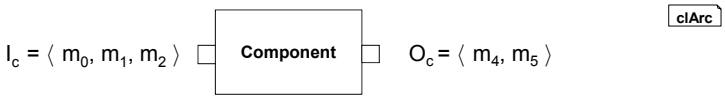


Fig. 8.2: A component receiving incoming messages from the channel I_C and sending outgoing messages to the channel I_C

Components are arranged in a decomposition hierarchy of components. In this hierarchy, leaf nodes represent the atomic building blocks of business logic while inner nodes compose, manage and supervise their immediate child components. Failures are propageted bottom-up through the hierarchy until they are handled or escalate at the root component.

This architecture style has several beneficial implications.

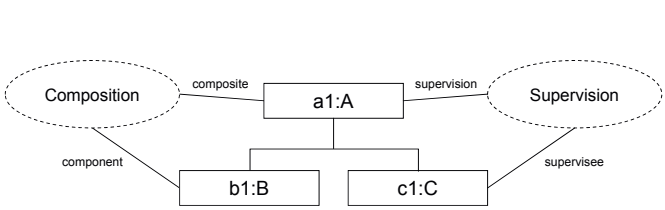


Fig. 8.3: A hierarchy of component runtime instances

- The execution semantics of the software do not depend on the actual physical distribution of components. Interactions are always and completely described by discrete, passed messages and always asynchronous in nature. Hence, components can be regarded as distributed by default.
- Component executions are self-contained. Every component has its own thread of control. Blocking operations or component failures do not influence other components directly. Failure recovery can be implemented within the local scope of the failed component.
- Components encapsulate a state and are, hence, in combination capable of representing the overall state of a continuously running system.
- Components do not need to have direct knowledge about the receivers their sent messages. They only need to know the message channels they are connected to. Channels, in turn, are implemented by a communication middleware that is independent from individual components. This middleware can deliver messages dynamically to different runtime instances of replicating receivers using different communication patterns (e.g. message queues) and technologies (e.g. different protocols and message formats). In this way, the middleware can implement several load balancing and scaling strategies and integrate heterogeneous infrastructures.
- Component implementations are technology-agnostic as their execution and interaction semantics are defined homogeneously and independent from concrete target technologies. Such components are easier to port between different infrastructures and easier to test and simulate.

This architecture style bears much resemblance to actor-based systems [2] but differs in some aspects. First, components may have not just one but many buffers for incoming messages. Second, these buffers are strictly typed. Third, communication channels are statically defined by the software architecture model and cannot be altered at runtime. Fourth, components do not control the instantiation of other components. Instead, the software architecture is statically defined by a given architecture model and automatically instantiated by the generated framework according to that model. Only the number of replicating runtime instances is dynamic.

8.4 Architecture Modeling

At the center of our language family is the cloud architecture description language (clADL). This language follows the *components and connectors* paradigm. It describes the structure of a software system in terms of system parts (components) and their mutual relationships (interfaces and connectors). [19] The clADL is derived from MontiArc, an architecture description language for distributed, interactive systems. [10] As MontiArc, the clADL is based on the semantics defined by the FOCUS method. [5] More precisely, our ADL is an extension of MontiArc that adds cloud software specific syntax and semantics.

The clADL describes cloud software architectures in terms of interacting *components*. A component is a distinct system part that implements a certain function. Components communicate with other components by exchanging *messages* as atomic units of information. Messages are exchanged via explicit *connectors* between components. Thereby, components and connections describe a network that represents the system architecture. In this network, components act autonomously and exchange messages asynchronously without an subordinately imposed control flow. This semantics applies to a *logical* point of view and does not enforce a particular *technical* realization.

Figure 8.4 shows the architecture of a simple cloud service in a graphical representation.

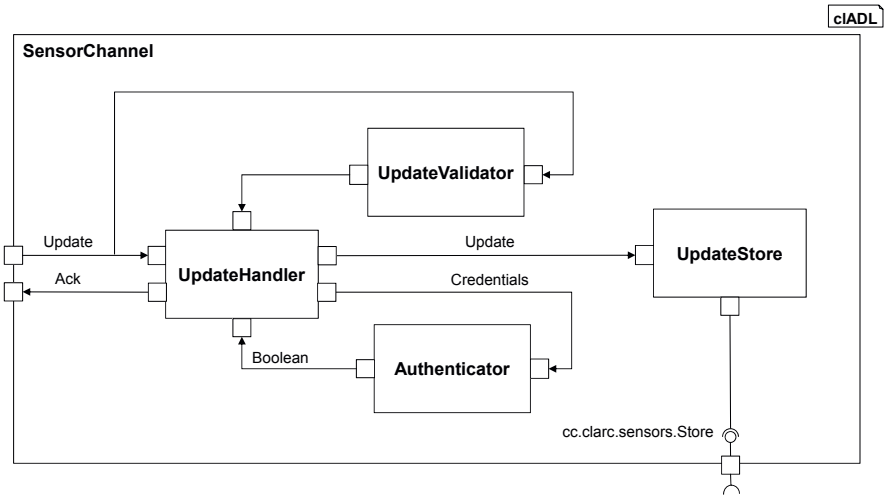


Fig. 8.4: The software architecture of a simple cloud service.

This service is modeled as a decomposed component named `SensorChannel`. It receives streams of sensor data represented by messages of type `Update` and acknowledges them by responding with `Ack` messages. The component is internally de-

composed into four subcomponents. The `UpdateHandler` receives all incoming `Update` messages from its `SensorChannel` parent component, interacts with the `Authenticator` and the `UpdateValidator` to analyze the update, sends valid updates to the `UpdateStore` and finally sends an acknowledgement to the `SensorChannel`'s respective outgoing port. The `Authenticator` checks whether the received Update has valid credentials. The `UpdateValidator` checks the received data for its validity. The `UpdateStore` uses a *service port* to write the update to a database provided by the service's underlying platform.

A component is syntactically defined by its name and its *interface*. Its interface is defined as a set of *ports*. Ports are the end points of *connections* between components and can either (as incoming ports) send or (as outgoing ports) receive messages. A component may be *decomposed* into further subcomponents and is, moreover, denoted as their parent component. Hence, components can again be understood as systems on their own. In fact, the system as a whole can be described as one single root composed component. Accordingly, we call components that are not decomposed into further subcomponents atomic components.

The semantics of a component defined by its externally observable (black box) *behavior* that is given by the relation between the sequence of received messages and the sequence of sent messages. The behavior of atomic components is given by a behavioral specification. It may be described in various ways, for instance, in declarative logic or functional/imperative programming languages. That specification includes the notion of an internal component *state* that changes depending on the sequence of incoming messages and implies the sequence of outgoing messages. The behavioral specification of composed components is inductively given by the aggregated behavioral specifications of its subcomponents and the topology of their connections.

Messages are syntactically defined by a name and a *type* that determines the kind of that specifies the kind of information they carry. Accordingly, ports reference a message type that determines the kind of messages they can communicate. Message types come with different internal syntactical structures that denote the basic structure of their carried information. They may be primitive types (e.g. a number or a string) or complex types that are composed of primitive types in a certain syntactic structure. Message types can be defined through external languages, for instance, Java and UML/P class diagrams. [22]

Connections syntactically connect exactly one outgoing port with one incoming port. Thus, connections haven an implicit direction in which messages can be communicated.

8.4.1 Replication

Subcomponents can be modeled as *replicating components*. Usually, the declaration of a component within the context of a parent component semantically implies the *fixed* existence of a *single* runtime instance of that component in the context of its

parent component. In contrast, the declaration of a replicating component implies a *variable* number of *multiple* instances. This notion allows us to describe quantitative system scalability in terms of dynamic replication of system parts. That means, the number of instances of replicating components may increase or decrease dependent on specific circumstances. Thereby, the system dynamically adapts to increasing or decreasing load demands.

By implication, replicating components may be dynamically created and destroyed. To represent this, every replicating component maintains a *lifecycle* state that corresponds to a lifecycle model. Basically, a components lifecycle determines if the component is idle and therefore a candidate for destruction or if it is busy and therefore protected from destruction.

8.4.2 Contexts

The semantics of channels where the receiver is a replicating component prototype are not in itself fully specified. The model does not define the mechanism that selects the replicating component's concrete runtime instance as the receiver of a given message. The semantics of channels only define the constraint that individual messages are only received by one receiver.

However, in many cases the concrete receiver of a message matters. A common real-world example are web systems that handle multiple user sessions. Runtime component states might be associated to such user sessions. Hence, interactions between such components should happen between those instances that have a state associated to that user session.

Contexts are a mechanism to resolve the ambiguities of receiver selection in such scenarios. A context is a type for *context tokens* and is declared in the scope of a component type. Context tokens are markers that can be assigned to components and messages.

Context tokens are assigned to messages that are sent through *context gates* of the respective context. Context gates can be defined on connectors and ports. Context gates can *open a context* by assigning a new token of that context to the message passing the gate. Furthermore, they can *close a context* by removing all tokens of that context from the message passing the gate. In addition, every outgoing port of a composite component implicitly closes all contexts defined in that component on messages passing this port.

The tokens assigned to messages that pass context gates serve as an identifier similar to session IDs in web systems. When messages with context tokens are received by a component, this component is also implicitly associated with these tokens.

8.4.3 Service Interfaces

Components may declare *service ports*. In contrast to other ports, service ports are not endpoints of message channels but represent *service interfaces* that provide *operations* that can be called on other software or by other software. Service ports can be required by a component (e.g. to call operations on the runtime infrastructure the component is being executed) or provided by a component, hence allowing other software running in the same runtime to call operations on the component.

8.5 Infrastructure and Deployment Modeling

Models of the clADL describe logical software architectures. They omit the details of its technical realization and physical distribution, that is, its complete *system architecture*. For a complete description of the actual system, additional information is necessary, in particular, information about the technical infrastructure it will run on (e.g. a Java EE Application Server), its technical configuration (e.g. the URL it will be bound to) and the concrete mapping of our software architecture onto this infrastructure. Alltogether, this information constitutes the *deployment* of an architecture implementation.

A deployment is described by *infrastructure models* and *mapping models*. Infrastructure models describe the technical infrastructure on which an archtecture implementation will run. For instance, an infrastructure can consist of application servers, and databases. Mapping models relate components in the software architecture to elements of the infrastructure architecture. For instance, a mapping model could specify that selected ports are accessible via a RESTful interface.

8.5.1 Target Description Language

The *Target Description Language* describes *physical infrastructure architectures* onto which software can be deployed for execution.

Targets are the essential building blocks of infrastructure architectures. Every target represents a discrete part of an infrastructure. Targets are called targets because components of the software architecture can be targeted at them for deployment.

Targets are defined by *target types*. Target types are defined in individual target models which in combination form the overall infrastructure architecture model. Target types can extend other target types. Thereby, they inherit all the structural and semantic properties of the other target type and enter a topological “is a” relationship with it.

The language defines a fixed set of target kinds.

- *Locations* represent physical locations (e.g. certain data centers or regions with particular legislations).
- *Resources* represent physical resources of information technology like computation and storage (e.g. hardware servers, virtual servers, storage systems).
- *Runtimes* represent containers for the execution of software (e.g. virtual machines, application servers, database management systems).
- *Artifacts* represent software documents or archives containing software documents (e.g. Java Archives, Web Archives, compiled binaries).
- *Modules* represent grouped software modules with shared properties (e.g. software parts executed in the same security sandbox).
- *Endpoints* represent resources for communication (e.g. web services, message queues).

Targets can contain subtargets which are target prototypes of a particular target type. Possible containments are constraint, for instance, a location can contain resources but a resource cannot contain locations. Subtargets can be declared as replicating subtargets and thus allow for multiple runtime instances of that subtarget.

Figure 8.5 shows an example of a hierarchy of nested target prototypes. In this example, a resource `Server` contains two other resources `VM_A` and `VM_B` (i.e. Virtual Machine). `VM_A` contains a runtime `ApplicationServer` which contains an artifact `WebArchive` which, again, contains two modules `Logic` and `UserManagement`. `VM_B` contains a runtime `MySQLServer` which contains an endpoint `MySQLAccess`. The `Server` is contained in an implicit location of the general target type `Location`.

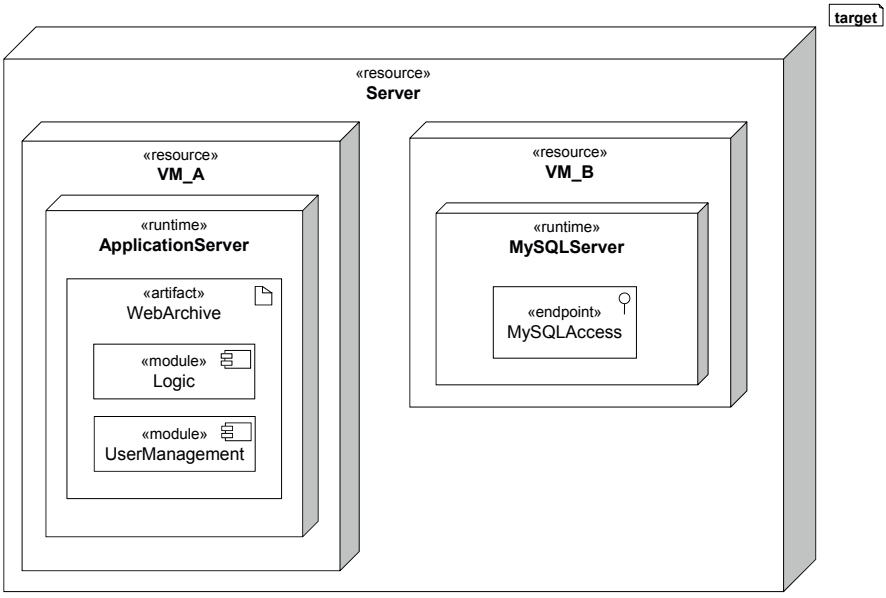


Fig. 8.5: An example for a hierarchy of targets

Target types may declare *target properties*. Target properties are variables that are assigned to string values in target prototypes. They can, for instance, describe TCP port numbers of services or identify the operating system of a server. Properties can be assigned to values in subtarget declarations.

8.5.2 Mapping Description Language

The *Mapping Description Language* defines *deployments* of a software architecture onto an infrastructure architecture.

Mappings are collections of concrete mappings between components and targets. Hence, a mapping relates a model of a logical software architecture defined by clADL models to a model of a physical infrastructure model defined by TDL models. Mapping declarations map a referenced component as well as all otherwise unmapped subcomponents of the hierarchy it defines to the referenced target.

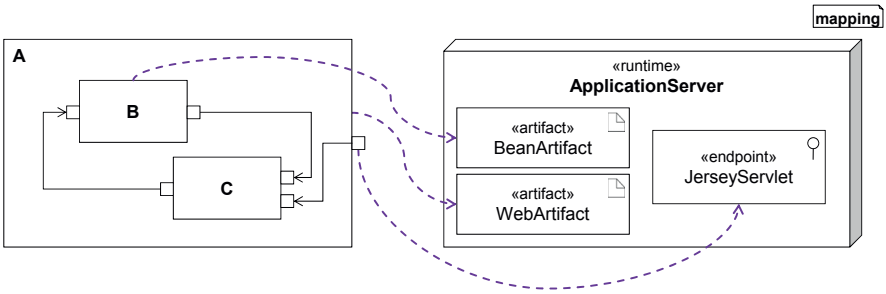


Fig. 8.6: A mapping model with two mappings

Figure 8.6 shows an example of a software architecture with a component `A` being decomposed into subcomponents `B` and `C` put next to an infrastructure architecture consisting of a runtime `ApplicationServer`, two artifacts `BeanArtifact` and `WebArtifact` and an endpoint `JerseyServlet`.

8.6 Model-based Testing

The *Architecture Scenario Description Language* describes exemplary interaction scenarios in concrete software architectures. The *Architecture Test Suite Definition Language* defines test setups of software architectures and corresponding scenarios.

A scenario describes a *valid*, chronologically arranged, partially ordered set of interactions in a software architecture. Interactions are described as messages passed

between component prototypes. Scenario descriptions bear resemblance to *Message Sequence Charts* [16]. Thus, scenarios are partial *protocol definitions* [23]

Figure 8.7 shows an interaction between the subcomponents of the `SensorChannel` component introduced in figure 8.4. This representation is similar to UML sequence diagrams. Components are depicted with ports and timelines while interactions are depicted as arrows between these timelines.

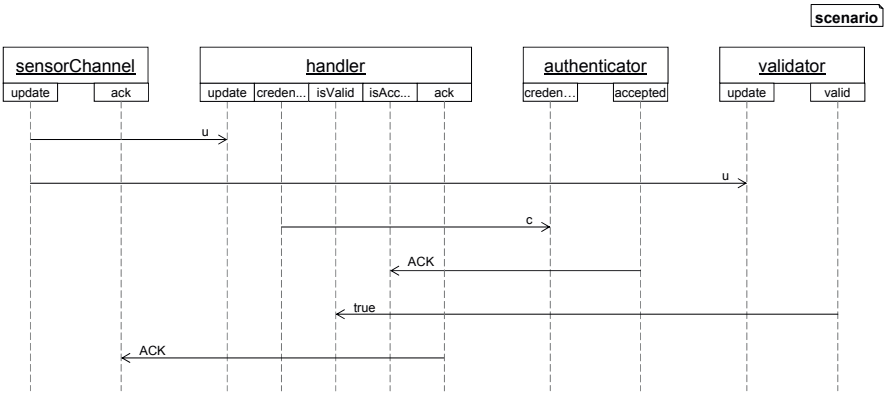


Fig. 8.7: A Scenario of Interactions inside SensorChannel

Scenarios can serve as specifications for model-based tests. From a scenario model, the expected behavior of each participating component can be derived. This behavior can be used (a) as a reference against which components under test can be evaluated and (b) as a basis for mocking components with whom components under test interact.

8.7 Language Execution

The model-based SDK for the SensorCloud consists of several tools that combine source code of traditional programming languages and models of the clArc language family.

At the core is a *modular code generator* that is composed of many generator modules, each being responsible for different aspects of the generated code (e.g. protocols, resource management). This code generator generates a deployment-specific service framework based on clADL, TDL and MDL models. This framework is customized with handwritten code and integrates with the SensorCloud's platform APIs. The result is a complete implementation of a SensorCloud service. The use of models lifts the abstraction of this implementation and makes it agnostic of the actual SensorCloud's API. Platform-specific code is strictly left to the code genera-

tor. In this way, the platform can evolve without affecting the implementation of its services.

In addition, a test-specific code generator can generate a functionally equivalent variant of a service's implementation that can be executed locally. In this way, the service can be functionally tested without the need for a testing infrastructure. Concrete test cases can be generated from scenario models.

8.8 Conclusion

In this paper we described a model-based SDK based on the clArc toolkit for developing cloud services for the SensorCloud platform. The SDK is in ongoing development and will enter its evaluation phase in the third and final year of the SensorCloud's research and development schedule.

References

1. SensorCloud. URL http://www.sensorcloud.de/
2. Agha, G.: Actors: A Model of Concurrent Computation in Distributed Systems. Dissertation, Massachusetts Institute of Technology (1986)
3. Armbrust, M., Stoica, I., Zaharia, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A.: A View of Cloud Computing. Communications of the ACM **53**(4), 50 (2010)
4. Booch, G., Rumbaugh, J., Jacobson, I.: The Unified Modeling Language User Guide, 2nd edn. Addison-Wesley Professional (2005)
5. Broy, M., Stø len, K.: Specification and Development of Interactive Systems. Focus on Streams, Interfaces and Refinement. Springer Verlag Heidelberg (2001)
6. Department of Software Engineering at RWTH Aachen University: MontiCore. URL http://www.monticore.de/
7. Engels, G., Whittle, J.: Ten years of software and systems modeling. Software & Systems Modeling **11**(4), 467–470 (2012). DOI 10.1007/s10270-012-0280-x. URL http://dblp.uni-trier.de/db/journals/sosym/sosym11.html\#EngelsW12
8. Friedenthal, S., Moore, A., Steiner, R.: A Practical Guide to SysML: Systems Modeling Language (2008). URL http://dl.acm.org/citation.cfm?id=1477660
9. Haber, A., Ringert, J.O., Rumpe, B.: Towards Architectural Programming of Embedded Systems. In: Tagungsband des Dagstuhl-Workshop MBEES: Modellbasierte Entwicklung eingebetteter Systeme VI (2010). URL http://www.se-rwth.de/publications/HRR10.pdf
10. Haber, A., Ringert, J.O., Rumpe, B.: MontiArc - Architectural Modeling of Interactive Distributed and Cyber-Physical Systems. Tech. rep., RWTH Aachen University, Aachen (2012)
11. Haller, P., Odersky, M.: Scala Actors: Unifying thread-based and event-based programming. Theoretical Computer Science **410**(2-3), 202–220 (2009). DOI 10.1016/j.tcs.2008.09.019. URL http://dblp.uni-trier.de/db/journals/tcs/tcs410.html\#HallerO09
12. Harper, R. (ed.): The Connected Home: The Future of Domestic Life. Springer London, London (2011). DOI 10.1007/978-0-85729-476-0. URL http://link.springer.com/chapter/10.1007/978-0-85729-476-0_1/fulltext.html

13. Iwai, A., Aoyama, M.: Automotive Cloud Service Systems Based on Service-Oriented Architecture and Its Evaluation. In: 2011 IEEE 4th International Conference on Cloud Computing, pp. 638–645. IEEE (2011). DOI 10.1109/CLOUD.2011.119. URL http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6008765
14. Krahn, H.: MontiCore: Agile Entwicklung von domänenspezifischen Sprachen im Software-Engineering. Dissertation, RWTH Aachen University (2010)
15. Krahn, H., Rumpe, B., Völkel, S.: MontiCore: A Framework for Compositional Development of Domain Specific Languages. International Journal on Software Tools for Technology Transfer **12**(5), 353–372 (2010). DOI 10.1007/s10009-010-0142-1. URL http://dblp.uni-trier.de/db/journals/sttt/sttt12.html\#KrahnRV10
16. Krüger, I.: Distributed System Design with Message Sequence Charts. Ph.D. thesis, Technische Universität München (2000)
17. Lee, E.A.: Cyber-Physical Systems - Are Computing Foundations Adequate? October (2006)
18. Manna, Z., Pnueli, A.: The Temporal Logic of Reactive and Concurrent Systems - Specification. Springer (1992)
19. Medvidovic, N., Taylor, R.N.R.: A Classification and Comparison Framework for Software Architecture Description Languages. IEEE Transactions on Software Engineering **26**(1), 70–93 (2000). DOI 10.1109/32.825767. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=825767
20. Rumpe, B.: Modellierung mit UML: Sprache, Konzepte und Methodik. Xpert.press. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
21. Rumpe, B.: Agile Modellierung mit UML: Codegenerierung, Testfälle, Refactoring. Xpert.press. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
22. Schindler, M.: Eine Werkzeuginfrastruktur zur Agilen Entwicklung mit der UML/P. Dissertation, RWTH Aachen University (2011)
23. Selic, B.: Protocols and Ports: Reusable Inter-Object Behavior Patterns. In: ISORC, pp. 332–339 (1999)
24. Selic, B., Gullekson, G., Ward, P.T.: Real-Time Object-Oriented Modeling. Wiley professional computing. Wiley (1994)
25. Taylor, R.N., Medvidovic, N., Dashofy, E.M.: Software Architecture: Foundations, Theory, and Practice (2009)

Chapter 9
TRESOR – Towards the Realization of a Trusted Cloud Ecosystem

Sebastian Zickau, Mathias Slawik, Dirk Thatmann, Sebastian Uhlig, Iwailo Denisow, and Axel Küpper

Abstract The TRESOR[1] project enables cloud computing solutions for the German health sector. This sector deals with sensitive medical information and is in general not suitable for current cloud-based solutions, which are lacking appropriate privacy and security features. The project evaluates and proposes new architectural components to address these shortcomings. These will be combined into a secure and trustworthy ecosystem that will enable the health industry and other sectors to take advantage of cloud computing. The architecture consists of components, such as a marketplace, a broker, a proxy and a PaaS-platform. TRESOR addresses privacy and data protection issues and aims at providing a standardized solution with reduced lock-in effects that can also be used in other domains. In this paper the specific tasks and the architecture of these components are presented, important challenges of the TRESOR project are highlighted and preliminary results, such as a secure transfer protocol, and policy integration are shown.

Key words: Cloud computing, broker, marketplace, proxy, description language, policies, location information, access control

Sebastian Zickau · Mathias Slawik · Dirk Thatmann · Sebastian Uhlig · Iwailo Denisow · Axel Küpper
Service-centric Networking, Technische Universität Berlin, Germany,
http://www.snet.tu-berlin.de/
e-mail: {sebastian.zickau,mathias.slawik,d.thatmann,axel.kuepper}@tu-berlin.de,{sebastian.uhlig,iwailo.denisow}@campus.tu-berlin.de

[1] **T**rusted **E**cosystem for **S**tandardized and **O**pen cloud-based **R**esources
http://www.cloud-tresor.de/

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_9

9.1 Introduction

The health sector is with more than 4 million members one of the largest employment sectors in Germany [11]. The technological progress demands new investments in the ICT departments of hospitals, but human and financial resources are limited. Not all medical institutions have updated their system to the latest technologies, which makes it an opportune moment to migrate to the new paradigm of cloud computing. The introduction of new solutions is one possibility for hospitals to reduce expenditures. With fast and connected systems the medical staff can benefit from new information technologies. There are also many obstacles with the usage of cloud computing.

The shortcomings range from vendor lock-ins and isolated solutions to the lack of management capabilities, the need for an extended service level agreement (SLA) monitoring, data protection and law-abiding systems. Service providers are usually interested that customers are only using their products by forcing them to use specific solutions (*lock-in effect*). By using remote services, which are managed by 3rd parties the communication and chain of management responsibilities are getting more complex (*lack of management capabilities*). The usual SLA monitoring (availability, scalability and performance) is too restricted when it comes to sensitive information and complex systems (*extended SLA monitoring*). The need for *data protection and law-abiding systems* is self-evident while treating patients and dealing with personal information. The medical staff also needs the assurance that they act within the boundaries of the law and are protected by it when new technologies are introduced.

These issues and challenges will be addressed in the TRESOR project. To reach these goals a cloud computing ecosystem will be developed by enriching current technological solutions with trusted features. Professional medical employees will evaluate the project results in two example scenarios. On the one hand the medical TRESOR partners[2] expect that the patients' treatment can be improved and hospitalization times can be reduced with the ecosystem. On the other hand the medical institutions hope that by the end of the project they have a better and easier data exchange among their overall IT infrastructure in place achieved by using common protocols and an enforced standardization[3].

In the next section the broker and marketplace are described in more detail, followed by a section dedicated to the proxy and its functionality. Section 9.4 summarizes the policy handling within TRESOR. Ideas regarding the location management of TRESOR users and devices are shown in Section 9.5. In Section 9.6 results of a location-based access control prototype are given. The paper concludes with a summary and an outlook.

[2] Deutsches Herzzentrum Berlin and Paulinen Krankenhaus

[3] Collected through interviews with the medical partners:
http://www.cloud-tresor.de/2012/05/14/experteninterview/

9.2 TRESOR Broker & Marketplace

The TRESOR broker and marketplace are two coupled components: while the back end TRESOR broker manages formalized service descriptions and customer requirements, the front end TRESOR marketplace provides means for the end user to interact with the broker, e.g., faceted searching, authoring of descriptions, and booking services. Additional functionalities of the broker are: recommending supplementary services, notifying users of new services meeting their requirements, and verifying service compositions. We envision a federation of different brokers implementing the same RESTful interfaces in the future. These brokers could, for example, specialize on certain industry sectors or service functionalities and enable new stakeholders to benefit from the brokering functionality.

The TRESOR service description language is used for formalizing service profiles and requirements. It contains elements representing empirically identified cloud requirements which consumers consider important for their service selection. These requirements originate from works such as the Cloud Requirement Framework [16]. Some examples of these requirements are: supported interfaces and standards, data portability, set-up and provisioning time, price components and contract terms, provider certifications and support options. These descriptions will be augmented by domain specific requirements collected from the health institutions, narrative service descriptions, and SLA statistics gathered by the TRESOR proxy. Our current description language is built on the Eclipse Modeling Framework (EMF) [19], as it enables definition of a rich data model without relying on concrete serialization formats and is surrounded by an ecosystem of diverse software tools. The EMF realizes linking different datasets through URLs and supports reflecting over the model of data.

The focus of our formalization on the consumer perspective contributes to lowering the time spent creating it, raising the impact of the service comparison functionality, and increasing the transparency of the cloud offers. Furthermore, it expresses typical requirements of service consumers, especially if requirements are combined in domain specific profiles. This should help prospective service providers to tailor their services to the explicit needs of service consumers, supporting a broader adaptation of useful cloud services.

9.3 TRESOR Proxy

The TRESOR proxy is a reverse HTTP proxy for TRESOR SaaS offers, which is augmented by a number of functionalities for managing cloud service consumption. The main purpose of the TRESOR proxy can be subsumed under the term ”regaining management capabilities”. The strict requirements for such capabilities, most often stemming from domains handling sensitive data, e.g., the health and finance sectors, are not fulfilled by current cloud computing ecosystems. In our previous work [18],

we have identified the main areas of unfulfilled requirements. The following sections outline some of the proposed concepts to address these issues:

Trusted Cloud Transfer Protocol.

Contemporary management proxies such as the TRESOR proxy carry out functions, such as load balancing, SLA monitoring, logging, intrusion prevention, or billing. These proxies act as TLS server connection ends and are therefore able to obtain access to the plaintext of HTTPS communication. Under German law, for example, such a disclosure of data would be a criminal act punishable with imprisonment for up to one year or a fine if the data contains medical information [8]. Within TRESOR, we have developed a data privacy preserving cloud service access method, the Trusted Cloud Transfer Protocol (TCTP) to address this issue. The method is described in [18]. TCTP builds upon TLS-functionalities to realize a HTTP-proxy independent end-to-end security mechanism, which enforces a secure transfer of body data between a user-agent and a cloud computing service. The HTTP-independence of TCTP enables this secure mechanism for existing (cloud) environments where RESTful services are in place. As the body information of the service data is encrypted and secured for any intermediate proxy, the header information can be used in an intermediary architecture to add monitoring services, such as compliance auditing and service logging. TCTP is currently being implemented and extended [17].

Central audit log.

Keeping records for a minimum duration is required by legal provisions, e.g., medical records for 10 years [5], or in some cases for up to 30 years [7]. In the majority of cases a log of service activities leading to a modification of such records also has to be kept as long. Meeting this requirement using SaaS solutions poses two main challenges: the inability to access service logs after the service contract duration, and the diversity of service-specific logging mechanisms. TRESOR will provide a central logging service whose contract terms are not bound to the use of any service. There are two sources of log data: HTTP requests for specific resources (e.g., patient data) captured by the proxy and service-specific information received by a consistent logging interface out-of-band.

Independent SLA monitoring.

A substantial boost of transparency is the ability of the TRESOR cloud proxy to assess SLAs of cloud services as an independent 3rd party. Among other things we plan to measure the availability, average response time, and quantity of server errors. These values will be presented to prospective users on the TRESOR marketplace enabling them to assess the service quality beforehand.

Identity Management out of the cloud.

Integrating identity management (IDM) systems of cloud consumers into current cloud services poses some challenges. As the TRESOR ecosystem relies on a central IDM system, these integration efforts are reduced: First, all TRESOR cloud services rely on identities managed by the cloud IDM system - without service-specific adapters. Secondly, customer IDM solutions, such as Microsoft Active Directory [12] or Kerberos [13] can be integrated into the cloud IDM system to be utilized by all TRESOR services.

Distributed authorization.

Another challenge is the enforcement of equal access policies within heterogeneous environments, such as state-of-the-art service landscapes built upon diverse SaaS offerings. The requirement of having those access policies is often mandated by domain specific regulations. Within the health sector for example there are essential rules governing patient data access by physicians, which have to be enforced on a multitude of services containing such data. As most distributed applications define their own set of authorization facilities, the manual maintenance of such rules is at least cumbersome. Distributed authentication technologies, such as the eXtensible Access Control Markup Language (XACML) [14], will be employed by TRESOR to enable all services to rely on the same policies authored by cloud consumers. As the TRESOR proxy carries out routing functionality, it can be used to enforce access control policies for RESTful [6] architectures, acting upon URLs and HTTP methods.

Open standards and APIs.

As the ecosystem introduces TRESOR-specific components, lock-in effects cannot be avoided completely. Nevertheless the reliance on open standards and APIs by these components lowers the most frequently observed service-specific lock-in effects considerably.

9.4 Cloud Ecosystems and Policies

Since TRESOR is an ecosystem supporting trading and consuming SaaS-Services, access control is an important aspect especially when consuming fee-based services. Driven by the need to be compliant with legal regulations, general requirements arise, such as the support of fine-grained audit-logs and controlling read and write access to resources. In addition, obligations, such as audit-proof logging systems must be carried out. We identified XACML [14] as a suitable candidate for handling

access policies in TRESOR. XACML represents a distributed approach for authorization. XACML introduces a policy description language, which allows defining a fine-grained access control. The decision is made within an architecture of distributed components, containing a Policy Decision Point (PDP), a Policy Information Point (PIP) and a Policy Enforcement Point (PEP). A Policy Administration Point (PAP) enables administrators to define authorization policies.

Policy Administration Point.

The Policy Administration Point (PAP) will allow TRESOR customers to create access rules for their employees based on SaaS specific templates provided by the SaaS provider. Since the XACML policy language can express privacy as well as security polices in a machine-readable format, the PAP will act as an abstraction layer in order to support policy creation in a simple manner. A sample for this is Axiomatics Language for Authorization (ALFA) [4]. In ALFA administrators can define XACML policies while using a syntax similar to JAVA. Compared to other PAP tools, such as the PAP provided by WSO2 Identity Server [2] or UMU-XACML-Editor [1], ALFA enables policy administrators to define complex rules in a more sophisticated way while reducing the loss of XACML's rich expressiveness. Inspired by ALFA a future goal of TRESOR is to allow policy administrators to create policies similar to ALFA. In addition auto complete and/or short-cuts will enable the policy administrator to link user or group IDs, provided by customer's local user databases, dynamically within a policy. This requires integration efforts including a connection from the PAP to the customer's user databases. A goal of the PAP is to give non XACML experts the ability to design and maintain XACML policies.

XACML policy templates and delegation.

Another challenge within TRESOR is to enable service providers to create policy templates for their services that can be used by customers for linking specific employees to particular service functions. Each service will expose its methods and functionalities. The SaaS provider will create a service specific template for each of his offered services, which is then handed over to the SaaS customer for defining user specific access policies. TRESOR will implement OASIS's XACML administration and delegation profile [10] in order to enable dynamic delegation. A sample scenario is "While doctor Bob currently is in the role 'doctor in duty', Bob gets read and write access to patient data of Alice" dynamically.

TRESOR XACML architecture.

The XACML 3.0 core specification [14] introduces a reference data-flow model. Actors, such as service customers, service providers, policy administrators and the

distributed components required for a decision (Policy Enforcement Point (PEP), Policy Decision Point (PDP), Policy Retrieval Point (PRP)) and a basic request/response format are introduced. The proposed data flow model must be merged and adapted with our architecture. Since the distribution of XACML components across the Cloud Ecosystem has significant impacts on security, privacy, legal aspects, overall performance and policy management, a challenge of TRESOR is to identify the most applicable component distribution. A high level architecture is depicted in Figure 9.1.

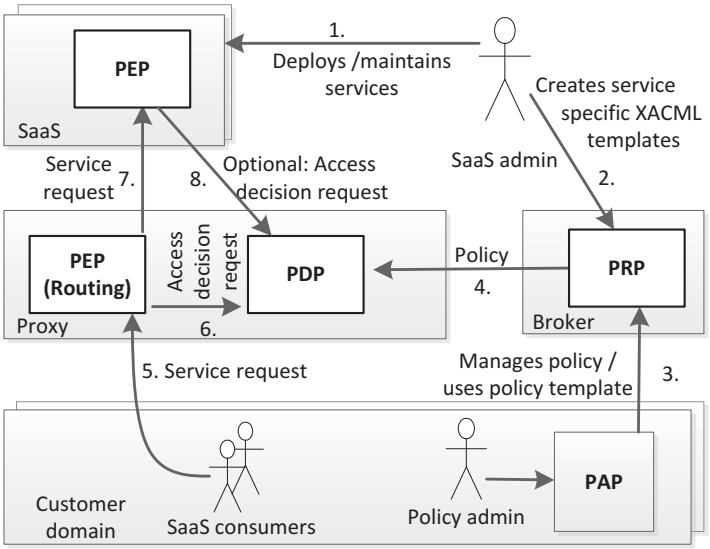


Fig. 9.1: High-Level XACML architecture in TRESOR.

Figure 9.1 is described as follows: first a SaaS must be available, thus be deployed by a provider. A SaaS specific policy template, containing information of the RESTful API, is handed over to a PRP (2) for further usage by the customer. In particular the policy administrator of the customer takes over the access rule adaptation (3). Having a business relation with a SaaS provider, the administrator of a customer creates and maintains access policies based on the provided templates, in order to allow or deny specific employees the use of the SaaS. The final policy is retrieved by the PDP (4). Whenever a SaaS consumer tries to access a resource using HTTP, the PEP within the Proxy intercepts the requests (5). The PDP will be asked for a decision. The HTTP header, the URI and the SAML token found the basis for the decision making process (6). If the PDP permits the access, the request is forwarded to the particular provider (SaaS) (7). Since the Proxy does not have access to the HTTP body and is just responsible for a preliminary routing decision, as described in Chapter 9.3, the SaaS may contain another PEP which can ask the

PDP for further authorization decisions. This, however this request depends on the concrete use-case and is optional (8).

9.5 Location Management

In recent years mobile devices were also introduced in domains, such as the health care sector. Doctors and other medical staff are using smartphones and tablets as well as laptops during their working hours. In the TRESOR scenarios the notion of location is also represented in the access of medical services. Simply speaking there is a scenario in which the doctor is located on the hospital premises and therefore allowed to read and change patients' data. In another situation the doctor operates from home and is only allowed to read a patient's record. These are just two simple examples how location information can be used in the TRESOR context. This idea is known under the term of LBAC (location-based access control) [3]. But location information within TRESOR is not limited to only control access. Enriched services, which use positioning of devices and users will also be in focus within the project.

The approach within TRESOR will implement two components, a location client and a location server. The location client is situated in the user agent, e.g., a browser or mobile operating system, such as Android OS. The location client communicates with the location server, which is located in the TRESOR proxy. The server receives various location information from the client and can use this information to influence the service consumption of the user on its user agent. The client will gather information, such as longitude and latitude, WLAN access points (via basic service set IDs, BSSIDs), Cell-IDs and also less common ones, e.g., Bluetooth, RFID or NFC tags located at known locations on the hospital grounds. The next section details the envisioned architecture and gives information about used standards. The possibility of location spoofing is given in the context of using the information for security purposes. The presented work does not provide a solution for this weakness and refers to the fact that location authenticity on mobile devices is a research topic on its own.

GeoXACML.

An extension to the presented XACML language is the GeoXACML standard version 1.0.1 [15], which adds geo-information to the policy framework. Currently, services use the geo-information represented by polygons within digital maps for securing geo-data. In TRESOR GeoXACML will mainly be used to enable systems to deny or grant access to different services provided by different service operators based on the customers' physical geo-position. The already mentioned location server, fed with data by the location client is representing a Policy Information Point (PIP), which deals with the evaluation of location data and provides this information to the Policy Decision Point (PDP). A challenge for TRESOR is to make the admin-

istration and management process within a Policy Administration Point (PAP) as simple as possible for policy creators. GeoXACML has still not been adapted to the current XACML version 3 standard.

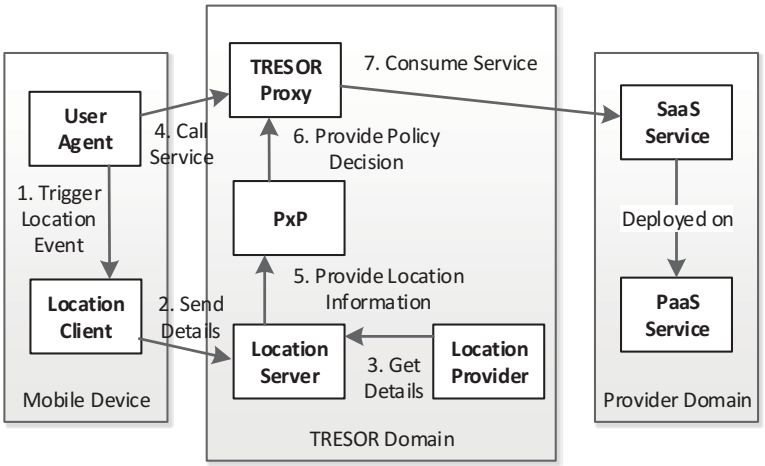


Fig. 9.2: High-Level Location Management Architecture in TRESOR

Location Client and Server.

As mentioned in the previous section and shown in Figure 9.2, a user interacts with a service via a user agent on a device. The user agents and devices can be manifold, e.g., mobile phones, tablets, laptop or stationary devices as well as browsers or native applications. As these devices have different location sensors the location client has to be very generic. The information gathered by this client will be the basis for the location information, which is processed by the location server (steps 1 and 2). As the location information provided can vary the location server may request additional information, such as address data or specific longitude and latitude values, from a location provider (step 3). As mentioned before all services will be consumed via the TRESOR proxy (step 4). The policy framework will provide the necessary information using its different policy points (PxP in steps 5 and 6) to provide an access decision to consume a service (step 7) on the PaaS-platform of the provider. It is also possible to combine certain location and/or user agent or client information to a broader meaning. As an example you can use GPS coordinates as well as a Bluetooth connection between a mobile device and a stationary sensor to pinpoint a more accurate location. Combining this data with the user and role iden-

tity information the PDP has a strong influence on how access to services will be enforced.

Stationary Devices.

Doctors with mobile devices are moving around while treating patients in their daily duty in hospitals. Also in the context of medical care various devices are used to record and compute patients' data. For a medical record it is useful to know where health records and diagnoses were made. With additional user information stored on doctors' mobile devices these records could be filled with additional information. As part of the TRESOR project a system has been developed to link mobile devices with stationary computers to provide accurate location information for services running on the static device [9]. This information can be used to tag medical records or identify the current location of the medical staff.

9.6 Preliminary Implementation Results

This chapter presents preliminary implementation results, which were collected during the realization of a location-based access control prototype in the TRESOR context. As shown in Figure 9.3 the prototype consists of five domains, the user domain, the service provider domain, the identity provider domain, the authorization domain and the location provider domain. Each domain consists of one or more (GeoX-ACML) components, which are described in the following subsection.

Prototype Architecture.

The basis for this scenario is a simplified Ruby on Rails (RoR) application that simulates the TRESOR example applications. The RoR application is used to exemplify the TRESOR use cases. The application allows different users, which are separated by their userIDs, such as *doctor*, *nurse* or *admin* to access medical data or can use a service to determine if a newly prescribed medications would be influenced by and previous medical conditions. The database for these applications is filled with example data. The notion of location-based access control (LBAC) was added subsequently.
For interoperability reasons, a Policy Enforcement Point (PEP) for Ruby on Rails is inserted into the RoR application and checks the permission of the request against a policy architecture (see Figure 9.3). This PEP is implemented on the basis of RoR for web services and proves the concept of integrating it into an existing TRESOR application.

As seen in Figure 9.3 the web service requester needs an identity token. The implementation of an identity provider is beyond the scope of the initial prototype

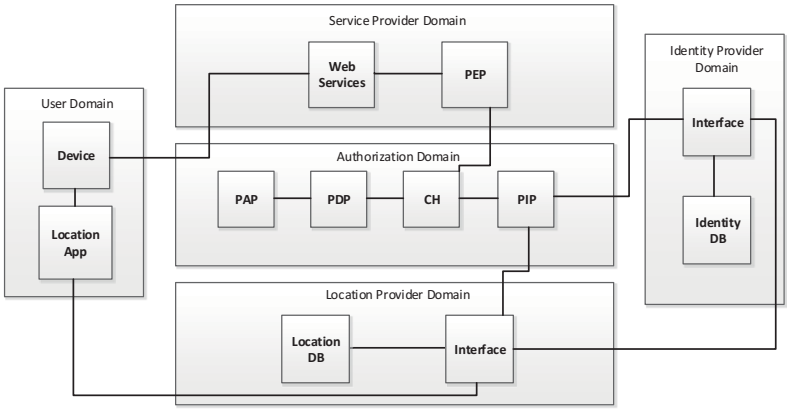


Fig. 9.3: LBAC Prototype Architecture

development. In practice there are many existing tools and platforms that can be easily deployed, such as SimpleSAMLphp[4], WSO2 Identity Server[5], external identity providers like Google or Facebook or other OpenID[6] providers.

9.6.1 User Domain

The LocationAPP on an Android client device (Smartphone), implemented using Java, Jersey and JAX-RS, collects all geolocation information from the local device and transmits it to the location provider. The transfer is only allowed after a successful login to an identity provider. The LocationAPP allows the selection of geolocation information, to give the user the opportunity to decide which data is transmitted. The geolocation services, such as WLAN or GPS can be enabled or disabled individually. Options within the Android app:

- Debug Mode: mode to fake the geolocation for testing purposes
- Manual Push: mode to push geolocation data immediately, which is useful for debugging

The following parameters are configurable:

- The Username (default: userReal)
- The REST URL of the location provider

[4] http://simplesamlphp.org/

[5] http://wso2.com/products/identity-server/

[6] http://openid.net/

The LocationAPP was built from scratch using only the Android SDK and GSON[7]. GSON is a Java library that can be used to convert Java Objects into their JSON (JavaScript Object Notation) representation. The application will push the position to the location provider every 300 seconds (default value). It needs to be discussed if a live system implemented in an actual medical environment can be based on an AndroidOS device, which hosts additional applications by Google or 3rd parties not certified for the usage within a healthcare domain. A dedicated healthcare version of the AndroidOS could be a possible solution.

A problem while using the LocationAPP is to determine if some other application is able to manipulate the recent position. It is possible to check if the location mock option is enabled in the system settings. This solution is not completely reliable, because users on a non-rooted device can still set a mock location then disable mock locations, with the result that the mock location is still active for a few seconds. In the state model a check is performed on this option before every position update. A second option is to perform a check if other applications are installed that require the permission for location mocking. While this check is not implemented in the prototype, it could easily be added and also used every time the position is pushed to the location provider.

9.6.2 Authorization Domain

The Authorization Domain consists of the main XACML components, including the Context Handler (CH), the Policy Decision Point (PDP), the Policy Administration Point (PAP) and the Policy Information Point (PIP)

The PDP decides over the legitimacy of an XACML request. For this task the PDP compares the XACML request with the defined policies. There is no usable Open Source GeoPDP solution available. The GeoServer[8] and the CHARON framework[9] have GeoPDP approaches, but they follow different goals and cannot be build from their current source code. Additionally the current version of GeoXACML is based on XACML 2.0, which makes it difficult to adapt to XACML 3.0 based solutions. For this prototype a commercial GeoPDP from Secure Dimensions[10] is used. This company offers a web service for GeoXACML. A customer gets access to a specific URL to which XACML requests can be sent via HTTP POST. With this web interface the customer is able to upload GeoXACML policies.

The Context Handler (CH), which is developed using Java, Jersey, JAX-RS and Tomcat, offers a REST interface to receive information from the PEP. The XACML request is transmitted to the PDP. If there are some attributes missing, the CH needs to gather further information from the PIP. After enriching the initial request with

[7] http://code.google.com/p/google-gson/

[8] http://geoserver.org/display/GEOS/GeoXACML-Integration

[9] http://www.enviromatics.net/charon/

[10] http://geoxacml.secure-dimensions.net/

attributes from the PIP, the altered request is transmitted again to the PDP. The answer sent by the PDP is returned to the requesting PEP. It was decided to put all additional attributes from the PIP into the XACML environment element. It follows a description of a common CH sequence:

1. The XACML request arrives per REST
2. The CH sends the request to the PDP - If there are missing attributes:
    a. The CH extracts the subject information from the request
    b. The CH contacts the PIP for additional attributes of the requesting identity
    c. The information from the XACML request and the PIP is merged together and sent to the PDP via REST
3. The PDP processes the request and generates an XACML response
4. The CH transmits the XACML response to the PEP

The following parameters are configurable:

- The REST URL of the PDP
- The REST URL of the PIP

The PIP, developed in Java, Jersey, JAX-RS, Tomcat, offers a REST interface to gather information for the CH. This is an intermediate interface, because the request is just forwarded to the location provider. The received parameters are transformed into a XML response and sent back to the Context Handler. The sequence is as follows:

1. The PIP request arrives per REST
2. The PIP sends a request to the location provider
3. The PIP receives the reply from the location provider
4. The PIP responds to the initial request from the CH

The following parameter is configurable:

- The REST URL of the Location Provider

9.6.3 Location Provider Domain

The location provider was built from scratch as a Servlet and uses a SQLite database for storage. The location provider expected by the LocationAPP, is able to process a POST request that contains the user in JSON format. The location provider offers a REST interface to receive information from the LocationAPP and offers geolocation information to the PIP.

1. The LocationAPP has to be registered with an identity provider (which is beyond the scope of this prototype)
2. The LocationAPP transmits the actual position data every 300 seconds

3. The location provider checks the validity (i.e. blocking injection attacks) of the data
4. The location provider inserts a timestamp into the data
5. The location provider checks if the user already exists, and decides between two cases:
    a. UPDATE an existing record (primary key is the identity string)
    b. INSERT as an new record (primary key is the identity string)
6. If an error occurs the location provider sends a HTTP response status code client error (>= 400)
7. If everything works fine, the location provider sends HTTP response status success codes (<300)

If the PIP requests information from the location provider, the provider will respond with a JSON object of a user. If the user is not found, all values will be empty. The Context Handler could access the location provider directly but this would violate the architecture of XACML.

9.6.4 Service Provider Domain

The service provider domain hosts the example TRESOR Ruby on Rails application and a Policy Enforcement Point (PEP), which grants or denies access to the application depending on the decision computed in the authorization domain. The PEP is a module, which intercepts the web service request. It is called directly within the application, i.e., it is placed before the main application. The process works as follows:

1. The service request arrives at the web service
2. The PEP extracts information from the HTTP header:
    a. The action (POST, GET, PUT, DELETE)
    b. The resource (target URL)
    c. The SAML identity (or other authorization attributes) – if necessary the requesting IP address
3. The PEP sends a request to the CH via REST
4. The CH processes the request (in coordination with other entities)
5. The CH gives the result back (nested in XML)
6. If the request has been denied: the PEP sends an HTTP status code 403
7. It the request has been granted: the PEP exits and the processing of the requested application will start

The communication between the PEP and CH relies on REST and XML. The following parameters are configurable:

- Behavior of the PEP in case an internal error occurs. (deny or permit; default: deny)
- Behavior of the PEP if the decision is "indeterminate", i.e., an error occurs in processing. (deny or permit; default: deny)
- Behavior of the PEP if the decision is "not applicable", i.e., no policy found for the request. (deny or permit; default: deny)
- Timeout of the PEP. After a timeout, the behavior is the same as not reachable. (default: 200 milliseconds)
- The REST URL of the CH

9.6.5 Identity Provider Domain

In our example the identity provider domain checks the identity of the user, which access the service against a database (Identity Database). The database is connected to the PIP that gathers information about different userIDs. In our example we used different userIDs, such as *doctor*, *nurse* and *admin*.

9.7 Conclusion and Outlook

The paper presents an overview of current research carried out in the TRESOR project. The main components are the proxy and the broker. It is shown that the requirements gathered during the first project phase led to a re-evaluation of current research in the fields of description languages and distributed policy systems. First results during an implementation of a location-based access control prototype using GeoXACML unveiled more research questions, e.g., the provisioning and revocation of known devices in the system, the usage of device or application identifiers and power consumption issues on smartphones. Additional ideas on how to use location information of users accessing medical cloud services are also being investigated during further realizations of the project goals. All these ideas and additional components, such as an open PaaS-platform and a marketplace for cloud services are developed under the proposition to build a trusted cloud ecosystem with the focus on the German health sector.

9.7.1 Acknowledgement.

The work presented in this paper is performed in the context of the TRESOR project. TRESOR is funded by the German Federal Ministry of Economics and Technology (BMWi) as part of the Trusted Cloud[11] technology program.

References

1. Umu xacml editor (2013). URL http://umu-xacmleditor.sourceforge.net/
2. Ws02 identity server (2013). URL http://wso2.com/products/identity-server/
3. Ardagna, C.A., Cremonini, M., di Vimercati, S.D.C., Samarati, P.: Access control in location-based services. Privacy in Location-Based Applications **LNCS 5599**, 106–126 (2009)
4. Axiomatics: Axiomatics Language for Authorization (ALFA) (2012). URL http://www.axiomatics.com/axiomatics-alfa-plugin-for-eclipse.html
5. Bundesärztekammer: Berufsordnung für die in Deutschland tätigen Ärztinnen und Ärzte - §10 Abs. 3. http://www.bundesaerztekammer.de/page.asp?his=1.100.1143 (2011)
6. Fielding, R.T.: Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine (2000). 2000
7. Geman Federal Ministry of Justice: Verordnung über den Schutz vor Schäden durch Röntgenstrahlen, § 28 Aufzeichnungspflichten, Röntgenpass. http://www.gesetze-im-internet.de/r_v_1987/__28.html (2011)
8. German Federal Ministry of Justice: StGB § 203 Verletzung von Privatgeheimnissen. http://www.gesetze-im-internet.de/stgb/__203.html (2013). German
9. Graf, T., Zickau, S., Küpper, A.: Enabling location-based services on stationary devices using smartphone capabilities (2013)
10. Lockhart, H., Parducci, B., Rissanen, E.: Oasis xacmlv3 administration and delegation profile (2010)
11. Merz, F.: Wachstumsmotor Gesundheit: Die Zukunft unseres Gesundheitswesens. Carl Hanser Verlag, München (2008)
12. Microsoft, Inc.: Access & Information Protection. http://www.microsoft.com/en-us/server-cloud/windows-server/identity-access.aspx (2013)
13. Neuman, C., Yu, T., Hartman, S., Raeburn, K.: The Kerberos Network Authentication Service (V5). RFC 4120 (Proposed Standard) (2005). URL http://www.ietf.org/rfc/rfc4120.txt. Updated by RFCs 4537, 5021, 5896, 6111, 6112, 6113, 6649
14. OASIS (Organization for the Advancement of Structured Information Standards): OASIS eXtensible Access Control Markup Language (XACML). https://www.oasis-open.org/committees/xacml (2013)
15. OGC (Open Geospatial Consortium): Geospatial eXtensible Access Control Markup Language (GeoXACML) (2011). URL http://www.opengeospatial.org/standards/geoxacml
16. Repschläger, J., Zarnekow, R., Wind, S., Klaus, T.: Cloud Requirement Framework: Requirements and Evaluation Criteria to adopt Cloud Solutions. In: Proceedings of the 20th European Conference on Information Systems (2012)
17. Slawik, M.: The trusted cloud transfer protocol. In: Proceedings of the 5th Intl. Conference on Cloud Computing Technology and Science (CloudCom 2013), pp. 203–208. IEEE, Bristol, UK (2013)

[11] http://trusted-cloud.de/

18. Thatmann, D., Slawik, M., Zickau, S., Küpper, A.: Towards a Federated Cloud Ecosystem: Enabling Managed Cloud Service Consumption, accepted. In: Proceedings of the 9th International Conference on Economics of Grids, Clouds, Systems, and Services, GECON 2012. Springer-Verlag Berlin Heidelberg, Berlin, Germany (2012)
19. The Eclipse Foundation: Eclipse Modeling Framework Project (EMF). http://www.eclipse.org/modeling/emf/ (2013)

Chapter 10
Towards Reliability Estimation of Large Systems-of-Systems with the Palladio Component Model

Fouad ben Nasr Omri and Ralf Reussner

Abstract The component paradigm aims at accelerating the construction of large scale systems-of-systems by facilitating the integration of third-party components. The size and the complexity of such large software make the reliability assessment process challenging. The different components building such large systems can be developed and maintained by multiple parties. In addition, the reliability testing process should focus on the integration logic connecting the components. The standard approach is to perform integration testing to uncover interaction faults between the different components building the software. However, integration testing of large systems-of-systems is in the most of the cases intractable. The multiplicity of the potential interactions between the different subsystems can be hardly systematically tested. In addition, standard integration test cases cannot be used to estimate the reliability of the tested software system. The integration test cases are not necessarily representative of the software usage model and consequently cannot be used to derive a sound reliability estimate. We propose a novel testing approach which supports both sound reliability estimation and high interaction coverage for reliable and interaction-intensive software.

Key words: Software reliability testing, statistical usage-based testing, symbolic execution

10.1 Introduction

Modern software applications involve the integration of autonomous software systems communicating with each other to deliver added-value services. The goal of the CLOUDwerker and PeerEnergyCloud projects (two trusted cloud projects where

Fouad ben Nasr Omri · Ralf Reussner
Chair for Software Design and Quality, Karsruhe Institute of Technology, Germany
e-mail: {fouad.omri,ralf.reussner}@kit.edu

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_10

the authors are involved), is to develop mechanisms and concepts to build large reliable systems-of-systems by integrating software services developed and maintained by different parties and vendors.

The reliability of a software is commonly assessed by generating and executing statistical usage-based test cases. Statistical usage-based testing establishes a basis for statistical inference about the software expected field quality [20]. However, statistical usage-based testing is in the most of the cases impracticable. The number of test cases executions required to reach a target reliability value is too large, even for very small software [12]. We propose a testing technique that should reduce the number of the test cases executions required to reach a target reliability value. We call our technique statistical white-box testing. Statistical white-box testing combines the standard statistical usage-based testing approach with the automatic generation of test patterns using symbolic execution. The standard statical usage-based testing approach is a black-box testing technique. Consequently, test cases may be redundant. However, statistical white-box testing controls possible test cases redundancy according to a given usage model.

Statistical usage-based testing involves generating test cases with random inputs according to a given usage model of the software under test. In the context of complex large software, it can be difficult to specify the usage model. We propose a hierarchical specification produced by the Palladio Component Model (PCM) environment. Using PCM, we can specify the expected behaviour of the single software components building the large software separately. We can estimate the reliability of the software components separately which allows to distribute the reliability testing effort on the software components.

The subsystems building a large systems-of-systems communicate with each other to deliver added-value services. Therefore, the reliability assessment process must take into account the possible interactions between the subsystems. Integration testing aims to uncover software failures in the interactions between components and their execution environment (i.e., other components or the underlying middleware platform). However, integration test cases are not necessarily representative for the software usage model and consequently cannot be used to estimate the reliability of the tested software. We propose an approach to automatically generate statistical usage-based test cases which supports both a sound reliability estimation and a high interaction coverage for reliable and interaction intensive software.

The paper is organized as follows: Section 2 illustrates a motivating example for the presented work. Section 3 provides background information for the paper. In Section 4, we propose our reliability testing approach. Section 5 discusses existing related works. We summarize our work in Section 6.

10.2 Motivating Example

The trusted cloud project CLOUDwerker, aims at developing a trustworthy open cloud platform for the primary target group craftsmen. The platform should allow

the integration of different Software-as-a-Service (SaaS) services such as, invoicing services, Customer Relationship Management (CRM) services and eGovernment services. The goal of the project is to deliver the craftsmen target group reliable added-value services.

In particular, the CLOUDwerker project aims at the development of an integrated trustworthy and reliable cloud service for distributed document creation (e.g., creation of offers and invoices). The service is the result of the integration different services such as an offer and invoicing service, a CRM service and an electronic mailing service as illustrated in Fig 10.1.

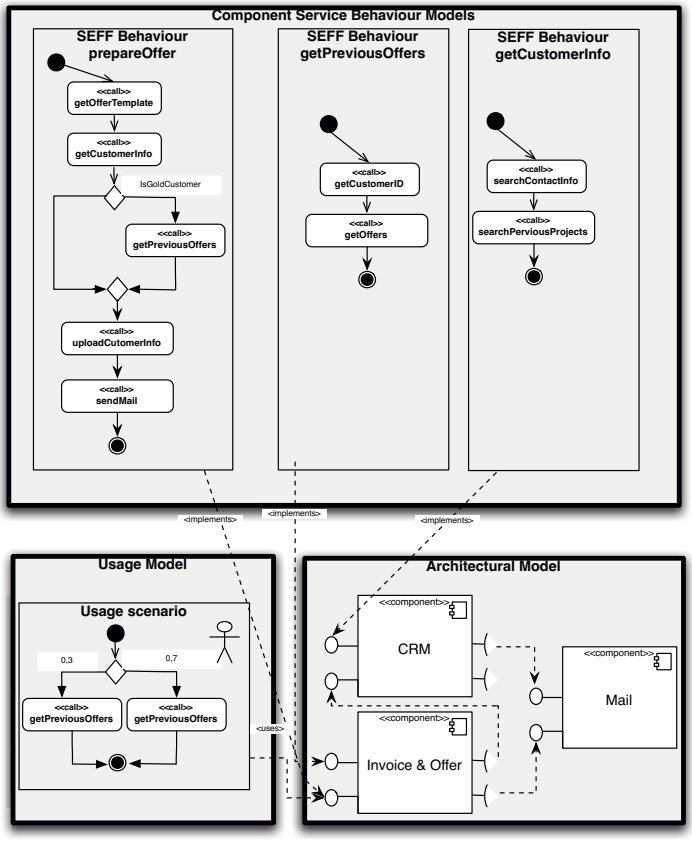


Fig. 10.1: PCM Instance of a CLOUDwerker Collaboration Service (Overview)

In order to estimate the reliability of the collaboration service, we need first to understand the various interactions between the subsystems building the service. In

addition, we need a usage model which describes the expected usage of the collaboration service.

The subsystems building the collaboration service implement complex functionalities. The standard approach is to perform integration testing to detect possible interaction faults. The integration test cases cannot be used to estimate the reliability of the tested software, since they are not necessarily representative for the software specific operational usage. Consequently, one would model the expected usage model of the software and generate extra statistical usage-based test cases from that model. The statistical usage-based test cases are executed and analysed to estimate the reliability of the software. However, the design of the test scenarios and the test input data may induce an analysis of reasonable complexity. In particular, the detailed analysis of the behaviour models of the software is in the most of the cases intractable, especially if the analysis is done manually and without the use of CASE tools for the specification and design of such complex systems.

Fig. 10.1 presents an overview of a PCM instance describing the collaboration service. We use the PCM instance to extract automatically a Markov usage model of the software. We generate automatically test cases from the Markov usage model. The test cases support both reliability estimation and high interaction coverage.

10.3 Background

This section provides background information for the rest of the paper.

10.3.1 Statistical Usage-based Testing

Statistical usage-based testing is the certification step of the *Cleanroom Software Engineering*. Cleanroom is a software development process to produce almost zero-defect software product with measurable reliability [10]. Statistical usage-based testing is focusing on how the software fulfils its intended functionalities from a users' perspective and not on the technical details of a software implementation (e.g., branch coverage, boundary-value testing, etc.,...). It is a black-box testing technique that randomly selects test cases from an expected usage model. A Markov chain usage model [20] describes the expected operational usage model of the software in its envisaged usage environment. We use in this paper the definition of [20]: the Markov chain usage model is a finite state, first order, discrete parameter and irreducible Markov chain. The chain has a unique start state, a unique final state, and a set of intermediate software usage states. The usage states are the actions that can be performed on the software. Given the present usage state, the next usage state of the software is independent of all past usage states (i.e., *Markov property*).

Definition 1 *Consider a Markov Chain with finite state space* Ω *a transition matrix P and a unique start state and a unique final state . Then a corresponding graphical*

representation is the weighted graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$*, where* $\mathscr{V} = \Omega$ *and* $\mathscr{E} = \{(x,y) \in \Omega \times \Omega | P(x,y) > 0\}$*. Self-loops are allowed since* $P(x,x) > 0$ *is allowed.*

Definition 2 *We define a usage-based test case as a walk* $\alpha = (e_0, e_1, \ldots, e_n)$*,* $e_k \in \mathscr{E}$*,* $k = (0, \ldots, n)$ *starting at the start node and ending at the end node of the graph. An Arc e can occur more than once in* α*.*

Definition 3 *The probability of execution of usage-based test case* $\alpha = (e_0, e_2, \ldots, e_n)$ *is defined as:* $P(\alpha) = \prod_{k=0}^{k=n} P(e_k)$*, where* $P(e_k)$ *is the weight of the edge* e_k *as specified in the transition matrix of the Markov chain.*

10.3.2 Software Reliability Estimation

The software reliability is defined as the probability that the software executes without to fail in a given usage environment during a specified execution time [11]. There are two types of software reliability models. The first type of reliability models is the "white-box" models which capture the information about the architecture of the software and the interactions between its components. The second type is the "black-box" models which only consider the interaction of the software with the external environment. "Black-box" models require a sequence of failure data. "Black-box" models are used in industry and academia [5] for their capability to capture the *reliability growth*. "White-box" reliability models need in addition to the failure and execution data as by "black-box" models, other more sophisticated data to predict the software reliability. In this work, we focus on the generation of the failure and execution data for reliability models. We explain our approach based on "black-box" models reliability models. However, our approach can also be used for "white-box" reliability models. Software reliability growth models try to statistically correlate the failure history with mathematical functions such as the exponential function. If the correlation is good, then the correlation function is used to predict the future failure behaviour (i.e., reliability) of the software.

We use symbolic execution to generate our test cases. Symbolic execution is a path generation technique (more details can be found in the following section). In this work, we call test cases redundant if they do not increase the path coverage in the tested software.

10.3.3 Symbolic Execution

Symbolic execution is a path generation technique. It executes a program with symbolic values and not with concrete ones. Symbolic execution evaluates the branching instruction, which is a boolean expression, to a symbolic value and not to true or false as in a regular program execution. Consequently, both branches can be taken. A

path in symbolic execution refers to a path in the control flow of the program. Each path is described with a path condition. The path condition is defined over a set of input variables $\mathscr{I}$. The path condition is a conjunction of branching decisions (i.e., boolean expression) made during the symbolic execution of a path in the program. The paths during symbolic execution are distinct. The different combination of decisions define the different paths to explore. Loops and recursion in a program can lead to an infinite number of paths. In standard approach, a user defined threshold can be set to bound the path length.

Constraint solvers are used to check the feasibility of the explored paths. A path is feasible if its path condition is satisfiable, i.e., there exists an input to the program that covers the path. If the path is feasible test inputs are generated to execute it.

We use the Java Pathfinder [13] symbolic execution engine for Java bytecodes. Java Pathfinder is developed by NASA and has been used in industry and different research project. Java PathFinder has already been used for test case input generation [19], but not for reliability testing. We extended Java Pathfinder to generate statistical white-box test cases.

10.3.4 PCM Architectural Modelling Capabilities

This section provides a description of the PCM models we use for this work. A complete and more detailed description can be found in [3]

10.3.4.1 Component Service Behaviour Models

The service behaviour of a component is specified in PCM in terms of service effect specifications (SEFFs). A SEFF is an abstract model of the usage of required services by the provided services. SEFFs model the control and data flow within the component service. SEFFs can include loops, forks and value-guarded or probabilistic branches. The control flow constructs in a SEFF are guarded using expression formed by input and component parameters.

The example in Fig 10.1 consists of four SEFF models, each modelling a provided service of a component from the architectural model.

10.3.4.2 Architectural Model

The components are connected by a software architect into an architectural model. The software architects determine the provided services that shall be exposed to system users or other systems.

10.3.4.3 Usage Model

The usage model describes the system's usage profile and consists of a set of usage scenarios. The usage scenarios represent different use cases of the system. A usage scenario specifies a sequence of system calls with probabilistic control flow constructs (e.g., branches or loops). The usage model is defined by a domain expert.

10.4 Approach

In this section, we present our approach for the estimation of the reliability of a software system modelled in PCM. Fig 10.2 presents an overview of the approach functioning.

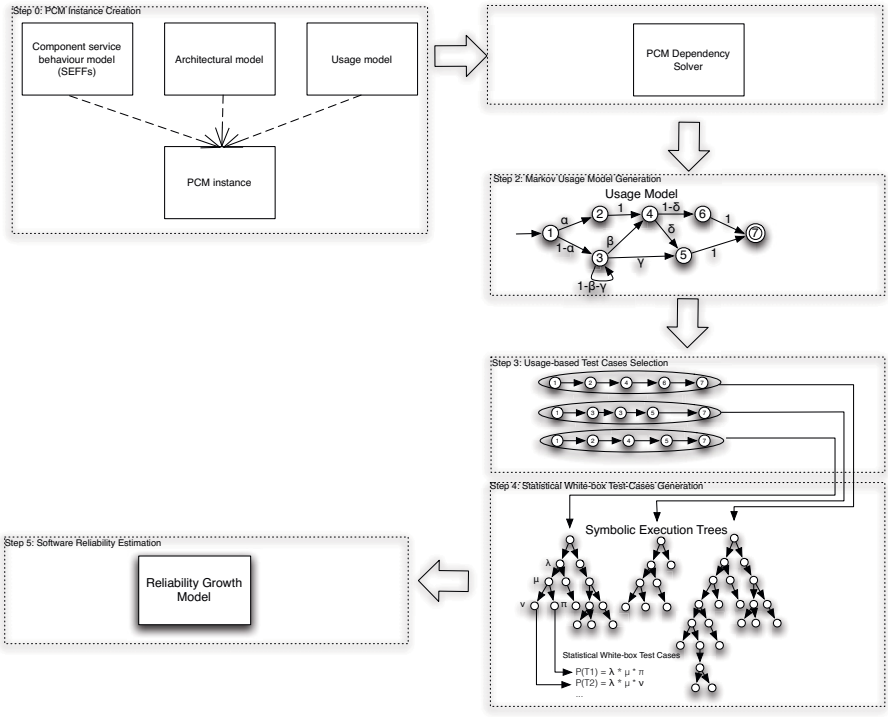


Fig. 10.2: Overview of the Reliability Estimation Approach with the PCM

The approach gets as input a full specified PCM instance combining the models as described in Section 10.3.4 and a user provided threshold $\mathscr{UT}$ (i.e., the user is interested in generating usage-based test cases which have a probability of execution higher than $\mathscr{UT}$ according the usage model).

The high-level behaviour of the software components is specified using service effect specifications(SEFFs). SEFFs may contain parameter dependencies which describe the influence of input parameters on the control and data flow of the method calls in the SEFFs. The parameters in the SEFFs are specified using parameter names and stochastic expressions. The domain expert can characterize the parameters with concrete values in the PCM usage model, depending on the expected software usage.

In step 1, we solve the parameter dependencies in the SEFFs. We obtain SEFFs with concrete parameter values (a concrete value is either a constant or a probability distribution) for the method calls, probability of the branches, length of loops, etc.,

We use all the SEFFs to generate recursively a Markov usage model of the modelled software in step 2. In step 3, we generate all usage-based test cases which have a probability of execution higher than the user pre-defined threshold $\mathscr{UT}$. Each usage-based test case is a sequence of method calls. The methods are either component's methods (i.e., implemented by the components) or integration layer's methods which connect the components to each other. If the source code of the components is not available, we call the method with concrete values and not symbolically. We use a mix of concrete and symbolic execution. In step 4, we execute symbolically each usage-based test case, after possible substitution of component's methods. Parameter values distributions as well as transition probabilities from the usage model are propagated during the symbolic execution. The result of the symbolic execution is a set of weighted feasible paths (see Section 10.4.4 for more details). The weight of a path expresses the relative frequency of execution of that path compared to the the other generated paths. For each path with weight ψ we generate exactly ψ different test cases which execute that path. We call these test cases statistical white-box test cases.

The generated statistical white-box test cases are then executed and their outcome (failure or success) is used in step 5 to estimate the reliability of the component-based software using a reliability growth model.

10.4.1 Resolving Parameter Dependencies

As described in Section 10.3.4.1, the component developer is expected to specify the behaviour of the component using SEFFs. The parameter dependencies expressing the impact of component input parameters on the control and data flow within the component must be solved. The probability distribution of the parameter is defined in the usage model of the software.

We reuse the existing PCM Dependency Solver to resolve all parameter dependencies in the PCM instance. A detailed description of the PCM Dependency Solver can be found in [3].

10.4.2 Generation of Markov Usage Model

The output of the PCM Dependency solver is a set of SEFFs with concrete values. Using theses SEFFs, we generate a Markov Usage model recursively starting from the usage model specification. We traverse the PCM instance and substitute recursively each action by its SEFF if available. The actions include internal actions, call actions, branches and forks. Loops actions are unrolled based on the solved parameter characterization of the loop count. Each SEFF is transformed to a Discrete Time Markov Chain (DTMC) by assigning probabilities to the transitions of the SEFF. Each action of the SEFF becomes a state of the DTMC. The transition probabilities are computed based on the usage model information as follows:

- The transition probabilities of branches and forks are computed based on the probability distribution of the guard expressions as specified in the PCM usage model.
- We assign to the other transitions the probability 1.
- We unroll loop actions based on the solved parameter characterization of the loop count. For example, if the loop count is probabilistically characterized as $P(count = 4) = 0.7$ and $P(count = 6) = 0.3$, then we branch the predecessor node in the Markov usage model of the loop action with two transitions. The transitions have respectively the transition probability 0.7 and 0.3. We append to each transition the unrolled loop as a linear sequence of the loop inner actions. The transitions in this sequence have the probability 1. In our Example, we repeat 4 times the inner loop actions in a sequence with transition probabilities equal 1 and we append this sequence to the transition with probability 0.7. We do the same for the other transition with a sequence of 6 repetitions of the inner loop actions.

The aggregation of the DTMCs of the different SEFFs build the Markov usage model of the software. We use the Markov usage model to generate statistical usage-based test cases.

10.4.3 Generation of Usage-based Test Cases

The aim of this section is to automatically generate a suitable sample of usage-based test cases through the Markov chain usage model. Statistical Software testing is viewed as a statistical experiment. In this experiment, a sample is statistically generated from all possible uses of the software. This sample is then used to draw

conclusions about the operational behaviour of the software. The population of uses of the software is characterized by a finite Markov chain as discussed in Section 10.3.1.

The goal of the generation process is to find the most likely usage-based test cases having a probability of execution higher than a user defined threshold and that visit the most possible transitions in the Markov usage model. We call this threshold usage threshold ($\mathscr{UT}$).

The goal of the generation process is similar to the *Directed Chinese Postman Problem* [17] with few modifications: we want to generate the fewest number of usage-based test cases that cover the most likely edges in the Markov usage model, and which have a probability of execution higher than a user defined usage threshold $\mathscr{UT}$. We call such generated usage-based test cases suite a *maximum edge coverage test cases* suite $\mathscr{MECTC}$. In the following, we present an algorithm for the generation of a $\mathscr{MECTC}$ suite from a Markov usage model. We first recall some graph theoretical notions we use in the algorithm.

Definition 4 *In a graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$, for each $v \in \mathscr{V}$ the indegree of v denoted as $deg^-(v)$ is defined as the number of edges going into v. The outdegree is defined as the number of edges pointing out of v and denoted as $deg^+(v)$. Let $\delta(v) = deg^+(v) - deg^-(v)$. If $\delta(v) = 0$, we say v is balanced.*

Definition 5 *An Eulerian path in a directed graph is a directed path traversing each edge exactly once. An Eulerian directed cycle is an Eulerian path which starts and ends on the start node.*

Theorem 1 *A Graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ has an Euler cycle if and only if $\forall v \in \mathscr{V} : \delta(v) = 0$.*

Algorithm 1 generates the shortest paths from the source to the sink in the Markov usage model. The total cost of each path should be smaller than $-log(\mathscr{UT})$. The cost of each transition is $-log(p)$; p the transition probability. The minimization of the cost of a path, which means the minimization of the sum of the negative logs of the transition probabilities, is equivalent to the maximization of the product of the transition probabilities along the path. Consequently, the algorithm generates the most likely paths which have a probability of execution higher than $\mathscr{UT}$.

The procedure addToAdjacencyMatrix$(v_i, v_j, -log(P(v_i, v_j)))$ adds an edge from node v_i to node v_j with a cost equal $-log(P(v_i, v_j))$ in the adjacency matrix representation of the new constructed Graph.

The procedure findShortestPaths uses the Floyds algorithm to find all shortest paths from $\mathscr{D}^+$ to $\mathscr{D}^-$ with a complexity of $O|\mathscr{V}^3|$. The function solveOptimization computes a one-to-one mapping of the unbalanced vertexes of the graph using the Hungarian algorithm [9] with a complexity of $O|\mathscr{V}^3|$. The computation of an Euler cycle has a complexity of $O(|\mathscr{V}||\mathscr{E}|)$. The algorithm 1 has then a total complexity of $O|\mathscr{V}^3|$. We extended the Folyds algorithm with a procedure to detect cycles in the Markov usage model. Cycles are not considered while computing the shortest paths.

Algorithm 1: Generation of an $\mathscr{MECTC}$ suite from a Markov usage model

Input: $\mathscr{G} = (\mathscr{V}, \mathscr{E}); \mathscr{UCT}$
Output: $\mathscr{MECTC}$
foreach $(v_i, v_j) \in \mathscr{E}$ **do**
  addToAdjacencyMatrix$(v_i, v_j, -log(P(v_i, v_j)))$
end
sets $\mathscr{D}^+, \mathscr{D}^-, \mathscr{MECTC} := \emptyset$
$\mathscr{D}^+ \leftarrow \{v \in \mathscr{V} | \delta(v) > 0\}$
$\mathscr{D}^- \leftarrow \{v \in \mathscr{V} | \delta(v) < 0\}$
foreach $v \in \mathscr{D}^+$ **do**
  findShortestPaths$(v, \mathscr{D}^-)$ //generates the spanning tree of the graph
end
$\mathscr{OPT} := solveOptimization(D, \mathscr{UT})$ // one to one mapping of the unbalanced vertexes using the Hungarian method
foreach $(v_i, v_j) \in \mathscr{OPT}$ **do**
  ω :=findShortestPath$(v_i, v_j, \mathscr{UT})$
  foreach $(v_k, v_l) \in \omega$ **do**
    addToAdjacencyMatrix$(v_k, v_l, -log(P(v_k, v_l)))$
  end
end
$\mathscr{MECTC}$:= findEulerCycles$(\mathscr{G})$

The algorithm outputs a set of sequences we call $\mathscr{MECTC}$. For each sequence $\mathscr{S} \in \mathscr{MECTC}$, if $\mathscr{S}$ contains cycles in the original Markov usage model then we iteratively add the cycles to $\mathscr{S}$. Each time, we add a cycle $\mathscr{C}$ to $\mathscr{S}$, a new sequence $\mathscr{S}^{'}$ is created if and only if $p(\mathscr{S}^{'}) = p(\mathscr{S}).p(\mathscr{C}) \geq \mathscr{UT}$. $p(\mathscr{S})$ is the probability of execution of the sequence $\mathscr{S}$. $p(\mathscr{C})$ the probability of execution of the cycle $\mathscr{C}$. Both probabilities are computed from the usage model as the product of the transition probabilities along the sequence and the cycle respectively.

10.4.4 Automatic Generation of Statistical White-Box Test Cases

Each generated usage-based test case is executed symbolically. Method calls with no source code available are executed concretely. A usage-based test case is a sequence of method calls. The symbolic execution tree consists of possible path conditions over a set of input variables $\mathscr{I}$. Each path condition is a conjunction of branching decisions as described in Section 10.3.3.

We enhanced Java Pathfinder Symbolic search algorithm to compute the probability that a branching decision holds true. We call such a probability, the probability of holding true. The probability of holding true is computed using the information about the parameter value distributions and the transition probabilities specified in the usage model. A branching decision with variables that have no parameter value distribution from the usage model have a probability of holding true equal $\frac{1}{2}$: both outcomes of the branching decisions are equal likely to be taken.

Each path in the symbolic execution tree is characterized by a path condition. A path condition is a conjunctive formula, ϕ, encoding the branch decisions made while executing the code symbolically. Let $\phi \equiv \omega_1 \wedge \omega_2 \wedge \ldots \wedge \omega_n$ where ω_i, $i = 1,2,\ldots,n$ a branching decision along the path $\mathscr{P}$ with path condition ϕ. Without the loss of generality, we index the branching decisions in ascending order from 1 to n in the order of their exploration by our search algorithm. Let $p(\omega_i)$ be the probability that the branching decision ω_i holds true. The probability of a transition from branching decision ω_i to ω_{i+1} denoted as $\omega_i \rightsquigarrow \omega_{i+1}$ is defined as

$$p(\omega_i \rightsquigarrow \omega_{i+1}) = \begin{cases} p(\omega_i), & \text{if } \omega_i \text{holds true} \\ 1 - p(\omega_i), & \text{if } \omega_i \text{holds false} \end{cases}.$$

The probability of execution of a path condition ϕ is defined as $p(\phi) = \prod_{i=1}^{n-1} p(\omega_i \rightsquigarrow \omega_{i+1})$.

The probability of execution of a path $\mathscr{P}$, $\mathscr{P} \in \mathscr{S}$ and $\mathscr{S}$ a usage-based test case, is defined as $p(\mathscr{P}) = p(\phi).p(\mathscr{S})$. $p(\mathscr{S})$ the probability of execution of $\mathscr{S}$ according the operational usage profile, and ϕ the path condition for $\mathscr{P}$.

We use $p(\phi)$ to control the search depth of the symbolic execution and only explore and solve the execution paths which have a probability of execution higher that a user set threshold ($\mathscr{U}\mathscr{T}$).

For each selected usage-based test case a set of paths is generated. The paths are feasible and have a computed probability of execution as explained earlier. Let $\mathscr{A}$ be the set of all paths we extracted for all usage-based test cases selected in Section 10.4.3. The weight of a path $\mathscr{P} \in \mathscr{A}$ is defined as $\psi(\mathscr{P}) = \lceil p(\mathscr{P}).|\mathscr{A}| \rceil$ where $|\mathscr{A}|$ the cardinality of the set $\mathscr{A}$ and $\lceil . \rceil$ the ceiling function.

A statistical white-box test case is an execution of a feasible path. A feasible path represents an equivalence class on the inputs which execute that path. The weight of a path defines the relative frequency of execution of that path to the rest of paths we generate. We generate exactly $\psi(\mathscr{P})$ different statistical white-box test cases to execute a path $\mathscr{P}$. The parameter generation is done by Java Pathfinder as explained in [19].

10.4.5 Reliability Estimation

Software reliability growth models require as input (i) the failure count in each testing interval and (ii) the completion time of each period where the software is under observation.

The observed reliability is not directly related to the number of faults in the software, but influenced by the usage model. Mills et al. [10] demonstrated that removing 60% of all faults in a software can decrease the failure rate of the whole system by less than 3%. We believe that test cases which have a higher probability of execution according to the usage model should have a higher impact on the estimated reliability since they represent highly possible usage situations of the software.

We do not propose to change the existing reliability growth models. However, our new testing generation and execution technique introduces a weighting effect on existing software reliability growth models without modifying the models themselves. As explained in Section 10.4.4, we exactly generate $\psi(\mathscr{P})$ statistical white-box test cases for each generated feasible execution path $\mathscr{P}$. Each execution of a statistical white-box test case is equally likely to detect a failure along the path it executes. A failure is a variation from the expected behaviour of the software execution observed by the user as a result of existing faults. The software reliability growth model counts $\psi(\mathscr{P})$ different executions, for each feasible execution path $\mathscr{P}$. The $\psi(\mathscr{P})$ executions are redundant covering the same path. The redundancy describes the importance of the path according to the usage model.

Each execution path can exhibit maximal one failure, since an execution path has exactly an expected output. If we execute a path $\mathscr{P}$, $\psi(\mathscr{P})$ times with different input parameters, and we detect n times the failure ($n \leq \psi(\mathscr{P})$), then n defines the importance of that failure according to the usage model. The software reliability growth models count then that failure n times. n describes the impact of the occurrence of the failure on the estimated software reliability.

10.5 Validation and Limitations

We tested our usage-based symbolic execution technique on several examples such as the Java open source libraries `JZlib` [1] and `JShell` [2]. The results were very promising. We could successfully generate statistical white-box test cases and conduct reliability estimations. We are planning to evaluate our approach on representative web applications to assess the effectiveness of our approach in generating statistical usage-based test cases which supports both a sound reliability estimation and a high interaction coverage. However, our approach inherits some limitation from the used Java Pathfinder framework. For example, Java Pathfinder can only execute self executable Java programs. Consequently, we cannot test Java web applications for example. In order to validate our approach, we need first to to generate models for the environment interacting with a web applications such as the J2EE servlets. The models are used to automatically generate small set of behaviours representing a subset of typical and atypical scenarios. The models are generated in form of drivers and stubs. The drivers and stubs are combined with the web application code to form an executable Java Code. This code can then executed symbolically. We apply then our approach for the generation of statistical white-box test cases for the new code. The generated test cases are then executed on the original web application code to estimate the reliability.

[1] http://www.jcraft.com/jzlib/

[2] http://geophile.com/jshell/

10.6 Related Work

10.6.1 Automatic Derivation of Statistical Usage-based Test Cases

We presented an approach to automatically generate executable test cases from a software usage model. [1, 7, 4] present similar work in the context of usage interface testing or protocol conformance testing. Other works such as [16, 14, 8] provide tools and usage model description formats to automate the generation of statistical usage-based test cases. However, these approaches do not consider the generation of test data and consequently the execution of the test cases (i.e., the execution process of the test cases is not fully automated). Our work, presents an approach to automatically generate statistical usage-based test cases from large Markov usage models. Our approach uses the information provided by the usage model about the input parameter distributions, propagate such information in the usage-based test cases, and generate adequate input parameters for the test cases.

10.6.2 Introducing White-box Program Information to Statistical Usage-based Test Cases

Statistical usage-based testing is a black-box testing technique [10, 20, 12, 15]. In order to execute statistical usage-based testing, test data are randomly selected based on a usage model of the software. [18, 6, 2] presented a testing technique they call statistical structural testing. Statistical structural testing aims at finding a probability distribution over the input domain of a program that maximizes a pre-defined structural code coverage criteria. They are deriving test cases based on code coverage criteria. Statistical structural test cases may have little connection to the actual use of the software. Consequently, their focus is more on finding faults in the program and not reliability assessment.

Our approach is the first to introduce white-box code information to statistical usage-based test cases. We propagate usage model information while generating white-box test cases using symbolic execution.

10.7 Conclusions and Future Work

We presented a modelling and testing approach to estimate the reliability of large systems-of-systems. Our approach uses the PCM to model and analyse the architecture and usage of a software. Based on the PCM instance of the modelled software, our approach automatically generates a Markov usage model. The usage model is used to automatically generate weighted statistical white-box test cases. The test cases are executed and their outcome is used by black-box reliability growth mod-

els to deliver reliability estimates. The tool for our approach is in development. An empirical validation of the approach is ongoing work and will be conducted on real large systems-of-systems in the context of the trusted cloud project. The main goals of the validation are (i) to verify to what extent our approach reduces the number of required test cases to reach a given reliability target with a given confidence, (ii) whether the reliability estimates delivered by our approach are reliable (i.e., Do our approach deliver an overestimation of the true reliability of the tested software).

Symbolic execution is computationally more expensive than random testing. Symbolic execution may not scale to large programs with many feasible program paths. We control the exploration and generation of execution paths during symbolic execution based on information derived from the usage model; we explore and solve only the execution paths which we judge likely to be executed according to the given usage model and the user pre-defined usage threshold.

We plan to develop a compositional reliability testing approach for component-based software. Our reliability testing approach relies on symbolic execution to generate automatically reliability test cases. We call such test cases statistical white-box test cases. We plan to encode the reliability test results of the single components building a component-based software into reliability testing summaries. The summaries can describe the pre- and postconditions of the provided component services, together with the reliability of the services and the exercised usage model. The reliability value of a service should be related to the exercised usage on the service. We want to re-use these summaries when components are integrated with other components to deliver added-value services. We want to avoid the re-testing of already tested usage scenarios.

The reliability of a software is also related related to the reliability of the deployment environment of the software (i.e., hardware, network, etc.,...). The PCM allows to model the deployment environment of a software. We will investigate how to integrate information about the reliability of the deployment environment in our reliability estimation approach.

In addition, we want to compare the estimate of black-box reliability models compared to white-box reliability estimation models. We believe that a model which considers the architecture of the software and the deployment model of the software should deliver a more reliable reliability estimate than black-box models.

References

1. Aho, A.V., Dahbura, A., Lee, D., Uyar, M.: An optimization technique for protocol conformance test generation based on uio sequences and rural chinese postman tours. Communications, IEEE Transactions on **39**(11), 1604–1615 (1991)
2. Baskiotis, N., Sebag, M.: Structural statistical software testing with active learning in a graph. In: Proceedings of the 17th international conference on Inductive logic programming, ILP'07, pp. 49–62. Springer-Verlag, Berlin, Heidelberg (2008)
3. Becker, S., Koziolek, H., Reussner, R.: The palladio component model for model-driven performance prediction. J. Syst. Softw. **82**(1), 3–22 (2009)

4. Belli, F., Budnik, C.J.: Minimal spanning set for coverage testing of interactive systems. In: Proceedings of the First international conference on Theoretical Aspects of Computing, ICTAC'04, pp. 220–234. Springer-Verlag, Berlin, Heidelberg (2005)
5. Chen, M.H., Lyu, M., Wong, W.: Effect of code coverage on software reliability measurement. Reliability, IEEE Transactions on **50**(2), 165–170 (2001)
6. Chevalley, P., Thevenod-Fosse, P.: Automated generation of statistical test cases from uml state diagrams. In: Computer Software and Applications Conference, 2001. COMPSAC 2001. 25th Annual International, pp. 205–214 (2001)
7. Csöndes, T., Kotnyek, B., Szabó, J.Z.: Application of heuristic methods for conformance test selection. European Journal of Operational Research **142** (2002)
8. Dulz, W., Zhen, F.: Matelo - statistical usage testing by annotated sequence diagrams, markov chains and ttcn-3. In: Quality Software, 2003. Proceedings. Third International Conference on, pp. 336–342 (2003)
9. Frank, A.: On kuhn's hungarian method – a tribute from hungary
10. Mills, H.D., Dyer, M., Linger, R.C.: Cleanroom software engineering (1987)
11. Musa, J.D., Okumoto, K.: A logarithmic poisson execution time model for software reliability measurement. In: Proceedings of the 7th international conference on Software engineering, ICSE '84, pp. 230–238. IEEE Press, Piscataway, NJ, USA (1984)
12. Poore, J., Mills, H., Mutchler, D.: Planning and certifying software system reliability. Software, IEEE **10**(1), 88–99 (1993)
13. Păsăreanu, C.S., Rungta, N.: Symbolic pathfinder: symbolic execution of java bytecode. In: Proceedings of the IEEE/ACM international conference on Automated software engineering, ASE '10, pp. 179–180. ACM, New York, NY, USA (2010)
14. Riebisch, M., Philippow, I., Götze, M.: Uml-based statistical test case generation. In: Revised Papers from the International Conference NetObjectDays on Objects, Components, Architectures, Services, and Applications for a Networked World, NODe '02, pp. 394–411. Springer-Verlag, London, UK, UK (2003)
15. Sayre, K.D.: Improved techniques for software testing based on markov chain usage models. Ph.D. thesis, University of Tennessee, Knoxville (1999)
16. Thelin, T.: Automated statistical testing suite for software validation (2004)
17. Thimbleby, H.: The directed chinese postman problem. Software: Practice and Experience **33**(11), 1081–1096 (2003). DOI 10.1002/spe.540. URL `http://dx.doi.org/10.1002/spe.540`
18. Thévenod-Fosse, P., Mazuet, C., Crouzet, Y.: On statistical structural testing of synchronous data flow programs **852**, 250–267 (1994)
19. Visser, W., Păsăreanu, C.S., Khurshid, S.: Test input generation with java pathfinder. SIGSOFT Softw. Eng. Notes **29**(4), 97–107 (2004)
20. Whittaker, J.A., Poore, J.H.: Markov analysis of software specifications. ACM Trans. Softw. Eng. Methodol. **2**(1), 93–106 (1993)

Part III
Platforms, Middleware and Integration

Chapter 11
Data Protection in the Cloud – The MimoSecco Approach

Jonas Lehner, Andreas Oberweis, and Gunther Schiefer

Abstract Cloud Computing is a technology with vast impact on IT systems. Costs can be significantly reduced through ondemand purchase of CPU time, memory and storage, offering high flexibility. The main reason to avoid cloud technology still is security. This leads to a lack of trust in cloud services. Most cloud providers secure their systems only against external adversaries by using firewalls and secure connections. Internal adversaries, however, remain a big threat in this scenario. Especially when using mobile devices as clients, usable security with a low performance impact remains a challenge. In this paper, we present concepts for using software as a service with mobile devices while guaranteeing a high level of data protection. MimoSecco uses an innovative encryption scheme and hardtoclone secure hardware to guarantee data protection. Top secret data is encrypted directly, processible confidential data is encrypted and fragmented by the database proxy and transferred to different servers. Contextbased access control makes the misuse of mobile devices for unauthorized data access difficult. These set of measures raises the privacy level of cloud computing significantly.

11.1 Introduction

11.1.1 Motivation

For many small and medium companies, cloud computing is very attractive. It offers scalability to cope with temporal high server loads. Building and maintaining

Jonas Lehner · Andreas Oberweis · Gunther Schiefer
Karlsruhe Institute of Technology, Institute AIFB
e-mail: {Jonas.Lehner,Andreas.Oberweis,Gunther.Schiefer}@kit.edu

Name of Second Author
Name, Address of Institute e-mail: name@email.address

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_11

an own computing center is very costly and can be avoided by using flexible cloud services. Therefore, costs can be cut while outsourcing maintenance to specialized cloud providers. Overall, this leads to an improved use of the available resources and lower costs. Especially in the last years, more and more mobile devices like smartphones or tablets are bought. While offering high mobility, they often lack in computing power. Cloud providers like Google respond by moving complex calculations or big data bases into the cloud, while using the mobile device only for the user interface and other simple tasks. This allows the devices to run longer while still offering the same service.

Once IT systems are outsourced into the cloud, the user loses control over his data. Using cloud services always leaks sensitive information, resulting in huge privacy issues but offers the possibility to access data from anywhere without the necessity of operating costly infrastructure for this purpose. Protection against external adversaries is not enough. Especially socalled insider attacks have to be taken into account when designing a cloud service that uses a public cloud. Usually administrators have lots of privileges that often allow them to read the client's data, e.g. for making backups. Data privacy laws, however, are impossible to hold for the big US cloud providers (Google, Amazon, Microsoft . . .), but also Chinese and Russian providers. In US, for example, authorized through the patriot act, the government can access all data of companies located in the US, even if stored outside the US. This also holds for data centers within the European Union.

11.1.2 Overview

MimoSecco aims at solving the cloud privacy dilemma by encrypting outgoing sensitive data before it is stored in the potentially untrusted cloud. We present a softwarebased solution with the option to integrate secure hardware for further security improvements. This article focuses on data protection in the terms of confidentiality, this means to avoid any information retrieval by a person which is not authorized to do so. Other security problems like data loss, availability, integrity and so on are not discussed here; most of them can be solved by system and/or data redundancy. A detailed description of related work and the technical architecture of the systems can be found on page 187. Sidechannel attacks to MimoSecco are discussed on page 35.

For a better understanding it is proposed to read page 187 first. This article focuses on additional mechanisms and legal aspects of MimoSecco.

11.1.3 Project Goals

The aim of the project MimoSecco is to protect data, processed by a cloud provider, better against internal attacks. Internal adversaries are all persons who have legal

access to the rooms or the IT systems of a cloud provider. This covers, for example, the administrator, the service technician and the cleaning personal. Protection against internal adversary includes automatically the protection against external adversaries. To do this, MimoSecco uses encryption technologies and separates the storage of the data from the processing. This reduces, for example, the risk, that data could be stolen by taking away a backup tape, because the provider who processes the data does not have a backup and the provider who stores the data cannot decrypt it.

By the integration of context based access control it is possible to grant data access only when specific context parameters are given. One example is that you are not allowed to access confidential technical documents while your calendar shows that you are on holiday.

11.2 Architecture

The MimoSecco architecture is derived from the outsourcing setting using cloud services (cf. Fig. 1). The hardware of the data owner is assumed completely trustworthy. The storage cloud is considered an honestbutcurious (passive) adversary. This means all queries are answered correctly, but the cloud provider tries to gain information from the queries and stored data. Between the trusted zone and the cloud storage provider is the semitrusted zone. It offers better privacy due to a more favorable jurisdiction than USbased providers or is provided by a known and trusted business partner. A private cloud is one example for this zone.

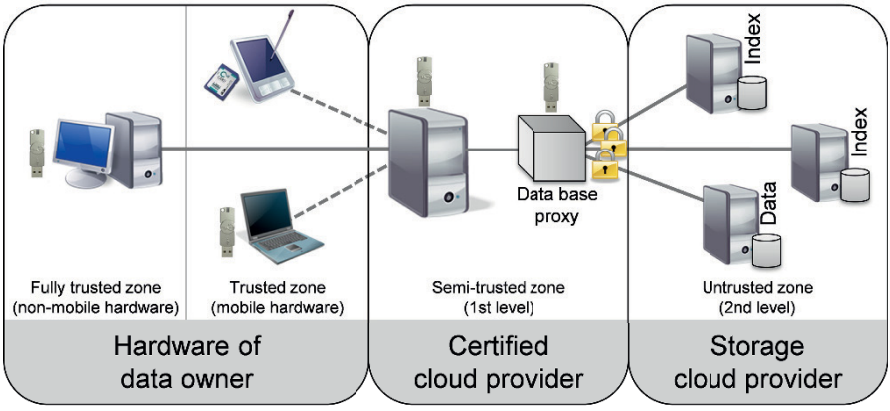


Fig. 11.1: Security zones in the MimoSecco scheme.

Most cloud security solutions only focus on transport encryption. MimoSecco uses authenticated end-to-end-encryption, too. Not only is the server authenticated,

like in common cloud-based services, but also the client. The certificates for this are generated within the company (fully trusted zone) and stored securely on a state-of-the-art smart card. We use the CodeMeter platform of WIBUSYSTEMS AG for secure storage. CodeMeter is a USB/SD/... dongle for software protection and offers password-protected storage.

MimoSecco also takes the potentially curious cloud provider into account and does not only rely on encrypted connections. Cloud providers pose high-value targets since they host services for multiple companies. If an adversary manages to corrupt such a provider, he can access the data of a potentially large number of clients. MimoSecco prevents damage in such a case by encrypting the data before uploading it to the second level cloud provider. The decryption is only possible with authentication by the user. The cryptographic keys required for encrypting and decrypting of data are distributed via the WIBUSYSTEMS CodeMeter License Central between the data owner's smart card and the cloud provider's smart card. MimoSecco only needs CodeMeter in the user's zone and for the certified cloud provider. Since a considerable contract between controller (user) and processor (certified cloud provider) is needed, the usage of a smartcard on the processor's server is a minor problem. Additionally, the ability to offer such an increase of security is a competitive advantage for the processor.

If a requested operation on the stored data is authenticated by the data owner and the Software as a service application (necessary for access control, see chapter 4) the smart card of the database proxy enables the proxy to use the cryptographic keys to encrypt or decrypt the data as needed.

11.3 Security Levels

One goal of MimoSecco is that a user retains control over his data. To ensure this, data has to be classified in different security levels, which is derived from mandatory access control (MAC, see chapter 4). Fig. 2 shows the different data channels for the security levels. Top secret data (black) is encrypted by the user's smart card on his device. This data can't be decrypted by the cloud provider. Therefor top secret data can't be processed by the cloud provider but can only be stored. The secret key for this type of data stays strictly at the user. Public data (gray) is not encrypted at all. This type of data is not worthy of protection or accessible easily from other source, e.g. published documents from the internet. This kind of data is not considered since it is not interesting in the MimoSecco context.

Confidential data (striped) is the mainly regarded kind of data in MimoSecco. To profit from the advantage of cloud computing these data has to be processible by the cloud provider who therefor needs to be able to decrypt the data, at least for a short period of time. The mechanism for decryption and encryption at this point and the technical implications are described on page 187.

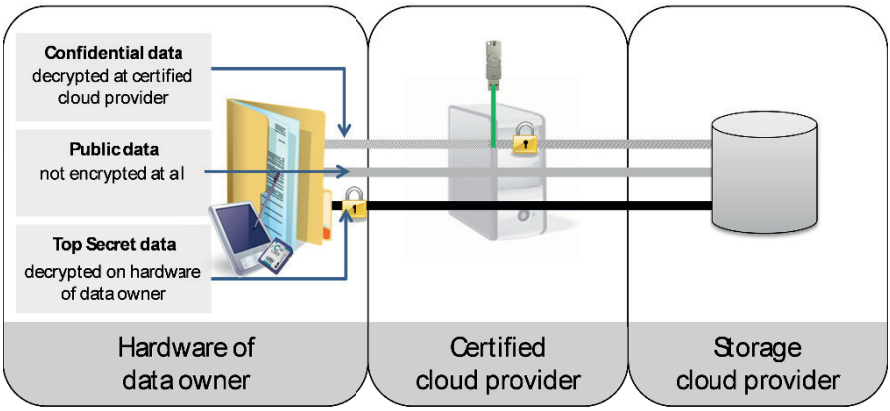


Fig. 11.2: Security levels of MimoSecco architecture.

11.4 Contextbased Access Control

A characteristic feature of the MimoSecco approach is the contextbased access control. The task of an access control in a computer system is to decide whether a user's request to execute a certain operation on a certain resource can be accepted [1]. Examples for resources are files, database objects or services (web services, print services etc.). Operations are commonly read, write and execute. The user is the active element and is also called subject, while the resource as the passive element is called object. Permission combines an object with an operation.

Access control models can be separated into generic models and application specific models. The latter are developed to be used in a certain application context like in a database management system or in a workflow management system whereas generic models don't have a defined context. There are three different relevant generic models for access control described in literature: mandatory access control (MAC), discretionary access control (DAC), and rolebased access control (RBAC).

11.4.0.1 Mandatory Access Control (MAC).

This model is mainly used in high security areas like military or intelligence and is based on the principle of different trust levels (e.g. public, confidential, secret or top secret) [2]. These trust levels are assigned to both users and resources. To access a certain resource a user has to have at least the same trust level (e.g. to access a secret resource, a user needs to be either secret or top secret).

11.4.0.2 Discretionary Access Control (DAC).

The principle of this access model is an access matrix, which contains the rights of each user to access each resource [3; 4]. A user, who creates an object (e.g. data) is initially entitled with all operation rights (e.g read, write or execute). He can then grant other users some or all of these rights. Users can be grouped to simplify configuration.

11.4.0.3 Rolebased Access Control (RBAC).

RBAC uses roles to define access rights [5]. Roles correspond with task descriptions for jobs or positions in the organization (e.g. marketing manager or board member). Access rights are never assigned to single users but always to roles.

In MimoSecco we took a DAC model as a basis since it is easy to implement yet still powerful enough to satisfy the project's needs. We then added a context aspect.

Context in MimoSecco is understood as dynamic information that is explicitly available during the runtime of the system and that can be used to adjust the application to the user's situation [6; 7]. Context-sensitivity means that permissions depend on context information, e.g. location, time or calendar data. For example, if a sales representative is located on the company area of a certain customer, he is not allowed to access data of a competitor.

To implement the contextsensitivity, we use so called context switches, which get context information as an input and make decisions based on rules defined by the company. As shown in Fig. 3 the context switches can affect the access control decision at two different points:

- Standalone permission: Permissions can be switched on and off according to actual context situation (e.g. reading of certain data is forbidden when using an unsecure wireless LAN connection)
- Type of object: In certain situations particularly sensitive data can be protected against some operations. I.e. no specific record is protected but an entire type of object (e.g. an employee can't access personal files while he is not within the company area). Fig. 3. Contextbased discretionary access control model used by MimoSecco

Some of the relevant context parameters we consider in MimoSecco are:

- Location: To protect sensitive data against access at places, which don't offer the required privacy, e.g. public airports, the mobile device can be located via GPS or interpretation of the IP address.
- Time: The context parameter time can be used to prevent access to sensitive data out of office hours. For global companies it can be necessary to determine the local time of an employee by using location data. In this case one context parameter is used as an input value for another context parameter.

- Calendar data: Calendar records can be used to determine whether an employee should have the permission to access certain data. For example, during his holidays an employee is not allowed to access any data.
- Type of authentication: Depending on the type of authentication a user can get different permissions. It is taken into account whether a user uses a username and password or if he is using a smartcard or hardware token.
- Type of connection: Different connection types can carry different risks that are regarded for the decision.

11.5 Points of attack

In MimoSecco we focus on socalled insider attacks. If we protect the data against adversaries inside the companies, this is also effective against adversaries from outside, therefore the arrows in the next Figure are striped grayblack. As shown in Fig. 4 there are four points of attack in our model. Fig. 4. Points of attack in the MimoSecco scheme (1) The 2nd level cloud provider is the host for all data. He only gets encrypted data (except the indices, where the keywords are in plain text). Therefore, nobody there can gain Information about the data. Any adversary can only learn something about the keywords which must be somewhere in the whole database, but he gets no information about the relationships. (2) Nowadays, the certified cloud provider in the semitrusted zone needs the decrypted data, to deal with it. (This can change in future). Therefore it is necessary to temporary decrypt the data in the memory. Since the data is not stored unencrypted in this zone and the keys are also only in memory during use, it is more difficult (but not impossible) to get some information here. It is feasible to implement some methods to control the datausage, which avoids the reading of the whole database. The TrustedCloudproject “Sealed Cloud” offers a solution to get more protection against an adversary at this point. For further information about Sealed Cloud see [[VERWEIS AUF PROJEKTÜBERBLICK Sealed Cloud]] (3) The contextbased access control and the usage of a hardware security token makes it significantly more difficult, to misuse the data on a mobile computer. There is no data stored on a mobile device, except cachingdata, which is automatically deleted after use. (4) The possibility to misuse the data by a person, who has the right to deal with it, is not changed through the MimoSecco model. Since this problem is independent from the usage of cloud computing it is not considered here.

By using cloud computing, the data is not persistent stored on the computers of the cloud user. Also the certified cloud provider does not store the data. Since there is no persistent data, there is nearly nothing to read, delete or modify for an adversary. The usage of a sufficient transport encryption is a matter of course and independent of wired or mobile connections.

11.6 Legal Aspects

One goal of MimoSecco is to enhance data security. Additionally there are a lot of use cases, where one uses personal data, for example in a CRM scenario where an enterprise stores information about their customers using a cloud service. This makes it necessary to deal with the questions concerning to personal data.

The upcoming General Data Protection Regulation (GDPR) from the European Union is still a draft. It can take several years since it becomes effective. Until then, the German Bundesdatenschutzgesetz (BDSG), which is based on the European Data Protection Directive 95/46/EC, is the main basis for dealing with personal data (in Germany). In the MimoSecco model, there is a certified provider that offers a cloud service as SaaS (semitrusted zone). This provider is a processor (Auftragnehmer) in the terms of the BDSG. According to the BDSG it is necessary to set out a contract in writing between the controller (Auftraggeber) and the processor. If the controller and the processor follow the full terms of § 11 BDSG, the processor is allowed to act with the personal data as if he is part of the controller. This proceeding is only allowed between an enterprise and a provider within the European Union (or some especially allowed countries).

For data storage, MimoSecco uses the database proxy to encrypt the data and store it within further cloud providers. These storage providers are set in the untrusted zone. This means, that the certified provider transfers the data to another party, the storage provider. According to the BDSG, the transfer of personal data is not allowed in most cases. In MimoSecco we have the transfer of highly encrypted data. The open question is, whether encrypted data is still personal data or not, since the transfer of personal data is mostly not allowed. To detail this question, it has to be determined, if encrypted data is pseudonymous or anonymous data. The aim of making data pseudonymous is to obscure the identity of personal data, but to have the possibility to re-identify the concerned person. The goal of anonymisation, however, is to make it impossible for everyone to re-identify the concerned person. According to this, anonymised data is no longer personal data, since pseudonymised data is still personal data. The Goal of anonymisation described here doesn't match with the definition which is given in § 3 (6) BDSG. The BDSG defines anonymisation as modifying of personal data in a way, that it is not possible to re-identify a concerned person or that it needs a disproportional effort of time, costs and manpower to do that. This last part of the definition matches with the result of a strong encryption, which is a strict version of pseudonymisation.

To answer the given question, the point of view is relevant. This has to be divided into a bifocal perspective (e.g. Stiemerling and Hartung [8]). From the subjective view of a provider, who receives encrypted data, this is no personal data at all, since he is not in possession of the decryption key (or any other possibility) to decrypt the data. It doesn't matter, if the original (unencrypted) data was personal data or not. From an objective view it is not possible to convert personal data through encryption into nonpersonal data. This is because there is still someone who has a key and the possibility to decrypt the data and to restore the personal information. Another open question is, if encrypted personal data is no longer personal data, when all keys to

decrypt it are destroyed. Since this is a separate question, it isn't discussed further here.

According to Stiemerling and Hartung is the mostly in Germany followed opinion, the subjective view. This means that encrypted personal data has not been treated as personal data by the provider, if he doesn't have the key at his disposal. There are still no court decisions which clarify this.

The GDPR (also directive 95/46/EC) follows the objective view. The given reason (23) at the beginning of the GDPR says: "The principles of protection should apply to any information concerning an identified or identifiable person. To determine whether a person is identifiable, account should be taken of all the means likely reasonably to be used either by the controller or by any other person to identify the individual. The principles of data protection should not apply to data rendered anonymous in such a way that the data subject is no longer identifiable." From this point of view it is not possible to convert personal data in nonpersonal data in any way (as long as a key to decrypt it exists). From this point of view, a 1st level provider (semitrusted zone) should use only storage providers (2nd level provider) within the European Union (or provider which have the same legal status according to the GDPR) and create in each case a relationship between the 1st level provider as a controller and a storage provider as processor according to the full terms of § 11 BDSG. This restricts the world wide cloud to a European cloud.

As a conclusion it could be said, that (following the dominant opinion in Germany) it is legal for a German provider to use the MimoSecco model for storing data on 3rd party clouds without following the rules of § 11 BDSG between the semitrusted and untrusted cloud provider. Since there are no court decisions about this, there is no legal certainty. For a careful, privacyaware enterprise, which wants predictability of legal decisions, the MimoSecco model should only be used in a European cloud as described above until further clarification of the legal situation.

11.7 Conclusion and Future Work

In this paper, we introduced a method to solve the cloud dilemma. With the MimoSecco transformation, outsourcing of sensitive information to the cloud is possible. Still, numerous challenges lie ahead. The legal aspects have to be clarified to give companies legal compliance. Other security problems like data loss, availability, integrity and so on have to be regarded. Some of them can be solved by system and/or data redundancy. This needs further research, which should also consider the performance of the whole system.

Another open question is the long time security of encrypted data. Nobody knows how long data, encrypted according to the state of the technology, is secured. How about a backup made by a storage cloud provider, which rests in a shelf for decades? Another planned feature is the use of the new German ID card (neuer Personalausweis, nPA) for authentication. This work has been partially funded by the Federal Ministry of Economics and Technology, Germany (BMWi, Contract No.

01MS10002). The responsibility for the content of this article lies solely with the authors.

References

1. S.D.C. Vimercati, S. Paraboschi, P. Samarati: Access Control: Principles and Solutions. Software — Practice and Experience, vol. 33, no. 5, 2003, p. 397-421.
2. M. Benantar: Mandatory-Access-Control Model. Access Control Systems: Security, Identity Management and Trust Models, 2006, p. 129-146.
3. B.W. Lampson: Protection. Operation Systems Review, vol. 1., no. 8, 1974, p. 18-24.
4. R.S. Sandhu, P. Samarati: Access control: principle and practice, Communications Magazine, IEEE , vol.32, no.9, Sept. 1994, p. 40-48.
5. R.S. Sandhu, E.J. Coyne, H.L. Feinstein, and C.E. Youman: Role-based access control models. Computer, 29(2), 1996, p. 38-47.
6. A.K. Dey: Understanding and Using Context. Personal and Ubiquitous Computing Journal, vol. 5, no. 1, 2001, p. 3-7.
7. G. Chen, D. Kotz: A Survey of Context-Aware Mobile Computing Research. Technical Report TR2000-381, Department of Computer Science, Dartmouth College, Hanover, NH, USA, 2000.
8. O. Stiemerling, J. Hartung: Computer und Recht: Zeitschrift für die Praxis des Rechts der Informationstechnologien, Köln, 1/2012, p. 60-68.

Chapter 12
Secure Database Outsourcing to the Cloud Using the MimoSecco Middleware

Matthias Gabel and Gerald Hübsch

Abstract Cloud Computing is a technology with vast impact on IT systems. Costs can be significantly reduced through on-demand purchase of CPU time, memory and storage. The main reason to avoid cloud technology is security. If data is outsourced to the cloud, control of the data is lost. This leads to a lack of trust in cloud services. Most cloud providers secure their systems only against external adversaries by using secure login mechanisms and secure end-to-end encrypted connections. Internal adversaries, however, remain a big threat in this scenario. Especially when using mobile devices as clients, usable security with a low performance impact remains a real challenge. In this paper, we present a cloud-storage technique for relational databases that is suitable for mobile devices. MimoSecco uses an innovative encryption scheme and hard-to-clone secure hardware to guarantee security. Relational databases are fragmented, encrypted and transferred to different servers. These measures make it infeasible for an adversary to extract the original data. The MimoSecco middleware acts as a proxy between the application and the untrusted cloud storage and transparently transforms SQL queries. We show the integration of the secure database proxy and the secure hardware tokens in a use case.

12.1 Introduction

For many small and medium companies, cloud computing is very attractive. It offers flexibility through scalability to cope with temporal high server loads. However, building and maintaining one's own computing center is very costly and can be

Matthias Gabel
Karlsruhe Institute of Technology (KIT)
e-mail: matthias.gabel@kit.edu

Gerald Hübsch
CAS Software AG
e-mail: gerald.huebsch@cas.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_12

avoided by using flexible cloud services. Therefore, costs can be cut while outsourcing maintenance to specialized cloud providers. Overall, this leads to an improved use of available resources and to lower costs. Especially in the last years, more and more mobile devices, especially smartphones and tablets have been bought. While offering high mobility, they often lack in power. Cloud providers like Google respond by moving complex calculations into the cloud, while using the mobile device only for the user interface and other simple tasks. This allows the devices to run longer while still offering the same service on the latest dataset.

But once calculations are outsourced into the cloud, the user loses control over his data. Instead of navigating offline with a common navigation device, the positions are uploaded to the provider, who finds the optimal route and supplies the client with the latest maps. In this cloud-based online navigation, the providers learn the client's positions for a long period of time. As in this example, using cloud services often leaks sensitive information, resulting in huge privacy issues.

Protection against external adversaries is not enough. Especially so-called insider attacks have to be taken into account when designing a cloud service that uses a public cloud. Usually administrators have lots of privileges that often allow them to read the client's data, e.g. for making backups. German and European data privacy laws, however, are impossible to hold for the big, US cloud providers (Google, Amazon, Microsoft, ...), but i.e. also Chinese and Russian providers, even if the data centers are located in Europe or even Germany. This is, because of the patriot act; the US government can access all data of companies located in the US, even if stored outside the US. This also holds for data centers within the European Union and violates EU privacy laws.

MimoSecco aims at solving the cloud privacy dilemma by encrypting outgoing sensitive data before it is stored in the potentially untrusted cloud. We present a software-based solution with the option to integrate secure hardware for further security improvements. In the MimoSecco project, CAS Software AG provides the sophisticated CRM called Pia and integrates the security components. WIBU-Systems provides CodeMeter, a state-of-the-art secure smart card for software protection. CodeMeter is used to secure the client and server code as well as secure storage of certificates. Karlsruhe Institute of Technology (KIT) is responsible for research and transfer of the theoretical results to help improve the security of the MimoSecco middleware.

Internal adversaries are all persons who have legal administrative or physical access the IT systems of a cloud provider. This covers for example the administrator, the service technician and the cleaning personnel. Protection against internal adversaries often includes protection against external adversaries. To do this, MimoSecco uses a modern encryption scheme and separates the storage of the data from processing. This reduces, for example, the risk, that data could be stolen by taking a backup drive, because the provider who processes the data does not have a backup and the provider who stores the data cannot decrypt it easily.

12.2 Related Work

Securing relation databases is one of the main goals of MimoSecco. In public research, this problem has been addressed before. Most authors look for a pragmatic approach resulting in a trade-off between security and performance. There are approaches mainly based on data distribution [2]. In this scenario, multiple servers are available. The servers are not aware of each other and consequently are not able to communicate with each other. The client then stores fragments of the original data on each isolated server. Even without encryption, server knowledge of the original dataset is limited because only a subset is learned. Agarwal et al. additionally use encryption – if and only if the privacy constraints cannot be met using data distribution alone.

There are approaches mostly based on encryption like Hacigümüs et. al [4, 5]. Each database row is encrypted independently. Furthermore, indices are generated for faster lookup of relevant rows. Not only are exact match queries ($attr = val$) possible, but also range queries ($val_1 < attr < val_2$). In this case, bucketization is used.

In 2009, Gentry presented the first fully homomorphic encryption (FHE) scheme [3]. Theoretically, this enables for semantically secure outsourcing to the cloud. The scheme offers homomorphic addition as well as homomorphic multiplication. Any algorithm can then be transformed into a circuit that is then evaluated using FHE. The cloud provider operates blindly on the encrypted data and yields the correct, encrypted result. He learns nothing about the input and output. Practically, this approach is currently not an option due to huge performance issues. The latest and best implementations are still orders of magnitude slower than just downloading all encrypted data to the client, decrypting, processing and encrypting it locally and finally uploading it again.

The MimoSecco scheme [1] was introduced in 2011 by Achenbach et al. This work is mainly based on the ideas of the original MimoSecco paper. For this reason, the work introduced in [1] is explained later in more detail.

Cumulus4j [6] was proposed by Huber et al. in 2013. Although Cumulus4j uses the same idea, Cumulus4j is more focused on performance than MimoSecco. As a result, the client uses a key server to send keys to the server for faster processing. The key is only stored in volatile memory and a lot of keys are used for encryption. Security is lower than with the MimoSecco scheme since key leakage is possible.

CryptDB [8, 9] is another practical approach to outsourcing relational databases and does not rely in an index. CryptDB uses a concept called onions and is managed monolithically by a proxy acting as an adapter between the user and the storage back-end. Each attribute in a relational table is initially probabilistically encrypted. This is the most secure (Ind-CCA secure) state. If certain queries for this attribute, e.g. `ORDER BY` are issued, layers of the onion are peeled off, resulting in another, less secure onion. CryptDB uses a novel scheme for order preserving encryption (OPE) [7] that leaks no information about the data besides order and thus allows for sorting encrypted data securely. The implementation achieves a low overhead and can be used with common relational database systems. The main drawback of

CryptDB is the security guarantee. No hard guarantee is offered to the client. It is only guaranteed that the untrusted server gets only the information that is necessary to process the query. This may cause every attribute to be reduced to the plain text in the worst case. Also, peeling off layers cannot be reversed, so a single query is sufficient to lower the security forever. Monomi [10] combines CryptDB and homomorphic encryption.

12.3 Architecture

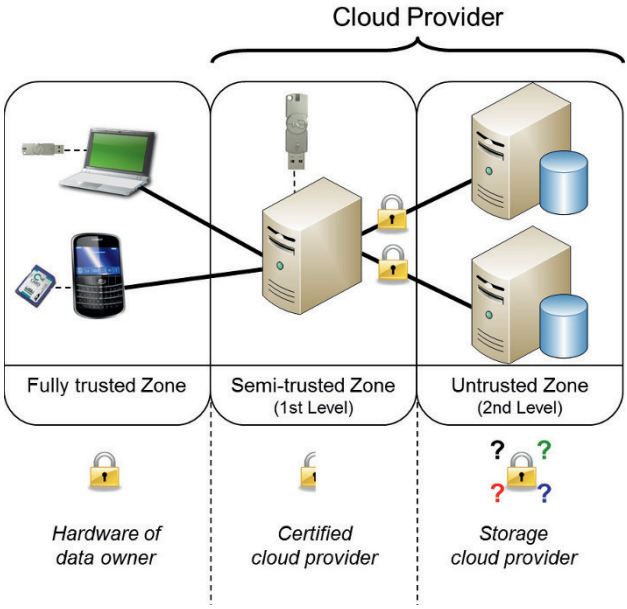


Fig. 12.1: The three security zones in the MimoSecco scheme.

The MimoSecco architecture is derived from the outsourcing setting using cloud services (cf. fig. 12.1). The client platform in the fully trusted zone is assumed completely trustworthy. If the client's security could not be guaranteed, most of the security mechanisms would be futile anyway. The cloud in the untrusted zone is considered an honest-but-curious (passive) adversary. This means all queries are answered correctly, but the cloud provider tries to gain information from queries and uploaded data. Between the fully trusted local zone and the cloud storage provider is the semi-trusted zone. It offers better privacy due to a more favorable jurisdiction than providers located in the US or is provided by a well-known and trusted business partner. A private cloud is one example for this zone.

Fig. 12.2: WIBU-Systems CodeMeter tokens for software protection and secure certificate storage. MimoSecco also takes the potentially curious cloud provider into account and does not only rely on encrypted connections. Cloud providers pose high-value targets since they host services for multiple companies. If an adversary manages to corrupt such a provider, he can access the data of a potentially large number of clients. MimoSecco prevents damage in such a case by encrypting sensitive information before uploading it to the cloud provider..

Nowadays, the number of sold mobile devices grows rapidly and devices with inferior speed supersede notebooks. As a consequence, there are lots of devices that cannot handle encryption and decryption sufficiently fast. In case of slower devices like smartphones, time-consuming operations should be outsourced to a private cloud in the untrusted zone. Most cloud security solutions only focus on transport encryption. Of course, MimoSecco also establishes authenticated end-to-end-encryption. For this, we use TLS (Transport Layer Security), which enables encryption and key agreement for privacy. Integrity is guaranteed by HMACs used in TLS. X.509 certificates for the server and client result in a two-way authentication. Not only is the server authenticated, like in common cloud-based services, but also the client. The certificates are stored securely on a state-of-the-art smart card. We use the CodeMeter platform of WIBU-Systems for secure storage. CodeMeter is a hardware dongle (cf. fig. 12.2) for software protection and offers password-protected storage. CodeMeter tokens are available in the micro-USB format and are used in the client's smartphone and other mobile devices.

12.3.1 Concept

12.3.1.1 Secure Database Proxy

When encrypting a database as a whole, privacy is easily achieved. This trivial approach, however, makes it impossible to evaluate queries on ciphertexts. Queries can only be executed by downloading and decrypting all data, which makes this approach horribly slow and infeasible. We seek a way to efficiently execute most SQL queries on encrypted data.

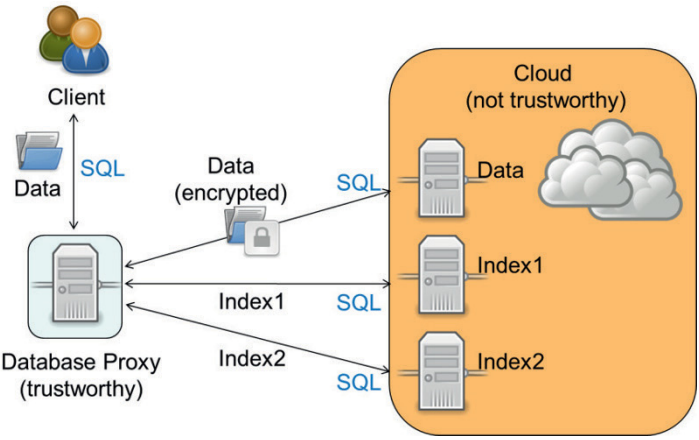


Fig. 12.3: Deployment using the secure database proxy. SQL is used as the interface to both the client and back-end side. The database proxy is deployed in the semi-trusted zone. It encrypts and distributes the data transparently for the client to the untrusted storage cloud.

The MimoSecco transformation [1] takes the original data and generates two classes of tables from it: the encrypted data table and the index tables. SQL-like exact match ($attr = val$) queries can be efficiently answered due to the index. The main intuition of this transformation is to hide attribute relations rather than the attributes themselves. The concept is best explained using a small example. Consider an exemplary database as depicted in fig. 12.4.

Row	Name	Surname	City
1	Jacob	Anderson	New York
2	Sophia	Anderson	Chicago
3	Emma	Connor	Los Angeles

Fig. 12.4: A simple example of a database; *original_data* before transformation. The *Row* column is introduced by the MimoSecco scheme because rows are referenced within the index tables.

The index tables *index_name* and *index_surname* are created from the base table *original_data* by creating a new table for each attribute containing each attribute value in the first column, and a list of its occurrences in the second column. Then the lists in the second column are encrypted. Figure 12.5 depicts the resulting index tables. The *encrypted_data* table is created by encrypting the base table per row (cf. fig. 12.6).

Keyword	RowList
Emma	Enc(3)
Jacob	Enc(1)
Sophia	Enc(2)

(a)

Keyword	RowList
Anderson	Enc(1 ‖ 2)
Connor	Enc(3)

(b)

Fig. 12.5: The index tables for the table depicted in fig. 12.4: *index_name* 12.5a and *index_surname* 12.5b. The keyword column is not encrypted; the row numbers are hidden using probabilistic encryption. An index table for an attribute can be omitted, which is the case for the attribute city in this example. Reversible concatenation is indicated by the symbol ‖ (the implementation uses escaping to solve this problem).

Row	EncData
1	Enc(Jacob ‖ Anderson ‖ New York)
2	Enc(Sophia ‖ Anderson ‖ Chicago)
3	Enc(Emma ‖ Connor ‖ Los Angeles)

Fig. 12.6: The encrypted data table *encrypted_data*. The original rows are stored as a probabilistically encrypted string. The String contains all attributes of one row.

Although the table data is now encrypted, efficient queries are still possible in most cases. Consider the following query:

```
SELECT * FROM original_data WHERE Name = 'Sophia' AND
Surname='Anderson'
```

In order to execute this query, the database proxy first queries the index table *index_name*:

```
SELECT * FROM index_name WHERE Name = 'Sophia'
```

The query yields the result Enc(2). In parallel, the database proxy queries the other index table table *index_surname*

```
SELECT * FROM index_surname WHERE Surname = 'Anderson'
```

which results in Enc(1 ‖ 2). The database proxy then decrypts the results and queries the *encrypted_data* table with the intersection $\{2\}$ of the two result sets $\{2\}$ and $\{1,2\}$:

```
SELECT * from encrypted_data WHERE Row = '2'
```

Due to the index tables, only the relevant rows of the data table are retrieved. This enables for search in sub-linear time, which is a common requirement for efficient databases.

If the query contains some form of aggregation, e.g.

```
SELECT AVG(Age) FROM ...
```

or

```
SELECT ... GROUP BY City
```

further processing is necessary. The relevant rows are then inserted into a locally hosted database and the aggregation part of the query is then applied. This results in the correct query and the local database is then emptied.

12.3.2 Security Improvements

In this section we will show some weaknesses of the MimoSecco scheme and present solutions for them.

12.3.2.1 Reducing Information Leakage through keywords

The attackers knowledge about information in the original table mainly comes from three sources:

- unencrypted (plain text) entries in the index tables
- knowledge of a part of the original query
- side channels: unsorted tables, access patterns, other statistics, ...

We will mainly focus on unencrypted index table entries in this paper since side channel are covered in separate work.

The unencrypted entries in the index tables do not violate the Ind-ICP security [6]. Informally speaking, Ind-ICP is achieved by hiding attribute relations of the original data. Since the disclosure of plain text fields from the original table does not reveal any higher information – given the attacker has no additional background knowledge – the Ind-ICP property holds for this transformation as proven in [6].

Some entries, however, are of very sensitive nature and leakage of a single field already poses a big threat. Credit card numbers are an example for critical data that still needs to be queried in some cases. MimoSecco can suppress index generation to hide sensitive data. But this leaves the problem of efficiently looking up a specific record. In the future, MimoSecco will allow for using hashed or encrypted index keywords. If a sufficiently long random salt and a secure hash function are used, keywords are hidden from the cloud provider. Because salting and hashing only emulates one-way deterministic encryption, it's probably preferable to use a secure deterministic symmetric encryption. Instead of finding the keyword w with the query `SELECT ... WHERE attr = w`, the database proxy simply queries `SELECT ... WHERE attr = Enc(w)`. The ciphertext $\mathsf{Enc}(w)$ is generated by the database proxy in the semi-trusted zone. MimoSecco uses AES-CBC for probabilistic encryption and will also use AES for deterministic encryption of index keywords.

This security improvements result in two minor disadvantages. First, additional time overhead is introduced by the encryption, as well as space overhead determined by the block size of the cipher. Second, approximate searches as SQL's *LIKE* queries, e.g.

```
SELECT * FROM table WHERE Surname LIKE 'Ma%er'
```

are not supported anymore. This is because hashing and encryption totally break the plain text's structure. Even though the symmetric key is hidden from the cloud provider, Ind-CCA security cannot be achieved with a deterministic cipher. The use of deterministic encryption only leaks duplicates because $\mathsf{Enc}(w)$ is always the same. But in each index every keyword is unique as re-occurring keyword hits are grouped in the same encrypted field, so this problem is mitigated. Depending on the scenario's security requirements, a different key can be used for each index, which makes the correlation of keywords between different index tables impossible.

12.3.2.2 Minimizing Leakage by Data Distribution

In this section we assume the keyword indices to be unencrypted to allow for LIKE-queries. Consider the following query:

```
SELECT * FROM table WHERE Name = 'Sophia' and Surname =
'Anderson'
```

From the query itself the server learns that someone with the name *Sophia Anderson* is assumed to be in the database. Note, that if approximate searches are necessary in some cases, the name and surname index must remain unencrypted. From the answer the cloud provider learns something too. If no lines are returned, the server can infer that no *Sophia Anderson* was in the database. Only if the result is not empty, there may or may not be a hit. There may be a *Sophia* and someone named *Anderson* in the database but no *Sophia Anderson*. Because this information is only found out by the adapter when intersecting the result sets, this information is hidden from the cloud provider. Still, the provider can see how long each returned keyword result is from the static data stored in the index tables. By the length of the encrypted ID list in the index the cloud provider may infer that a longer ciphertext (meaning there are more rows this keyword relates to) infer a higher probability that the query result is not empty.

If certain keywords have to be stored in plain text, correlation between keyword of different attributes can be reduced. This is easily achieved by storing the attributes on different servers. In this case, each server only learns a part of the original query.

12.3.3 Key Management and Distribution

The secure handling and storage of the cryptographic keys used for the database encryption is absolutely crucial for MimoSecco. A key management solution for MimoSecco must protect the cryptographic keys used in the semi-trusted zone against theft, manipulation and illegal creation. Stolen or copied keys would allow adversaries to compromise the encrypted data stored in the untrusted zone. Manipulated or illegally created keys could be used in a similar way.

Multi-tenancy must also be supported, because it is a typical feature of cloud applications. In multi-tenant scenarios, it is highly desirable to use one unique key per tenant for several reasons. Different keys generally enhance the isolation between the tenants' data pools. If one tenant key must be disclosed for legal reasons, or if it stolen, only the data of one tenant is compromised. All other tenants remain untouched. If a tenant loses confidence in the semi-trusted zone for whatever reason, access to the encrypted data can be withdrawn by deleting the tenant's key.

The operator of a cloud application is responsible for the generation and the management of all tenant keys and must protect them against loss. The key management solution must therefore provide an option for the save transfer of tenant keys between the semi-trusted zone (typically a data center) and the premises of the operator.

MimoSecco uses the WIBU-Systems CodeMeter License Central as the key management system. Cryptographic keys are protected against theft by programming them into protected storage areas of WIBU CodeMeter smart cards. Credentials in the form of PIN codes must be presented to read them. Manipulation and illegal creation of keys is prevented by a system of firm codes. Firm codes are unique identifiers issued by WIBU-Systems for every customer. In our case, the customer is the operator of the cloud application. The firm code is programmed into a Firm Security Box smart card by WIBU-Systems and handed over to the customer. The Firm Security Box is required to program the protected storage areas of CodeMeter smart cards using the assigned firm code. Without physical possession of the Firm Security Box, an adversary is therefore unable to create new keys or manipulate existing keys.

Under one firm code, multiple unique product codes can be independently created, modified and deleted. Product codes are identifiers which are associated a protected storage area. In MimoSecco, the multi-tenancy key management is implemented simply by mapping one product code to one tenant. CodeMeter smart cards are programmed by generating licenses that can be serialized into files. The licenses contain the firm code, the product code and the content (i.e. the key) to be programmed into the protected storage area. These license files can be archived to protect them against loss. They can also be easily transferred over a secure channel between the creator and the data center where the smart card to be programmed is located. Keys can be removed from the smart card by creating a license that overwrites the programmed key.

12.3.4 Application Server Integration

The MimoSecco middleware has been integrated into CAS Open, a Java application server for Software-as-a-Service solutions hosted in cloud environments. CAS Open supports multi-tenancy by mapping tenant data to individual, per-tenant databases. For encrypted data stored in the untrusted zone, an individual and unique AES key exists for every tenant. CAS Open has a classical three-tier (UI, business logic, data)

architecture and uses a relational data model to store business objects. It supports web frontends for thin clients and offers webservice interfaces for mobile applications. In the following, the extended server architecture developed in MimoSecco is presented.

Figure 7 depicts the extended server architecture. It also shows how the three tiers map to the zones defined in MimoSecco. The end devices are located in the trusted zone of the user. TLS is used to secure the connection to the server. The UI and business logic tiers are placed in the semi-trusted zone, where the protected data from the cloud storage is transiently available in plain text for processing and presentation.

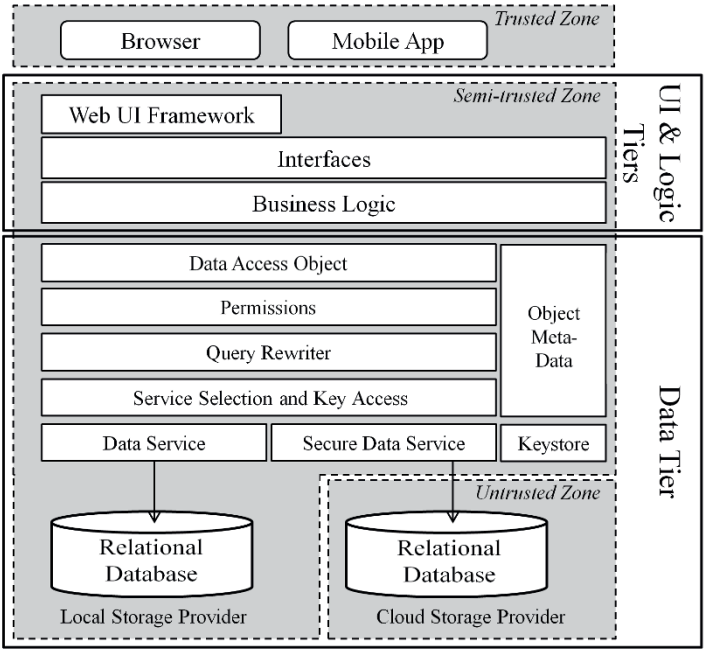


Fig. 12.7: Application server architecture integrating the secure database

The data supply is implemented by the data tier, which was modified to utilize the MimoSecco middleware. The modifications are isolated in the components below the Data Access Object. This is important because the interface compatibility of the data tier towards the logic tier is maintained. Note that the data tier uses a dual stack consisting of a local storage provider in the semi-trusted zone and a cloud storage provider in the untrusted zone where encrypted data is stored. The dual stack design was motivated by the following aspects. The MimoSecco approach requires a local database as a cache for processing queries that contain aggregations. The application server must furthermore manages meta data such as data object permissions and information about the structure of the data objects. Storing this

information in the untrusted zone is cumbersome. Unlike index tables, permission information is hard to distribute over different hosts. External adversaries could also try to manipulate permissions or try to identify valuable data records by analyzing the defined data object permissions. Storing the meta data in encrypted tables is also not an option, because there is at least one set of permissions for each data object. Even a minimal overhead for the decryption would therefore additionally reduce the overall performance of the server. The local storage can also be used to store data objects for which security can be traded against performance and availability.

The Data Access Object component supports classical create, read, update and delete operations for single data objects as well as SQL queries for mass data operations. CAS Open uses CAS SQL, a SQL language similar to the MySQL dialect. It adds extensions for the handling of object relations and special filter conditions.

The Permissions component is responsible for user level permission checks. The effective data object permissions are provided by the Object Metadata component which retrieves them from the local storage.

The Query Rewriter performs two conversions on CAS SQL queries. The first conversion extracts the tenant information from the invocation context and determines the name of the tenant's database using the Object Metadata component. For example, the query

```
SELECT * from ADDRESS WHERE TOWN='Karlsruhe'
```

would be transformed into

```
SELECT * from tenant.ADDRESS WHERE TOWN='Karlsruhe'.
```

Queries referencing databases different from the one allowed in the invocation context are rejected. The second conversion transforms the CAS SQL query into the SQL dialect of the target database. This step makes it possible to support different databases. In case of MimoSecco, the local database is a MySQL database, while the MimoSecco middleware uses PostgreSQL.

The target database is the database being queried, i.e. either the local storage or the cloud storage. It is determined by the Service Selection and Key Access component for the combination of the tenant name and the table name. If the target database is the cloud storage, the Service Selection and Key Access component creates a credential to read the tenant's cryptographic key from the keystore. The necessary information to create the credential is provided by the Object Metadata component. In MimoSecco, the credential is a triple (PIN, firm code, product code). The keystore is a WIBU CodeMeter smart card. Firm code and product code together represent the address of the AES key on the smart card. The smart card only reveals the AES key stored under this address if the correct PIN is presented. The component rejects queries for the cloud storage if no credentials exist.

Finally, the query is passed together with the credentials to the Secure Data Service. This component wraps the secure database adapter. It decrypts the result of the query with the AES key returned by the keystore for the provided credentials. The query fails if no key is returned from the keystore.

12.4 Use Case Example

This section presents the complex use case scenario that has been defined to validate the practicability of the MimoSecco approach for real world applications. The use case is centered around an extended relationship management (xRM) application that supports the planning, installation and operation of solar plants worldwide. The requirements for the use cases are provided iPLON, an industrial partner that plans, installs and maintains solar plants in Europe and India. The prototype of the xRM application integrates with the solar plant monitoring infrastructure operated by iPLON. The software platform of the prototype is the extended CAS Open server presented in sect. 12.3.4.

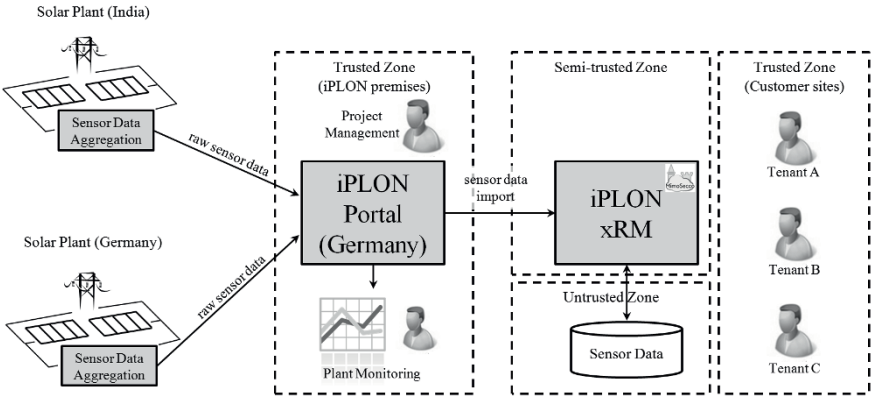


Fig. 12.8: Overview of the iPLON use case

Figure 8 gives an overview of the iPLON use case. The iPLON xRM covers two major requirements. It is used to manage data related to iPLON solar plant projects in the planning and construction phase, such as the contacts of the involved stakeholders, technical documentation, and contracts. The latter often have the form of electronic documents that can be secured by state-of-the-art document encryption techniques and are therefore not in the focus of the following discussion. Stakeholder contacts are often publicly known and can therefore be stored in the semi-trusted zone without additional protection.

The interesting part of the iPLON xRM in terms data security in the cloud is the monitoring of solar plants that are already in operation. The monitoring data have economic value, because they document current and past solar energy production output and the effectivity of the plants. At the same time, large quantities of monitoring data (around five gigabyte per plant and year) are produced. This makes the use of cheap cloud storage attractive from an economic perspective if the data can be adequately protected.

A typical solar plant is equipped with multiple sensors for environmental data (ambient temperature, irradiation), the current energy output, and the total amount of energy produced by the plant over a period of time. The measured values are collected by a sensor data aggregation component installed on the plant's site. It assigns time stamps and sensor IDs before it transmits the collected raw sensor data over a secure channel to the iPLON portal (cf. Figure 8) in regular intervals of several minutes. IPLON uses this data to monitor the plants' status for anomalies that require immediate actions like on-site maintenance.

The iPLON xRM automatically imports sensor data sets from the iPLON portal once per day or upon request by the user. These imported sensor data sets are quintuples of the form

```
(timestamp, solar radiation, ambient temperature,
total energy, AC Power).
```

They describe the most important parameters of the solar plant and are therefore well suited for archiving purposes and allow plant performance analysis over historical data. The import logic of the application server converts them into data objects and stores them in the encrypted database. The import logic is configured with the mapping between plant identifiers and tenants. If, for example, plant 'A' maps to 'Tenant A', the import logic will store the sensor data objects of this plant in the context of tenant A. When this context is set, the data tier will encrypt the data with the cryptographic key of tenant A as described in sect. 12.3.4.

IPLON can not only use the sensor data in the xRM system for internal purposes. It can also create individual user accounts for its customers, i.e. plant owners, or for maintenance staff. They can then use the sensor data analysis functions provided by the iPLON xRM for the archived sensor data. In the prototype, the calculation and visualization of the plant performance ratio has been implemented as an example analysis technique (cf. figure 8). It is a key indicator for the health status of a solar plant, as it is the ratio between the actual energy output of the plant and its theoretical output under the measured environmental conditions. If it deteriorates over time, typically maintenance is required to clean the plant's solar panels or to exchange aged modules.

12.5 Conclusion and Future Work

In this paper, we introduced a method to solve the cloud dilemma. With the MimoSecco transformation, outsourcing of sensitive information to the cloud is possible. Due to the index data, most of the important SQL queries can be efficiently processed. By hiding relations among attributes, MimoSecco achieves privacy and only leaks the existence of attributes (only if an index is created).

Still, numerous challenges lie ahead. Support of the full SQL standard can be achieved by improving the database proxy and transformation process. Then, Mi-

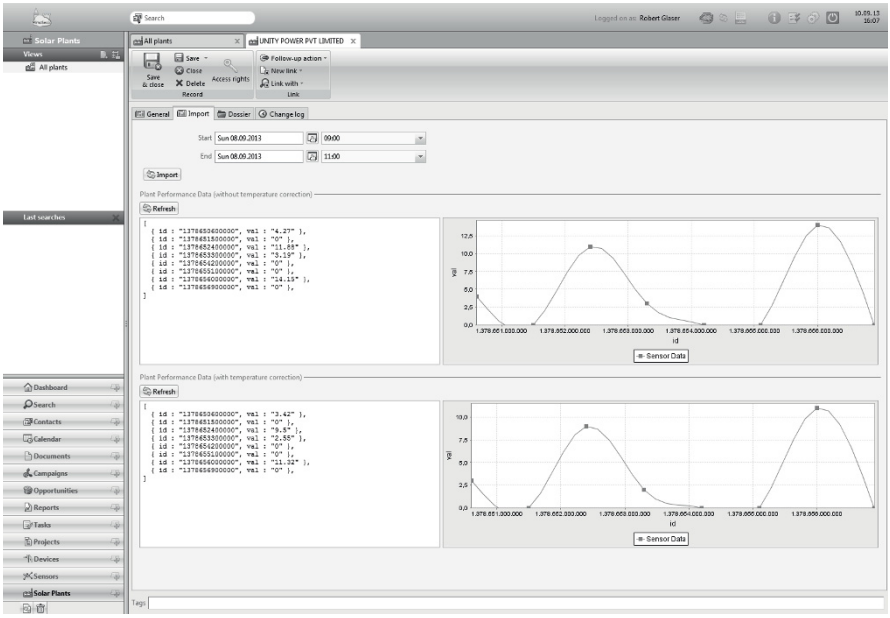


Fig. 12.9: Screenshot of the iPLON xRM prototype

moSecco can be used almost transparently for the client with only minimal restrictions and modifications to the application. Performance can be improved by making the transformed data scheme more elaborate. Consistency can also become a challenge if data is distributed to different servers and one of the servers does not respond. Then, the database proxy – after detecting the inconsistency – has to query the non-responding server again to have the current revision of the distributed dataset. If such an update is impossible, a global rollback has to be performed. Another planned feature is the use of the new German ID card (neuer Personalausweis, nPA) for authentication.

This work has been partially funded by the Federal Ministry of Education and Research, Germany (BMWi, Contract No. 01MS10002). The responsibility for the content of this article lies solely with the authors.

References

1. Achenbach, D., Gabel, M., Huber, M.: Mimosecco: A middleware for secure cloud storage. In: Improving Complex Systems Today, pp. 175–181. Springer (2011)
2. Aggarwal, G., Bawa, M., Ganesan, P., Garcia-Molina, H., Kenthapadi, K., Motwani, R., Srivastava, U., Thomas, D., Xu, Y.: Two can keep a secret: A distributed architecture for secure database services. CIDR 2005 URL `http://ilpubs.stanford.edu:8090/659/`

3. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st annual ACM symposium on Theory of computing, STOC '09, pp. 169–178. ACM, New York, NY, USA (2009). DOI 10.1145/1536414.1536440. URL http://doi.acm.org/10.1145/1536414.1536440
4. Hacigümüs, H., Iyer, B., Li, C., Mehrotra, S.: Executing SQL over encrypted data in the database-service-provider model. In: Proceedings of the 2002 ACM SIGMOD international conference on Management of data, pp. 216–227. ACM (2002)
5. Hacigümüs, H., Iyer, B., Mehrotra, S.: Providing database as a service. In: ICDE '02: Proceedings of the 18th International Conference on Data Engineering, p. 29. IEEE Computer Society, Washington, DC, USA (2002)
6. Huber, M., Gabel, M., Schulze, M., Bieber, A.: Cumulus4j: A provably secure database abstraction layer. In: CD-ARES Workshops, pp. 180–193 (2013)
7. Popa, R.A., Li, F.H., Zeldovich, N.: An ideal-security protocol for order-preserving encoding. In: IEEE Symposium on Security and Privacy, pp. 463–477 (2013)
8. Popa, R.A., Redfield, C.M.S., Zeldovich, N., Balakrishnan, H.: Cryptdb: protecting confidentiality with encrypted query processing. In: Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, SOSP '11, pp. 85–100. ACM, New York, NY, USA (2011). DOI 10.1145/2043556.2043566. URL http://doi.acm.org/10.1145/2043556.2043566
9. Popa, R.A., Redfield, C.M.S., Zeldovich, N., Balakrishnan, H.: Cryptdb: processing queries on an encrypted database. Commun. ACM **55**(9), 103–111 (2012)
10. Tu, S., Kaashoek, M.F., Madden, S., Zeldovich, N.: Processing analytical queries over encrypted data. PVLDB **6**(5), 289–300 (2013)

Chapter 13
SensorCloud: Towards the Interdisciplinary Development of a Trustworthy Platform for Globally Interconnected Sensors and Actuators

Michael Eggert, Roger Häußling, Martin Henze, Lars Hermerschmidt, René Hummen, Daniel Kerpen, Antonio Navarro Pérez, Bernhard Rumpe, Dirk Thißen, Klaus Wehrle

Abstract Although Cloud Computing promises to lower IT costs and increase users' productivity in everyday life, the unattractive aspect of this new technology is that the user no longer owns all the devices which process personal data. To lower scepticism, the project SensorCloud investigates techniques to understand and compensate these adoption barriers in a scenario consisting of cloud applications that utilize sensors and actuators placed in private places. This work provides an interdisciplinary overview of the social and technical core research challenges for the trustworthy integration of sensor and actuator devices with the Cloud Computing paradigm. Most importantly, these challenges include i) ease of development, ii) security and privacy, and iii) social dimensions of a cloud-based system which integrates into private life. When these challenges are tackled in the development of future cloud systems, the attractiveness of new use cases in a sensor-enabled world will considerably be increased for users who currently do not trust the Cloud.

13.1 Introduction

Recent advances in human/nonhuman interaction networks, so-called Cyber-Physical (Social) Systems (CPS) which integrate computational, physical, and social pro-

Michael Eggert · Roger Häußling · Daniel Kerpen
RWTH Aachen, Lehrstuhl für Technik- und Organisationssoziologie
e-mail: {meggert,rhaeussling,dkerpen}@soziologie.rwth-aachen.de

Martin Henze · René Hummen · Dirk Thißen · Klaus Wehrle
RWTH Aachen, Communication and Distributed Systems
e-mail: {martin.henze,rene.hummen,thissen,wehrle}@comsys.rwth-aachen.de

Lars Hermerschmidt · Antonio Navarro Pérez · Bernhard Rumpe
RWTH Aachen, Software Engineering
e-mail: {hermerschmidt,perez,rumpe}@se-rwth.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_13

cesses [11, 27], continue to blur the boundaries between the physical and digital world [21, 19] for a vast range of applications like health care, mobility, infrastructure, and manufacturing [30]. These CPS often generate a huge amount of data that has to be stored and processed. Additionally, it is desirable to aggregate data from multiple sources in order to generate higher value information.

The central approach of Cloud Computing with seemingly infinite computing and storage resources that can be scaled elastically has the potential to meet the requirements of CPS [21, 28]. From the service developer's perspective, this ease of use is largely to be attributed to the separation of responsibilities and the flexible, on-demand billing model introduced with Cloud Computing. More precisely, one or more providers are made responsible for maintaining and scaling the central computing and network infrastructure to the demand of their resource consumers. As a result, service developers are unburdened from managing individual compute units and the reliable interconnection between each other as well as with the Internet. However, to fully utilize the Cloud Computing paradigm, distributed programming models must be employed in developing a service. This renders the development of clouds a challenging task and demands for further development tool support.

Furthermore, the involvement of multiple providers in the provisioning of clouds opens new vectors of attacks against potentially sensitive data that is stored or transmitted within the cloud environment. As such, Cloud Computing stands in stark contrast to previous IT outsourcing activities that involve a single clearly defined entity offering its IT services. Cloud Computing, however, inherently implies multi-tenancy of the offered system. Each tenant (e.g., a service provider or an administrator of the cloud infrastructure provider) may be interested in another tenant's data. This makes Cloud Computing highly challenging from the security perspective.

Finally, clouds are commonly consumed by non-expert users. Hence, the presentation of data as well as the underlying security model must be intuitive and understandable by the common user. Otherwise, adoption barriers prevent the usage of a technically sound, highly secure, but unusable system. Thus, user acceptance testing is a major concern for massive-scale services.

This work is structured as follows: After this introduction, we first present the vision and scenario of the SensorCloud project. Afterwards, we identify and discuss research challenges when outsourcing storage and processing of sensor data to the cloud. We identify three core research challenges: i) ease of development, ii) security and privacy, and iii) social dimensions of a cloud based system which integrates into our private life. Thus, we focus on these challenges and discuss approaches to address them. These approaches will help to develop cloud solutions that users can confide there sensitive sensor data.

13.2 SensorCloud Vision and Scenario

The SensorCloud project, incorporating partners from industry and academia, is a member of the project cluster Trusted Cloud which is funded by the German Fed-

eral Ministry of Economics and Technology. Trusted Cloud aims to develop and evaluate innovative, robust, secure and law conform Cloud Computing solutions. Pilot projects will apply these solutions to industrial demands and thus demonstrate the benefits and trustworthiness of Cloud Computing. These pilots will especially address the current skepticism towards Cloud Computing observable in small and medium enterprises and their consumers.

The goal of SensorCloud is to develop a cloud-based platform that integrates basically every kind of distributed internet-connected sensor and actuator devices. Such devices may be located at a variety of places like personal homes, offices, cars, industrial plants, or outdoor areas. As scarcity of limited resources gets an ever growing issue, the controlled use of these resources gets a success factor for modern industry and future smart homes. In the SensorCloud project, we use the smart home scenario to capture requirements of home users and to validate assumptions necessary when designing technical solutions. The smart home scenario foresees a not to far future, where all devices, handles, and general interaction points residents or visitors have with a house are equipped with a sensor resp. actuator. A house with sensors and actuators is not smart in itself but enables an IT system to have a rich interaction with the users of the house. In order to make the house smart, SensorCloud serves as a central hub to which such devices can be connected. It aggregates their data and controls their functions. Thereby, it manages devices and data from different domains, places and owners at the same time. On top of its base of devices and data, SensorCloud provides an extensible and flexible service platform that is populated by "apps" from third-party developers. These apps leverage the potential of the centrally aggregated devices and data, providing a rich, diverse and innovative supply of end user functionality. Bringing together owners and suppliers of internet-connected sensors and actuators, developers of services working on these devices, and the provider of the integrating cloud platform, the SensorCloud project enables a marketplace for innovative customer solutions that minimizes costs and hurdles for all involved players.

13.3 Research Challenges

We identify three major challenges when integrating storage and processing of sensor data with the cloud paradigm: i) development tool support is needed in order to assist the developer in implementing efficient, distributed services that can handle large amounts of data, ii) security and privacy mechanisms have to be considered as first-class citizens in the system design in order to cater to the security requirements of the inherently multi-tenant cloud system, and iii) acceptance of the end-user has to be assured in order to overcome potential acceptance barriers of a secure cloud system. We now further detail the respective challenges.

13.3.1 Ease of Development

Cloud Computing unites approaches and technologies from different areas [29, 24, 33]: The management of the physical cloud infrastructure in an infrastructure layer often referred to as Infrastructure as a Service (IaaS) is the most mature and most commonly used cloud technology. Virtualization and dynamic resource allocation are the main technologies in this area. From a software engineering point of view the specification of the infrastructure resources used by an application as well as the quality attributes and costs of these resources are of special interest [29]. On top of this layer a platform layer called Platform as a Service (PaaS) is located which provides common functionality. This layer is typically realized with component-based middleware which utilizes the Service Oriented Architecture (SOA) pattern.

Although Cloud Computing facilitates scalable and cheap operations, it adds additional complexity to the design and implementation of services based on such a cloud. The service's software has to be designed appropriately to deliver the desired cloud properties of massive scaling and parallelization over heterogeneous computation and networking resources. Thereby, cloud specific software engineering faces known challenges at a larger scale as well as new challenges.

In order to successfully design, develop, deploy, and operate Cloud Computing services for specific tasks, a service developer needs diverse skills, ranging from understanding the behavior and background of cloud service users, over concepts and architectures to ensure security and privacy, up to the correct and scalable implementation of such concepts which can run for years in the cloud. This is a non-trivial and error prone task. Therefore, approaches are needed to simplify the tasks of the developer. For instance, in model-driven software engineering [12] models are used to abstract from technical details (e.g., how to build a system) and to focus on higher level requirements from the stakeholders. So, models are used to capture the requirements in a more natural, non-technical way. Several modeling languages or domain specific languages (DSL) have been introduced for this purpose (e.g., [36]), each focusing on a specific problem domain. The models are then used to automatically or semi-automatically generate the final technical system. To do so, they need to have a precise semantic.

The PaaS lacks mature development approaches which cope with cloud specific aspects. Especially the question of proper modeling of and programming languages for cloud systems is an open question. Only specialized solutions, e.g., MapReduce [6], are widely used. cloud specific agile methods are an open topic as well. Especially the question of efficient testing of service platforms and services is important to answer in order to adopt agile processes to cloud service development [34].

13.3.2 Security and Privacy

Data sensed by CPSs often contains sensitive information [21]. This is not only restricted to the sensed raw data but also applies to meta-information, e.g., time or

location. Thus, the owner of the data often does not want to reveal her data unconditionally to others. Instead, she may strive to share her sensitive information only with a few, carefully selected cloud services, while concealing her data from the cloud provider or other cloud services that she does not fully trust. Hence, the cloud provider must be able to guarantee the confidentiality and protection of data while being stored and processed by the data owner-defined set of clouds. Otherwise, the data owner may completely disregard the option of storing and processing her sensor data in the cloud. The root cause of this adoption barrier, as previously identified in [3, 20, 31, 18], is the fact that the *data owner loses control over her data* in the cloud.

Typical state-of-the-art approaches to securing storage and processing of data in the cloud aim at providing hard security guarantees. For this purpose, they leverage technologies such as (fully) homomorphic encryption or trusted platform modules [35, 10, 38]. We argue that these approaches are not sufficient for storing and processing sensitive sensor data in the cloud. These approaches either lead to an excessive encryption overhead when applied to rather small (only a few bytes) sensor readings [5] or do not offer the necessary control of users over their outsourced data. Additionally, it has been shown that encryption alone is not sufficient to guarantee privacy in Cloud Computing [7].

Thus, a core research challenge when outsourcing the storage and processing of possibly sensitive sensor data to the cloud is the design of a *practically viable security architecture*. This security architecture has to enable the data owner to stay in control over her data. To this end, it has to offer transparent measures to protect data already in the sensor network, to grant fine-grained access to user-selected cloud services, and to isolate cloud services at the platform level.

13.3.3 The Socio-technical Condition

The Cloud Computing paradigm constitutes a relatively new approach to information and communication technology. Its qualities, from a user perspective, differ greatly from hitherto well-known and institutionalized ways to handle data and computer technology. Based on the idea of releasing the storage and processing of individual data to the – spatially and organizationally distributed – cloud, the user is confronted with a significantly higher degree of complexity of the system used. Thus, the task of developing a potentially successful cloud architecture exceeds the mere fulfillment of well-defined technological demands. Instead, it should be understood as the challenge to materialize the vision of a certain socio-technical system, especially in the case of a technology like SensorCloud which will serve as a platform for the connection of a multitude of sensor and actuator equipped devices constantly analyzing and influencing even our physical everyday environment.

Constraints to implementation should be addressed as early as possible in the course of the development process. Hence, one of our major points is that, besides all technical feasibilities, the potential success of innovative cloud-based systems and

services needs to be explored in the context of particular social dimensions such as user acceptance, usability, and environmental conditions already in the early stages of development.

Unfortunately, to our knowledge only few related sociological work has currently addressed Cloud Computing related issues. Though recent reviews refer to some cloud-related contributions from a social science perspective [37, 40], particularly sociology has so far only rudimentarily dealt with the topic. Nevertheless, a holistic analysis of socio-technical constellations and relationships is especially needed in our context of information and communication technologies, because such a perspective allows to take human and non-human (inter-)action and (inter-)activity into account [15]. Investigating on the technology's acceptability and chances for its contextualization is therefore decisive for the success of the development project.

Assuming major developments in human-technology relationships implicated by the development of cloud technologies, indeed, it seems not sufficient to simply identify drivers and barriers for the diffusion of respective technologies and services. Moreover, we seek to sketch a concept of a human-cloud interaction [4, 14, 17] which is embedded in social practices and social structures [2, 32] in order to understand and support future developments in the area of Cloud Computing. This, however, cannot be realized without giving regard to the processes and actors which are responsible for the future of Cloud Computing themselves. Hence, an encompassing analysis of the SensorCloud innovation process completes the bundle of research challenges posed to our sociological interest by the technological project.

13.4 Development of Cloud Software

Up to now we discussed challenges which have to be solved when developing a cloud. As discussed the development of cloud service platforms and services implemented on top of these platforms is a software engineering challenge itself.

A cloud service platform is a generalized execution environment for software instances, which implement a particular functionality and share technical and/or business domain specific similarities. Software instances may be called applications, components, services, and so forth, but we refer to all of them as *apps*. The cloud service platform supports developers in developing apps by providing the commonalities as ready-made functional building blocks which can be accessed over simple and clearly defined interfaces. At run-time, the cloud service platform provides the apps with infrastructural resources (processing power, data persistence, ...). As part of the project SensorCloud we developed a multilayered architecture of cloud service platforms which utilizes the Cloud Computing paradigm. Figure 13.1 gives an overview of the layered cloud architectures. The cloud architecture consists of four layers: The *Infrastructure as a Service* layer (IaaS), the *Cloud Service Framework* layer, the *Sensor/Actuator* layer and the *Apps* layer. The *Sensor/Actuator* layer uses the cloud-specific basic functionality provided by the Cloud Service Framework and adds sensor and actuator domain specific reusable functions, which are part of sev-

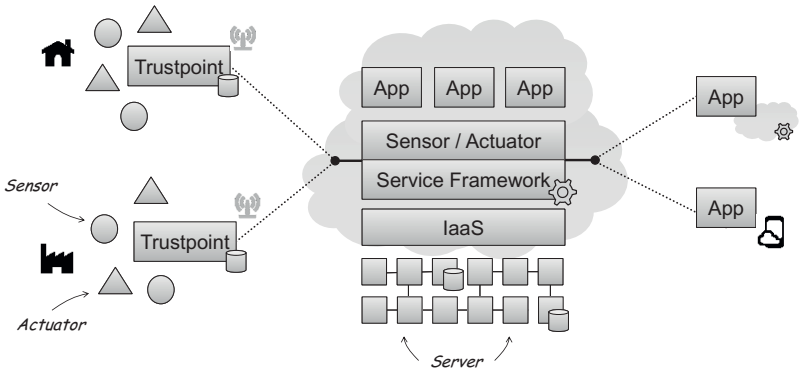


Fig. 13.1: Overview of the cloud architecture

eral use cases. An app developer thus can concentrate on the use cases which are specific to her app. The development of apps which use the cloud service platform and the development of the platform itself are performed in two separate development processes by distinct groups of developers. To improve the development of the cloud service platform and the services, we provide both development teams with software engineering methods, concepts, and tools. This support is focused on the software engineering principals agile methods, model-driven software engineering, and component-based software architecture.

13.4.1 Agile Software Development

The fundamental principal of agile software development is to set inflexible, bureaucratic processes with long-running up front planed development steps aside. Instead, the software development is broken down to activities, which are performed in short iterative cycles. At the end of each cycle the result of each activity is reflected and the next cycle is planed taken this reflection into account. The activities are guided by agile methods and tools, which aim to reduce organizational efforts and maximize the effort spend in constructive work. Every cycle starts by breaking down the current requirements to the final product down to activities for the starting cycle. The cycle ends with a reflection about the results, which may cause a change or addition of requirements. A central goal at the beginning of the project is to reach a state where an early version of the final software system is running. From this point on the software should be always in a state where it can be released containing all features added in the last cycle. This running system is then used to validate customer requirements as soon as possible against the actual software system. Changes in requirements resulting from this early validation can then be implemented with minimal effort. Agile software development is best fitting for projects where the challenges are unclear and new requirements arise during the

project and others change. The method enables a project team to validate requirements in short intervals and lowers the effort for changes resulting from changing requirements. Thereby agile methods are valuable especially for highly innovative research projects like SensorCloud where new technologies are build. Development decisions in these projects are mostly of explorative nature and typically have to be implemented with small effort and in different versions which then are compared and only one alternative may be accepted for future development.

13.4.2 Model-Driven Software Engineering

The foundation of model-driven software engineering is to represent the different aspects of the software system with suitable explicit, dedicated models which use abstraction to focus on the relevant aspects. The models are used as foundation for analytic, constructive, and communicative software engineering activities. The modeling language in which models are expressed geared to the mental and vocabulary of concepts of the problem domain. As models are more abstract and concise then source code of software they ease the understanding of the software to be developed and thereby analytical software engineering activities.

In the SensorCloud project we focus on constructive activities which aim to automatically derive a significant part of the final system implementation from models. One part of this is the code generation where the system implementation is partly or completely generated from models. The increased automation leads to enhanced quality in the resulting software. For example wide spread changes on a system, which require significant manual effort in conventional development can be realized by small changes on the model with less effort in model-driven development. As model-driven software engineering enhances development efficiency it supports agile methods and therefore benefits to SensorCloud as discussed before. In addition the development of complex systems where several complicate aspects interact like in SensorCloud benefits from the dedicated, compact and understandable representation of single aspects in specialized models.

13.4.3 Component-Based Software Engineering

The fundamental basis for component-based software engineering is the fundamental software engineering principal “separation of concerns”. The key to handle the development of complex systems therefore lies in the modularization of the system in small components, where each component fulfills a precisely defined task. The system is then constructed by combining these components. Components may contain other components which leads to a hierarchical decomposition of the system. Components implement the functionality of the system by collaborative interaction among each other. Components hide there internal structure and communicate with

other components over channels with defined interfaces. This component network represents the overall architecture of the software, which abstracts from the implementation details. In addition, components in a distributed system represent the atomic units which can be distributed. For cloud based architectures, aspects at the overall architectural level are of special relevance. These aspects include distribution of components, realization of the communication between components, elastic replication of components, fault tolerance of components, monitoring of components, as well as the logging of interactions between components.

In the project SensorCloud we study the combination of the principals agility, model-driven and component-based development in cloud based software in general and in the specific context of SensorCloud. Our approach integrates agile development principles with model-based development techniques. At its core is a set of cloud-specific architecture-oriented modeling languages. Those languages allow for the formal specification of the software architecture of cloud software as well as properties of that architecture like distribution, elasticity, robustness, and monitoring. Based on these languages, we develop a set of tools that allow for analysis/simulation, code generation, and testing.

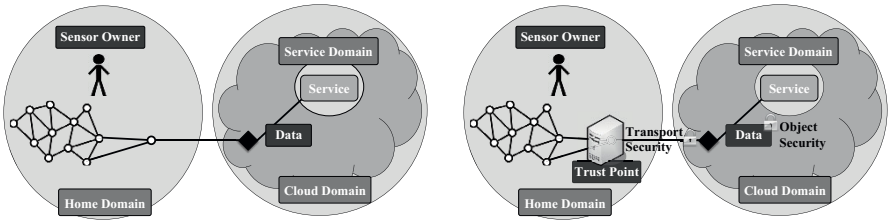
13.5 Practically Viable Security Architecture

As identified in Section 13.3, one core research challenge that has to be addressed when integrating sensor data with the cloud is the design of a practically viable security architecture. The main design goal of our security architecture is to enable the data owner to stay in control over her data when outsourcing storage and processing of sensor data to the cloud.

13.5.1 Trust Domains

As a basis for the development of our security architecture, we identified three high-level trust domains that are essential for our security architecture: the home domain, the cloud domain, and the service domain. Figure 13.2a gives an overview of these domains and depicts their interaction. We now discuss these trust domains and the underlying assumptions in more detail.

The *home domain* of a sensor owner consists of a multitude of sensor and actuator devices that are interconnected in a local sensor network. The sensor network is connected to the internet via a gateway located at the border of the home domain. We assume that only authorized devices can participate in the sensor network and that third parties cannot overhear communication within the home domain. This can, e.g., be achieved using wired connections or secure wireless communication. Thus, the sensor owner can fully trust all devices that are located within her home domain. Still, trusted communication ends as soon as the gateway passes data to



(a) Data flows through different trust domains.

(b) The trust point applies transport security and object security.

Fig. 13.2: Data flowing from the home domain up to the service domain has to be secured before it leaves the home domain which is controlled by the sensor owner.

entities outside the sensor network. For the *cloud domain*, the cloud provider has to guarantee trust into the cloud platform. To this end, it has to take the necessary technical (e.g., cryptography), organizational (e.g., security policies), and legal (e.g., contracts) measures. For both, data owner and service provider, the cloud domain is a trusted domain. The *service domain* consists of isolated service instances. Each service instance is a trust domain for the respective sensor owner. It is the task of the cloud provider to establish trust into the service domain using technical (e.g., virtualization), organizational (e.g., policies), and legal (e.g., contracts) measures. The trust domain ends as soon as data is forwarded to a different entity and thus leaves the service instance.

Inside the respective trust domain, the responsibility to keep data available, confidential, and unmodified can be attributed to a single entity. For example, a service provider is responsible for keeping data that the service may access undisclosed from other third parties. However, data in transit is at risk. To securely bridge between the identified trust domains, we propose to use a combination of transport and object-related security mechanisms. Specifically, we employ transport security in order to protect data during transfer from the gateway to the cloud. Moreover, our security architecture additionally protects data at rest (i.e., storage) and during transmission in the cloud by means of additional object-based security that is applied per data item.

13.5.2 Trust-Point-Based Security Architecture

When bridging trust domains, we aim to allow the data owner to stay in control over her data in the cloud. Hence, there exist two main requirements when securely outsourcing data storage and processing to the cloud:

1. Storage and transmission of sensor data must be protected against illegitimate access. Most importantly, unauthorized services or cloud-external entities must not be able to access or modify sensor data unnoticeably.
2. Access to sensor data in the cloud requires explicit approval by the data owner. As a result, only data owner-selected services have access to sensor data, thus limiting permission to a small set of privileged services.

From these requirements, we draw two conclusions. First, data has to be secured whenever it is stored or forwarded outside a trusted domain. Second, the control over the data access has to take place in the control domain of the sensor owner, i.e., at her home domain. Thus, we introduce the *trust point* as a control instance at the (network) border of the home domain (see Figure 13.2b). The trust point maintains a secure connection between the home domain and the cloud domain. Additionally, it manages access of service instances to sensor data of the respective sensor owner towards the cloud domain and the service domain. The trust point thus acts as a representative of the sensor owner. Introducing a trusted entity similar to our trust point has successfully been proposed for different scenarios in the past, e.g., in the context of Wi-Fi-sharing communities [16] or for intelligent energy networks in Germany [9].

The communication between trust point and SensorCloud takes place authenticated and encrypted at all times (transport security). In addition to the transport security measures, the trust point applies object security measures to the sensor data. This allows for secure storage in the cloud PaaS and realizes the access control for service instances. Before the trust point forwards sensor data to the cloud, it thus encrypts sensor data (and possibly meta data) according to the demand of the sensor owner and applies integrity protection mechanisms. The permission to access data in the cloud can hence only be granted by the trust point. For this, the trust point provides, upon request of the sensor owner, a service instance with the necessary keying material for decrypting the selected sensor data. By introducing the trust point as a local control instance, we offer the sensor owner control and transparency over the access to her data even when they are stored and processed in the cloud.

13.6 Socio-technical Innovations for the Cloud

The following section outlines central foci of interest stemming from the perspective of sociological analysis of human-cloud interaction. They are derived from the already outlined sociological research challenges and present intermediate results of our current project.

The aim of the sociological work within the project SensorCloud may roughly be seen as comprising the following different streams of interest: On the one hand there are the prospective users of SensorCloud technologies, their estimations towards the technological propositions and thus the acceptability and consequently the acceptance of SensorCloud within a certain context. These aspects derive from the wider context reflecting Cloud Computing technologies' potential to change human-

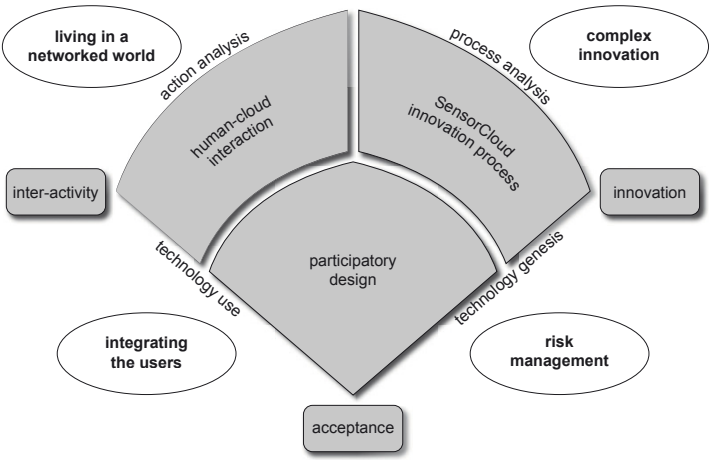


Fig. 13.3: Dimensions of sociological analysis

computer interactivity. On the other hand, it is the development process itself which is subject to our sociological interest stemming from a huge tradition of sociological innovation research [8].

Structured around the three poles of research on (1) interactivity between humans and technology, (2) innovation processes, and (3) social acceptance of emerging technologies, these streams result in a design consisting of the research strategies sketched in Fig. 13.3 and outlined in the following sections. By pursuing this strategy, on the one hand, we seek to actively contribute to the success of the SensorCloud development project. We accomplish this by informing the partners about the prerequisites for acceptability of the technological system to be developed. On the other hand, we aim at gaining a comprehensive picture of the SensorCloud innovation process as a whole. From this efforts we expect us being able to draw general conclusions for future research on both the question of human-cloud interactivity and the emergence of innovations for Cloud Computing technologies and smart environments.

13.6.1 Living in a Networked World

Cloud Computing and ubiquitous networking of an ever increasing multiplicity of diverse artifacts undoubtedly change the way in which people deal with technology and implicate transformations in the human-object-relationship. Technology may no more be conceived as a mere tool to achieve certain goals, but increasingly becomes a kind of a partner in interaction in all kinds of contexts [26]. At the same time, technology vanishes to the background and fulfills its task without being actively influenced by the users [39]. Within the SensorCloud project, we observe these

transformations and explore this novel way of dealing with technology by means of technographic field studies in order to identify and characterize this new quality in the relation between humans and their technological environments.

13.6.2 Complex Innovational Setting

Technological innovations do not arise in a vacuum, and the idea of a single genius inventor may consequently be regarded as obsolete, too. Instead, complex innovations like the SensorCloud project result from joint efforts of heterogeneous actor-networks [13] from science, (corporate) research, political actors, and intermediaries, all of which try to pursue their own individual goals in developing and implementing such a technological project. Therefore, the sociological analysis of technology development critically accompanies the entire SensorCloud development process from its very beginning. It examines how the actors balance their respective interests in order to come to mutually shared understandings of process-related problems, decisions, and common solutions. Already from the launch of the project, such a sociological perspective of technology development may obtain evidence on the chances for innovation under the condition of conflicting political, legal, social and economic interests and requirements.

13.6.3 Participatory Design

One of the crucial elements for the success or failure of an innovation is its potential users. Especially in the context of a technology such as SensorCloud, whose benefits as a network effect good [23] become more evident the larger its user base grows, it is therefore essential to integrate the users' perspective into the development process as early as possible. Against the background of a possible SensorCloud deployment in a smart home scenario, our interest focuses on users belonging to the so-called "New Middle Class Milieu"[1] with their hopes and expectations, fears and concerns. In terms of user-centered participatory design, qualitative and quantitative studies help us to identify drivers and barriers for the acceptance of the SensorCloud technology. In short, users are directly involved in the project and become an active element of technology development.

[1] http://www.sinus-institut.de/en

13.6.4 Managing Risks

In general, engineers and designers may be considered as the driving forces behind technology innovation projects. Therefore, observation of engineers' and designers' working practices, for example in the laboratories of their research institutes, should be given high priority. But their work is also often confronted with the risk of incurring concepts and ideas which may miss actual needs on the user side. Another problem may be the encounter of cultural, structural, and organizational barriers in the process of development and diffusion [25]. Here, the sociology of technology can contribute significantly to the SensorCloud innovation process by addressing such uncertainty by continuously reflecting the results to the involved project partners. Furthermore, we help to spur the project's course by implementing joint future workshops of developers, users, and other relevant groups. In such workshops, the different stakeholders meet, share time discussing topics of technological innovation, and (more or less) mutually agree on and incorporate their different views, ideas, and proposals about the topic [22, 1]. However, such future workshops do not only function as arenas to thoroughly deliberate ideas and consequences of the project in question. In addition, such dialogue serves as the basis for developing a trustful relationship between the different stakeholders.

13.7 Conclusion

As Cloud Computing enables CPS scenarios such as SensorCloud, where potentially privacy relevant data is processed, the development of such systems faces technical and non technical challenges, as we stated in our work. Within this paper, we presented the three directions of research which integrate efforts to build trustworthy cloud platforms and services. From the socio-technical perspective, we presented a framework useful for the sociological analysis of human-cloud interaction, focusing on the interactivity between humans and technology, on innovation processes, and on the social acceptance of emerging technologies. From the technical security and privacy perspective we showed a practical solution for users of cloud services to better control there data in the cloud. And from the software engineering perspective we showed that foundations like component-based and model-driven software engineering can be combined to support agile methods when developing cloud systems. These approaches will help to develop trustworthy cloud services which users can confide there data from all kind of CPS.

13.7.1 Acknowledgments

The authors would like to thank all members of the SensorCloud consortium for the inspiring discussions on the concepts described in this paper. A special thanks goes

to all participants of the sociological acceptance study. The SensorCloud project is funded by the German Federal Ministry of Economics and Technology under project funding reference number 01MD11049. The responsibility for the content of this publication lies with the authors.

References

1. Bell, W.: Foundations of Futures Studies, *Human Science for a New Era*, vol. History, Purposes, and Knowledge. Transaction Publishers, New Brunswick, London (1997)
2. Cerulo, K.: Nonhumans in social interaction. Annual Review of Sociology **35**, 531–552 (2009)
3. Chow, R., Golle, P., Jakobsson, M., Shi, E., Staddon, J., Masuoka, R., Molina, J.: Controlling Data in the Cloud Outsourcing Computation without Outsourcing Control. In: 2009 ACM Workshop on Cloud Computing Security (CCSW), pp. 85–90 (2009)
4. Dahm, M.: Grundlagen der Mensch-Computer-Interaktion. Pearson, New York (2006)
5. Danezis, G., Livshits, B.: Towards Ensuring Client-Side Computational Integrity. In: 3rd ACM Workshop on Cloud Computing Security (CCSW), pp. 125–130 (2011)
6. Dean, J., Ghemawat, S.: MapReduce Simplified Data Processing on Large Clusters. Communications of the ACM **51**(1), 107–113 (2008)
7. van Dijk, M., Juels, A.: On the Impossibility of Cryptography Alone for Privacy-Preserving Cloud Computing. In: 5th USENIX Workshop on Hot Topics in Security (2010)
8. Dougherty, D.: Organizing for innovation in the 21st century. In: S.R. Clegg, C. Hardy, T.B. Lawrence, W.R. Nord (eds.) The Sage Handbook of Organization Studies, 2 edn., pp. 598–617. Sage, London, Thousand OaksCA, New Delhi (2006)
9. Federal Office for Information Security, Germany: Protection Profile for the Gateway of a Smart Metering System. v01.01.01(final draft) (2011)
10. Gentry, C.: Computing Arbitrary Functions of Encrypted Data. Communications of the ACM **53**(3), 97–105 (2010)
11. Giese, H., Rumpe, B., Schatz, B., Sztipanovits, J.: Science and Engineering of Cyber-Physical Systems (Dagstuhl Seminar 11441). Dagstuhl Reports **1**(11), 1–22 (2011)
12. Greenfield, J., Short, K.: Software Factories Assembling Applications with Patterns, Models, Frameworks and Tools. In: 18th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA), pp. 16–27 (2003)
13. Haussling, R.: Sozialwissenschaftliche Innovationsforschung. Soziologische Revue **30**, 369–382 (2007)
14. Haussling, R.: Video analyses with a four level interaction concept a network-based concept of human-robot interaction. In: U.T. Kissmann (ed.) Video interaction analysis. Methods and methodology, pp. 107–131. Peter Lang, FrankfurtMain (2009)
15. Haussling, R.: Design als soziotechnische Relation. Neue Herausforderungen der Gestaltung inter- und transaktiver Technik am Beispiel humanoider Robotik. In: S. Moebius, S. Prinz (eds.) Das Design der Gesellschaft. Zur Kultursoziologie des Designs, pp. 263–288. transcript, Bielefeld (2012)
16. Heer, T., Jansen, T., Hummen, R., Gotz, S., Wirtz, H., Weingartner, E., Wehrle, K.: PiSA-SA Municipal Wi-Fi Based on Wi-Fi Sharing. In: 19th International Conference on Computer Communications and Networks (ICCCN), pp. 1–8 (2010)
17. Heinecke, A.: Mensch-Computer-Interaktion. Fachbuchverlag, Leipzig (2004)
18. Henze, M., Grossfengels, M., Koprowski, M., Wehrle, K.: Towards Data Handling Requirements-aware Cloud Computing. In: 2013 IEEE International Conference on Cloud Computing Technology and Science (CloudCom) (2013)
19. Henze, M., Hummen, R., Matzutt, R., Catrein, D., Wehrle, K.: Maintaining User Control While Storing and Processing Sensor Data in the Cloud. International Journal of Grid and High Performance Computing (IJGHPC) **5**(4) (2013)

20. Henze, M., Hummen, R., Wehrle, K.: The Cloud Needs Cross-Layer Data Handling Annotations. In: 2013 IEEE Security and Privacy Workshops (SPW), pp. 18–22 (2013)
21. Hummen, R., Henze, M., Catrein, D., Wehrle, K.: A Cloud Design for User-controlled Storage and Processing of Sensor Data. In: 2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom), pp. 232–240 (2012)
22. Jungk, R., Mullert, N.: Future Workshops. Institute for Social Inventions, London (1987)
23. Katz, M.L., Shapiro, C.: Network externalities, competition, and compatibility. The American economic review **75**(3), 424–440 (1985)
24. Khajeh-Hosseini, A., Sommerville, I., Sriram, I.: Research Challenges for Enterprise Cloud Computing (2010). ArXiv1001.3257 [cs.DC]
25. Knorr-Cetina, K.: Epistemic Cultures. How the Sciences Make Knowledge. Harvard University Press, Cambridge (1999)
26. Kornwachs, K., Stephan, P.F.: Das Mensch-Ding Verhaltnis. In: O. Herzog, T. Schildhauer (eds.) Intelligente Objekte. Technische Gestaltung – Wirtschaftliche Verwertung – Gesellschaftliche Wirkung, pp. 15–22. Springer, Berlin, Heidelberg (2009)
27. Lee, E.A.: Cyber Physical Systems Design Challenges. In: 11th IEEE Symposium on Object Oriented Real-Time Distributed Computing (ISORC), pp. 363–369 (2008)
28. Li, F., Vogler, M., Claessens, M., Dustdar, S.: Efficient and scalable IoT service delivery on Cloud. In: 2012 IEEE 5th International Conference on Cloud Computing (CLOUD) (2013)
29. Li, X., Li, Y., Liu, T., Qiu, J., Wang, F.: The Method and Tool of Cost Analysis for Cloud Computing. In: 2009 IEEE International Conference on Cloud Computing (CLOUD), pp. 93–100 (2009)
30. Liu, Z., Yang, D.s., Wen, D., Zhang, W.m., Mao, W.: Cyber-Physical-Social Systems for Command and Control. IEEE Intelligent Systems **26**(4), 92–96 (2011)
31. Pearson, S., Benameur, A.: Privacy, Security and Trust Issues Arising from Cloud Computing. In: 2nd IEEE International Conference on Cloud Computing Technology and Science (CloudCom), pp. 693–702 (2010)
32. Rammert, W.: Hybride Handlungstragerschaft Ein soziotechnisches Modell verteilten Handelns. In: O. Herzog, T. Schildhauer (eds.) Intelligente Objekte. Technische Gestaltung – Wirtschaftliche Verwertung – Gesellschaftliche Wirkung, pp. 23–33. Springer, Berlin, Heidelberg (2009)
33. Rellermeyer, J.S., Duller, M., Alonso, G.: Engineering the Cloud from Software Modules. In: 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing, pp. 32–37 (2009)
34. Riungu, L.M., Taipale, O., Smolander, K.: Research Issues for Software Testing in the Cloud. In: 2nd IEEE International Conference on Cloud Computing Technology and Science (CloudCom), pp. 557–564 (2010)
35. Santos, N., Gummadi, K.P., Rodrigues, R.: Towards Trusted Cloud Computing. In: USENIX Workshop on Hot Topics in Cloud Computing (HotCloud) (2009)
36. Van Deursen, A., Klint, P., Visser, J.: Domain-Specific Languages An Annotated Bibliography. ACM SIGPLAN Notices **35**(6), 26–36 (2000)
37. Venters, W., Whitley, E.A.: A Critical Review of Cloud Computing Research Desires and Realities. Journal of Information Technology **27**, 179–197 (2012)
38. Wallom, D., Turilli, M., Taylor, G., Hargreaves, N., Martin, A., Raun, A., McMoran, A.: myTrustedCloud Trusted Cloud Infrastructure for Security-critical Computation and Data Managment. In: 2011 Third IEEE International Conference on Coud Computing Technology and Science (CloudCom, pp. 247–254 (2011)
39. Weiser, M.: The computer for the 21st century. Scientific american **265**(3), 94–104 (1991)
40. Yang, H., Tate, M.: A Descriptive Literature Review and Classification of Cloud Computing Research. Communications of the Association for Information Systems **31**, 35–60 (2012)

Chapter 14
Testbed for the Sensor Cloud

Gregor Büchel, Henning Budde, Maria Bunina, Sven Elbrandt, Martin Fehre, Georg Hartung, Tobias Krawutschke, Andreas Lockermann, Thomas Partsch, Daniel Scholz, Andre Schüer, and Lothar Thieling

Abstract SensorCloud is a cooperative project of research institutes and companies in Germany which is aimed at using secure cloud technologies to connect sensors and actors from industry and private homes to the cloud. This article focuses on the construction of a testbed for SensorCloud with the application 'smart private home'. Newly developed sensors e.g. a vision sensor form the base of a system which is connected to the SensorCloud through a Location master gateway which combines several functions namely secure communication with the cloud using the SensorCloud protocol, interworking of different sensor technologies based on sensor ontologies. Special attention was given to the construction of a federated data base consisting of local components and Cloud components. The article presents the system built so far and shows some possible applications.

14.1 Introduction

SensorCloud is a cooperative project of research institutes and companies in Germany which is aimed at using secure cloud technologies to connect sensors and actors from industry and private homes to the cloud. This is part of the so called Internet of Things (IoT) idea which extends the worldwide reachability from computers to devices of all kinds. The advantages of this idea are manifold and will be shown here in some simple examples: A maintenance company specialized in maintaining complex machines at remote sites may get the information from sensors inside the machines which allow them to do maintenance routines when they

Gregor Büchel · Henning Budde · Maria Bunina · Sven Elbrandt · Martin Fehre · Georg Hartung · Tobias Krawutschke · Andreas Lockermann · Thomas Partsch · Daniel Scholz · Andre Schüer · Lothar Thieling
Cologne University of Applied Sciences, Faculty of Information, Media and Electric Engineering, Institute of Communications Engineering
e-mail: {Gregor.Buechel,Georg.Hartung,Lothar.Thieling}@fh-koeln.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_14

are really needed. An owner of an 'intelligent' house with shutters may allow a weather forecasting company to operate them in case of strong wind to avoid damages. A company specialized on house security may use existing cameras to analyze the situation and send personnel when sensors detect an break-in attempt. But the last example shows that the internet connection of sensors and actors has to be secure - otherwise burglars would penetrate a house control system, open the door locking system and pretend to the security system that all is normal.

On base on the possibilities but also the threats of the IoT a consortium of German industry and university was formed with the goal to build a prototype of a secure IoT system: the SensorCloud. This project is part of the Trusted Cloud initiative of the German federal ministry of economics and technology. The project partners have identified the following areas of research:

- Defining a secure and 'trusted' cloud protocol as well as mechanisms by which the owner of data may control the data usage inside the cloud.
- Development of tools for the development of secure cloud applications.
- Sociological investigation of the trustworthy of a SensorCloud especially in the field of domotic application.
- Building a SensorCloud node and its cloud interface.

This article focuses on the last topic. The authors currently build a domotic sensor/actor example system and its cloud data base counterpart. This includes the development of new sensors, especially in the field of vision, the development of the SensorCloud gateway and of a data base (db) concept which allows scaling from a small db inside the gateway into a db for *big data* in the cloud.

This article is organized in five Chapters:

- Chapter 2 describes comparative work
- Chapter 3 describes sensors which we developed as part of the project, especially our vision sensor
- Chapter 4 describes our gateway and the implemented sensor/actor *ontology*
- Chapter 5 describes the data base aspects of the testbed and its cloud counterpart
- Chapter 6 summarizes first experimental applications of the system

14.2 Comparative work

VisionSensor The VisionSensor is based on concepts and technologies of embedded systems and can therefore be classified by the term "embedded vision". The limited hardware resources of embedded systems are almost always the main challenge.

The main research in embedded vision focuses on the development of new algorithms, in conjunction with new embedded hardware for very specific areas of application. (stereo vision: [15, 1, 20], robot control: [10, 14], tracking: [36, 2], object detection: [37, 3, 4, 26]).

The extent of research activities that are not focused on specific applications is significantly smaller. The approach of an open embedded vision-system, but without FPGAs (Field Programmable Gate Arrays), is introduced in [34, 33, 32]. The aspects of sensor networks are treated in [32, 7, 11]. Systems with FPGA-based image preprocessing, but without the aspects of networking, are presented in [17] und [35].

Room sensor A room sensor records different climate parameters like temperature, humidity, presence of persons and air quality. Since air quality is by far the most complicated parameter to measure, air quality sensors for sensor networks have been researched in different areas of application. In [28] the author chose CO_2, CO, methane and propan gas as indication, because they are common in urban environments and become a potential life threat. The sensor module consists of a thick film gas sensor to measure the targeted gases, e.g. TGS242 for CO_2 measurement. As in case of [27] the authors use Total Volatile Organic Compounds (TVOC) as indication for the air quality as they are produced by indoor building materials, e.g. furniture, floor covering, wall paint. A TGS-2602 Metal Oxide Semiconductor Gas Sensor indicates the presence of TVOC gases by responding with a change in its resistance value. A special sensor for CO_2 measurement in subway stations is descripted in [23] based on a non-dispersive infrared (NDIR) sensor for CO_2 monitoring.

SensorCloud Gateway The idea to connect a domotic system to internet clients via a Cloud gateway was investigated in numerous projects. Zamora-Izquierdo et al from Murcia (Spain) built a system connecting devices from different house bus families (KNX, ZigBee, ...) with the internet through OSGi services with the idea to enhance comfort and allow several applications e.g. a green house or user-friendly adaptation to elderly people [39]. K. Gill et al from Loughborough, England, built a ZigBee based domotic system connected to the internet by a specialized gateway using the idea of a 'virtual home' [16]. They also focuse on security aspects and propose a suitable architecture for the gateway. An ontology based system was constructed by D. Bonino et al. from Torino (Italy) which is based on OSGi and the OWL ontology [5]. Like the Spain group they integrate devices from different house bus families and allow access through OWL-based commands and scripts. Based on these research results the idea was taken up by different "big players". The German Telecom marketed the Qivicon system [29] with the aim to define a OSGi-based Internet access to domotic networks. Microsoft research project HomeOS tries to integrate remote domotic control into the Windows 'ecosystem' [25]. Additionally, Loxone [24] is marketing a domotic system connected to cloud services with own sensors and actors. Our system distinguishes from these approaches by the ideas of openess and trust: it allows an easy integration of any kind of sensors and actors and defines mechanisms for secure Cloud services.

Federated Database System Federated Database Systems are weakly linked heterogeneous data base systems [31]. Different local data bases are managed by a global schema [38, 19]. The process of the construction of a global schema on a set

of given local schemes contains several semantic problems [12, 30]. [38] proposes an ontology driven way to solve these problems.

Database Systems for sensors and for the "internet of things" are on a general level discussed in [18]. Beenken el. Al. investigates data base system for offshore wind parks [8]. Special technologies using database system for the information management of wireless sensor networks are investigated by Klan and Sattler [21]. But these technologies are only working with a small local database management system like TinyDB. The big table concept (see [9]) is for the moment not introduced for wireless sensor networks.

In the focus of Cloud database systems some big questions are generally unsolved like "How to Partion the Data?" and "Consistency vs. Availability?" [22]. Building up a federated database system for a cloud of sensor networks could be a contribution to give a partial answer to these questions.

14.3 Sensors for a Smart Home

This section describes sensors which we developed as part of the project, especially our vision sensor.

14.3.1 Vision Sensor

Motivation, Aims and Resulting Conceptual Decisions Vision systems are now primarily designed and marketed for the purpose of industrial applications. Thus, the typical application areas and resulting requirements are very different from those of a bulk product. In connection to SensorClouds, there are versatile new fields of applications for vision systems as bulk products, particularly in the context of *smart private home* (e.g. safety-related buildings and room surveillance, home monitoring of people in need of care, reading of conventional meters for gas, water and electricity). All these different applications share a common requirement of preferably low cost, size, and power requirement, as well as preferably high communication ability.

These requirements cannot be realized in a single all-rounder product. Therefore, the target solution is a modular system that is scalable by combination and configuration of the modules. Expressed in numbers, this means the following: price (80 to 300 €), size (30 to 100 cm^3), communication skills (35 kb/s, 300 Mb/s), power requirement (2 to 15 W). The essential conceptual basis of the system is the following:

- Image processing via FPGAs (Field Programmable Gate Arrays): FPGAs are predestinated for the implementation of simple but fast processes. Preprocessing algorithms (e.g. FIR-filter) are not very complex, but must be performed on the big amount of image data. Therefore, they can be implemented opti-

mally on an FPGA. The specification of the algorithms is done in VHDL (Very High Speed Integrated Circuit Hardware Description Language), which allows the implementation of a re-usable module library for fast image preprocessing. VHDL code can be synthesized on different FPGAs by different producers, granting a guaranteed future of the design.

- Image evaluation via processors: The implementation of complex algorithms (e.g. for classification) must be done on processors. Here, the ARM technology has established itself as a quasi-standard for processors, that guarantees scalability of processing power at the highest possible compatibility.
- Usage of SoPC (System-on-a-programmable Chip): As part of this technology, the processor and the VHDL design are implemented in a single programmable chip, significantly increasing the cost-performance ratio. The processor is often available as a soft-core processor. The hard-core processors (attached to the silicon of the FPGA) are used to an increasing extent.
- Conceptual anchoring of components from the consumer sector: Vision and communication components (camera, WLAN, ethernet, ZigBee) are today contained in mass products (e.g. smartphones, webcams, and wireless thermometer). Components of that type are cheap, small and energy-efficient.
- Use of standards: To ensure a wide range of applications and the guaranteed future of the system, standard technologies are applied, as far as possible. These are, for instance, camera interface (CCIR 656), WiFi (IEEE802.11), ZigBee (IEEE 802.15.4), ARM processor technology.

Hierarchical software concept: Even with a consequent use of standard technologies, rescaling also always means hardware modifications, which cause repercussions on the software. Therefore, a hierarchical software concept is pursued, in which the underlying hardware-dependent layers provide hardware-independent software interfaces, in an upward direction.

Scalable and Modular Hardware Concept The basis of the system is a scalable and modular hardware.

The VisionSensor system can be configured from the following subsystems: camera, ARM System and/or SoPC. Each of these subsystems is also scalable. In principle, three different system configurations can be identified.

Low-end system. This version of the system consists of a camera and a SoPC. The digital image data stream of a camera is preprocessed and the outcome (typically images) is stored in shared memory. The on-chip processor can access the outcome via this memory. Complex algorithms of image interpretation, configuration of image preprocessing (via shared memory), camera configuration (over I^2C interface), as well as communication are performed on the on-chip processor. The communication is performed via (W)LAN, USART or ZigBee.

The image preprocessing unit is realized on the basis of reusable VHDL modules. These modules operate in the image-data stream; contain a uniform interface for a Vision Data Stream (VDS) and Configuration Data (CD). Thus, the modules can be interconnected to arbitrary pipelines (including intermediate storage) or to more complex processing chains.

So-called base modules, which are used for conversion (CCIRtoVDS, VDStoCCIR), generating (READ), control (MUX) and storage (WRITE) of image data streams, are available. Upon request of the on-chip processor, the configuration unit reads configuration data from the shared memory and parameterizes the modules. This parameterization is synchronous to the frame-rate clock. So-called filter modules (FM) (e.g. FM-FIR, FM-Subsample, FM-Subtract) perform signal processing on vision data streams.

The simple exemplary configuration shown above implements the pixel-by-pixel subtraction of two consecutive images captured by a camera. For this purpose, the current image is stored in the image memory and simultaneously subtracted from the previously stored image. The resulting difference image is stored in shared memory, where it can be read by the processor and further evaluated/analyzed. In addition, the difference image is converted into CCIR format and optionally can be read via DMA interface of an ARM system as preprocessed camera image. An appropriate dynamic configuration of the WRITE and the READ modules (setting the base address) ensures that the modules operate on a double buffer. Furthermore, the resolution of an image can be varied by appropriate configuration of the MUX, the FM-Subsample and the FM-FIR (low pass filtering) modules.

Median system. A medium realization of the system consists of a camera and an ARM System (Fig. 14.1). The image-data stream of a camera is stored over the

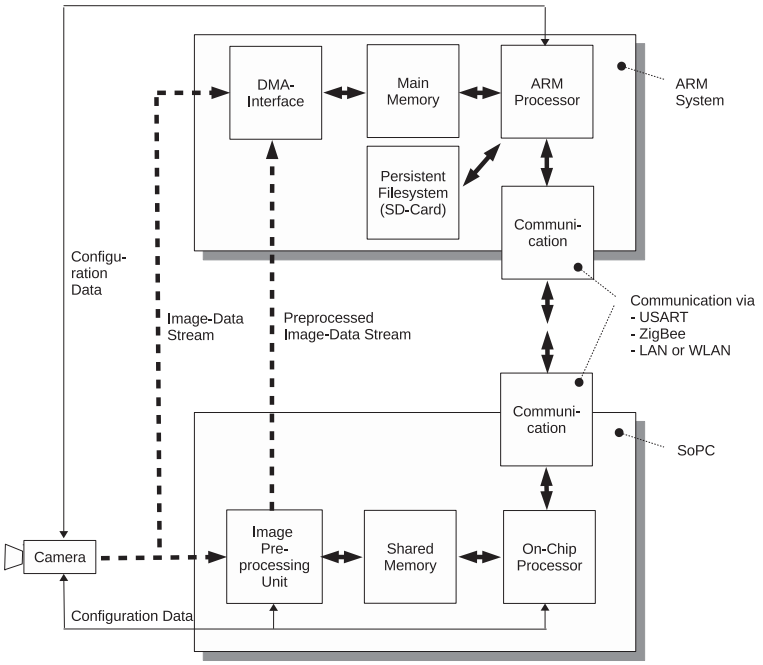


Fig. 14.1: Overview Hardware

DMA interface, directly into the main memory of the ARM processor. In addition to the image pre-processing, the ARM processor handles the image analysis, camera parameterization and the communication. The communication occurs again via (W)LAN, USART or ZigBee.

High-end system. The high-end system consists of a camera, as well as SoPC and an ARM system. Image preprocessing takes place on an SoPC. The preprocessed image data is available for both: the ARM and the on-chip processors. Hence, image analysis can be performed parallelly on both processors. Typically, the ARM processor takes over the task of camera parameterization and communication. Generally, the ARM system possesses, in this case, a persistent filesystem, e.g. in the form of an SD card.

Hierarchical Software Design. The VisionSensor is modeled corresponding to the model of a service, and implemented, according the client-server concept.

The top layer of software reflects the already introduced modeling of the VisionSensor as a service. Each VisionSensor provides a minimum vision-basic-service (e.g. identification, status request, listing of all offered services, ...). In addition, there exists a service for image acquisition (capture-service) and, per application, an application service (e.g. room-monitoring application). It is essential that an application service provided by a VisionSensor can use (as a client-process) the service of another VisionSensor (e.g. remote-file-service). This ensures a transparency re-

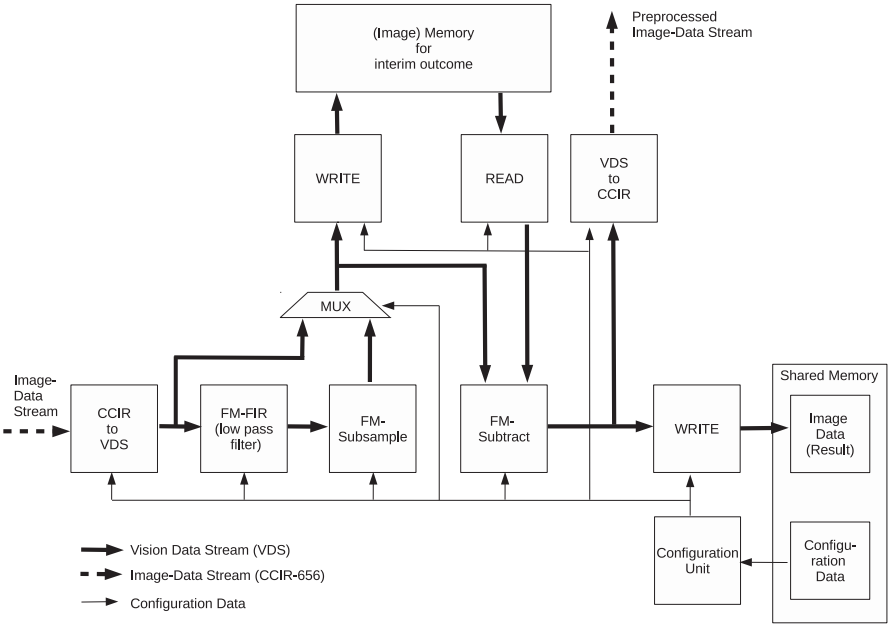


Fig. 14.2: Exemplary Image Preprocessing Unit

garding the "location" where the service is rendered, and thus an easy scalability, also at system level.

The vision-communication-lib can be understood here as a "virtual library", under which all client functions of all services are subsumed. In practice, depending on various application areas from the VisionSensor, meaningful subsets are defined and bound in real libraries. The client functions are implemented according to the concept of RPCs (Remote Procedure Calls). Depending on the used underlying communication interface (SensorCloud-API or packet-communication-lib) the service of the VisionSensor can be reached via SensorCloud or local network.

The vision-algorithm-lib can also be understood as a virtual library, under which all standard functions for image processing and image analysis are summarized. It is important that all these algorithm functions use a uniform access to the image data, like provided by the IBV-Lib (a library for industrial image processing).

The vision-file-lib provides a uniform interface for file access. It ensures a transparency regarding the location (local, remote) of the file and the type of the storage (persistent e.g. on SD-Card, non-persistent e.g. on RAM-Disc).

The packet-communication-lib provides a uniform packet-oriented interface for the communication via different communication mediums (LAN, WLAN, USART, ZigBee), between the different system platforms (SoPC, ARM system using Linux, PC using Windows). The capture-lib and the camera-config-lib provide a uniform platform-independent interface to the camera. The lowest software layer is platform-dependent and mostly self-explanatory, based on the previously given context.

14.3.2 Room Sensor

The room sensor is designed for measuring different environmental conditions and sending them to the SensorCloud gateway for further processing, e.g. the regulation of the indoor climate. By this way we achieve a raise in living and working comfort, through fresh and temperature-controlled air. It also supports the green thinking, which acts as a strong driving force in politics, and offers the opportunity to lower costs by saving energy.

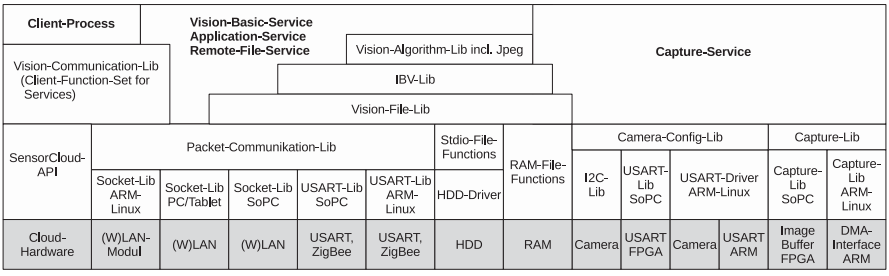


Fig. 14.3: Software Layers of the Vision Sensor

To achieve consistent data of the In Air Quality (IAQ), we need to measure basic air parameters like temperature, humidity and air pollution. Volatile air pollution compounds are especially important, because they have the ability to pass the nasal mucous membrane and to stimulate the olfactory receptor [6]. Other requirements to the room sensor device are given by a small form factor, inexpensiveness and low-power parts for comprehensive and location independent installation. This led to the following system architecture:

- A low-power, high-performance 8-bit microcontroller forms the base of the room sensor. It is connected to a TI CC1101 transceiver capable of data transfer in the 868.3MHz frequency band.
- A passive infrared sensor for movement detection. Beside its primary detection function it helps adjusting the interval of which an (energy consuming) IAQ measurement is taken. Whilst one measurement every 10 minute in an empty room is sufficient, this rate needs to be doubled if motion is detected.
- The chosen temperature and humidity sensor STH10 is an inexpensive low-power component with good measurement characteristics. It is the smallest sensor of the SHT1x family that meets the requirements with a temperature operating range of -40°C to 123.8°C and a humidity operating range of 0 to 100 % RH.
- We gave special attention to the choice of a gas sensor. Detecting different gases for air pollution measurements is important. There are different technical solutions, but none that meets all requirements mentioned like small form factor, inexpensiveness and low-power parts. We choose the MQ-135 gas sensor based on SnO_2 (Tin Dioxide), since it is an inexpensive sensor with a high variety of detectable gases like NH3, CO_2, CO, NOx, alcohol, and benzene. However, this gas sensor needs a preheat phase of 24 hrs with a relative high power consumption of approximately 800mW. To lower the power need of the MQ-135 gas sensor, we carried out experiments with reduced heating phase and found that a good prediction of the air pollution is reached with a preheat phase of 20 min prior to the first measurement and afterwards a cyclic heating of 1 min before reading. This reduces the energy need by 80 to 90 %.

Within the microcontroller the culfw firmware [13] is used which is distributed under the GPL license agreement. It was designed for common tasks in the household (e.g. switch, lamps, shutters, ...) and supports different RF protocols like FS20, FHT, HMS, ESA2000 and AskSin. For the data transfer to the SensorCloud gateway we chose the AskSin protocol since its data rate is much higher compared to the other protocols and its radio characteristics are similar to BidCoS® which makes it potentially easy to integrate into a HomeMatic network.

14.4 The intelligent gateway Location Master

This section describes our gateway and the implemented sensor/actor *ontology*.

14.4.1 System architecture

The gateway of our testbed named *Location Master* fulfills multiple tasks

- It serves as an endpoint of the SensorCloud. All communication from/to the cloud is passing through it, and it separates the secure cloud communication world from the less secure internal world of a home. Thus, a lot of encryption/decryption tasks are assigned to it.
- It connects several house automation buses to allow interworking of sensors and actors using different standards (e.g. KNX and Homematic sensors). This implies integrating several transceivers for various home bus standards as well as translation modules which convert manufacturer-dependent sensor/actor data representation into a manufacturer-independent data representation.
- It buffers data during times of unreachability. Although gateway and network technology has reached a nearly faultless state we decided to foresee measurements for unreachability times. This buffering is done in a local data base (see Section 14.5).

From these items, it may clearly be seen that a small microcontroller (e.g. from the Atmel Mega series) cannot fulfill these tasks. Thus, a Linux single computer board was chosen which offers the possibility to run several applications in parallel and to install (free) software written by the Linux community, especially data base systems and development tool chains (on a Linux PC development system). Moreover, connection to the IP net is already integrated. Last not least, we can change the hardware in latter stages exploiting multicore or SOC techniques.

Our first choice was a "beagle board xM" with an ARM single core processor (1 GHz clock rate), 512 MB local memory, USB, Ethernet and WiFi connectors. It works since project start without problems. We installed several additional home bus transceivers mostly connected through the USB interface. All installations including the Linux extensions were done in a small amount of time.

To reflect the point that our system provides the SensorCloud with access to the sensors and actors in a location, we choose the name *location master* for it.

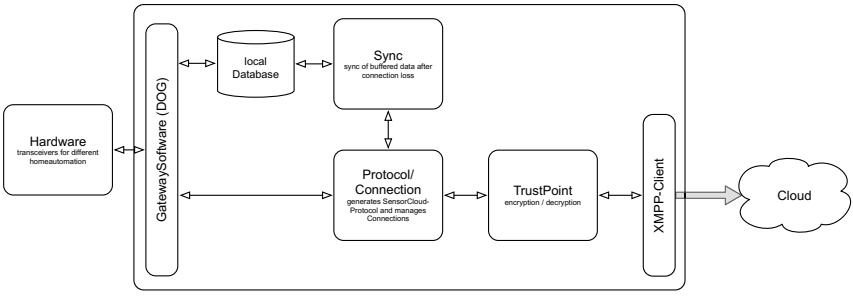


Fig. 14.4: Overview Location Master

14.4.2 Sensor interworking based on sensor ontologies

Since our testbed builds the endpoint of the SensorCloud it has to provide a uniform sensor/actor interface to the cloud which serves as base for SensorCloud services. E.g. a service provider who is allowed to operate shutters to avoid damages by strong wind only knows of shutters but not of different shutter controls. Thus, a uniform view on all shutters has to be presented to the SensorCloud.

This central idea of interworking with different sensor/actor families and standards is the reason why we started to search for a uniform way to describe sensors/actors and their interworking. As the result, we gracefully adopted the already mentioned work of Dario Bonino and others from the University of Torino (Italy) [5] and installed the DOG system, an ontology based system on our location master. We added new drivers especially for the 868MHz-based AskSin protocol which is used by our room sensor. DOG presents a uniform interface to our sensors/actors usable by SensorCloud applications. Currently we are looking for ways to offer DOG services in the SensorCloud.

14.4.3 Data management inside the Location Master

Aspects of information interchange in SensorCloud require a description of sensors and actuators in a communication protocol oriented format like JSON. The following example for the entity sensor that belongs to the entity group sensor/actuator shows the structure of the global schema in JSON-Syntax:

```
{ "global": {
       "entityGroup":[
       {      "nameOfGroup":"Sensor/Aktor",
              "entity":[{
                     "nameOfEntity":"Sensor",
                     "attributes": [
                     {      "name":"SenID",
                            "friendlyName":"Sensor ID",
                            "datatype":"text",
                            "constraints":["PRIK","UNIQUE", "UUID","notNull"],
                            "description":"Unique ID for a sensor"
                     }, ...]
              }, ...]
       }, ...]
}}
```

SensorCloud will support many types of sensors and actuators from different manufactures. This leads to the problem of having different representations of sensed values or having more than one receiving value in one measurement. To solve this issue a parsing regulation is needed for each SensorProduct in SensorCloud. The parsing regulation[1] for every SensorProduct has the following structure:

$$N + \{PHYNAM + W + EH + DT + SEM + ERL\}$$

[1] The parsing regulation is compatible to SenML (http://tools.ietf.org/html/draft-jennings-senml-10)

N is the number of values receiving from one Sensor measurement, PHYNAM is the physical name. The range of values will be described in W, EH is the physical unit, DT is the datatype (e.g. boolean, float, int), SEM is the semantic explanation and ERL is a general description.

The Multi-Room-Sensor for example is a sensor that is a SensorProduct sending four different values (motion detector, humidity, temperature and air quality) with different units at one measurement. The sensed value could look like this: 0;24;20;38 . To interpret this sensed value a parsing regulation should look like the following:

```
4;
motion detector;0,1;;boolean;0:no movement,1:movement;;
humidity;0-100;%;int;;;
temperature;0-50;°C;int;;;
air quality;0-1024;;int;;VOC
```

For example a service that generates a chart of temperature values collected in a given period needs the parsing regulation to extract the right values from the sensed values.

14.5 The data base system inside the testbed

The data base system of the SensorCloud is designed as a federated database. Additionally to DOG ontological information is stored in the **f**ederated **d**ata **b**ase **s**ystem (FDBS) of SensorCloud. The global schema called 'JSON Global Schema' describes the cloud components and the local databases of the FDBS and all their entities. This additional ontology is written in JSON format. The compact format of JSON allows a simple structure to represent the database entities. The global schema should be understandable for machines and humans and describes all properties of SensorCloud database system. These properties could be relationships, namespaces, data types and further important information.

The main goal is to generate database schemas like tables with columns in relational databases automatically. Further it is possible to generate OWL Code for the DOG System out of the JSON Global Schema. Services offered by SensorCloud should be able to get the knowledge about the actual structure of the database from the JSON Global Schema and it supports a schema monitoring service for the different databases of the FDBS.

The high requirements on horizontal and vertical scalability and the heterogeneity of sensors and actuators demand different database management systems for local systems on gateway computers and the global distributed databases in the cloud. There will be a relational database system for all data which needs relationships between each other. For the moment this will be a PostgreSQL database. Additionally there will be a NoSQL database for the massive amount of measurement data. Apache Cassandra is chosen as NoSQL database.

The database is designed as a top-level ontology. All entities of the database are assigned to six different entity groups.

Entity Group	**Number of entities**	**Excerpt of entities**
Sensor/Actuator	24	• Sensor • SensorType • SensorConfiguration • Actuator
Users	13	• UserData • UserSecurity • Groups
Location	12	• Address • Location • Room
Measurement	7	• SensedValue • MeasureType • MeasureLine
LocationMaster	7	• LocationMaster • LocationMasterConfiguration • LocationMasterComponent
Events	4	• Event • EventMessage

Table 14.1: entity groups of our top-level ontology

The top-level ontology defines a base set of concepts with all necessary types of entities that should be implemented as persistent database segments. Later, the scheme could be extended in different ways. One option is to derive new entities from former existing ones but also all other known schema extensions possibilities are allowed.

In SensorCloud all information should be managed persistent which serve the purpose of watching objects X (X= rooms, machines, grids, ...) with sensors S thereby monitor their states Z to control actuators A based on rules R.

The concept of hierarchy can be made clear by showing an example of the entity group location.

The entity location is the root of this group. All other entities derive their properties from the root or have a relationship with it. For example a room belongs to a building and the building belongs to a location. A second example is a windmill which belongs to a power-generating machine and this machine is on a location.

Another example is the entity group "Sensor/Actuator". It contains all entities that have a semantic relation to sensors and actuators. SensorProduct, SensorConfiguration, SensorService and SensorType are some entities from this group.

All of these entities have their own properties with their own datatypes. Each of them has a unique identifier which follows the concept of UUIDs (**U**niversal **U**nique **Id**entifier). As an example for all entities the entity "Sensor" will be described:

Every entity can be converted into a relational table, an object, a JSON element or an Apache Cassandra ColumnFamily.

The database within the LocationMaster must consider the limited resources of the hardware. Therefore the implemented scheme on the testbed is only an excerpt of the whole global data scheme of the SensorCloud. Only the entities that are necessary for an offline mode will be implemented on the LocationMaster. If the connection to the SensorCloud network is up, databases will be synchronized by services. These services are

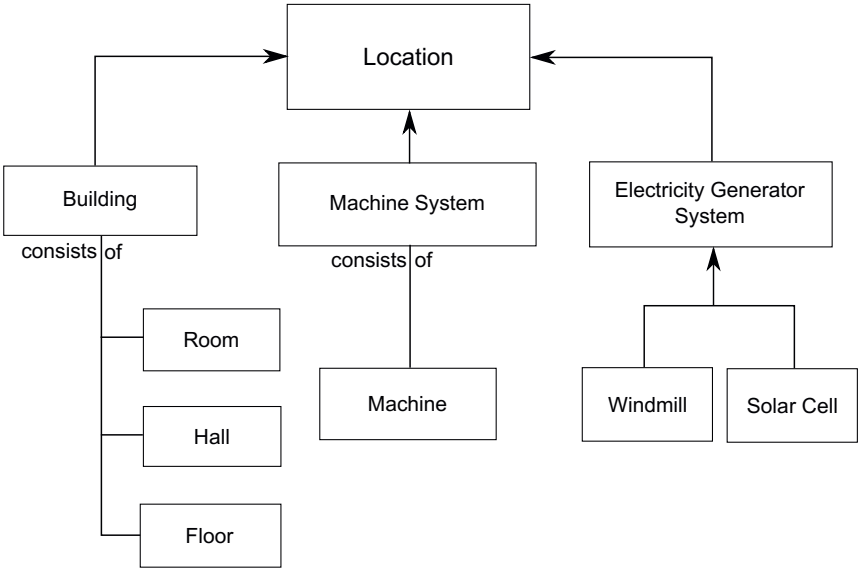


Fig. 14.5: Excerpt of the entity-group "Location"

Entity "Sensor"

name	datatype	descritpion	relationship / integrity
SenID	text	Sensor ID Unique ID for a Sensor	PRIK
SenSenProID	text	Sensor Produkt ID Describes the product of the sensor	FKEY: SensorProdukt.SenProID
SenSenTypID	text	Sensor Typ ID Describes the type of the sensor	FKEY: SensorTyp.SenTypID
SenRauID	text	Raum ID Describes the room where the sensor belongs to	FKEY: Raum.RauID
SenNutStaID	text	Nutzerstammdaten ID Describes the owner/user of the sensor	FKEY: NutzerStammdaten.NutStaID
SenLocMasID	text	LocationMaster ID Describes the LocMaster of the sensor	FKEY: LocationMaster.LocMasID
SenSouID	text	Sensor Source ID Manufactur Adress of the sensor	/
SenBez	text	Sensor Bezeichnung Friendly name for the sensor	/
SenPos	text	Sensor Position Describes the position of the sensor	/
SenDatEin	timestmp	Sensor Datum Eintritt Date of entry of the sensor to SensorCloud	/

Table 14.2: Properties of the entity "Sensor"

- a scheme monitor which observes permanently the local and global schemes of the SensorCloud and detects every change on it and creates scripts to make changes on the global schema
- a tool for data synchronization which permanently synchronizes the databases
- a prototype for event driven control of local actuators

14.6 First results and applications

This section summarizes first experimental applications of the system.

14.6.1 First applications of the VisionSensor

Room Monitoring using VisionSensors. Room monitoring is performed by VisionSensors that capture images in or in front of buildings. The images are explored as a whole or in parts (e.g. in front of doors or windows). Exploration is done in respect to temporal changes. The detection of certain changes in certain localities causes an alarm, as well as the storage and presentation of relevant images.

The monitoring is based on the comparison of a cyclically recorded actual image with a setpoint image. The two images are divided into equal-sized tiles, and tiles at the same position are compared. An alarm is given if a sufficient number of defined tiles display differences high enough, for an adequately long duration, between the actual image and the setpoint image.

An interactive parameterization of the system can be done via network, using an Android-App as the client for the application-service "room-monitoring". This app also receives and visualizes the alarms. Upon receipt of an alarm, the user can request relevant images from the scene and thus, pursue the incident online.

The first implementation of this application is based on a median system configuration (S3C6410-ARM-11-processor, WLAN-module, OV9650-camera-module). The scene can be analyzed with a rate of 20 Hz. Image JPEG-compression is done by software and needs typically 0.5s. The transmission and visualization of a compressed image on a Galaxy-Note needs about 0.1s.

We are working on a second implementation based on a low-end system configuration (Zynq-7020, WLAN-module, OV9650-camera-module). For this configuration, we already designed a VHDL-module for hardware-based JPEG-compression as part of the image preprocessing unit. Thus, the image compression could be speeded up to 0.05s. The next step is to implement a VHDL-module for image comparison. It is remarkable that the low-end system will be up to ten times faster than the median system. This results from the implementation of the JPEG-compression in hardware. This development of this hardware-based JPEG-compression cost about 1500 person hours but reduces the possible production costs of about 100 Euro per part.

14.6.2 Integrating the test node into the SensorCloud

In our testbed, different HomeMatic and KNX sensors and actuators are connected to the Location Master. Furthermore a LEGO robot is controlled by the system. We first realized small applications which show the capability to couple sensors and actors from different manufacturers and connect via different domotic buses (e.g. KNX switch to HomeMatic actor) by means of a DOG script.

At this stage, the deployment of the cloud database and the local database is conducted. The design for the database as described in chapter 14.5 consists of about 70 entities with entity groups of multiple entities. The implementation of entities is not finalized yet. For that, entities and their attributes can still be added or changed.

The schema is designed for Cassandra as NoSQL database in the cloud and for PostgreSQL as local relational database system. The local database system is used as buffer in the case of connection loss.

Regarding to the database the focus is currently set on the entity Messwert (Measurement). Data from different sensors are stored in the entity Measurement and are interpretable with the parsing regulation described in chapter 14.4.3.

The room sensor designed in this project (see 14.3.2) sends simultaneously multiple values. This data is stored in the Cassandra database and can be evaluated and visualized on the cloud side by a web interface.

References

1. K. Ambrosch, W. Kubinger, M. Humenberger and A. Steininger, in EURASIP Journal on Embedded Systems. Flexible Hardware-Based Stereo Matching, Article ID 386059, (2009)
2. S. Apewokin, B. Valentine, R. Bales, L. Wills and S. Wills, in Proceedings of 2nd IEEE Workshop on Embedded Computer Vision (ECVW), CVPR 2008, Fort Col-lins, CO, USA. Tracking Multiple Pedestrians in Real-time Using Kinematics, (2008)
3. C. Arth, H. Bischof and C. Leistner, in Proceedings IEEE Conference on Com-puter Vision and Pattern Recognition (CVPR), IEEE Workshop on Embedded Computer Vision (ECVW), New York, USA. TRICam – an Embedded Platform for Remote Traffic Surveillance, (2006)
4. C. Arth, F. Limberger and H. Bischof, in IEEE International Conference on Computer Vision and Pattern Recognition (CVPR '07). Real-time License Plate Recognition on an Embedded DSP-platform, (2007)
5. Bonino, Dario; Castellina, Emiliano; Corno, Fulvio: "DOG: An Ontology-Powered OSGi Domotic Gateway" in 20th IEEE International Conference on Tools with Artificial Intelligence: IEEE, pp. 157–160, 2008
6. F. Bitter, A. Dahms, J. Kasche, B. Müller, D.Müller and J. Panaskova, in Raumklimatechnik – Band 2 (2008). Sensorische Bestimmung der Luftqualität, (2008)
7. M. Bramberger, A. Doblander, A. Maier, B. Rinner and H. Schwabach, in IEEE Computer. Distributed Embedded Smart Cameras for Surveillance Applications, vol. 39, no. 2, (2006)
8. Beenken, P.; Schwassmann, St.; Albrecht, M.; Appelrath, H.-J.; Heisecke, S.: "Speicherstrategien für die Sensordaten des Offshore-Windparks alpha ventus", in: Datenbank-Spektrum 28/2009, S.22-30
9. Chang, F.; Dean, J.; Ghemawat, S.; Hsieh, W.; Wallach, D.; Burrows, M.; Chandra, T.; Fikes, A.; Gruber R.: "Bigtable: A Distributed Storage System for Structured Data" OSDI 2006. `http://labs.google.com/papers/bigtable.html`
10. E. Cervera, F. Berry, P. Martinet in Proceedings of the 15th IFAC World Congress on Automatic Control, IFAC'02 Barcelona, Spain, July2002.Image based stereo visual servoing: 2D vs 3D features. (2002)
11. P.W. Chen, P. Ahammad, C. Boyer, S. Huang, L. Lin, E.J. Lobaton, M.L. Meingast, S. Oh, S. Wang, P. Yan, A. Yang, C. Yeo, L. Chang, J. D. Tygar and S.S. Sastry, in Third ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC). CITRIC: A Low-bandwidth Wireless Camera Network Plat-form, pp. 1–10, (2008)
12. Conrad, St.: "Schemaintegration – Integrationskonflikte, Lösungsansätze, aktuelle Herausforderungen", in: Informatik – Forschung und Entwicklung, 17 / 2002, S.101-111
13. Koenig, Rudolf, "Features", Home of culfw, Sep. 27. 2013. `https://www.culfw.de`
14. de la Fuente, M.I.; Echanobe, J.; del Campo, I.; Susperregui, L.; Maurtua, I. International Symposium on Industrial Embedded Systems (SIES), 2010, Devel-opment of an embedded system for visual servoing in an industrial scenario, (2010)

15. R. Gerndt, S. Michalik, S. Krupop, in Proceedings of 9th IEEE International Con-ference on Industrial Informatics (INDIN), Caparica, Lisbon. Embedded Vision Systems for Robotics and Industrial Automation, (2011)
16. Gill, K.; Shuang-Hua Yang; Fang Yao; Xin Lu: A zigbee-based home automation system, IEEE Transactions on Consumer Electronics, Vol 55., Issue 2, pp.422-430, 2009
17. J. Hiraiwa, H. Amano, in Proceedings of 27th IEEE Workshop on Advanced In-formation Networking and Applications, AINA, Barcelona, Spain. An FPGA Im-plementation of Reconfigurable Real-Time Vision Architect, (2013)
18. Karnouskos, St.: "Efficent Sensor Data Inclusion in Enterprise Services", in: Datenbank-Spektrum 28/2009, S.5-10.
19. Kemper, A.; Eickler, A.: "Datenbanksysteme", (6. erw. Aufl.) München, Wien (Oldenbourg) 2006.
20. B. Khaleghi, S. Ahuja and Q. Wu, in Proceedings of 2nd IEEE Workshop on Embedded Computer Vision (ECVW), CVPR 2008. An Improved Real-time Minia-turized Embedded Stereo Vision System (MESVS-II), (2008)
21. Klan, D.; Sattler, K.-U.: "AnduIN: Anwendungsentwicklung für drahtlose Sensornetzwerke" in: Datenbank-Spektrum April 2011, S.15-26.
22. Kossmann, D.; Kraska, T.: "Data Management in the Cloud: Promises, State-of-the-art, and Open Questions", in: Datenbank-Spektrum Dezember 2010, S.121-129
23. Jongwon Kwon; Gwanghoon Ahn; Gyusik Kim; Jo Chun Kim; Hiesik Kim, "A study on NDIR-based CO2 sensor to apply remote air quality monitoring system", ICCAS-SICE, 2009, pp.1683,1687, 18-21 Aug. 2009
24. http://www.loxone.com
25. http://research.microsoft.com/en-us/projects/homeos/
26. Y.M. Mustafah, A. Bigdeli, A.W. Azman and B.C. Lovell, in Recent Advances in Security Technology (RNSA). Smart Cameras Enabling Automated Face Recognition in the Crowd for Intelligent Surveillance System, (2007)
27. Peng, I-Hsuan; Chu, Yen-Yin; Kong, Cui-Yu; Su, Yu-Sheng, "Implementation of Indoor VOC Air Pollution Monitoring System with Sensor Network," Complex, Intelligent, and Software Intensive Systems (CISIS), 2013 Seventh International Conference on , vol., no., pp.639,643, 3-5 July 2013
28. Preethichandra, D.M.G., "Design of a smart indoor air quality monitoring wireless sensor network for assisted living" Instrumentation and Measurement Technology Conference (I2MTC), 2013 IEEE International , vol., no., pp.1306, 1310, 6-9 May 2013
29. http://www.qivicon.com
30. Rahm, E.: "Mehrrechner-Datenbanksysteme – Grundlagen der verteilten und parallelen Datenbankverarbeitung", Bonn, Paris, Reading MA, USA (Addison-Wesley), 1994
31. Ritter, N.: "Verteilte und föderierte Datenbanksysteme", in: Kudraß, Th. (Hg.): "Taschenbuch Datenbanken", Leipzig (Fachbuchverlag) 2007. S.394-426
32. Anthony Rowe, Dhiraj Goel, Raj Rajkumar, FireFly Mosaic: A Vision-Enabled Wireless Sensor Networking System, Real-Time Systems Symposium (RTSS), December 2007
33. Anthony Rowe, Adam Goode, Dhiraj Goel, Illah Nourbakhsh: CMUcam3: An Open Programmable Embedded Vision Sensor, Carnegie Mellon Robotics Institute Technical Report, RI-TR-07-13 May 2007
34. Anthony Rowe, Charles Rosenberg, Illah Nourbakhsh: A Second Generation Low Cost Embedded Color Vision System Embedded Computer Vision Workshop, CVPR 2005, 2005
35. M. Samarawickrama, A. Pasqual, R. Rodrigo: FPGA-Based Compact and Flexible Architecture for Real-Time Embedded Vision Systems in Proceedings of 4th International Conference on Industrial and Information Systems, ICIIS, Sri Lanka., 2009
36. J. Schlessman, C. Chen, W. Wolf, B. Ozer, K. Fujino and K. Itoh: Hardware/Software Co-Design of an FPGA-Based Embedded Tracking System in Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop, Washington, DC, USA, 2006
37. M. Sen, I. Corretjer, F.H.S. Saha, J. Schlessman, S.S. Bhattacharyya and W. Wolf: Computer Vision on FPGAs: Design Methodology and its Application to GestureRecognition in Proceedings of IEEE Workshop on Embedded Computer Vision (ECVW), CVPR, San Diego, CA, USA, 2005

38. Vossen, G.: “Datenmodelle, Datenbanksprachen und Datenbankma-nagementsysteme”, (5. erw. Aufl.) München, Wien (Oldenbourg) 2008
39. Zamora-Izquierdo, Miguel A.; Santa, José; Gómez-Skarmeta, Antonio F.: “An Integral and Networked Home Automation Solution for Indoor Ambient Intelligence”, IEEE Pervasive Computing, Vol 9, Issue 4, pp 66-77, 2010

Chapter 15
An Architecture for Trusted PaaS Cloud Computing for Personal Data

Lorena González-Manzano, Gerd Brost, and Matthias Aumueller

Abstract Cloud computing (CC) has gained much popularity. Large amounts of data, many of them personal, are consumed by CC services. Yet, *data security* and, derived from that, privacy are topics that are not satisfyingly covered. Especially usage control and data leakage prevention are open problems. We propose the development of a trusted Platform as a Service CC architecture that addresses selected *Data security* and privacy threats (*Data breaches*, *Insecure interfaces and APIs*, *Malicious insiders* of service providers and *Shared technology vulnerabilities*). Services that consume personal data and are hosted in the proposed architecture are guaranteed to handle these data according to users' requirements. Our proof of concept shows the feasibility of implementing the presented approach.

15.1 Introduction

CC providers offer services according to the well-known service models Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a service (IaaS) [17]. As the usage of Cloud Computing (CC) progresses, so does the distribution of personal data. As a result, data is managed in a decentralized way by services running on nodes with a particular set of features. This leads to security questions that have to be answered [35]. The possibilities of having huge storage and com-

Lorena González-Manzano
University Carlos III of Madrid (Leganés, Spain)
e-mail: lgmanzan@inf.uc3m.es

Gerd Brost
Fraunhofer Research Institution for Applied and Integrated Security
e-mail: gerd.brost@aisec.fraunhofer.de

Matthias Aumueller
Fraunhofer Research Institution for Applied and Integrated Security
e-mail: matthias.aumueller@aisec.fraunhofer.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_15

putational capabilities are unquestionable benefits of CC. However, the persistent compliance of users' requirements or the liability of delivering data in a trusted service are some of the security problems which have to be solved.

Moreover, the increase of services based on cloud computing, like data storage services or upcoming services like e-government and health care, which use huge amounts of personal data, highlights the need of developing services that care about the security of personal data. In order to protect the user's privacy, data protection has to be in the focus of service- and application development [20]. Though some users are willing to disclose personal data in return for benefits [15], this is not true for all of them and laws and rights highlight the need of privacy preservation.

As an example, users who manage their contacts and appointments with a cloud based application, have to be sure about the trustworthiness of the node where the service is running and the trustworthiness of the service itself.

Likewise, using a health care application where medical history is managed, the trustworthiness of the node and the service that use these data must ensured. Note that according to the Trusted Computing Group "*an entity can be trusted if it always behaves in the expected manner for the intended purpose*" [1].

The scenario that this paper tries to protect against are service providers who offer malicious services, meaning services that do not work as intended by the users and, by that, threatening the security of the user's data. This could happen accidentially by a broken design or intentionally by a malicious service provider.

This can be solved by offering a PaaS service platform which limits functionality to a provided API and by designing this platform with special regard to mechanisms that provide data security. The contribution of this proposal is the development of a trusted PaaS CC architecture.

Users that are uploading personal data to services running in the proposed architecture, can be sure that their data is handled according to their requirements, stated in form of policies. A proof of concept shows the feasibility of implementing the proposal.

The document is structured as follows. The threat model is described in Section 15.2. Related work is disucssed in Section 15.3.Section 15.4 presents the overview and the design of the proposed architecture. In Section 15.5 a use case is depicted. Section 15.6 presents the developed proof of concept. In Section 15.7 we discuss our approach. Finally, conclusions and future work are outlined in Section 15.8.

15.2 Threat model

The Cloud Security Alliance defined the top cloud computing threats in [9]. These Threats can be classified under these nine categories:

a) *Data breaches* refers to the acquisition of data by entities which are not compliant with users' expectations.
b) *Data loss* corresponds to the loss of data intentionally, e.g. a malicious data administrator, or accidentally, e.g. an earthquake.

c) *Account or service traffic hijacking* refers to attack methods such as phishing or exploration of software vulnerabilities.
d) *Insecure interfaces and APIs* corresponds to the appropriate design of interfaces or APIs to prevent accidental and malicious attempts to circumvent policies.
e) *Denial of service* bases on using huge quantities of CC resources until leaving services unavailable.
f) *Malicious insiders* refers to the existance of users who are able to manage data in an unauthorized way. However, it must be distinguished between malicious insiders in cloud nodes and in service providers.
g) *Abuse of cloud services* bases on using CC for illicit purposes such as brute forcing decryption keys.
h) *Insufficient due diligence* refers to the ignorance or a lack of understanding of techniques, procedures, responsibilities, etc. related to CC.
i) *Shared technology vulnerabilities* corresponds to shared components that offer insufficient isolation on infrastructure (IaaS), platform (PaaS) and software (SaaS) level.

As stated in the introduction, the scenario that this paper tries to protect against are service providers who offer malicious services, meaning services that do not work as intended by the users and, by that, threatening the security of the user's data.

Regarding this, a), d), f) and i) is the addressed set of threats in this proposal, though highlighting that f) focuses on malicious insiders on the service provider side.

Consequently, attackers outside of the domain of the provider (external attacks, firewall hacking, etc.) and malicious insiders of cloud nodes (physical access to server rack etc.) are out of the scope of the proposal.

15.3 Related work

CC is a significant area of research where a lot of authors have contributed. In sum, a total of 15 proposals have been analysed, identifying mechanisms applied to address CC threats related to *Data security* (*Data breaches*, *Insecure interfaces and APIs*, *Malicious insiders* and *Shared technology vulnerabilities*). Table 15.1 presents a summary of the analysis.

A total of 5 proposals address *Data breaches*. In particular, [16] and [8] apply cryptography to protect data in the storage service, thereby data cannot be delivered to untrusted parties. Moreover, Maniatis et al. propose a tracker module to follow data and persistently control them [16]. However, this contribution presents a conceptual approach without specifying details on the execution and performance of encryption and data tracker procedures. Similarly, Zhang et al. proposes the encryption of the whole virtual disk [33]. By contrast, other works base on applications that are certified by their vendors [26, 30]. Then, an appropriate data management is ensured as long as users trust these vendors.

Proposals	Data breaches	Insecure interfaces and APIs	Malicious insiders	Shared technology vulnerabilities	Service model	Threat model
[22]				Attribute based encryption to match user requirements concerning node settings	IaaS	Attacker is an agent who has access to the node's management interface.
[16]	Data is cryptographically bundled to access and usage policies. Secure execution environment which tracks data and enforces policies.	Secure execution environment which tracks data and enforces policies.	Secure execution environment which tracks data and enforces policies.	*Node attestation protocol required+ Track data + Access control enforcement	*IaaS/ PaaS	Users that act either: Intentionally, Accidentally, Unknowing.
[6]		Certify programs running		Certify programs running	PaaS	A service provider launches a malicious or faulty application service.
[21]				Node attestation protocol according to a set of configuration settings and also node attestation while migration	IaaS	Insiders that administer the software system pose a serios risk.
[30]	Execution monitor in kernel of service provider platform, client and service commitment protocol.	Service commitment protocol: certify services by vendors		Client and service commitment protocols	PaaS	**Attacker gains root priviledge over the server and trick the user into having requests processed by untrusted code.
[25]		Fine-grained attestation, attest critical pieces of code		Fine-grained attestation, attest critical pieces of code	PaaS	Prevent software attacks
[11]				VM attestation	IaaS	**Malicious devices using hardware.
[8]	Encrypting data in the storage service		Encrypting data in the storage service	VMM registration + nodes attestation through TPM and PCR values	IaaS	**Malicious cloud administrators.
[29]				Attest nodes integrity, regarding users' requirements, when VMs launch, migrate and return results.	IaaS	**Attestation platform is prone to be attacked.
[26]	Certificates to attest applications	Certificates to attest applications		Certificates to attest applications	PaaS	**Storage replay, modification and exhaustion attacks .
[23]				Attest integrity of VMs and hosts. Data remain encrypted until the trustworthiness of the nodes and VM are verified.	IaaS	**Attack the VMs where services run.
[33]	Whole virtual disk encryption		Whole virtual disk encryption	VM attestation, the hypervison guarantees integrity and confidentiality of software running in guest VMs + whole virtual disk encryption	IaaS	Local and remote adversaries in full control of a VM.
[28]				Attesting VMs over a single TPM	IaaS	**Man-in-the-middle attacks.
[4]				Node attestation and VM trustworthiness verification	IaaS	
[19]			Labels within file to define how they should propagate	Labels within file to define how they should propagate	SaaS	User who inadvertently uploads sensitive data to the cloud and then shares or accesses the data in a way that violates company policy.

* Briefly mentioned
** Not explicitly defined

Table 15.1: Cloud Computing security state of art

Concerning *Secure interfaces and APIs*, 5 out of 15 works address this threat. The main mechanism focuses on the use of certificates [6, 30]. Certifying applications or services leads to assuming the right execution of interfaces and the corresponding APIs. Otherwise, Shi et al. look for fine-grained attestation. They propose the attestation of critical pieces of code [25]. Then, if such critical pieces of code refer to interfaces or certain parts of APIs, this threat is addressed to some extent.

Malicious insiders have been addressed by 4 out of 15 proposals, two of them focus on malicious insiders of the SP [19, 16] and the other two on malicious insiders administering the cloud node [8, 33]. Cheng et al. apply cryptography to encrypt data in the storage service [8] and Zang et al. proposes the encryption of the whole disk of a VM [33]. It can be assumed that data is only partially protected with this approach against malicious insiders because data is not protected at runtime. Maniatis et al. proposes the tracking of data to be in persistent knowledge of their location [16]. Similarly, but at software level, Papagiannis et al. work focuses on filtering HTTP connections to manage data propagation [19]. This technique helps to determine if data is controlled by an attacker or not, though requiring the installation of a browser extension.

All proposals address *Shared technology vulnerabilities* to some extent and assorted mechanisms are applied. Specifically, apart from mechanisms that address the remaining set of threats, node attestation [21, 22, 29], virtual machine (VM) attestation [11, 23, 33, 28] and combined attestation [4, 8] are the main ones developed.

Lastly, a great variety of threat models are distinguished, even not being explicitely defined in all proposals (6 out of 15 do not explicitely describe a threat model [11, 8, 29, 26, 23, 28]). For instance, some threat models base on VM attacks [23, 33] and others on software attacks [25, 21, 6].

In the light of this analysis, a lot of publications have gone towards the development of trusted CC. Nonetheless, none of these proposals addresses all threats related to *Data security* (note that [16] is a conceptual approach where only a high level description is provided).

Several mechanisms concerning the prevention of data leakage have already been developed. First of all, some techniques focus on automatically identifying personal data, such as expression-pattern-matching algorithms to identify strings like credit cards [5]. By contrast, other sets of mechanisms search data leakages knowing personal data beforehand and using them as input. The most common technique is taint tracking, where a variable is tainted when accepting user data and all instructions that make use of them are also tainted [34]. For instance, many approaches are related to data taint analysis of smartphone applications, Android applications in particular [10]. Besides, from a different perspective, [7] presents an algorithm that instruments untrusted C programs regarding places where policy violations may exist.

Users want to be in persistent control of their data, thereby requiring the enforcement of access control policies along the whole usage process. To address this issue the concept of sticky policies comes into play. It means that policies have to be always attached to the data they apply to [13].

Multiple mechanisms in different contexts have been proposed to address this matter. In general, two different groups can be distinguished. The first group corresponds to the application of Attribute or Identity Based encryption where policies are embedded in keys or ciphertexts which helps to control access to data [13, 18]. A second group of proposals focus on verifying access control policies at runtime, re-

quiring the installation of a small piece of software, e.g. a plug-in, at the client-side [12, 3].

Attestation of an entity is a proof of specific properties of that entity [1]. It is usually related to digital signatures and to software and hardware integrity.

Assorted attestation protocols have been proposed to attest nodes, virtual machines (VM) or nodes and VMs simultaneously. Santos et al. apply Attribute-Based Encryption to encrypt data left on a node, as long as the node satisfies a set of attributes [22]. Another approach based on cryptography is proposed by Garfinkel et al. [11]. It attests the integrity of VMs and hosts, encrypting data satisfying user's requirements. Similarly, though avoiding the use of cryptography, other works base on attesting nodes integrity regarding user conditions [29] and on attesting nodes according to a set of configuration settings as well as on attesting nodes during migration [21]. Velten et al. present a node attestation procedure to attest an arbitrary amount of VMs using a single TPM [28].

15.4 An architecture for trusted PaaS cloud computing for personal data

This Section presents the overview (Section 15.4.1) and the design (Section 15.4.2) of the proposed trusted PaaS CC architecture.

15.4.1 System overview

Trusted PaaS for personal data bases on creating services implemented through a provided API. These services are inspected to avoid data leakage and are running on attested nodes that guarantee the enforcement of access control along the whole process of data usage. To address this issue the proposed architecture consists of the following objects and entities (see Figure 15.1):

- *User* uses services and uploads personal data to services. Users have a key pair (K_{pub_u} and K_{pv_u}) used to sign delivered personal data and access control policies.
- *Personal data store (PDS)* is a storage of personal data controlled by data owners. For instance, it can correspond to a physical device like a smartcard within a pen drive, a mobile phone [2] or a cloud storage service [14]. Specifically, the PDS contains personal data of the users, access control policies related to each used service and data and user's key pairs.
- *Services* are pieces of software created through the use of a provided API.
- *Service provider (SP)* creates services and sends them to the Leakage Inspector to guarantee their trustworthiness and appropriate enforcement of access control policies. Once a service is successfully analysed and signed, the SP uploads

it to the cloud node to make it available to users.

- The *Leakage inspector (LI)* certifies that a particular service does not contain data leakages, that is, any kind of personal data leaves the service without the data owner's consent. Moreover, the LI instruments the services' code at runtime, so the SPA can guarantee a persistent enforcement of access control policies. This entity has a pair of keys ($K_{pub_{LI}}$ and $K_{pv_{LI}}$) to sign services after inspection and instrumentation. The LI is attested by the node when running a signed service.
- *Sticky-policy analyzer (SPA)* guarantees, in each service, the persistent enforcement of access control policies. The SPA is attested by the node before being called for the first time.
  Note that for the sake of simplicity just a single SPA and LI are mentioned but many of them are assumed to be managed.
- *Cloud node* is a particular PC/server within the cloud computing environment that offers the appropriate infrastructure to execute services. Besides, it includes the management of the provided API. Its trustworthiness has to be attested by users before delivering data to running services.

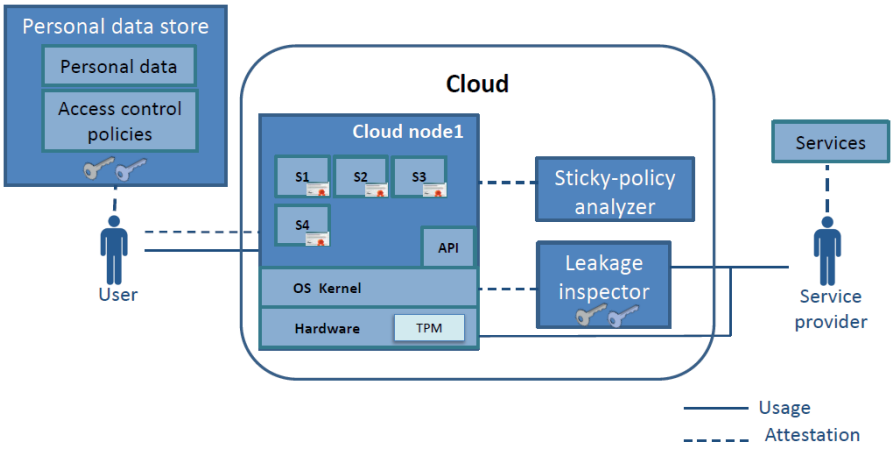


Fig. 15.1: Architecture

In general, SPs implement services using a provided API. Each service is uploaded to the LI that inspects, instruments and signs the code if the inspection has been successful. Afterwards, the SP uploads the service to the cloud node, making it available to users. Users who want to use a cloud service attest the node where the service is running to confirm it being in the expected, trustworthy configuration.

This guarantees the node is only running signed services, that it is connected to a trustworthy LI and SPA and that it delivers valid service signatures. Checking the

services signature ensures it has been inspected by the LI. If the verifications are successful, the service is used and personal data is delivered when required. While the service is running, access control policies are verified by the SPA, so access control enforcement is performed along the whole service usage guaranteeing no data leaves the system if not allowed by an attached policy. The enforcement process requires the SPA to verify signed policies and signed data, retrieved from users' PDSs, to ensure that both type of elements belong to the same user.

Note that signatures refer to digital signatures where the public keys are included in certificates issued by certification authorities. Thus, a public key infrastructure is assumed to exist.

15.4.2 System design

This section presents the description of applied mechanisms. In particular, the provided API is briefly introduced (Section 15.4.2.1), applied access control policies are defined (Section 15.4.2.2), the functionality of the LI is described (Section 15.4.2.3), the execution of the SPA is defined (Section 15.4.2.4) and the applied attestation protocols are presented (Section 15.4.2.5).

15.4.2.1 Proposed API

Assuming that the development of a complete API is a matter of future work, this section presents the first steps towards the development of the final API. It consists of nine methods implemented in Java. This API must be used by SPs that want to take advantage of the proposed architecture.

- *Object receiveFromPDS(String PDSLocation, String dataType)* obtains a list that contains the requested object and the set of policies attached to the object. The data is retrieved from a PDS located in a particular place.
- *Object sendToEntity(Entity entity, Object data)* sends data to a particular entity.
- *Object receiveFromEntity(Entity entity)* retrieves data from a particular entity.
- *Boolean dataStoreDB(Object DB, Object table, Object data)* stores data in a particular database and a particular table.
- *Object dataRetrieveDB(Object DB, Object table, Object cond, Object request)* retrieves requested data from a particular database and a particular table given a set of conditions.
- *Object searchDB(Object DB, Object table, Object cond)* searches data in a particular database and a particular table given a set of conditions.
- *Boolean writeFile(Object data, File file)* writes data to a given file.
- *Object readFile(File file)* reads a given file.
- *Object retrieveEnvironmentalAtt(LinkedList¡LinkedList¡Data, Policy¿ ¿ policies)* retrieves the necessary environmental attributes to evaluate established policies.

It should be noted that the analysis performed by the LI focuses on methods *writeFile* and *sendToEntity*. On the one hand, the LI inspects the use of *writeFile* because it causes data leakages when the written data is sensitive. On the other hand, it instruments the code according to the use of *sendToEntity* because this method may send personal data to entities without the consent of users.

15.4.2.2 Access control policies description

Access control policies (ρ) for CC services can be complex [32, 27, 31]. Considering that the search of fine-grained policies is a desirable feature and attribute management facilitates this issue [31], proposed access control policies are constructed under the Attribute Based Access Control Model. This model manages access control evaluation rules against data attributes, entities attributes (either a subject, a group or a general entity) and environmental attributes [32]. In this respect, proposed access control policies are composed of the type of data to which they apply (TD), entity conditions to satisfy (EntC), environmental conditions to satisfy (EnvC) and the granted right (GR). Applying BNF notation [24], proposed policies are depicted as follows:

- $\rho ::= \{TD,\ EntC,\ EnvC,\ GR\}$
  - $TD ::= DataType|DataType.att_i\{\wedge|\vee DataType|DataType.att_i\}^*$
    - $DataType ::= Medical|Contact|Hobbies|Multimedia|...$ refers to the type of data under which personal data are classified.
    - $DataType.att_i ::= Medical.medicalHistory|Medical.lastPrescription|Contact.name|...$ corresponds to an attribute of a particular data type.

    For example, $Medical \wedge Contact.name \wedge Contact.surname \wedge Contact.address$ expresses that the policy is linked to *medical data* and to users' name, surname and address which are considered *contact data*.
  - $EntC ::= EntityType|(EntityType.att_i\ FunctionalOpe\ [\neg]EntityType.att_iValue)\ \{\wedge|\vee EntityType|(EntityType.att_i\ FunctionalOpe\ [\neg]EntityType.att_iValue)\}^*$.
    - $EntityType ::= Receptionist|Doctor|Ministry|...$ refers to the type of the entity that may use personal data.
    - $EntityType.att_i ::= Receptionist.age|Doctor.yearsExperience|...$ corresponds to attributes of a particular entity type.

    For instance, $Doctor.yearsExperience > 10 \wedge Doctor.age > 40$ expresses that the subject who uses a particular data has to be a doctor older than 40 and with more than 10 years of experience.
  - $EnvC ::= EnvAtt_i\ FunctionalOpe\ [\neg]EnvAtt_iValue\{\wedge|\vee\ Env.att_i\ FunctionalOpe\ [\neg]Env.att_iValue\}^*$ refers to applied environmental restrictions, that is attributes.

    For instance, $ExpirationDate > 30/05/2012 \wedge ExpirationTimeDate < 30/05/2013$ expresses that the policy is valid along a year, from 30/05/2012 to 30/05/2013. Note that $\emptyset$ implies that restrictions do not exist.

– $GR := Read|Write|Download|Print|...$ corresponds to the set of operations that can be performed over data.

Operators are distinguished between the following pair of types. Note that brackets ("()") can be applied to manage precedence in the order of attributes.

- $LogicOpe ::= \wedge|\vee|\neg$ where $\wedge$ means a logic conjunction, $\vee$ a logic disjunction and $\neg$ means negation. The first pair can be applied to two elements and the last one can be applied to particular attribute values.
- $FunctionalOpe ::= < | > | \leqslant | \geqslant | =$ refers to operators applied over two attributes. For example, given the attribute $YearB$, the birthday year of a particular user, $User1$, can be compared by building the expression $YearB(User1) < 2013$ to identify if $User1$ was born before 2013.

Access control policies are established by users, per each service, and located in Personal Datastores. Alleviating the burden of defining policies per each piece of data, policies are linked to data types. For instance, a policy can be simultaneously linked to *medical data* (medical history, previous prescriptions, etc.) and *contact data* (name, surname, phone, etc) instead of defining as many policies as pieces of personal data. However, a more detailed definition of access control policies, including the definition of existing data types, entities and attributes, as well as the management procedure, is a matter of future work. Analogously, the simplification of access control policies establishment is another open issue.

15.4.2.3 Data leakage inspection

Despite the variety of existing algorithms (see Section 15.3), this paper contributes defining a general mechanism, based on [7], to identify personal data leakages in CC services.

It should be noted that services are implemented using a provided API and operations that cause data leakages as well as operations that require access control enforcement are known in advance. Under these assumptions, the proposed mechanism applies a Data Leakage Inspector (LI) to inspect, instrument and sign the service code after a successful inspection. In particular, once SPs send services to the LI, the process is split into:

A. Data leakage inspection:

1. All data received from a PDS are considered personal data.
2. Personal data are tracked along the whole service:
   a. If they are used within an allowed operation, the analysis continues.
   b. If they are used within a disallowed operation, the analysis aborts and the SP is notified.

B. Instrumentation regarding access control enforcement:

1. Places where access control enforcement has to be performed are marked.

2. A call to the SPA is inserted in each place previously noticed. In general, the SPA is called through the method *sendToSPA*:
   Boolean sendToSPA(LinkedList¡Data¿ data, LinkedList¡Object¿ envAtt, String right, Entity entity, LinkedList¡LinkedList¡Data,Policy¿ ¿ dataPolicies) makes a call to the SPA to verify policies at runtime. Specifically, *data* refers to the request data which is signed by data owners, *envAtt* is the set of existing environmental conditions, *right* is the right to execute, *entity* is the entity which has the granted right over data and *dataPolicies* refers to signed data and access control policies to verify.
3. Once the inspection is successful and instrumentation is finished, the LI signs the service's code and hands it back to the SP.

15.4.2.4 Sticky policies analysis

For the persistent control of data, policies always have to follow the data to which they are applied to enable later enforcement. In this regard, this paper presents a general mechanism to address sticky policies. In contrast to existing approaches (see Section 15.3), the application of cryptography and the installation of software at client-side is avoided. The mechanism applies a Sticky Policy Analyzer (SPA) in charge of verifying access control policies at runtime. Given calls introduced in the services' code by the LI, the SPA is able to evaluate access control policies at runtime. The process consists of:

1. Signed data and policies are verified to ensure that both types of elements refer to the same user.
2. Policies are verified and enforced.

15.4.2.5 Node attestation

In order to be able to trust a certain service and therefore being willing to provide sensitive data to it, users must be assured that their sensitive information is processed in a way they are comfortable with. To achieve that level of assurance, the node as well as the limiting and enforcing security components (LI, SPA) are attested by different attestation protocols: the proposed node attestation protocol ensures that the node is free from malicious system files, faulty configuration files or backdoors and that it has not been booted into a system state that is considered insecure; the SPA attestation protocol certifies that the SPA is running, working correctly and that is has not been altered and therefore that the policies of the user are enforced and not bypassed. The LI attestation protocol ensures the correct signing and inspection process and also ensures the service provider that its source code that is revealed to the LI is inspected with regard to confidentiality. The node attestation protocol verifies the service container is in a trusted configuration and only runs signed code.

The problem of configuration management (e.g. how to verify the trustworthiness of a node after a software update) is not considered here and is a matter of future

work, as well as the large trusted computing base (meaning a whole system running software componets is the trusted computing base).

Two scenarios would be possible: Attesting a node configuration which incorporates SPA and LI, or placing each component on a single node and attesting it then. Due to flexibility (e.g. having multiple SPAs and LIs) we chose the second approach and will illustrate it here:

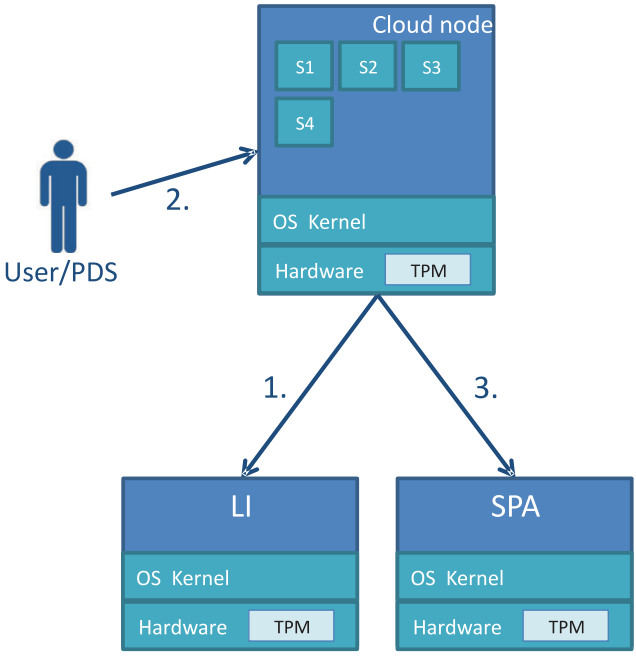


Fig. 15.2: Attestation flow

1. When a service is instantiated on the cloud node and signed code is about to be run, the node will attest the PCR values of the LI bundled with the signed source code. By verifying the signature with the attestation identity key of the LI, it ensures it was in a trusted configuration when signing the code. Thus, the LI is indirectly attested.
2. Before the user uploads personal data to a node, she can verify the configuration by attesting the node. This makes sure only ”trusted” nodes receive personal data.
3. Before calling a SPA for the first time, the node attests the configuration of the SPA to make sure it works in the expected manner (thus, not simply ignoring policies etc.).

However, self-attestation does not guarantee freshness. So an attacker could attest the LI to get a trusted configuration, manipulate the LI so it performs arbitrary signing operations and thus obtain malicious service code bundled with a trusted PCR value. To avoid this, the node running the service would need to re-attest the associated LI before running it and trigger a re-signing process if fingerprints do not match.

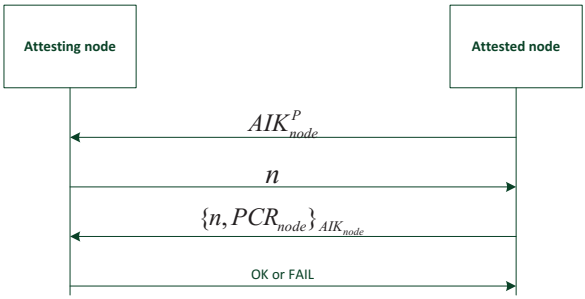


Fig. 15.3: Attestation protocol

We use a very simple attestation protocol close to [22] and use the same notation. To attest the node the user first retreives the public attestation identity key AIK^{P}_{node} and sends a nonce. The node answers with a quote that contains this nonce and its current PCR values that are stored inside the node's TPM. This quote is signed by the node's AIK to ensure the integrity of the response. After the receipt of the payload from the node the user can compare the nonce to check the freshness of the attestation and, if this matches the expected value, compare the received PCR values with a library of trusted node configurations. This protocol is used for attesting cloud node and SPA.

15.5 Use case: health care service

The proposed use case shows a realistic scenario, regarding a Health Care Service, where the presented trusted computing architecture is applied. The proposed service functionalities, implemented making use of the API presented in section 15.4.2.1, are described (section 15.5.1). After that, the definition of entities and attributes applied in access control policies associated with this particular service are briefly presented, as well as an example of applicable policies (section 15.5.2). Concluding the section, the process of sending the service's code to a LI is defined (section 15.5.3).

15.5.1 Service description

A Health Care Service, developed in Java, offers users the possibility of requesting medical appointments and taking part in online consultations.

Figure 15.4a depicts the method which allows users to request medical appointments. A retrieval of the user's name, surname and address from the user's PDS is performed. If the user is not found, medical history data is requested and her name, surname, address and medical history are stored in the DB, registering the user. After that, the type of the appointment (that is the type of the consulting room) and the desirable date and time for the appointment are sent to a receptionist (a subject in charge of managing appointments) who processes the request. When the receptionist responses, the appointment is returned to the user.

Figure 15.4b depicts the method that allows users to take part in online consultations. Again, it is verified if the user has previously used the service and if not, she is registered. Subsequently, the request type, that is the type of doctor to whom the request should be forwarded, is identified and the request is sent accordingly. Finally, when the right doctor responses, the answer is forwarded to the user. Note that, for simplicity reasons, this service can only make requests to general medicine doctors and to allergists.

15.5.2 Service access control policies

The proposed Health Care Service offers services related to medical issues and entities involved refer to doctors, doctor assistants, nurses and receptionists. Besides, entity attributes such as age or years of experience are also involved. Personal data, as the user's name and address, as well as the medical history data, which refers to *medical data* is the data that is consumed. Environmental attributes such as the expiration time of a policy can be also applied.

Under the mentioned set of entities, personal data and environmental attributes, manyfold policies can be defined. However, for the sake of clarity, the following set of policies has been created as an example. Note that environmental conditions are not applied.

- *Policy1-Medical-1(medical, doctor.yearsExperience¿10, ∅, read)*. It means that personal data categorized as *medical data*, can only be read by subjects who are doctors with more than 10 years of experience.
- *Policy2-Medical-1(contact.address, receptionist, ∅, read)*. It means that the user's address, which is categorized as *contact data*, can only be read by subjects who are receptionists.
- *Policy3-Medical-1(contact.name AND contact.surname, receptionist OR doctor, ∅, read)*. It means that the user's name and surname, which are categorized as *contact data*, can only be read by subjects who are receptionists or doctors.

15.5.3 Service inspection

Before uploading a service to a cloud node the Health Care Service, composed of the methods *requestMedicalAppointment* and *requestMedicalInfoOnline*, is sent to the LI. Recalling that *writeFile* may apply personal data, which refer to data retrieved from a PDS, and it may cause data leakages sending personal data out of the service, its application is inspected. Thus, in case *writeFile* applies personal data, data leakages are found and the LI notifies the SP of all identified ones. Afterwards, the SP has to remove all data leakages and sends the code back again to the LI. Otherwise, the LI instruments the code. It tracks personal data and inserts calls before the use of *sendToEntity* in case personal data are used. These calls are introduced regarding *sendToEntity* because this method may deliver personal data to external entities. Figures 15.5a and 15.5b present the result of executing both processes, though considering that data leakages should be avoided before the code's instrumentation is performed. Finally, when the inspection and instrumentation finishes, the service's code is signed and sent to the SP.

15.6 Proof of concept

A proof of concept has been developed, using J2SE 1.7, to prove the feasibility of implementing the proposal. A pair of web services has been developed. One of them involves the basic implementation of methods *requestMedicalAppointment* and *requestMedicalInfoOnline* presented in the proposed use case (section 15.5). Basic means that the functionality of processing requests, either medical appointments or online consultations, is not currently implemented but the execution of inspected and instrumented code is proven to be successful.

On the other hand, the other implemented web service corresponds to the LI. It allows the inspection of java files pointing out statements that cause data leakages, and the instrumentation of java files to ensure an appropriate access control enforcement.

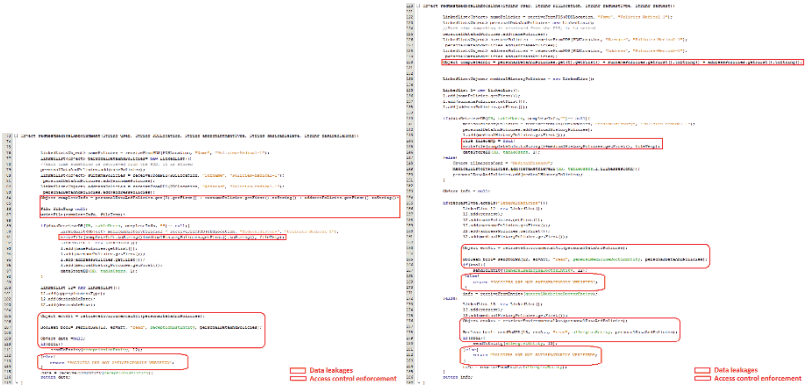
15.7 Evaluation of our approach

The proposed approach focuses on the development of a trusted PaaS CC architecture which is supposed to address all noticed *Data security* threats (*Data breaches*, *Insecure interfaces and APIs* , *Malicious insiders* and *Shared technology vulnerabilities*). We will briefly assess if these threats have been covered.

Data breaches are addressed by the use of the LI. The service code is inspected to detect places where personal data is sent to external elements (e.g. written to a file), hindering data breaches.

(a) Method requestMedicalAppointment (b) Method requestMedicalInfoOnline

Fig. 15.4: Health Care Service's code



(a) Method requestMedicalAppointment (b) Method requestMedicalInfoOnline

Fig. 15.5: Processed Health Care Service's code

Insecure interfaces and APIs is addressed by the development of an API used to implement running services, as well as the inspection of data leakages. An API is provided to facilitate the identification of possible places where users' privacy may be violated. Where malicious uses of the provided API can be performed, the LI analyses the code to detect disallowed statements that cause data leakages.

Malicious insiders is another addressed threat, though only malicious insiders in the service provider have been considered. The verification of users' policies at runtime avoids the disclosure of data against the consent of the user, thereby mitigating the existence of malicious insiders. Besides, the fact that users verify the LI signature attached to the used service helps to assure the lack of malicious alterations in the service's code. The enforcement of API usage also limits the possibility to run malicious code.

Last but not least, *Shared technology vulnerabilities* is mainly addressed through the development of an API to implement services. This leads to a limited set of

functionalities and avoids shared technology vulnerabilities, since services cannot run arbitrary commands and do not have direct access to (virtualized) ressources.

Following this, we will briefly discuss strenghts and weaknesses of our approach:

On the one hand, the presented proposal has several weaknesses. It requires a public key infrastructure which brings complexity into the system. Also the range of services that could use our architecture is limited: Services like social networks, in which a huge amount of personal data is processed and accessible to many entities, would require a lot of access control policies verifications causing a large number of calls to the SPA, thus reducing the service performance or even worse, provoking a denial of service in the SPA. Furthermore, the development of a complete API to allow the implementation of any type of service is difficult and successfully securing the code gets complex. Likewise, it should be noticed that existing services would have to be re-implemented if they want to be executed in the proposed trusted CC architecture. Attestation is currently feasible for a specific configuration, but is getting somehow difficult with several supported configurations, even made worse by having multiple components that need to be handled. This would lead to a trusted repository which would have to be managed as well. Defining independent components for SPA, Node and LI may lead to performance penalties (in regard to response times) and online/offline situations must be handled.

Nonetheless, the proposed architecture has multiple strengths. Limiting functionality to a provided API, a common mechanism in PaaS solutions like Google AppEngine, makes the detection of malicious code easier. The usage of components like the LI and SPA enable parallel or cascading designs to limit performance problems (in regard of traffic load) and enables segregation of duties. Concerning sticky policies, the proposed mechanism bases on instrumenting the code before being executed. Access control management is performed at the server side instead of applying more cumbersome mechanisms such as plug-ins or cryptographic procedures. Another advantage is that the development of an API simplifies services' code instrumentation and analysis because methods which may violate users privacy are known beforehand. Attestation ensures the trustworthiness of used components and guarantees policy enforcement and the integrity of the whole system. This even guarantees the SP that his code is not leaked to the outside world.

15.8 Conclusions and future work

CC services are used quite commonly and make use of large amounts of data, many of it personal. In this regard, *Data security* is of vital concern. Personal data has to be managed according to users requirements and free from privacy violations. To address the discussed threats, this paper presented a PaaS architecture concerning the development of attached mechanisms. In particular, it bases on inspecting and instrumenting services before uploading them to a cloud node to ensure an appropriate access control enforcement at runtime. Moreover, it also ensures the trustworthiness of the node the services are deployed to. Last but not least, a security analysis at-

tests the coverage of all threats and a prototype shows the feasibility of the proposal implementation.

Concerning future work, the proof-of-concept prototype will be developed into a fully working showcase. This is accompanied by the specification by a full-fledged service API suitable for real world service types. The definition of allowed and disallowed calls that are evaluated when inspecting code will be formalized to work with a greater set of use cases. The integration with different platforms and/or protocols has to be studied. A detailed definition of access control policies, including management procedures and techniques to simplify their creation are also matters of future work. This is also true for configuration management of the software components and a more property based attestation workflow with a reduced trusted computing base.

References

1. Achemlal, M., Gharout, S., Gaber, C.: Trusted platform module as an enabler for security in cloud computing. In: Network and Information Systems Security (SAR-SSI), 2011 Conference on, pp. 1–6. IEEE (2011)
2. Allard, T., Anciaux, N., Bouganim, L., Guo, Y., al. et: Secure personal data servers: a vision paper. Proceedings of the VLDB Endowment **3**(1-2), 25–35 (2010)
3. Beato, F., Kohlweiss, M., Wouters, K.: Scramble! your social network data. In: Privacy Enhancing Technologies, pp. 211–225. Springer (2011)
4. Bertholon, B., Varrette, S., Bouvry, P.: Certicloud: a novel tpm-based approach to ensure cloud iaas security. In: Cloud Computing (CLOUD), 2011 IEEE International Conference on, pp. 121–130. IEEE (2011)
5. Brodie, B.C., Taylor, D.E., Cytron, R.K.: A scalable architecture for high-throughput regular-expression pattern matching. In: ACM SIGARCH Computer Architecture News, vol. 34, pp. 191–202. IEEE Computer Society (2006)
6. Brown, A., Chase, J.S.: Trusted platform-as-a-service: a foundation for trustworthy cloud-hosted applications. In: Proceedings of the 3rd ACM workshop on Cloud computing security workshop, pp. 15–20. ACM (2011)
7. Chang, W., Streiff, B., Lin, C.: Efficient and extensible security enforcement using dynamic data flow analysis. In: Proceedings of the 15th ACM conference on Computer and communications security, pp. 39–50. ACM (2008)
8. Cheng, G., Ohoussou, A.: Sealed storage for trusted cloud computing. In: Computer Design and Applications (ICCDA), 2010 International Conference on, vol. 5, pp. V5–335. IEEE (2010)
9. Cloud_Computer_Alliance: The notorious nine cloud computing top threats in 2013 (2013)
10. Fritz, C.: Flowdroid: A precise and scalable data flow analysis for android. Master's thesis, Technische universitat Darmstadt (2013)
11. Garfinkel, T., Pfaff, B., Chow, J., Rosenblum, M., Boneh, D.: Terra: A virtual machine-based platform for trusted computing. In: ACM SIGOPS Operating Systems Review, vol. 37, pp. 193–206. ACM (2003)
12. Ghorbel, M., Aghasaryan, A., Betgé-Brezetz, S., Dupont, M., Kamga, G., Piekarec, S.: Privacy data envelope: Concept and implementation. In: Privacy, Security and Trust (PST), 2011 Ninth Annual International Conference on, pp. 55–62. IEEE (2011)
13. González-Manzano, L., González-Tablas, A., de Fuentes, J., Ribagorda, A.: Security and Privacy Preserving in Social Networks, chap. User-Managed Access Control in Web Based Social Networks. Springer (2013)

14. Kirkham, T., Winfield, S., Ravet, S., Kellomaki, S.: A personal data store for an internet of subjects. In: Information Society (i-Society), 2011 International Conference on, pp. 92–97. IEEE (2011)
15. Li, H., Sarathy, R., Xu, H.: Understanding situational online information disclosure as a privacy calculus. Journal of Computer Information Systems **51**(1), 62 (2010)
16. Maniatis, P., Akhawe, D., Fall, K., Shi, E., McCamant, S., Song, D.: Do you know where your data are? secure data capsules for deployable data protection. In: Proc. 13th Usenix Conf. Hot Topics in Operating Systems (2011)
17. Mell, P., Grance, T.: The nist definition of cloud computing (draft). NIST special publication **800**(145), 7 (2011)
18. Mont, M.C., Pearson, S., Bramhall, P.: Towards accountable management of identity and privacy: Sticky policies and enforceable tracing services. In: Database and Expert Systems Applications, 2003. Proceedings. 14th International Workshop on, pp. 377–382. IEEE (2003)
19. Papagiannis, I., Pietzuch, P.: Cloudfilter: practical control of sensitive data propagation to the cloud. In: Proceedings of the 2012 ACM Workshop on Cloud computing security workshop, pp. 97–102. ACM (2012)
20. Pearson, S.: Taking account of privacy when designing cloud computing services. In: Software Engineering Challenges of Cloud Computing, 2009. CLOUD'09. ICSE Workshop on, pp. 44–52. IEEE (2009)
21. Santos, N., Gummadi, K.P., Rodrigues, R.: Towards trusted cloud computing. In: Proceedings of the 2009 conference on Hot topics in cloud computing, pp. 3–3 (2009)
22. Santos, N., Rodrigues, R., Gummadi, K.P., Saroiu, S.: Policy-sealed data: A new abstraction for building trusted cloud services. In: Usenix Security (2012)
23. Schiffman, J., Moyer, T., Vijayakumar, H., Jaeger, T., McDaniel, P.: Seeding clouds with trust anchors. In: Proceedings of the 2010 ACM workshop on Cloud computing security workshop, pp. 43–46. ACM (2010)
24. Scowen, R.S.: Extended bnf-a generic base standard. Tech. rep., Technical report, ISO/IEC 14977. http://www. cl. cam. ac. uk/mgk25/iso-14977. pdf (1998)
25. Shi, E., Perrig, A., Van Doorn, L.: Bind: A fine-grained attestation service for secure distributed systems. In: Security and Privacy, 2005 IEEE Symposium on, pp. 154–168. IEEE (2005)
26. Sirer, E.G., de Bruijn, W., Reynolds, P., Shieh, A., Walsh, K., Williams, D., Schneider, F.B.: Logical attestation: an authorization architecture for trustworthy computing. In: Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, pp. 249–264. ACM (2011)
27. Takabi, H., Joshi, J.B.: Semantic–based policy management for cloud computing environments. International Journal of Cloud Computing **1**(2), 119–144 (2012)
28. Velten, M., Stumpf, F.: Secure and privacy-aware multiplexing of hardware-protected tpm integrity measurements among virtual machines. In: Information Security and Cryptology–ICISC 2012, pp. 324–336. Springer (2013)
29. Xin, S., Zhao, Y., Li, Y.: Property-based remote attestation oriented to cloud computing. In: Computational Intelligence and Security (CIS), 2011 Seventh International Conference on, pp. 1028–1032. IEEE (2011)
30. Xu, G., Borcea, C., Iftode, L.: Satem: Trusted service code execution across transactions. In: Reliable Distributed Systems, 2006. SRDS'06. 25th IEEE Symposium on, pp. 321–336. IEEE (2006)
31. Yu, S., Wang, C., Ren, K., Lou, W.: Achieving secure, scalable, and fine-grained data access control in cloud computing. In: INFOCOM, 2010 Proceedings IEEE, pp. 1–9. IEEE (2010)
32. Yuan, E., Tong, J.: Attributed based access control (abac) for web services. In: Web Services, 2005. ICWS 2005. Proceedings. 2005 IEEE International Conference on. IEEE (2005)
33. Zhang, F., Chen, J., Chen, H., Zang, B.: Cloudvisor: retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization. In: Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, pp. 203–216. ACM (2011)

34. Zhu, D.Y., Jung, J., Song, D., Kohno, T., Wetherall, D.: Tainteraser: protecting sensitive data leaks using application-level taint tracking. ACM SIGOPS Operating Systems Review **45**(1), 142–154 (2011)
35. Zissis, D., Lekkas, D.: Addressing cloud computing security issues. Future Generation Computer Systems **28**(3), 583–592 (2012)

Chapter 16
Privacy-Preserving Cloud Computing for Biomedical Research

Martin Beck, V. Joachim Haupt, Janine Roy, Jan Moennich, René Jäkel, Michael Schroeder, and Zerrin Isik

Abstract Applications in the biomedical sector have a rising demand for computational power due to the rapid growth of biological data. However, the setup and maintenance of a sufficient computational infrastructure is costly. Cloud computing allows distributed computing over a network and maximizes the efficiency of the shared resources. Nevertheless, security concerns rise when valuable research data is transferred to a public Cloud. Herein, we describe – from the application and security perspective – three biomedical case studies from different domains: Patent annotation, cancer outcome prediction, and drug target prediction. We elaborate on different strategies to secure the data and results in the Cloud as well as on the infrastructure needed. The proposed Cloud solutions can help to adapt similar algorithms from different domains to benefit from Cloud computing.

16.1 Introduction

Data in life sciences – such as sequence, structural, pharmacological/drug data and biomedical literature – are ever growing and its usage demands appropriate computational resources. Cloud services offer such resources, but require adjusted algorithms and data management. Security concerns rise when valuable research, business data or personally identifiable information is transferred to a public Cloud.

Martin Beck
TU Dresden, Institute of Systems Architecture, Germany
e-mail: martin.beck1@tu-dresden.de

Joachim Haupt · Jan Moennich · Michael Schroeder · Zerrin Isik
TU Dresden, BIOTEC, Germany
e-mail: michael.schroeder@biotec.tu-dresden.de

René Jäkel
TU Dresden, Center for Information Services and High Performance Computing (ZIH), Germany
e-mail: rene.jaekel@tu-dresden.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_16

Moreover, resulting data should be protected in such a way, that nobody except from the submitting party is able to evaluate the results.

Nonetheless, the use of High Performance Computing (HPC) for biomedical research has a long history [29] and its transition to Cloud Computing (e.g. Cloud storage systems for DNA sequencing [31]) – going away from in-house solutions – becomes an ever increasing pressure for smaller companies due to increased computing needs. This brings new problems, such as vendor lock-in and the use of potentially untrustworthy environments. Contrary to the increased economical efficiency and flexibility of Cloud Computing, there are still major drawbacks related to privacy issues, possible data leaks, and unauthorized data access [13], which prevent broad adoption.

Solutions to these issues can be approached by different directions. One of these are examinations and audits leading to certificates regarding the implemented security solutions. Examples are ISO 27001, NIST SP800-144, EuroCloud, Cloud Security Alliance (CSA) and TÜV Trust IT. Other, more rigorous audits are planned, like the German BSI Cloud certificate. Another direction would be to check promises made by providers regarding implemented security solutions, backed up contractually and not by any available certificate. This can include technical advances recently made for excluding unauthorized personal to access sensitive data. However, this still requires a lot of trust that the provider really achieves the desired privacy guarantees.

From another perspective, the Cloud users can improve privacy by themselves using techniques for secure computation. Some of these techniques require the Cloud provider to cooperate and install some sort of trusted hardware, while others are solely based on computational techniques defined by the user. While the latter possibly achieves the highest possible privacy level, as there is no dependency on trusting the Cloud provider, it may also be the most involved or most expensive. Many proposals on how to enhance data privacy in public and hybrid Cloud computing scenarios in such a model can be found [36].

This includes the use of trusted platform modules (TPM) for trusted computing [39], effective separation of data depending on the secrecy level [8], the complete use of fully homomorphic encryption [21] or multi-party computation between several non-colluding parties [45]. Nevertheless, these techniques possess advantages and disadvantages, that need to be considered before application. Some methods like homomorphic encryption provide a high security level, but data transformation is extremely time consuming, thus increasing execution time dramatically. In contrast, techniques – like anonymization – have only a small impact on the computational complexity, but do not provide a high level of security. The balance between different security mechanisms and the efficiency of data transformation is shown in Figure 16.1.

To overcome issues in data security on identical complexity levels [4], we develop strategies for service operation in a potentially untrustworthy environment to be able to use any computation platform, e.g. a public Cloud, without the necessity to rely on security promises by the chosen provider. The presented strategies – involving (homomorphic) encryption, minimization and anonymization – are tai-

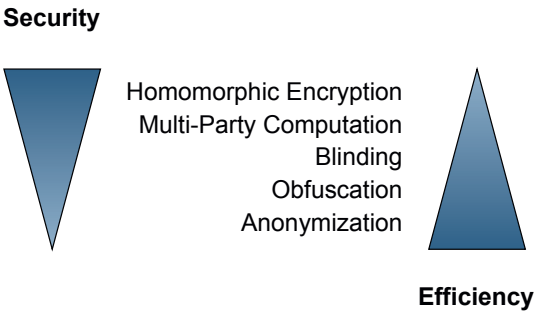


Fig. 16.1: Balance between security and efficiency

lored to three case studies from different domains: Patent annotation (text mining), cancer outcome prediction (translational bioinformatics), and drug target prediction (structural bioinformatics).

A typical application of Cloud services to biomedical research is disease prognosis for personalized medicine. Transcriptional bioinformatics aims to interpret the vast amount of clinical data (e.g., gene expression/regulation data, protein-protein interaction data, pathway information) to discover combinations of causative disease genes [27]. Computational outcome prediction has proven successful in many cancer types [12, 26, 43], particularly when Google's PageRank algorithm (NetRank) is run on protein-protein interaction and gene expression data [38]. Using the gene expression data obtained from the patient's tissue sample as input for computation in a Cloud service could provide physicians a valuable tool to support the selection of treatment options. However, patient data is confidential and thus has to be secured appropriately while being processed in the Cloud.

Most knowledge in the biomedical domain is predominately presented in scientific articles rather than in structured representations like databases, limiting access to information to literature search [17]. More than 22 million articles are indexed in MEDLINE to date. Patents bare valuable information – years before the results are published in the scientific literature – and grow rapidly in numbers with up to 900,000 granted patents per year [18]. The annotation and mining of such texts are tedious and computationally expensive. Here, we present two text-mining Cloud services, allowing for patent annotation with knowledge-extraction and the ranking of results of search queries with two entities by assessing their relevance. Such ranked results facilitate the completion of notoriously incomplete database entries [14].

The third application represents the domain of drug development, which is a very costly and failure-prone endeavor with about 81% of the drugs failing [15]. Major causes for failure are lack of drug efficacy and off-target binding. Thus, it is fundamental to discover targets causing disadvantageous effects early in the drug discovery pipeline. A promising strategy is to infer off-targets from detected binding site similarity to known target proteins [24], yielding information for lead optimization at the same time. To address this, we implemented a pipeline for binding site similarity detection and assessment in a Cloud environment [23]. Another applica-

tion of this pipeline is drug repositioning, a drug development strategy proposing approved (or late stage clinical trial) drugs for alternative indications, saving up to 40% of drug development costs [11] and yielding treatment options for tropical and rare diseases [24].

We present for all services a fully data-centric security approach, utilizing customized algorithms to achieve the desired trade-off between privacy-level and performance. In our approach, sensible data is preprocessed and secured locally in full control of the user and subsequently transferred to the Cloud platform. Ideally, the Cloud services run on encrypted data, evading an access by an intruder and ultimately decreasing privacy related usage restrictions of Cloud infrastructures. To perform the presented analysis, we use a private Cloud infrastructure based on Open Source solutions and a HPC cluster.

16.2 Cloud Infrastructure

In this section, we describe the architectural model as well as the infrastructure used to elaborate on the different solutions available and on the approaches necessary to run biomedical research in a Public Cloud.

Contrary to the increased economical efficiency, flexibility and all the other advantages a company gains by utilizing the Cloud Computing model, there are still a few major drawbacks, which prevent broad adoption. At least three of the top ten threads to Cloud Computing that were presented by the Cloud Security Alliance [13] are directly related to privacy issues, possible loss, leakage and unauthorized access to user supplied data. The literature gives many proposals on how to enhance data privacy in the public and hybrid Cloud Computing scenario. A general overview can be found in [36].

Beside other approaches to overcome these major privacy concerns, which are more infrastructure related [25], we present in this paper a fully data-centric security approach. This includes the use of trusted platform modules (TPM) for trusted computing [39], effective separation of data depending on the secrecy level [8], the complete use of fully homomorphic encryption [21] or multi-party computation between several non-colluding parties [45].

We aim to realize services operating on highly vulnerable data in the pharmaceutical domain. Therefore, any Cloud environment used for calculations not under our direct control concerning access rights and mechanisms, is regarded as a potentially non-trustworthy environment. To provide necessary data for the computation in the Cloud, the services need access to encrypted or blinded data, which have to be generated before any computation in the Cloud. This preparation step has to be executed in a trusted environment, such as a small computing cluster or even a private Cloud in a company, where the user has full control over the used infrastructure. Afterwards, the data can be transferred and stored on any Cloud platform for further processing.

The security mechanism relies on the fact that the executed Cloud service works directly on encrypted data, which means, even in case an attacker is able to get access to the Cloud infrastructure, the potentially obtained information can not be analyzed directly and therefore not be used to infer some knowledge out of the service operation. Therefore, the previous hard restriction in the trustworthiness of using a Cloud infrastructures can be decreased immensely, since the strong privacy requirements can be assured by the security model itself.

In the following sections we describe an architecture that does not rely on specific precautions, selected protocols, standards or components on the provider side to ensure privacy of the supplied data. The realized infrastructure is based an widely used Cloud models, Open Source implementations and standards for accessing the virtualized hardware components for computation. Our Cloud services from the biomedical domain rely each on an application specific privacy solution, which is tailored to the algorithm used and the security level required.

16.2.1 Cloud Architecture and Infrastructure

Within our project, we realized a Cloud infrastructure based on Open Source solutions. We have tested two widely used Cloud middlewares, namely OpenNebula[1] and OpenStack[2], more in detail. Nowadays, the Cloud middlewares have evolved towards a rather mature state and provide a common set of general management functionalities, such as user handling and rights management, full virtual machine (VM) life cycle, monitoring, and user and developer interfaces.

The Cloud middleware operates on top of a virtualization layer, which is controlled independently by an additional instance, which is often referred as hypervisor in case of KVM[3]. In our production system, we are using OpenNebula and KVM as default, since it has a long and stable history as open source project and a broad distribution. However, the middleware can equally used in conjunction with Xen[4].

Figure 16.2 shows the general usage workflow to interact with the Cloud environment, either by directly accessing the virtual machine (VM) level via previously registered VM's, or by starting the service via its registry entry. The middleware provides a convenient user interface for multiple purposes. Depending on the given user rights steered by the administrator, the user can not only register and run predefined VM's, but can also upload specific VM templates, or even define networks for his own VM's. The access to the distributed VM instances can be done via the graphical user interface or by its network route defined in the VM template This way, even complex usage scenarios are possible, e.g. to define a VM for hosting

[1] Open Source Data Center Virtualization, information online: `http://opennebula.org`

[2] Open Stack Cloud Software, online via: `http://openstack.org`

[3] Kernel Based Virtual Machine, accessible via `http://www.linux-kvm.org`

[4] Xen Project: information online, `http://xenproject.org`

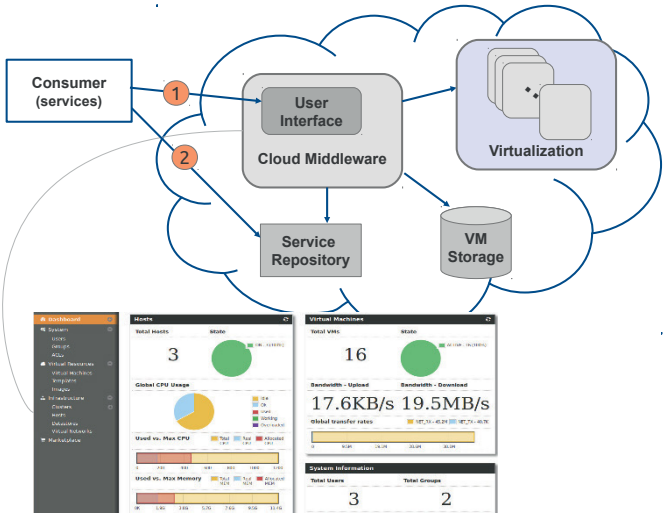


Fig. 16.2: User interaction with the Cloud infrastructure: (1) via its provided graphical user interface (small inset below) or (2) by calling a service from the repository via a provided client

databases, which later on can be accessed by a set of further VM's for computation tasks. The user of the Cloud system, or alternatively a registered service, can also access encrypted data at the home institution and to address said data from the running VM instance via standard Internet connection.

Based on the realized Cloud infrastructure, different cloud models can be realized. In the first place we use the system as a Private Cloud for testing purposes, but since the computing needs are potentially very large, other models can be realized as well.

In our case, which is based on OpenNebula, the middleware supports interfaces to access other Cloud infrastructures via servers. This allows for an elastic scaling of needed computing demands by using either the proprietary EC2 interface (e.g. to submit VM's to the Amazon Cloud) or the open quasi-standard OCCI [33].

Based on this basic model, we can realize not only a private Cloud model, but also extend our computing needs to other infrastructures – such as the local HPC cluster – and realize a hybrid Cloud solution for our services. This is a big advantage in terms of interoperability, especially for smaller companies to avoid a dependency to a particular Cloud provider. With such a flexible approach even a federated Cloud [9, 37] could be realized in order to build a fully functional Community Cloud, maintained by several providers.

16.2.2 Service Infrastructure

In general, there are versatile ways to develop and provide services in the Cloud, either by using vendor-specific platforms to create services or by using available open source APIs, e.g. libcloud[5] or SimpleCloud[6]. This marks a unique way to abstract the interaction with the virtualization layer in order to be able to interact directly with the VM for service invocation. In our reference architecture based on OpenNebula we can define services, which addresses the underlying hardware via the API's provide by the middleware, which are based on XML-RPC, Java or Ruby.

For service registering and invocation, we have also examined and set up a community solution based on the XML-RPC interface of OpenNebula. The SVMSched project [10] abstracts the service registry in an own server instance and replaces the OpenNebula scheduler by its own one. In this way, the project hides the configuration details from the user, who only needs to specify the hardware options for the used VM's and the path to the data by invoking the provided client. This has the advantage that arbitrary services can be run in a more generalized VM version, which is beneficial in terms of administration and scaling and distribution of multiple instances.

As already introduced in Figure 16.2, the user is than simple able to address a service via a client to the service repository. Based on our infrastructure, we can realize our services in different ways and stay very flexible, since the aimed services are technically highly heterogeneous.

16.3 Use Cases

In the following sections, we briefly describe three case studies from the application and security perspective for different domains: Patent annotation [18], cancer outcome prediction [43, 38], and drug target prediction [24, 23]. Please refer to the original studies for detailed information.

16.3.1 Text Mining

Text mining is used to automatically extract knowledge from text for various purposes such as improving document search or discovering new connections between entities. This section describes two different areas that we are researching to solve these challenges. The first one is the analysis of diverse corpora, especially patents,

[5] Python library to access various cloud provider APIs, information via `http://libcloud.apache.org/`

[6] PHP based API to Cloud infrastructure services, accessible via `http://simplecloud.org`

which are different from scientific publications. The second area is the extraction of relations between entities.

16.3.1.1 Patent analysis.

While many different platforms have been developed to offer advanced functionality for literature search (e.g. GoPubMed [17], GeneView [40]), they usually only provide the option to search scientific publications such as the abstracts from the MEDLINE database. Patents might be another valuable application for text mining. In contrast to biomedical literature, information in patents might be published earlier than actual publications. The reasons are manifold:

- patented information must not be revealed before granting the patent due to legal issues
- putative information, which are not fully proved, can be patented to get a broader claim

Some companies are concentrating on creating patents but they are not publishing in scientific journals (e.g. Oxis, Imusys). One such an example is the expression of the protein ELTD1 as biomarker for Glioblastoma. This discovery was published in late 2012, whereas the patent application was started in 2009 and was granted in early 2011. There are still many other patented biomarkers for Glioblastoma (e.g. NID1), that are currently not covered by MEDLINE abstracts.

We are developing GoPatents (see Figure 16.3), a new patent search platform, that will incorporate advanced search functionality to facilitate the search in 2 million full text patents and classify them into MeSH and IPC-Ontology [18].

The text processing is done using different algorithms. We use alignment algorithms [17], rule based approaches [22] and also supervised trained machines [16] for different types of concepts.

The processing of a patent requires 200 times more computational time than one PubMed abstract due to the longer text and also the more complex sentence structure. For a corpus of 2 million patents 6,000 CPU h are required, which requires a computer cluster for an efficient analysis.

Computational complexity is linear in the count of sentences of a document, but $\mathcal{O}(n^3)$ on the sentence level (n is the word count in a sentence). The sentence length has especially large impact on computational time in patents, since the sentences are usually longer compared to other publication types [42].

16.3.1.2 Sentence analysis.

In the biomedical domain, information about the relations between different entities, such as proteins, drugs and diseases, is extremely important. However, this information is usually not easily accessible from a database, it has to be collected from many different journal publications. We have developed a system that retrieves

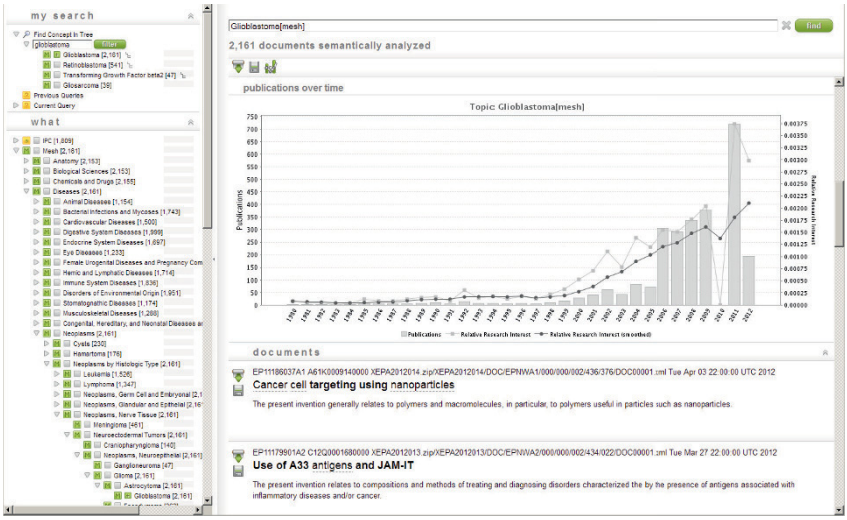


Fig. 16.3: GoPatents: search results (right lower side) are automatically classified into ontologies (left side) and statistically analyzed (right upper side).

a list of statements about the relation between two different entities of the given drugs, proteins, and diseases. After a search for two entities, all documents are automatically fetched and all sentences with both entities are extracted. These sentences are ranked according to their importance using a maximum entropy classifier. While such analysis for combinations of drugs, proteins or diseases is already available, we are developing an online application that identifies all counterparts for one known entity. As these calculations are computationally intensive (appr. 3 CPU months for 22 million MEDLINE abstracts), access to a Cloud environment can shorten the overall execution time considerably due to parallel processing of search tasks.

In the future, we want to develop an online application that runs an analysis with only one known entity to identify all its unknown counterparts. This will imply a huge amount of processing time, hence need to be run on a Cloud environment. Currently, a demonstrator for a few thousand abstracts is available. However, we need to adjust this application for the Cloud environment to run the analysis for the whole MEDLINE database.

16.3.1.3 Security Analysis.

In such literature and patent search platforms, the query of a user is assumed to be confidential, since the user might validate some experimental or original results by searching available resources. Hence, such search platforms should guarantee the confidentiality of input and output data in the Cloud environment. The algorithms used for extracting information out of literature are rather involved and complex in

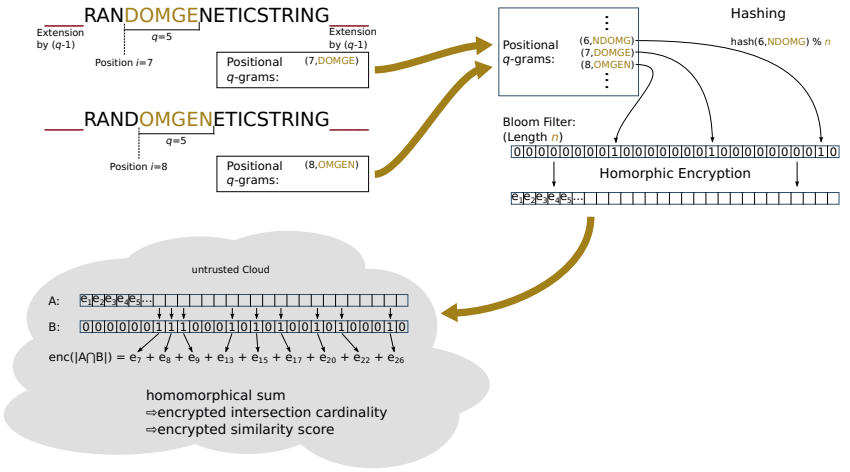


Fig. 16.4: Secure text matching with Bloom filters and homomorphic encryption, achieving a linear complexity in the longest string and secure computation at untrusted parties

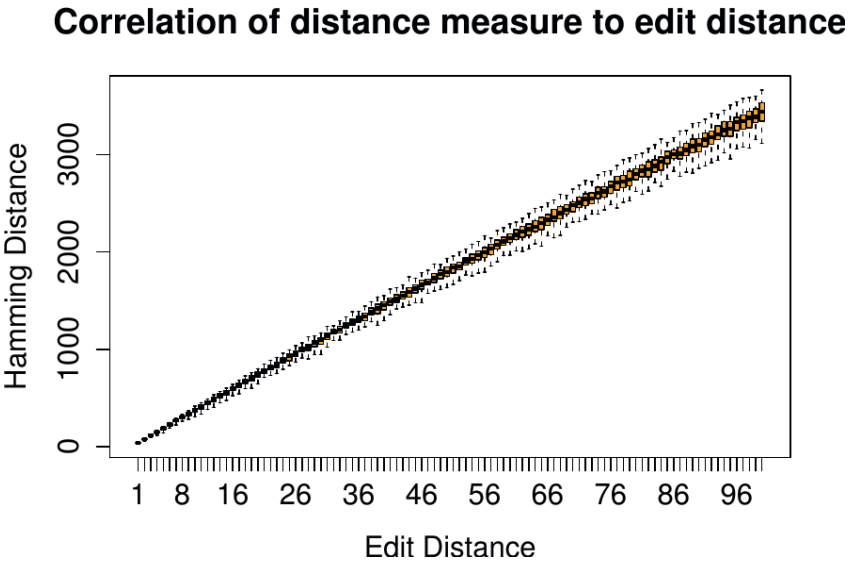


Fig. 16.5: Correlation of the approximated Levenshtein distance to the actual Levenshtein distance (Edit distance) for different distances

terms of converting them into privacy-respecting versions. Some useful parts were already discussed within the security community, like homomorphically encrypted

machine learning [30], oblivious execution of finite state machines [41] and oblivious evaluation of hidden Markov models [20].

A simpler primitive that is always needed is secure string comparison as known from private record linkage [7] and DNA matching [28]. All these proposals do not scale well with increased string lengths, so we built a privacy-preserving scheme that approximates the Levenshtein Distance and has linear complexity in the longest string [4]. In the following sections we focus on the Levenshtein distance, which defines the minimum number of *insert*, *replace* and *delete* operations necessary to transform one text string into another. Figure 16.4 shows an overview of the secure matching algorithm.

The basic idea is that searching for a text string is similar to evaluating the local similarity of a search query against a document. There are many means to calculate such a similarity like the Levenshtein distance, Smith-Waterman or other dynamic programming algorithms. These are approximated by transforming the query and the document into sets by calculating variable length grams upon them and checking the dissimilarity of those sets.

As we want to arrive at a privacy preserving comparison method, the sets are again approximated into binary vectors representing them. Upon these vectors set intersections and unions are very easy and - more importantly - very fast in linear time to be computed. This is done by calculating element-wise the AND or OR operation upon the binary values between the two vectors. Our algorithm uses the dissimilar elements and thus computes the XOR between the binary vectors and further the cardinality of the resulting set representation. Fortunately, the XOR and cardinality calculation can both be evaluated using an additive homomorphic cryptosystem, thus making it privacy preserving and still be rather efficient. The resulting encrypted cardinality correlates very well with the edit distance between the two original strings (Figure 16.5). For each Levensthein distance, 100 sequence pairs with that distance were choosen randomly from the test set and our approximated distance value was calculated upon them. These 100 approximated distances are printed as box plots for each Edit distance value to show the distribution of the results and how their mean increases nearly linear with the actual Edit distance. Further the standard deviation is rather low and increases slowly with increasing edit distances.

The result will only be accessible to one of the two parties, which fits nicely into the Cloud Computing scenario, where we want to borrow computational capacity from a remote machine that we do not trust.

In cases where approximate search is not applicable and exact matches are necessary, we apply two solutions to perform secure searches. One version uses the results from Blass *et al.* [5], who transform huge documents into hashed tables and apply techniques from single-server private information retrieval for deciding, if a searched term can be found within such a document. The other solution is a trivial one, which takes documents and search terms as input. Both are split into appropriate tokens: words, word based n-grams or character based n-grams. All tokens are processed using a keyed hash, and sent to the Cloud for executing the actual search – a one to one comparison between all documents and query tokens. The matches

are returned to the client. This solution is potentially faster compared to Blass *et al.* [5].

Building upon these exact and approximate matching blocks, privacy-preserving versions of simple text mining algorithms can be constructed to securely extract information out of confidential literature like full text corpora of patents or scientific publications.

16.3.2 Cancer Outcome Prediction

High-throughput experiments aim to explore predictive signature genes that would provide identification of clinical outcome of diseases. Microarray data analysis helps to reveal underlying biological mechanisms of tumor progression, metastasis, and drug-resistance in cancer studies. The field of cancer outcome prediction aims to address the problem of learning how to forecast disease progression from gene expression and thus allows refined treatment options. The gene signatures obtained in such analyzes could be addressed as biomarkers for cancer progression.

The experimental or computational noise in data and limited tissue samples collected from patients might reduce the predictive power and biological interpretation of such signature genes. Network information efficiently helps to improve outcome prediction and reduce noise in microarray experiments [19]. For this purpose, network information has been integrated with microarray data in various studies in the last decade.

Results.

We developed an algorithm – called NetRank – that employs protein-protein interaction networks and ranks genes by using the random surfer model of Google's PageRank algorithm. The performance of the algorithm was evaluated in two studies. In the first study, NetRank was applied on gene expression data obtained from 30 pancreas cancer patients and seven candidate biomarker genes could be identified [43]. The goal of this study was to predict the survival time of patients after tumor removal. Compared to the genes found by other methods, NetRank improves the outcome prediction accuracy by 7%. When experimentally validating the prognostic value of the seven biomarkers on new samples, a 70% accuracy was achieved, which is superior to established clinical prognostic factors.

The second study was a systematic evaluation of network-based outcome prediction on 25 cancer datasets [38]. The datasets were diverse as they cover 13 different types of cancer. They also contain different outcome variables e.g. response to treatment, prediction of cancer progression, tumor subtyping, initial diagnosis. In this study, NetRank performed better than random signatures, leading to an average improvement of 12%. In nearly all cases, NetRank was also better than the classical methods (i.e. *t*-test, fold change), with an average improvement of 6%. The

results prove that fully automated integration of network information and expression data leads to more accurate outcome predictions compared to the signatures of previous studies and classical methods. Network-based gene expression analysis provides more detailed understanding of cancer-related processes. In addition, reproducibility of biomarker signatures found by NetRank significantly increases between different datasets of the same cancer type (data not shown).

Algorithm.

The NetRank algorithm assigns a rank value to a gene that is based on a combination of its score (gene expression) and the scores of genes linked to it. Thus, the rank r_j^n of gene j in the n^{th} iteration is defined as:

$$r_j^n = (1-d)s_j + d\sum_{i=1}^{N} \frac{m_{ij}\, r_j^{n-1}}{deg_i} \qquad 1 \leq j \leq N \tag{16.1}$$

where $d \in (0,1)$ is a parameter describing the influence of the network on ranking of genes, s is the gene expression value, $m \in \mathbb{R}^{N\times N}$ is a symmetric adjacency matrix for the gene network and deg is the degree of genes in the network. Basically, if a gene is linked to other highly ranked genes, it will also get a high rank due to the boosting effect of neighboring genes. Hence, the algorithm provides an integration of the topological information (i.e. connectivity and random walk) and microarray data (i.e. node score) to explore crucial signature genes for outcome prediction.

NetRank performs a supervised learning process combined with a Monte Carlo cross-validation scheme to predict biomarker genes (Figure 16.6). The full dataset is a gene expression matrix with the genes as rows and the patients as columns. For each patient, the outcome (poor or good) is given (1). The dataset is randomly divided into a training and a test set (2). Within the training set, genes are ranked by how different they are between patients with poor and good outcome (3). The most different genes are selected (4). These genes are used to train a machine learning classifier on the training set (5). After training, the classifier is asked to predict the outcome of the test set patients (6). The predicted outcome is compared with the true outcome and the number of correctly classified patients is counted (7). Steps $2-7$ are repeated 1,000 times, and the resulting final accuracy is obtained by averaging over the 1,000 accuracies of Step 7.

The parameter d in Equation 16.1 is set as a part of the Monte Carlo cross-validation workflow. For this purpose, we added an additional inner cross-validation loop (see Figure 16.6, Steps 2a to 2d). In this inner cross-validation, a part of the training set samples was set aside, and different values of d ranging from 0 to 1 in steps of 0.1 were used to run NetRank on the remaining training set samples. Accuracies of the top-ranked genes were then obtained on the samples previously set aside. As a result of the inner cross-validation, a d value was chosen and used for deriving a signature based on the whole training set, and evaluating it's accuracy on the test set. It is important to note that no information of the test set is used for

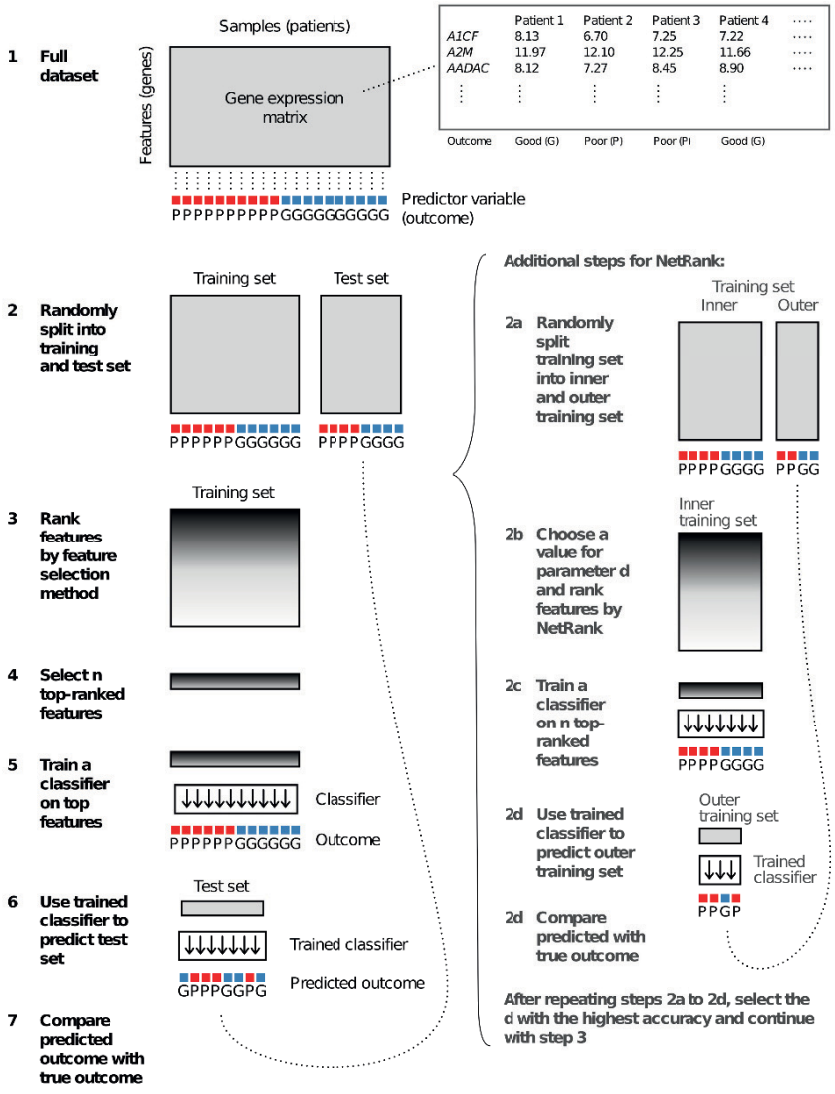


Fig. 16.6: Workflow of NetRank. Figure is adapted from [43]

the selection of d, so the value of d in the inner cross-validation does not rely on any prediction accuracy in the test set data.

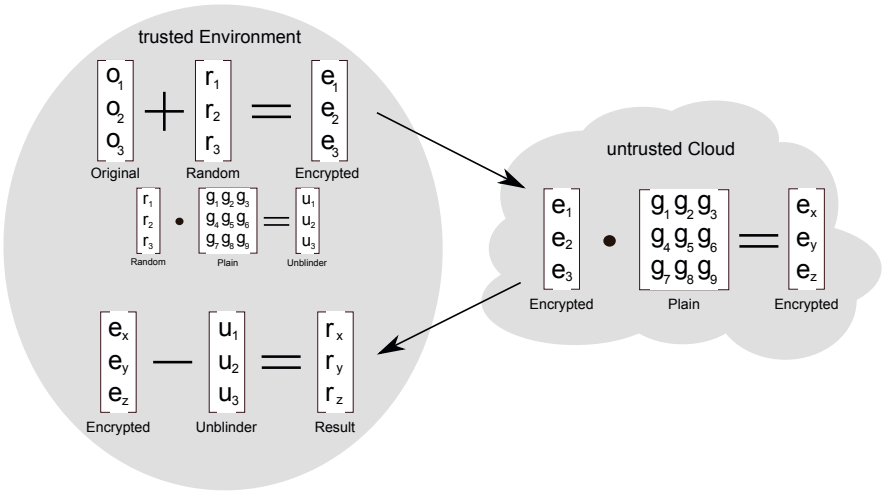


Fig. 16.7: Randomization of vectors for secure outsourcing of matrix-vector multiplications

Complexity.

The main component of the algorithm is a matrix (network) – vector (patient data) multiplication, thus having a computational complexity of approximately $\mathcal{O}(n^2)$. Depending on the amount of patients and network size, the running time of one biomarker prediction can easily reach up to 20,000 CPU h. In order to reduce this running time each cross-validation step is submitted to the Cloud as one worker process. The gene expression data obtained from patients and some type of interaction networks are highly confidential. Therefore, the security of the input data and the predicted biomarkers should be guaranteed in the Cloud environment. Extra computation time is needed depending on the applied security approach (e.g. blinding, homomorphic encryption, randomization) on the data.

For an efficient cross-validation the network and patient data are loaded into memory, which introduces a memory-related overhead. In the current version, running a matrix-vector multiplication with a network of approximately 100,000 edges, consumes at least 1GB of memory. For multiple networks this might be potentially growing up to TBs, making the memory consumption the major bottleneck of the algorithm. Since these factors cause the algorithm to be very memory intensive, an optimization is necessary before efficiently running it on a public Cloud.

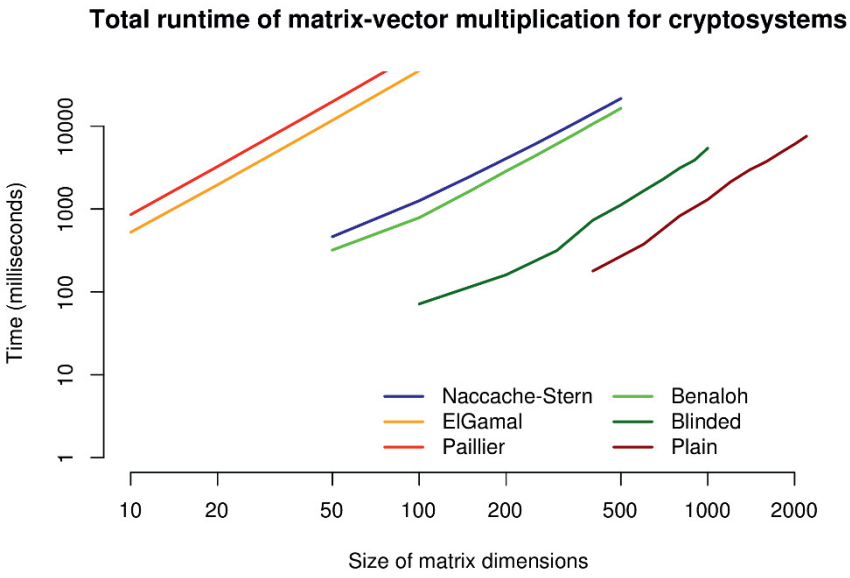


Fig. 16.8: Runtime of encryption, calculation and decryption for a single matrix-vector multiplication of different cryptographic systems for several matrix sizes

16.3.2.1 Security Analysis.

At the core of the NetRank algorithm are two important primitives, matrix-vector multiplications and machine learning algorithms working on confidential data. Privacy-preserving solutions for matrix-vector multiplications are available based on different security assumptions like semi-honest multi-party computation [2] or blinding [1, 3]. We implement a solution using blinding, which achieves a high efficiency and increased privacy, but still leaks some information. Protecting privacy will be implemented by perturbing the data of either the matrix or the vector, not both at the same time. The perturbation is performed by adding random values upon the sensitive data. This step needs to be performed within a trusted environment.

Depending on the actual data this randomization can be performed for example over the real numbers in $\mathbb{R}$, in which case it is necessary to choose an appropriate distribution for the random numbers. Working in $\mathbb{R}$ will leak information as the perturbed distribution can be compared with the expected distribution and assumptions can be made based on their differences.

Another approach would be to represent all data in $\mathbb{Z}_p$, the finite field over the integers modulo a prime p. For practical considerations, p needs to be large enough to hold the resulting values of the matrix multiplication without overflowing. This can however be estimated very easily by taking the size of the vector s_v, its largest

element $v_{\max}$ and the largest element in the matrix $m_{\max}$. We now choose a $p > s_v \cdot v_{\max} \cdot m_{\max}$. The choice of the random distribution is easy, as the random values need to be uniform over $\mathbb{Z}_p$ in order to perfectly hide all information about the original distribution of the data and the actual values itself.

Based on the perturbed data we can efficiently calculate the matrix-vector multiplication in the untrusted Cloud environment. The blinded result is transferred back to the trusted systems and decrypted via subtraction by a prepared unblinding vector (Figure 16.7).

Another implemented solution is based on additive homomorphic encryption [34]. This approach also perfectly preserves privacy, as the blinding in $\mathbb{Z}_p$ does, but is less efficient. It comes in two possible flavors, using an additive encryption scheme like [34] to protect one side of the matrix multiplication, similar to the blinding, or a scheme from pairing-based cryptography [6], that protects all input data with less efficiency. The advantage of using homomorphic cryptography over blinding is that we do not need to perform the unblinding preparation on the trusted party all the time.

A performance comparison of different additive homomorphic cryptographic systems against the blinding solution and no privacy can be seen in Figure 16.8. The x-axis describes the size of the used square matrices. A value of 500 therefore denotes that matrices with 500 rows and 500 columns were used. Each curve represents a single cryptographic system and on the y-axis one can see the overall processing time throughout all steps necessary to do a secure matrix vector multiplication. This includes encryption, the operations itself and the decryption. The key setup time however is omitted, as this is a constant offset and does not depend on the size or number of matrices used. One can easily see that curves further on the right side of the plot are working on larger matrices within the same amount of time and thus are more efficient. The fastest algorithm is obviously the plain calculation, without any security enhancements. The blinding solution is roughly 4.1 times slower on average compared to the plain version, but preserves privacy pretty good, as it works in $\mathbb{Z}_p$, as discussed above. The cryptographic algorithms were all set to use the same security parameter. As fastest cryptographic solutions, the Benaloh and Naccache-Stern systems are very close to each other. This is of course not surprising, as their construction is very similar and the Naccache-Stern system can be reduced to simulate the Benaloh system. This usage of cryptography is again roughly 15.4 times slower on average than blinding. The use of these two systems is therefore anticipated if cryptographic guarantees are necessary. ElGamal and Paillier are the slowest systems in this benchmark, both due to the fact that they either double the length of the ciphertext (Paillier), or use two ciphertexts to represent a single value (ElGamal). Depending on the requirements the speed factors between the cryptographic systems might change if the speed of the encryption, actual calculation or decryption is of higher importance.

The machine learning part of NetRank was also targeted by specific privacy-preserving solutions. Graepel *et al.* [30] propose a new class of machine learning algorithms based on leveled homomorphic encryption. New binary classifiers are build with a multiplicative circuit depth of low polynomial degree. This ensures

rather efficient evaluation through somewhat homomorphic encryption schemes. Another efficient and less privacy-preserving solution based on blinding is to randomize the data, while preserving important properties like the order and approximate relative distance. The blinding based approach also yields comparable classification results on the original data with nearly no performance impact. The achievable privacy margin is on the opposite rather low.

16.3.3 Drug Target Prediction from Binding Site Similarity

Drug development is a very costly and failure-prone endeavor with about 81% of the drugs failing [15]. Major causes for failure are lack of drug efficacy and off-target binding. Thus, drug target prediction is a valuable tool to support drug discovery, helping to predict disadvantageous off-targets and finding new uses for already known drugs. The latter strategy is termed drug repositioning (saving up to 40% of drug development costs [11]) and yielding treatment options for tropical and rare diseases [24]. We implemented a structural bioinformatics drug target prediction and assessment pipeline. It uncovers local similarities between pairs of protein structures and scores the alignments according to the spatial proximity of the bound drug ligands [23].

Protein structures and bound drugs are represented in atomic coordinate files from the Protein Data Bank (PDB). Atoms are points in space and sparsely (implicitly or explicitly) connected by atomic bonds. The spatial atom arrangement is abstracted to a graph, with groups of atoms as nodes and distances between the atoms as edge labels [24, 44]. The local alignment of a pair of protein structures is then given by the maximum weight common subgraph of their graph representation abstracted from the molecular structure.

Here, we faced two problems: First, this task is computationally expensive since the subgraph search is NP-hard (100 CPU h per 1000 alignments on average) and second, $n = 2,284$ PDB structures had to be aligned pair-wise (giving $n_a = n(n-1) \cdot 2^{-1} = 2,607,186$ alignments) and read from disk. For performance reasons, operations on files were performed on a RAM disk, which can optionally be encrypted for increased security. The n_a alignments $\{a_1, \ldots, a_{n_a}\}$ were split into j jobs such that the number of accessed coordinate files was minimized for each job, thus reducing concurrent read operations to a minimum. This was achieved by splitting the upper triangular matrix of the n_a alignments into j equally sized submatrices. The submatrices were optimized in size due to the triangular form of the full matrix. Suppose such a submatrix has $lm \leq l^2 \approx \frac{n_a}{j} = n'_a$ elements with $l \geq m$ and $l - m \leq 2$. Thus, $l + m = n'$ coordinate files have to be loaded to the RAM disk to compute n'_a alignments. In total, the number of disk read accesses for j jobs is bound by $jn' = j(l+m) \leq 2jl = 2j\sqrt{\frac{n_a}{j}} = 2j\sqrt{\frac{n(n-1)}{2j}} = \sqrt{2j}\sqrt{n(n-1)} \leq n\sqrt{2j} \in \mathcal{O}(n)$ for $j \ll n$ and constant, which is optimal. Therefore, the disk read overhead is constant with a factor of $\sqrt{2j}$ and thus dependent on the number of jobs j. In con-

trast, a naïve approach without the RAM disk would need $n_a \in \mathcal{O}(n^2)$ disk reads. However, j has to be set in such a way, that n' coordinate files can be stored on the RAM disk.

After having computed the local structure alignments, successful alignments were scored by the quality of the spacial superposition of their ligands. This measure is termed LigandRMSD [23] and computes the difference in root mean square deviation (RMSD) in comparison to an optimal alignment of the ligands. The RMSD is computed from atomic distances (d is the Euclidean distance) of matched atoms a and a' in a set of matched atoms $A = \{(a_0, a'_0), \ldots, (a_n, a'_n)\}$:

$$\text{RMSD} = \sqrt{\frac{1}{|A|}\sum_{i=1}^{|A|} d(a_i, a'_i)^2}$$

However, for the scoring of the superposition of two protein structures, the two molecular graphs – of the bound ligand molecules in each of the structures – have to be matched, giving mapped atoms (a, a') being used for the RMSD calculation. In the case of identical graphs or subgraphs (ligands L, L' with $L' \subseteq L$) this corresponds to a graph isomorphism, computed with OpenBabel/Pybel [32]. For dissimilar ligands, the maximum common subgraph is computed with the Small Molecule Subgraph Detector [35]. Having the mapping between the two ligands L and L', two RMSDs are calculated. RMSD$'$ is the RMSD of the bound ligands given by the protein structure superposition as computed above. Subsequently, the RMSD$''$ is obtained from an optimal superposition of the ligands, i.e. the coordinates of ligand L' are transformed such that RMSD$''$ is minimal. Finally, LigandRMSD $=$ RMSD$'$ $-$ RMSD$''$.

16.3.3.1 Security Analysis.

We first tried to build a privacy-enhanced version of the binding site-alignment step discussed in the previous section. However, the used tools and algorithms are not available for modification and can only be used in a black-box setting, preventing the development of a privacy-preserving version. As it is still essential to enhance the security while working with black-box algorithms, we analyzed the input and output data to reduce it to a minimum. Protein structure determination is costly and time consuming. Thus, unpublished structures have to be secured appropriately. Besides that, predicted drug targets are potentially valuable and should be secured. In case of the binding site alignment tool, structural information about the compared proteins are supplied as input in form of PDB files. These files carry much more information than needed for computation of the binding site alignment. Thus, we apply a series of filters to these files to remove all but the essential information as follows:

First, remove all lines that are no atom or ligand entries and all entries with coordinates further than d Å away from any ligand's center of mass. This is followed by a removal of all entries, which are further than $d' < d$ Å away from any ligand

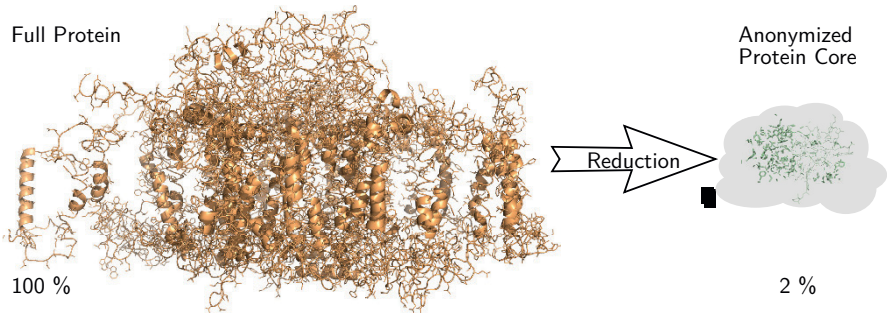


Fig. 16.9: Full protein structures are anonymized locally for processing in the Cloud.

atom. Second, permute atoms randomly per amino acid and amino acids per chain. Third, rotate and translate all entries randomly and set all columns except for the coordinates, chain, atom, and residue number to zero. In a nutshell we applied the methods of fuzzing and removal of information to only keep the absolutely necessary information.

We applied the described method to a set of proteins and removed on average 91% of the structural information, while the binding site alignment was still calculating correct results. Figure 16.9 shows an example of what remains after the anonymization step. A test set of 2,171 protein structures was anonymized this way. To test, how easy these structures can be recovered, we ran a BLAST search over all PDB entries. This search returned the anonymized structure on average at position 25 on the ranked result list, while 182 were not found. Thus, this approach offers no full security, but instead hides the results making it tedious for an attacker to recover the actual hits.

16.4 Results and Conclusions

We described three different biomedical applications (text/patent mining, cancer outcome prediction and drug target prediction) in a Cloud environment. The underlying infrastructure was set up particularly with these applications in mind and the employed Cloud middleware is purely based on open standards and capable of easy deployment of virtual machines e.g. to the local HPC cluster or other Cloud providers. Particular focus was put on how to secure these approaches to work privacy-preserving. This is important when computing is done on valuable research data or on patient data. We applied different approaches on securing the computation as well as the input/output data: Homomorphic encryption, hiding, blinding and anonymization.

We secured our text mining algorithms using approximate matching and homomorphic encryption such that information can be securely retrieved from confiden-

tial texts without increasing the computational complexity. The cancer outcome prediction – running on confidential medical data – was secured with blinding in one approach and additive homomorphic encryption in another. Although the latter offers best security while still being in $\mathcal{O}(n^2)$, we preferred the first for production use. This is due to a constant factor in the order of 10^6 introduced by the homomorphic encryption, making it slow in practice. The drug target prediction approach involves black-box like libraries, which could not be secured. Instead, we concentrated on reducing the input data to a minimal amount leading to an anonymization of the input coordinate files. This makes the mapping of the results back to the original data tedious for an attacker without increasing computational complexity.

As the proposed security solutions are independent of the actual Cloud infrastructure, the services can be run on virtually any Cloud platform. This allows more flexibility, interoperability and efficiency compared to platform dependent solutions. The implemented solutions are tailored to the applications, but the ideas can easily be adopted to build security solutions for similar tasks. As a result, the security risks in using public Cloud Computing upon private data decreases and new applications might arise.

Due to the fact that our main security concern is the integrity of the Cloud provider itself, or rather the possibility of an unauthorized person within the Cloud provider accessing viable information, we do not depend or rely on security and privacy mechanisms implemented by that potentially untrusted party. However, if the respective provider is trusted to a certain degree, applied security mechanisms and contractual bindings become much more important, as partly discussed in section 16.1.

16.5 Acknowledgements and Funding

Funding by the German Federal Ministry of Economics and Technology is kindly acknowledged as GeneCloud is part of the Trusted Cloud Initiative.

References

1. Atallah, M., Pantazopoulos, K., Spafford, E.: Secure outsourcing of some computations. Computer pp. 1–24 (1996). URL http://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=2327\&context=cstech
2. Atallah, M.J., Frikken, K.B.: Securely outsourcing linear algebra computations. In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security - ASIACCS '10, p. 48. ACM Press, New York, New York, USA (2010). DOI 10.1145/1755688.1755695. URL http://dl.acm.org/citation.cfm?id=1755688.1755695
3. Atallah, M.J., Pantazopoulos, K.N., Rice, J.R., Spafford, E.H.: Secure outsourcing of scientific computations. ADVANCES IN COMPUTERS pp. 215 – 272 (1998). URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.158.9763

4. Beck, M., Kerschbaum, F.: Approximate two-party privacy-preserving string matching with linear complexity (2013)
5. Blass, E., Pietro, R.D., Molva, R., Önen, M.: PRISM–Privacy-Preserving Search in MapReduce. Privacy Enhancing Technologies (2012). URL http://www.springerlink.com/index/0720TJP211W4Q6U7.pdf
6. Boneh, D., Goh, E.J., Nissim, K.: Evaluating 2-DNF formulas on ciphertexts. Theory of Cryptography (TCC) '05 **3378**, 325–341 (2005). URL http://www.springerlink.com/index/wtt5caxkr94laxkg.pdf
7. Bonomi, L., Xiong, L., Chen, R., Fung, B.: Privacy Preserving Record Linkage via grams Projections. arXiv preprint arXiv:1208.2773 (2012). URL http://arxiv.org/abs/1208.2773
8. Bugiel, S., Nürnberger, S., Sadeghi, A.R., Schneider, T.: Twin Clouds: Secure Cloud Computing with Low Latency. In: 12th Communications and Multimedia Security Conference (CMS'11), *LNCS*, vol. 7025. Springer (2011). URL http://thomaschneider.de/papers/BNSS11.pdf
9. Buyya, R., Ranjan, R., Calheiros, R.: InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services. In: C.H. Hsu, L. Yang, J. Park, S.S. Yeo (eds.) Algorithms and Architectures for Parallel Processing, *Lecture Notes in Computer Science*, vol. 6081, pp. 13–31. Springer Berlin / Heidelberg (2010). URL \url{http://dx.doi.org/10.1007/978-3-642-13119-6_2}
10. Chakode, R., Yenke, B.O., Mehaut, J.F.: Resource Management of Virtual Infrastructure for On-demand SaaS Services. In: CLOSER2011: Proceedings of the 1st International conference on Cloud Computing and Service Science, pp. 352–361 (2011)
11. Chong, C.R., Sullivan, D.J.: New uses for old drugs. Nature **448**(7154), 645–646 (2007). DOI 10.1038/448645a. URL http://dx.doi.org/10.1038/448645a
12. Chuang, H.Y., Lee, E., Liu, Y.T., Lee, D., Ideker, T.: Network-based classification of breast cancer metastasis. Mol Syst Biol **3**, 140 (2007). DOI 10.1038/msb4100180. URL http://dx.doi.org/10.1038/msb4100180
13. Cloud Security Alliance: Top Threats to Cloud Computing (March), 1–14 (2010)
14. Daminelli, S., Haupt, V.J., Reimann, M., Schroeder, M.: Drug repositioning through incomplete bi-cliques in an integrated drug-target-disease network. Integr Biol (Camb) **4**(7), 778–788 (2012). DOI 10.1039/c2ib00154c. URL http://dx.doi.org/10.1039/c2ib00154c
15. DiMasi, J.A., Feldman, L., Seckler, A., Wilson, A.: Trends in risks associated with new drug development: success rates for investigational drugs. Clin Pharmacol Ther **87**(3), 272–277 (2010). DOI 10.1038/clpt.2009.295. URL http://dx.doi.org/10.1038/clpt.2009.295
16. Doms, A.: GoPubMed : Ontology-based literature search for the life sciences. Dresden, Techn. Univ., Diss., 2009 (2008). URL http://dx.doi.org/10.1073/pnas.0704422105http://d-nb.info/992853664
17. Doms, A., Schroeder, M.: Gopubmed: exploring pubmed with the gene ontology. Nucleic Acids Res **33**(Web Server issue), W783–W786 (2005). DOI 10.1093/nar/gki470. URL http://dx.doi.org/10.1093/nar/gki470
18. Eisinger, D., Tsatsaronis, G., Bundschus, M., Wieneke, U., Schroeder, M.: Automated patent categorization and guided patent search using ipc as inspired by mesh and pubmed. Journal of Biomedical Semantics **4**(Suppl 1), S3 (2013). DOI 10.1186/2041-1480-4-S1-S3. URL http://www.jbiomedsem.com/content/4/S1/S3
19. Fortney, K., Jurisica, I.: Integrative computational biology for cancer research. Hum Genet **130**(4), 465–481 (2011). DOI 10.1007/s00439-011-0983-z. URL http://dx.doi.org/10.1007/s00439-011-0983-z
20. Franz, M., Deiseroth, B., Hamacher, K., Jha, S., Katzenbeisser, S., Schröder, H.: Towards Secure Bioinformatics Services. Financial Cryptography and Data Security - Lecture Notes in Computer Science **7035**, 276–283 (2012)
21. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st annual ACM symposium on Theory of computing, STOC '09, pp. 169–178. ACM, New York,

NY, USA (2009). DOI http://doi.acm.org/10.1145/1536414.1536440. URL http://doi.acm.org/10.1145/1536414.1536440
22. Hakenberg, J., Plake, C., Royer, L., Strobelt, H., Leser, U., Schroeder, M., Group, A.B., Centre, B., Dresden, T.U., Knowledge, G.: Open Access Gene mention normalization and interaction extraction with **9**(Suppl 2) (2008). DOI 10.1186/gb-2008-9-S2-S14
23. Haupt, V.J., Daminelli, S., Schroeder, M.: Drug promiscuity in PDB: Protein binding site similarity is key. PLoS One (2013)
24. Haupt, V.J., Schroeder, M.: Old friends in new guise: Repositioning of known drugs with structural bioinformatics. Brief Bioinform **12**(4), 312–326 (2011). DOI 10.1093/bib/bbr011. URL http://dx.doi.org/10.1093/bib/bbr011
25. Jansen, W., Grance, T.: NIST Special Publication 800-144, Guidelines on Security and Privacy in Public Cloud Computing (2011)
26. Johannes, M., Brase, J.C., Fröhlich, H., Gade, S., Gehrmann, M., Fälth, M., Sültmann, H., Beißbarth, T.: Integration of pathway knowledge into a reweighted recursive feature elimination approach for risk stratification of cancer patients. Bioinformatics **26**(17), 2136–2144 (2010). DOI 10.1093/bioinformatics/btq345. URL http://dx.doi.org/10.1093/bioinformatics/btq345
27. Kann, M.G.: Advances in translational bioinformatics: computational approaches for the hunting of disease genes. Brief Bioinform **11**(1), 96–110 (2010). DOI 10.1093/bib/bbp048. URL http://dx.doi.org/10.1093/bib/bbp048
28. Katz, J.: Secure text processing with applications to private DNA matching. Proceedings of the 17th ACM conference on **1**, 485–492 (2010). URL http://dl.acm.org/citation.cfm?id=1866361
29. Krishnan, A.: Gridblast: a globus-based high-throughput implementation of blast in a grid computing framework. Concurrency and Computation: Practice and Experience **17**(13), 1607–1623 (2005). DOI 10.1002/cpe.906. URL http://dx.doi.org/10.1002/cpe.906
30. Kwon, T., Lee, M.K., Kwon, D.: ML Confidential: Machine Learning on Encrypted Data. Information Security and Cryptology – ICISC 2012 Lecture Notes in Computer Science **7839**, 1–21 (2013). DOI 10.1007/978-3-642-37682-5. URL http://link.springer.com/10.1007/978-3-642-37682-5
31. Morgan, J.C., Chapman, R.W., Anderson, P.E.: A next generation sequence processing and analysis platform with integrated cloud-storage and high performance computing resources. In: Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine, BCB '12, pp. 594–594. ACM, New York, NY, USA (2012). DOI 10.1145/2382936.2383033. URL http://doi.acm.org/10.1145/2382936.2383033
32. O'Boyle, N.M., Morley, C., Hutchison, G.R.: Pybel: a python wrapper for the openbabel cheminformatics toolkit. Chem Cent J **2**, 5 (2008). DOI 10.1186/1752-153X-2-5. URL http://dx.doi.org/10.1186/1752-153X-2-5
33. Open Grid Forum Working Group: OCCI – Open Cloud Computing Interface (2009)
34. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. Advances in Cryptography - Eurocrypt '99 **1592**, 223–238 (1999). DOI 10.1007/3-540-48910-X. URL http://www.springerlink.com/index/10.1007/3-540-48910-X
35. Rahman, S.A., Bashton, M., Holliday, G.L., Schrader, R., Thornton, J.M.: Small molecule subgraph detector (smsd) toolkit. J Cheminform **1**(1), 12 (2009). DOI 10.1186/1758-2946-1-12. URL http://dx.doi.org/10.1186/1758-2946-1-12
36. Ren, K., Wang, C., Wang, Q.: Security Challenges for the Public Cloud. IEEE Internet Computing **16**(1), 69–73 (2012). DOI 10.1109/MIC.2012.14. URL http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6123700
37. Rochwerger, B., Breitgand, D., Epstein, A., Hadas, D., Loy, I., Nagin, K., Tordsson, J., Ragusa, C., Villari, M., Clayman, S., Levy, E., Maraschini, A., Massonet, P., Muñ andoz, H., Tofetti, G.: Reservoir - When One Cloud Is Not Enough. Computer **44**(3), 44 –51 (2011). DOI 10.1109/MC.2011.64
38. Roy, J., Winter, C., Isik, Z., Schroeder, M.: Network information improves cancer outcome prediction. Brief Bioinform (2012). DOI 10.1093/bib/bbs083. URL http://dx.doi.org/10.1093/bib/bbs083

39. Santos, N., Gummadi, K.P., Rodrigues, R.: Towards trusted cloud computing. In: Proceedings of the 2009 conference on Hot topics in cloud computing, HotCloud'09, p. 3. USENIX Association, Berkeley, CA, USA (2009). URL http://portal.acm.org/citation.cfm?id=1855533.1855536
40. Thomas, P., Starlinger, J., Vowinkel, A., Arzt, S., Leser, U.: Geneview: a comprehensive semantic search engine for pubmed. Nucleic Acids Res **40**(Web Server issue), W585–W591 (2012). DOI 10.1093/nar/gks563. URL http://dx.doi.org/10.1093/nar/gks563
41. Troncoso-Pastoriza, J.R., Katzenbeisser, S., Celik, M.: Privacy preserving error resilient dna searching through oblivious automata. In: Proceedings of the 14th ACM conference on Computer and communications security - CCS '07, p. 519. ACM Press, New York, New York, USA (2007). DOI 10.1145/1315245.1315309. URL http://dl.acm.org/citation.cfm?id=1315245.1315309
42. Verberne, S., D'hondt, E., Oostdijk, N.: Quantifying the Challenges in Parsing Patent Claims. 1st International Workshop on Advances in Patent Information Retrieval (2003)
43. Winter, C., Kristiansen, G., Kersting, S., Roy, J., Aust, D., Knösel, T., Rümmele, P., Jahnke, B., Hentrich, V., Rückert, F., Niedergethmann, M., Weichert, W., Bahra, M., Schlitt, H.J., Settmacher, U., Friess, H., Büchler, M., Saeger, H.D., Schroeder, M., Pilarsky, C., Grützmann, R.: Google goes cancer: improving outcome prediction for cancer patients by network-based ranking of marker genes. PLoS Comput Biol **8**(5), e1002,511 (2012). DOI 10.1371/journal.pcbi.1002511. URL http://dx.doi.org/10.1371/journal.pcbi.1002511
44. Xie, L., Bourne, P.E.: Detecting evolutionary relationships across existing fold space, using sequence order-independent profile-profile alignments. Proc Natl Acad Sci U S A **105**(14), 5441–5446 (2008). DOI 10.1073/pnas.0704422105. URL http://dx.doi.org/10.1073/pnas.0704422105
45. Yao, A.C.C.: How to generate and exchange secrets. In: Foundations of Computer Science, 1986., 27th Annual Symposium on, pp. 162–167 (1986). DOI 10.1109/SFCS.1986.25

Part IV
Social Aspects, Business Models and Standards

Chapter 17
Designing a Business Model for a Cloud Marketplace for Healthcare

Nicolai Hanner, Tatiana Ermakova, Jonas Repschlaeger, and Ruediger Zarnekow

Abstract Cloud platforms providing healthcare cloud services are supposed to start a new era in the healthcare industry. Nevertheless, appropriate marketplace business models have not been examined in this domain so far. This paper contributes by presenting a first business model draft for such a marketplace. Following the ac-tion research principles, we collaborate closely with hospital workers, software providers and developers. Based on the existing literature and findings from the TRESOR research project on the adoption of cloud computing in healthcare, by using the business model canvas we create a business model showing the main value proposition, the customer segments and the value architecture of the cloud marketplace. Furthermore the problem of multisided markets is addressed, as the business models show how it can be prevented.

17.1 Introduction

Adopting cloud computing (CC) in healthcare is extensively discussed both in literature [1] and practice. A cloud platform providing healthcare cloud services is believed to deliver new benefits for the healthcare industry [2, 3]. One benefit is due to the multisided nature of such a platform. A multisided platform depends on the number of participants of its different customer segments (e.g. in the case of a cloud market-place - users and providers of cloud services). Interaction between those segments is essential for the value the interacting parties receive [4, 5]. In the context of multisid-ed platforms, Rochet and Tirole state that "the choice of a business model seems to be key to the success of a platform" [6]. Business models also help to understand value creation in e-business [7]. According to the definition by Oster-

Nicolai Hanner · Tatiana Ermakova · Jonas Repschlaeger · and Ruediger Zarnekow
Technische Universität Berlin
e-mail: {nicolai.hanner,tatiana.ermakova,j.repschlaeger,ruediger.zarnekow}@tu-berlin.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_17

walder et al. we refer in this paper to “a business model describes the rationale of how an organization creates, delivers, and captures value” [8]. Hence, the creation, delivering and captur-ing could be described as a value chain within the business model.

Nevertheless, the previous literature review on CC in healthcare conducted by Er-makova et al. [1] shows, that the current research does not cover business models of cloud marketplaces for healthcare. Based on the lack of such study, the goal of this paper is to develop and present a first business model draft for a healthcare cloud marketplace based on the example of the TRESOR project (Trusted Ecosystem for Standardized and Open cloud-based Resources). Thus, we address the following research question in our paper: How can a business model for the TRESOR ecosystem look like and how can its value chains be described?

Our work is part of the TRESOR project within the Trusted Cloud funding program and conducted in accordance with the action research principles. The paper is struc-tured as follows. Section 2 provides a background on CC for healthcare by presenting the current research issues in this field, while section 3 introduces the research design used in this paper and the business model theory. In section 4, we present our busi-ness model based on the business model canvas (BMC) by Osterwalder. We further discuss the findings draw conclusions, and make recommendations for future work.

17.2 Background

The adoption of CC in healthcare is becoming one of the frequently addressed topics in the research community. The CC paradigm is expected to enrich the healthcare sector by new opportunities [2, 9–13]. This can be exemplarily demonstrated through the use cases the Trusted Cloud research projects in the healthcare sector are aimed to realize. The TRESOR research group works on enabling interorganizational pa-tients’ data exchange and drugs interaction checking, whereas the Cloud4health and GeneCloud research groups focus on embedding CC computational capacities to perform text mining on clinical data and simulations for development of medical therapies, respectively. Furthermore, Ermakova et al. [14] provide an overview over cloud-based solutions to address the non-value-added activities in the hospital pro-cesses previously determined using established methods. In particular, the solutions make it possible to reduce the waiting time when patients’ data are being transferred, to prevent overproduction and rework caused through media breaks.

Through a systematic literature search, Ermakova et al. [1] reveal 48 publications published till year 2012 in the area of CC in healthcare. Based on this sample, the authors derive four main research directions. Along with security mechanisms and the feasibility of the approach in the domain, cloud marketplace components such as services, platform and broker (see Table 1) are defined as being paid attention to in the current research. However, the studies on the healthcare cloud marketplace com-

ponents are characterized as design-oriented and thus providing no details about their business models.

Healthcare Cloud-Marketplace Component	Sources
Healthcare Cloud Application Scenario	[11, 12, 15–28]
Healthcare Cloud Platform	[2, 3, 9, 10, 16, 17, 29]
Broker for Healthcare Cloud Platform	[13, 30–32]

Table 17.1: Healthcare Cloud Marketplace Components as Current Research Issues

17.3 Research Design

17.3.1 Action Research

In the IS literature action research has been widely discussed and requires the researcher to participate in an organization's change situation (e.g. [33–35]). A well-known definition of action research is provided by Baskerville and Wood-Harper: "[..] Action research as understood in Information Systems research is a form of qualitative research in which the researcher engages with the client organization to resolve the client's problems, with both researcher and client learning through this process" [36]. Within the TRESOR project two German hospitals, two IT service providers and an infrastructure provider cooperate in order to achieve an enhancement for the health sector by using CC. Therefore the action research approach is chosen in order to design a new solution in a collaborative context and to understand underlying causes and evaluate the findings in real use cases within the hospitals.

17.3.2 Business Model Theory

Current research states that business models can be described in many ways and can focus on many aspects [37, 38]. Chesbroug argues that companies have a business model, no matter if the company knows its business model or not [39], though being unable to adopt a suitable business model to environmental changes might be a reason for the company's failure [40].

Recent research shows that there is also a manifold of business models in CC. Labes et al. [41] analyzed the business models of 29 cloud firms, revealing they can be clustered in at least four different business model groups. However, none of them can be transferred as it is into a suitable business model for a cloud marketplace. And as we mentioned in the introduction, to our knowledge there is no other literature that describes a business model for a cloud marketplace in healthcare. Hence, we

need to develop a new business model to provide a helpful understanding of the TRESOR marketplace's value creation and its value offering [42].

Burkhart et al. summarize in their findings, though a business model is designed at a high level of abstraction, it should still be distinguishable from the business models of competitors [37]. Despite different approaches [37, 38], most definitions include four to five aspects of a business model. These are value creation, value, customers and distribution as well as the revenue part [40, 42, 43]. Hence, a method is needed to translate these aspects into a concrete business model of a firm. For our TRESOR business model, we use the BMC [8] due to its easy understandable nature. This in turn eases the discussion on the business model, therefore helping us to follow our action research approach.

The BMC separates the business model into nine fields which answer all the above stated aspects of a business model. Each of them describes a single aspect of a busi-ness model and is logically connected with others. First, the approach introduces the customer segments and the value proposition which is to be delivered to those seg-ments. The way the value proposition is delivered to the customer is described in the channels a company uses. The other part of the connection between the company (value proposition) and the customers is the customer relationship and depicts how the company establishes and maintains the relationship to its customer segments. To satisfy the customer's needs it is essential that a company has key resources that allow it to perform key activities, which then create the value proposition. If, for any reason, a key activity cannot be performed within the company, the company might source it from its key partners. The financial part of the value creation can be de-scribed in the cost structure of the company. The backbone of this value creation process is the revenue stream the company gets for their sold products [8].

17.4 TRESOR Business Model Design Proposal

In this section, we present the nine BMC blocks of the status quo of our business model based on the previous research findings on adopting CC in the health care sector and the results of our action research.

Customer Segments: Since the TRESOR project actually focuses on IT provisioning for the healthcare sector, the TRESOR customers are hospitals which will mainly use the services pro-vided through TRESOR. Furthermore, customers can be registered doctors or the pharmaceutical industry. On the other side, we address all software-providers to offer their services in the TRESOR marketplace and deploy them on the TRESOR PaaS platform.

Value Propositions: One of the value propositions to be granted by TRESOR is the connection of service providers and service users on the cloud marketplace. This implies bringing new buy-ers to the providers and new applications to the users. The incentives to join the mar-ketplace will also be given by further value propositions. A value is to be proposed through the comparability and transparency of services.

The marketplace itself builds the visual part of TRESOR and allows the software providers to present their services. The cloud broker eases the finding of services by making them comparable to the predefined requirements of users. This creates value for the users, since they are able to choose a service under the most possible transpar-ency conditions, e.g. the price, mandatory certificates for data security and scope of performance.

Secondly, trust and security is a value proposition of TRESOR. According to this point, certificates will guarantee that mandatory security requirements are observed. The proxy will observe that safety requirements will be maintained. Furthermore, the TRESOR consortium considers introducing an independent monitoring tool for ser-vice level agreements (SLA) between users and service/platform. Therefore the proxy creates a value for the user by easing the monitoring process and reducing the risk of fraud or data loss, which is essential for cloud services in the healthcare sector. The enabling of cloud services is another value proposition. The platform for hosting or developing services (PaaS Platform) will also be provided by TRESOR. This will enable for example scalability, use of thin-clients and reduced hardware management, especially in healthcare where cloud services are rare. This can help to increase the number of services. The TRESOR provides the platform for software not available as SaaS so far. In particular, it enables the provision of individual software as SaaS, even for small software-providers. Further TRESOR can guarantee that the platform fulfills the requirements for IT-security. This allows the small software pro-viders to focus on their core competencies and to access a complete new market. The platform adds another value that can be described as a service-oriented approach. The media breaks by transferring patient data can be reduced by enabling a through-going data usage. It can ease the support by reducing the contacts to a single point of contact a user has in case of a problem. Additionally, it is planned to deliver a central billing and accounting part, where users see the costs for all their applications in an aggregated way, therefore reducing complexity and easing the comparability of costs.

Customer Relationships: The marketplace's web page and the way it presents the services is the primary tool to establish and maintain customer relationships. Furthermore, the TRESOR should provide a central support structure for both services and the marketplace itself. This can include a trough going ticket system that users can access over the marketplace surface. Due to the complexity of hospitals IT-Infrastructure where will be the possi-bility to interpose the hospitals IT experts in the support process. We also see the fo-cus on healthcare as a mediator for familiarity which will generate a reputation with-in the healthcare sector [44].

Channels: The sales channel will be the marketplace itself. Hence, all values will be transferred through that channel. To be precise, the users access all applications, as well as the support, the monitoring of SLAs or the central billing system through the market-place. The marketplace and its services will be only accessible over the internet. Ac-cording to this, the service providers will sell their products over the marketplace reaching new customers.

Key Activities: This field includes the most activities that are needed to provide the value proposi-tions mentioned before. These will be the brokering, the moni-

toring of services, the matching of services and the supporting and accounting part of the platform. Fur-thermore, the evaluation of the transparency of services can be seen as an internal certification process for services. On the other side, there will be a third-party certifica-tion for data security and the strict obedience of privacy policy. Both will be essential for a trustable and secure cloud service.

Key Resources: Key resources will be mainly the technologies which will be developed by the part-ners during the project period. As mentioned before, these will be the proxy, the bro-ker and the marketplace. All these technologies will enable the activities needed to create the value and bring it to the customer. We also identify industry knowledge of the healthcare sector as one of the key resources for TRESOR since the product itself will be developed in close cooperation with all partners.

Key Partners: We identify two activities that should be sourced from key partners. These will be the certification process (of security certificates) and the providing of the PaaS or IaaS platform. The external sourcing of the PaaS/IaaS platform could also reduce the lock in effect. The sourcing of third party certification also contributes to a certain level of trust, if the certification will be provided by an independent organization.

Cost Structure: The costs can be differentiated where they occur: internally for key resources and externally for key partners. Certainly most of the internal occurred costs will be gen-erated by the provided infrastructure for the proxy, broker or the marketplace and the going of support and accounting. More expenses will be necessary for marketing or sales activities. Moreover costs from external parties will be third-party certifica-tion and the PaaS platform. The cost will be defrayed through a mixed calculation from the revenue streams.

Revenue Streams

We argue that mainly the service-providers have to generate the revenue streams for the platform. Hence the software user will be charged by the provider for the usage of a service and the marketplace will charge a percentage share from the provider.

For instance, a software provider hosts its software over the PaaS Platform of TRESOR and it will be available in the marketplace. The software provider decides that the software should be “pay-per-use”, e.g., for every check of medication interaction the user has to pay a fee. At the end of a period the user generated a certain amount of revenue. The marketplace gets its share from that part. The height of that fee can depend on the certification level the service has, making it more expensive if it is higher ranked. The platform can realize revenue streams by charging the software provider for using the platform. E.g. a software-provider booked the platform infrastructure to provide his service. Therefore, it will be charged another fee, most likely on a “pay-per-use” basis. The height of the share could depend on the guaranteed service level of the platform. Hence, a lower service level could reduce the costs for both service provider and service user. Depending on the usage of the matching service, users can be charged for a match-ing of services. For instance, if a user matches services to his requirements he pays a fee if he actually books one of the proposed services.

17.5 Discussion

According to our business model draft, a cloud marketplace can be an enabler for explicit value propositions to users and providers of services. We also show how they can be structured in a business model, how the value creation process can be realized and what resources and activities are needed. Figure 1 shows as an exemplary part of the business model: the certification-provider as a key partner. It enables the certification of services as a key activity, therefore creating the value proposition of trust and security. This value proposition will be delivered through the marketplace and addresses the service-users.

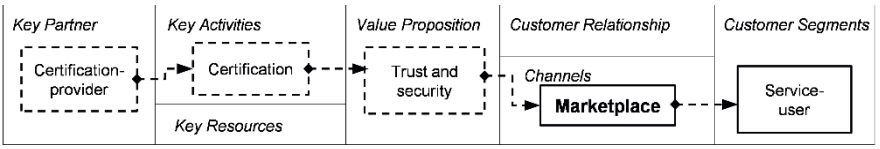


Fig. 17.1: Exemplary Value Chain of a Value Proposition

Figure 2 shows our main findings in form of value chains depicted into the BMC and each denoted by a special pattern of line.

As shown in Figure 2, some value propositions can be realized through the key ac-tivities and can therefore be guaranteed by the marketplace. On the other side, there could be the difficulties typical for two-sided markets. This addresses mainly the value proposition for both users and providers in terms of new customers and new applications. The TRESOR loses parts of its value if one or both of them cannot be acquired in a sufficient number for the marketplace. Nevertheless, the TRESOR mar-ketplace still offers a wider range of other values propositions to its customers. This should be focused as a strategy to prevent a strong dependence on the network of users/providers. Especially the high security standards will be a reason for service users to join the market place even if there are only a small number of applications. According to our findings in section 2, we value the security aspect as a major con-cern of potential users. Yet realizing an inter-organizational patients' data exchange without a cloud service seems to be unrealistic. This can lead hospitals to start using the marketplace for this service once integrated they will likely buy more services. Regarding service providers features like the PaaS platform will help them to develop new applications. Further the features provided by the proxy will reduce the devel-opment effort of service providers. Therefor they have an inducement to develop and to deploy their services within the TRESOR ecosystem.

Another finding is that the status quo of the TRESOR shows a wide scope of value propositions and the depth of value creation. The management of such a wide scope can be inefficient for one single organization - therefore some parts should be sourced from key partners when possible, making the sourcing processes a key

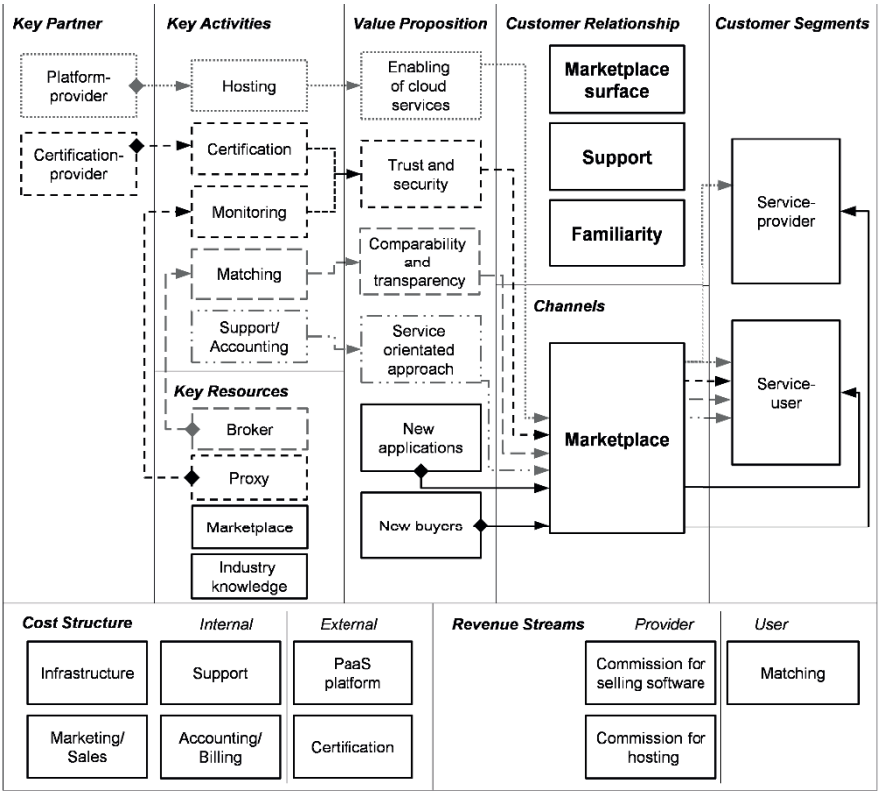


Fig. 17.2: Value Chains in TRESOR

activity for the marketplace provider and thus unloading her or him from to many complex tasks. This could be the PaaS platform or the certification process.

This leads to the following questions which arise from this first draft. A decision for a specific business model depends on the operating company's abilities to capture value. Being more technologically oriented might be ideal for providing technological resources like the proxy or the broker as a service for a variety of different market-places. Firms being more marketing/distribution oriented should concentrate on ac-cumulating know–how on marketplaces, offering of services and bringing the two sides of customers together and managing their needs.

17.6 Conclusion and Future Work

In the background section, we presented the current research issues on CC in healthcare and thus motivated the potential of cloud marketplaces for healthcare. Follow-

ing our research design, we presented a first draft for a business model for the TRESOR ecosystem. In our discussion we revealed our major findings:

- The TRESOR business model shows a dependence on the numbers of participants of different customer segments (provider and user) similar to other multi-sided platforms
- Technology like the broker, proxy or the PaaS platform, as well as the providing of se-curity and trust can reduce the risk of failure gaining a sufficient number of users.
- Due to the complexity of technological components, the TRESOR platform will be likely provided by more than one company

It should be however noticed that this paper has some limitations. Firstly, following the action research design we interviewed mostly the practitioners involved into the TRESOR project, thus limitedly taking into consideration the possibly available knowledge from outside, especially those from research scientists in this field. Secondly, the BMC has its limitations in the way a business model can be presented and described. To address these limitations and to foster future work, we emphasize the need of the research on the business model issues itself and more scenarios how the compo-nents can be provided separately within an ecosystem. This can be done by case study research. Additionally, it can be examined what measures could increase the efficiency of a cloud marketplace, e.g. different PaaS platform providers or the com-petition of services.

References

1. Ermakova, T., Huenges, J., Erek, K., Zarnekow, R.: Cloud Computing in Healthcare – A Literature Review on Current State of Research. In: Proceedings of the Americas Conference on Information Systems (2013)
2. Guo, L., Chen, F., Chen, L., Tang, X.: The Building of Cloud Computing Envi-ronment for E-Health. In: Proceedings of the International Conference on E-Health Networking, Digital Ecosystems and Technologies (2010)
3. Wang, X., Tan, Y.: Application of Cloud Computing in the Health Information System. In: Proceedings of the International Conference on Computer Application and System Modeling (2010)
4. Caillaud, B., Jullien, B.: Chicken & Egg: Competition among Intermediation Service Providers. The RAND Journal of Economics 34, 309–328 (2003)
5. Armstrong, M.: Competition in Two-Sided Markets. The RAND Journal of Economics 37, 668–691 (2006)
6. Rochet, J.-C., Tirole, J.: Platform Competition in Two-Sided Markets. Journal of the European Economic Association 1, 900–1029 (2003)
7. Amit, R., Zott, C.: Value Creation in E-business. Strat. Mgmt. J. 22, 493–520 (2001)
8. Osterwalder, A., Pigneur, Y., Clark, T.: Business Model Generation Wiley, Hoboken, NJ (2010)
9. Basu, S., Karp, A., Li, J., Pruyne, J., Rolia, J., Singhal, S., Suermondt, J., Swaminathan, R.: Fusion: Managing Healthcare Records at Cloud Scale. IEEE Computer Special Issue on Move Toward Electronic Health Records (2012)

10. Chang, H.H., Chou, P.B., Ramakrishnan, S.: An Ecosystem Approach for Healthcare Services Cloud. In: Proceedings of the IEEE International Conference on e-Business Engineering (2009)
11. Hoang, D.B., Chen, L.: Mobile Cloud for Assistive Healthcare (MoCAsH). In: Proceedings of the IEEE Asia-Pacific Services Computing Conference (2010)
12. Ratnam, K.A., Dominic, D.D.: Cloud Services - Enhancing the Malaysian Healthcare Sector. In: Proceedings of the International Conference on Computer & Information Science (2012)
13. Wu, C.: E-Healthcare Web Service Broker Infrastructure in Cloud Environment. In: Proceedings of the IEEE 8th World Congress on Services (2012)
14. Ermakova, T., Weimann, P., Zarnekow, R.: Reduzierung von Schwachstellen in Prozessen der Krankenhauseinrichtungen durch cloud-basierte Lösungen am Beispiel der Patientenversorgung. Informatik 2012 - Workshop „Datenmanagement und Interoperabilität im Gesundheitswesen“ (2012)
15. Abbadi, I., Deng, M., Nalin, M., Martin, A., Petkovic, M., Baroni, I.: Trustworthy Middleware Services in the Cloud. In: Proceedings of the 3rd International Workshop on Cloud Data Management (2011)
16. Berndt, R.-D., Takenga, M.C., Kuehn, S., Preik, P., Sommer, G., Berndt, S.: SaaS-Platform for Mobile Health Application. In: Proceedings of the 9th International Multi-Conference on Systems, Signals and Devices (2012)
17. Deng, M., Nalin, M., Petkovié, M., Baroni, I., Marco, A.: Towards Trustworthy Health Platform Cloud. In: Secure Data Management, Lecture Notes in Computer Science, 7482, 162–175 (2012)
18. Deng, M., Petkovié, M., Nalin, M., Baroni, I.: A Home Healthcare System in the Cloud - Addressing Security and Privacy Challenges. In: Proceedings of the IEEE 4th International Conference on Cloud Computing (2011)
19. Chiang, W.-C., Lin, H.-H., Wu, T.-S., Chen, C.-F.: Bulding a Cloud Service for Medical Image Processing Based on Service-Oriented Architecture. In: Proceedings of the 4th International Conference on Biomedical Engineering and Informatics (2011)
20. He, C., Jin, X., Zhao, Z., Xiang, T.: A Cloud Computing Solution for Hospital Information System. In: Proceedings of the IEEE International Conference on Intelligent Computing and Intelligent Systems (2010)
21. Kanagaraj, G., Sumathi, A.: Proposal of an Open-Source Cloud Computing System for Exchanging Medical Images of a Hospital Information System. In: Proceedings of the 3rd International Conference Trend in Information Sciences and Computing (2011)
22. Karthikeyan, N., Sukanesh, R.: Cloud Based Emergency Health Care Information Service in India. Journal of Medical Systems, 36, 4031-4036 (2012)
23. Koufi, V., Malamateniou, F., Vassilacopoulos, G.: Ubiquitous Access to Cloud Emergency Medical Services. In: Proceedings of the 10th IEEE International Conference Information Technology and Applications Biomedicine (2010)

Chapter 18
SensorCloud: Sociological Contextualization of an Innovative Cloud Platform

Michael Eggert, Daniel Kerpen, Kirsten Rüssmann, and Roger Häußling*

Abstract Cloud Computing's tremendous implications on individual, organizational, and societal levels call for a decisively socio-technical perspective of analysis. This paper contributes to such a regard by applying a framework which focuses on human-technology interaction, innovation processes, and social acceptance of emerging technologies. By reviewing main strands of cloud related research, central sociological research foci are identified which lead to an exploratory qualitative acceptance study focusing on Cloud Computing in general and its application in a smart home context in particular. In 24 in-depth interviews, security of the data, privacy, and minimization of misuse possibilities, reliability and stability of the system, as well as the fear of autonomy losses are identified as central problems. The interviewees report on safety and security, improvement of energy efficiency and gains in comfort as the most important fields for Cloud Computing applications in smart homes. The contribution concludes with an outlook on future work.

18.1 Introduction

Today, information and communication technologies (ICT) have tremendous socio-economic implications [39]. One could say that our age of information and globalization (at least beyond the so-called digital divide [38, 14, 19]) goes along with the demand for massive computing power on all levels of societal, organizational, and individual action. Such computing power is not only demanded by industry giants, but also by a growing number of small and medium-sized enterprises (SMEs) to generate businesses and competitive advantage [20, 46], and even individuals and

Michael Eggert · Daniel Kerpen · Kirsten Rüssmann · Roger Häußling
Sociology of Technology and Organization, RWTH Aachen University, Germany
e-mail: {meggert,dkerpen,kruessmann,rhaeussling}@soziologie.rwth-aachen.de

* We would like to thank our interviewees for participating in the qualitative study.

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_18

private households increasingly demand computational power in everyday life (e.g. in the context of the development of AACC ("Anytime, Anywhere Communication and Computing"), cf. [12]). Thus, new approaches to data processing are needed, and Cloud Computing promises to offer a viable approach to meet this demand.

Cloud Computing can be defined as "an IT deployment model, based on virtualization, where resources, in terms of infrastructure, applications, and data are deployed via the internet as a distributed service by one or several service providers. These services are scalable on demand and can be priced on a pay-per-use basis" [30, sect. 2.2]. In this view, Cloud Computing promises to provide on-demand computing power with quick implementation, low maintenance, fewer IT staff, and consequently lower costs [53, 54]. However, non-technical barriers that may hinder successful implementation of cloud infrastructure, applications, and services are prevalent. Therefore, a decisively socio-technical perspective is regarded useful for addressing and tackling such issues.

Nevertheless, in many cases such a perspective is consulted only retrospectively to identify sources for the failure of innovative technological projects. But addressing respective issues already in early stages of a technology's development may help to sensitize developers and designers for the needs, desires, practices, and perceptions of the technology's prospective users, i.e. the contexts, in which the innovation will or should be used and therefore needs to be integrated in. This approach allows for significantly increasing the prospects of an innovation in terms of social acceptance and therefore the diffusion of the technological system. Within the SensorCloud project, we contribute to such a perspective by applying a framework which focuses on the three poles of (1) interactivity between humans and technology, (2) innovation processes, and (3) the social acceptance of emerging technologies. In this paper, we focus on the latter by exploring potential end-users' perceptions and attitudes concerning Cloud Computing in general and its application in a smart home context in particular. Moving from the rather abstract concepts of Cloud Computing and intelligent environments to the more tangible idea of cloud-based smart home installations, we will shed some light on the acceptance of Cloud Computing technologies for the realization of smart devices and environments in the context of private households. Especially three questions are central to this analysis: The first one asks for the end-users' awareness of the concepts Cloud Computing and smart home in a very general sense. Second, we are interested in the fields of application which private users envision for the implementation of Cloud Computing technologies in smart homes. And finally, we want to learn about the perils that people associate with the use of smart technologies in order to identify barriers that may hinder acceptance.

The remainder of this paper structures as follows: First, we give an overview of Cloud Computing related main fields of research but with a focus on the research originating from a social science perspective (Sect. 2). Here, we especially identify central research gaps from a certain sociological point of view. Subsequently, Sect. 3 will give an overview of objectives and design of our own sociological acceptance study and the realization of its exploratory phase. Finally, we present our first step of sociological contextualization of Cloud Computing by summing up the results of

our explorative interviews (Sect. 4) and by giving a short outlook on future work (Sect. 5).

18.2 A Short Review of Cloud Computing Research

Although adoption of Cloud Computing continuously grows, research on this topic is still at an early stage with issues not yet fully addressed, while new challenges emerge on the horizon. As recent Cloud Computing related reviews [48, 54] show, mainly the following topics have been subject to research up to now: conceptual and technological issues, business issues, as well as domains and applications. The remainder of this section will address these issues in detail.

18.2.1 Conceptual and Technological Issues

When reviewing current literature on Cloud Computing, it is obvious that conceptual and technological issues are mainly addressed from a Computer Science perspective.

18.2.1.1 Conceptual Issues.

Concerning *conceptual issues*, an earlier review [54] refers to works that provide a general view of Cloud Computing practice and research in the sense of a general understanding of either both foundations and introductions to this area or future predictions. Regarding *foundational* and *introductory works*, articles pool around themes like definitions and key features of Cloud Computing [1, 35], its benefits and obstacles, strengths and weaknesses, as well as comparing the Cloud Computing paradigm with other concepts such as cluster/grid/virtual computing [4]. Regarding predictions, studies forecast the future of Cloud Computing and suggest potential implications, as well as future research directions [1, 33].

18.2.1.2 Technological Issues.

Concerning *technological issues*, we draw on a classification proposed by [56, pp. 14–17]:

Generally, it is not obvious, how a service provider can (de-)allocate resources from the cloud to fulfill service level objectives while simultaneously minimizing operational costs. Another issue is *virtual machine (VM) migration* which encompasses *workflow scheduling and load balancing and the improvement of dynamic resource allocation*. Along with migration technology goes the issue of *server con-*

solidation to effectively maximize resource utilization, as well as the improvement of energy and traffic management. Considering *energy efficiency,* the *design of data centers* has been approached from several directions with the key challenge remaining to achieve a trade-off between energy savings and application performance. In order to fit requirements needed for making management and planning decisions directed at optimal customer experiences, existing *traffic measurement and analysis methods* depend on continuous cloud performance evaluation and optimization.

Considering the software perspective, applications should be scalable and fault-tolerant. Research ranges from *developing distributed and parallel software in Cloud Computing environments,* over *component-based approaches for developing composite applications,* to *restructuring traditional applications* into distributed/-partitioned cloud-based ones. Currently, novel approaches are sought for problems that come along with large data centers' *architectures,* as well as their *data storage* and *management approaches*, because traditional commercial cloud implementation still focuses on centralized setups and frameworks.

As highlighted in our SensorCloud project (see this volume), the involvement of multiple providers in the provisioning of cloud services opens new vectors of attacks against potentially sensitive data that is stored or transmitted within the cloud environment. Therefore, *data security* is a central and challenging research topic in Cloud Computing. Here, topics like restrictions and audits as *general security mechanisms* are discussed, as well as *multi-tenancy authorization, third-party assurance,* and *cloud-based security services.* Other *specific issues* address data encryption and coloring, software watermarking for multi-way authentications, data-partitioning schemes for implicit security, as well as *network security* (e.g., intrusion detection and cloud level defense against DoS-attacks). Furthermore, cloud services are commonly consumed by non-expert users. Hence, the presentation of data as well as the underlying security model must be *intuitive and understandable by the common user.* Otherwise, adoption barriers might prevent individuals from applying a technically sound, highly secure, but unusable system. Thus, data security in Cloud Computing is a concern not only for technophiles and developers, but also for a wider public.

18.2.2 Business Issues

Key findings of the work focusing on current trends in the perception, assessment, and adoption of Cloud Computing within organizations, e.g. enterprises, state the significantly higher importance of business drivers for the adoption of Cloud Computing in comparison to technological issues [41]. While such work mainly handles the technology of Cloud Computing as a black-box, its domain comprises the following issues [48, 54]:

First, economic benefits from a cloud-user perspective may be examined. This includes estimating and monitoring *costs* for making buy-or-lease decisions on cloud services [49]. Others focus on *pricing strategies* of cloud providers [55]. Neverthe-

less, not only costs and pricing, but *legal issues* are associated with Cloud Computing, too [26]. For instance, articles introduce general legal risks of adopting Cloud Computing [24], or address specific topics such as uncertain jurisdiction for geographically distributed cloud data centers [51]. Furthermore, ethical discussions consider the way of decision making from different standpoints of applied ethics [36].

Other studies explore Cloud Computing *adoption*. Whereas some assess cloud readiness, practitioners' experiences, and compliance risks in large, multinational companies [32], others explicitly focus SMEs and consider their pros and cons of Cloud Computing adoption [7, 47].

Trust and *privacy* are an inevitable concern for businesses, too. In order to facilitate storage of sensitive business and personal information by cloud users, cloud providers are encouraged to display clear policies about how customer data is used in remote data centers which are managed by third parties [44]. Therefore, transparency and public auditability are stressed [50].

18.2.3 Domains and Applications

Besides business questions, the following domains and applications are considered by recent work: For instance, the computing power and low costs of Cloud Computing are attractive for *e-Sciences*. Here, huge loads of data that need to be quickly processed call for new infrastructure and new ways of scientific workflow management [13]. Further, this development might trickle down in universities from research to *education* and spur *e-Learning* environments [6]. Besides that, *e-Government* may arise as a future main domain, if the associated risks and security concerns may adequately be addressed [57]. Another area is to combine Cloud Computing and *mobile computing*, e.g. implementing a health-monitoring system based on a combination of cloud infrastructure, mobile phones, and sensors [40]. Further domains may include open clouds by merging Cloud Computing and *open source* to create an interoperable 'network of networks' [37]. Last but not least, the construction of smaller, cheaper, and smarter robots [15] and the development of intelligent urban transportation systems [31] are fields of research in this area.

18.2.4 Research Challenges for a Sociological View on the Cloud

Currently, to our knowledge only few works from Social Sciences, and particularly from Sociology, address cloud related issues [48, 54]. For instance, the German Federal Ministry of Education and Research (BMBF) carried out the "Smart Objects" project between 2006-2009 [18]. Settled within Germany's National Academy of Science and Engineering (acatech) thematic network "information and communication technology", the project did address Cloud Computing related research and

technology and major internet of things (IoT) trends in Germany at least in parts from an explicitly sociological perspective. One other study [21] is concerned with Cloud Computing related economic and organizational benefits in the context of sociological actor-network theory (ANT; [29]). By focusing on corporate Cloud Computing deployment, it shows in the cases of Amazon, Google, IBM, and Microsoft that potential benefits are generated by dynamic interactions of information technology artifacts, services, organizations, and stakeholder interests. Furthermore, implications such as questions of standards and efficiency are discussed.

Although sociological research of Cloud Computing has remained a desideratum, its relevance is obviously evident concerning multiple dimensions of sociological interest: One of the most evident and our major point in the present paper is, that, besides all technical feasibilities, potential success of innovative cloud-based systems and services should be explored in the context of particular social dimensions such as user acceptance, usability and environmental conditions—both material and structural as well as cultural ones. Such an analysis of socio-technical constellations and relationships is particularly valuable in the context of the development of ICT, because it allows to take human and non-human (inter-)action and (inter-)activity into account [17]. Taken for granted the assumption that there actually will be a difference between the hitherto common use of computers and that of Cloud Computing, we assume that there is a certain kind of inter-activity developing between users and the technologies in question. Analogously to the concepts of "man-machine interaction" and "human-computer interaction" [5], we propose to call this emerging kind of inter-activity "human-cloud interaction", regard its embeddedness in social practices and social structures [3, 42], and prospectively use this term to identify and describe its specialties in relation to the former concepts.

Once implemented to a certain degree, cloud technologies unquestionably have the potential to strongly impact on the structures they are embedded in. Most basically as a means of communication, cloud technologies like social networks sites offer new ways for human-to-human communication and interaction. Cloud storage and cloud-based software services (Software-as-a-Service—SaaS) like Google Drive (`http://drive.google.com`) or Dropbox (`https://www.dropbox.com`) facilitate information sharing and cooperation over large distances. The permanent availability of personal data and individual documents allows for a restructuring of working processes and the development of new forms of the organization of work and private life. This is especially true for technologies like the SensorCloud, which intersect the technological developments of Cloud Computing and an ever increasing networking of diverse artifacts.

On a more abstract—but nevertheless relevant—level, Cloud Computing technologies change the way how people not only interact with one another, but also with technology in general and with connected artifacts in particular [11]. The human-technology relationship becomes problematic as traditional ways of conceptualizing technology become more and more obsolete. Technological artifacts no longer can be seen as mere tools which are useful to reach certain goals, but become increasingly interwoven in every action and inscribed into the structures of everyday life. They fulfill tasks of increasing complexity, become partners in inter-

action, and increasingly gain agency [43]. Thomas Hughes' "seamless web" [22] becomes increasingly visible—not only to theorists of technology and society—and can be more and more experienced in everyday life. Thus, it is clear that cloud technologies and such of ubiquitous computing can hardly be understood as something external to society, but must be conceptualized as parts of society or rather as elements of socio-technical networks consisting of both human and non-human actors, see e.g. [28]. At the same time, technology becomes increasingly invisible, moves to the background, and acts more and more without the need for direct user instructions. This development fits quite precisely with Mark Weiser's vision of "Ubiquitous Computing", in which he states that "[t]he most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it" [52]. This indistinguishableness is one of the central challenges that Cloud Computing technologies—especially in the context of them being the backbone of the UbiComp-vision—pose to sociology.

18.3 Conceptualizing and Operationalizing User Acceptance

Having summarized the main strands on Cloud Computing research in general, and the challenges it poses to sociology in particular, we now turn to a first aspect of an attempt to contextualize cloud technology in the socio-technical arena, which we have identified as one of our three poles of sociological research on Cloud Computing—that is user acceptance.

18.3.1 Why Investigate User Acceptance?

The goal of this first exploratory stage of the acceptability study was to determine those parameters of the SensorCloud development, which appear as the crucial ones from an end-user perspective, and to consequently identify demands which are posed to an application of SensorCloud technologies in the context of an intelligent home. Additionally, it allows for an outlook to possible application fields and use cases for the SensorCloud, and it provides the basis for the further steps of the acceptability and acceptance analyses. Finally, from the results of this stage, first sketches may be drawn on the concept of human-cloud interaction to be developed.

Our introductory aim was to gain an encompassing insight into how private users adhere to SensorCloud technologies within a smart home scenario, what areas of application they might imagine, and what they consider as being problematic. Concretely, we would survey the people's awareness of the basic technological concepts Cloud Computing and smart home and their attitudes towards them. Important aspects of this complex are experiences already gained with correspondent or related technologies, and the interviewees' ideas, on whether and how they could imagine such technologies might be applied in their actual situation. Another focus was put

on possible fears regarding the technologies in question, in order to specify related fields of concern, and to address potential diffusion barriers as early as possible to the project partners in charge for the technological development.

18.3.2 Conception and Realization

Those topics we intended to examine had not been subject to explicit sociological analyses, yet. Thus, a study based on qualitative interviews seemed to be the adequate instrument which would enable us to identify central ideas without running the risk of ignoring important aspects due to a too strongly pre-formed perception of the relevant matters. The resulting exploratory character of the study forced a largely open study design, being as free of suppositions as possible. Thus, the guide designed for the interviews was mainly used to ensure that all relevant topics were addressed, but not to rigorously structure the interviews themselves. This facilitates the creation of a comfortable climate for conversation, and empowers the interviewees to weight the different aspects according to their own views.

Further considerations led to the decision not to confront the interviewees with both of the underlying concepts, but to split the interviews into two blocks, each of them leading to the SensorCloud scenario from either the idea of Cloud Computing or the idea of smart environments. From this approach, we expected a more thorough insight into existing imaginations, as well as a more differentiated understanding of SensorCloud relevant aspects from different points of view.

Based on the scenario of the home automation of a "New Middle Class" milieu family[2] on the one hand, and the desire to reflect a wide range of awareness, attitudes, and believes on the other hand, the potential interviewees were defined as living in households with children and representing different social milieus with about two thirds of them being expected members of the focused group. Recruitment was realized via different institutions of child care and children's education, situated within different areas of Aachen featuring different compositions of the respective inhabitants. Except of getting a summary of the results, no incentives were offered to the participants of the study.

In sum, 24 interviews were conducted (12 starting from the concept of Cloud Computing, 12 from the idea of the smart home), with each of them having a duration of 20 to 55 minutes. In order to estimate the interviewees' socio-demographic position, besides the thematic conversation some demographics were collected for both the interviewees themselves and their respective partners (gender, age, current occupation, education, possible migration background) as well as for their household (number, gender, and age of children, plus available monthly income). Addi-

[2] The "New Middle Class" is described by *Sinus-Sociovision* as "The modern mainstream with the will to achieve and adapt: general proponents of the social order. [S]triving to become established at a professional and social level, seeking to lead a secure and harmonious existence." Child-friendly milieu, generally living in households with multiple persons, having an average income, and being moderately or slightly higher educated.[45]

tionally, the interviewees‚Äô innovativeness concerning ICT was queried by asking for their general interest in ICT and their information sources on new technological developments. Our sample of interviewees consisted of 14 women and 10 men (Cloud Computing: 8 women/4 men, smart home: 6 women/6 men), aged 25 to 52 with a mean of 35.5 and a standard deviation of 6.3 years (CC: 30—52/38.1/7.0, SH: 25—40/33.0/4.4). The majority of the interviews were conducted within the respective institutions, a smaller portion of them at the Institute of Sociology or at the interviewees' homes. The interviews were recorded electronically, transcribed word-by-word, and afterwards analyzed according to Mayring's concept of qualitative content analysis [34].

18.4 Results

The following section presents central findings of our qualitative study, structured according to the guide used: first focusing the basic concepts of either Cloud Computing or smart home technologies and subsequently leading over to the idea of SensorCloud.

18.4.1 Central Concepts: Awareness, Past Experience, and Ideas

To start with, we can observe that both Cloud Computing and smart home are known to only some of the interviewees—at least concerning the terms themselves. Asked directly, more than half of each group declare not to be familiar with the concepts of the respective technologies. On the other hand, for both concepts, there are individuals who are not only familiar with the terms, but also have quite concrete and encompassing conceptions about the technological approaches.

18.4.1.1 Cloud Computing—Vague Images, Unconscious Use.

The little the concept of Cloud Computing is known to the interviewees, the little it evokes adequate associations spontaneously. The ideas expressed impulsively are rather nebulous and cover a wide range, reaching from general data procession to robotics. Interesting to note is, that, repeatedly, ideas related to smart home solutions are mentioned in the context of Cloud Computing. Furthermore, two of the interview partners spontaneously outline the field of traffic management as an application for cloud technologies—one of the prospective large-scale applications of SensorCloud imagined by our industrial partners.

One of the interviewees who was unaware of the concept at first, tried to gather some information on Cloud Computing in the run-up to the interview. From her internet research, she draws the conclusion that Cloud Computing is an "umbrella term

for all kinds of things" (CC-07:09—interview sequences are referenced as follows: "Interview-ID:paragraph") and after reading a bit about the topic, she still qualifies it as being too abstract. This impression is also confirmed in the other interviews. Even after a short presentation of the basic idea, the concept as such remains too "technical" for most of them.

Looking at the question, what kinds of cloud services our interviewees can imagine to use, we can identify the use of cloud storage to be most likely imaginable—either to be able to access certain data by means of different devices, or as a backup solution. Apart from individual interviewees who already use SaaS solutions such as Google Drive, or at least can imagine to do so, mostly, there seems to be no need for a broader application of cloud services. Regarded from this perspective, the use of cloud services appears not to be very wide-spread in the private realm.

A slightly different picture results from looking at the actual use of cloud services, which might not be connected with the term itself. Almost all of our interviewees use cloud services as a means of communication. For a significant part of them, communication via e-mail or social network sites is even the most important reason to use computers at all. For the smartphone users among the interviewees this is complemented by a certain amount of "hidden use", as CC-08 puts it. Thus, we can state, that cloud technologies are already applied on a broad basis, but they are often not recognized as such or at least not labeled with the term "cloud".

18.4.1.2 Smart Home—Imaginable But Rarely Spread.

Compared with the concept of Cloud Computing, the idea of a smart home seems to be more comprehensible in general. Even those interviewees who have never heard the term before, can envision the concept or have a rather appropriate idea of it. Especially for the less technophiles among our interviewees, a connected home with e.g. a smart heating, automatically regulated jalousies, or other smart appliances seems more imaginable than the rather technologic and economic idea of the cloud.

The ideas expressed in this context also cover quite a broad range of possible applications, but they consistently focus on smart home technologies. The differences among them mainly relate to the imagined depth of intervention or the expected technological pervasion of everyday life. Nevertheless, we find a dominant concept also in this domain: supervision and control of different functions and appliances in the house from outside via mobile devices. However, actual experiences with smart home technologies are extremely rare.

18.4.2 Attitudes Towards Cloud Computing *and* Smart Home

The qualitative difference of the ideas of Cloud Computing and smart home technologies in the interviewees' perception has already been suggested. It becomes more obvious when dealing with the question whether our interlocutors could imag-

ine to increasingly make use of respective technologies in their everyday life. Indeed, none of the interviewees categorically refuses a future use of Cloud Computing or smart home technologies, but the reasons for not yet doing so, differ considerably. Whereas for a more intense use of Cloud Computing, there simply seems to be no desire or need, it is mainly economic reasons that oppose an application of smart home technologies.

18.4.2.1 Cloud Computing—Data Security and Privacy.

In general, both technological approaches are appreciated by the interviewees. They consider both Cloud Computing and smart home applications as "useful", but especially for Cloud Computing restrict this valuation partially by pointing to the lack of a currently visible benefit. Furthermore, the majority of interviewees considers the use of cloud services as implicating privacy problems. Thus, actual users and non-users agree on this point, e.g. data storage in the cloud can only be considered for data which are perceived as non-sensitive.

The fears concerning questions of privacy and data security are based on different issues. On the one hand, the interviewees suspect a higher risk for their data to be looked at by unauthorized third parties and thus to be "stolen". On the other hand, there is a certain distrust about the cloud service providers in the sense, that they are generally suspected to use customers' data in other ways than expected—for example by interlinking or selling them.

Almost consistently, our interviewees declare to handle their data on the internet as sparingly as possible and just to rarely give away sensitive data like bank data or credit card numbers. However, in this context, a discrepancy between attitudes and behavior can be observed similar to the one that can be stated in the field of pro-environmental behavior [27]: albeit the interviewees are clear about the fact that giving away sensitive data is always critical, the risk is quite often taken—although "with a strange feeling" as they are aware "that it is not always completely safe" (CC-07:129)—, if there is at least a small but immediate advantage like, e.g., faster order procession or a discount in the case of shopping on the internet, or the higher comfort when using online banking.

A similar situation can be observed when it comes to terms with social networking sites, especially facebook. The more active facebook users among our interlocutors also publish photos on the platform, but at the same time are concerned about their privacy and that of their children. The consequences the users draw from those worries can be of highly different nature: as the problems and concerns are not necessarily specific to a certain provider, CC-10, for example, pays attention to not publish photos or other data of his private life on the internet (CC-10:109)—especially not of his children, as they are still too young to decide, which information they are willing to disclose (CC-10:179). On the other side, there is for example CC-05 who worries about privacy issues as well. But for her, communication via facebook is essential to such an extent that the only consequence she could render for herself is "to try to post as little as possible" of her children (CC-05:82).

Altogether, from the interviews we can state that the everyday life of our interviewees can no longer be thought without the use of modern—and to a certain degree cloud based—communication media. But this media use is always risky to some extent. Partially, this risk is accepted with a certain resignation and considered inevitable: especially in the realm of information and communication technology, our interviewees are convinced, "there cannot be 100% security" (SH-07:129), whilst, on the other hand, to be left behind by societal development when not using communication media—like, e.g., social networking sites—and other technologies is considered a realistic threat (cf. CC-01:40).

18.4.2.2 Smart Home—Reliability and Technology Dependence.

Just like Cloud Computing technologies, the smart home idea is also predominantly appreciated by the interviewees. However, there are also critical voices who point to aspects that might possibly be problematic. Other than in the area of Cloud Computing, there are not so much questions on privacy or data security which arouse worries. Rather, the concerns regard primarily the functioning of the technology itself, or its susceptibility to failure, respectively. On another dimension, the concerns relate to handling the technology and its consequences, and thus tackle the question, how the respective technologies' application might influence everyday life.

Interviewees conceive problems for smart home technologies for example in the point that computer technology to some extent being prone to failure (e.g. in case of a virus infection (SH-05:71; SH-10:15)) and having the tendency—in contrast to mechanical devices—"to fail without any announcement and lead time" (SH-10:79). Hence, in case of a breakdown, the people who had delegated certain tasks to technology and relied on them, were confronted with the situation that, "all of a sudden, nothing worked any more at all" (SH-11:51). More far-reaching is the second area of concern, the smart home idea is countered with. This area raises the issue of increasing networking, informatization, and technization of everyday life, and poses the (normative) question, whether this development did not point to the wrong direction. The delegation of ever more substantial and encompassing tasks to technological artifacts, from this point of view goes along with the fear of a technology's gain in power. For the interviewees, this leads to the question, whether it would still be technology which is adapted to humans or if the opposite would be the case: that people subordinated to some kind of technocratic regime. Some of our interlocutors in this context fear the loss of certain human qualities and competences, as well as some social, emotional, or even physical degeneration (e.g. CC-08:83—85; CC-12:119; SH-09:43, 67, 77; SH-12:117, 223). At this point, it should be noted that this attitude not necessarily goes along with a general lack of interest or even rejection of technology, as even one of our most technophilic interviewees (CC-08) elaborates on such a position.

18.4.3 Possible Fields of Application for SensorCloud Technologies

Possibilities for SensorCloud application in the scope of a smart home scenario are identified by the interviewees in different areas. First, there is the question for security, where SensorCloud technologies are expected to be useful for admission control, surveillance, or the simulation of an inhabited state during holidays. Also, protection against natural forces or weather phenomena is seen as a realistic and important application field. Besides those safety and security issues, the possibility to supervise and control diverse states and appliances within the residence from outside seems reasonable. In this context, SensorCloud technologies are outlined as a means to support the users' own smartness by offering the possibilities to control "whether the stove has really been turned off" (CC-03:39) or to make up for the forgotten turning-on of the washing machine, e.g. via smartphone.

Further potentials for the application of SensorCloud technologies within private residences, according to the interviewees can be found in the field of energy savings and improving the household's energy efficiency. Besides smart systems for regulating climate and illumination, load-dependent control of large appliances like washing machines or dish washers is mentioned in this perspective. However, as the most important and encompassing field for SensorCloud applications, our interviewees expect the possibilities to create gains in comfort by using smart devices.

Especially interesting to note in this context is the fact that the use of respective technologies for means of security, safety, and energy efficiency is approved universally, whereas the possible comfort gains are perceived quite critically. In line with the concerns regarding smart home technologies, it is expected that those could also lead to loss in quality of life on the other side.

18.4.4 Worries Concerning SensorCloud Use

As a last point in this section, we will throw a short glance on the interviewees' concerns regarding SensorCloud technologies in particular, on which a certain focus was put in the interviews.

Notably regarding this topic, the decision to treat the areas of Cloud Computing and smart home technologies in different interviews proved to be fruitful. In our case, the SensorCloud approach represents a cross-sectional technology combining both the concepts of Cloud Computing and smart home technology. Thus, it bears the potential to relate to itself both the privacy and security concerns brought up against the Cloud Computing approach, and the questions of reliability as well as the skepticism concerning technological pervasion, articulated relative to the smart home idea. This assumption is verified by the interviews as well as the supposition that interviewees would weight the areas of concern differently, depending on the respective idea from which they were led to the SensorCloud concept.

At least, this is true for the interviewees with the smart home focus. Indeed, most of them also identify privacy problems when dealing with SensorCloud technologies

and fear to become even more transparent. But compared with the Cloud Computing group, in their concerns, we may identify a second focus which emphasizes a possible technological determination.

To sum up, we can identify three central areas of concern which, in terms of a user-centered technology design, should be respected as early as possible both in the development of the technological platform and in the future design of corresponding services: (1) the most frequently and most intensely articulated area of concern comprises the generation, processing, and storage of the SensorCloud data. The protection of users' *privacy,* the *security of the data* generated and stored, as well as the *minimization of misuse possibilities* constitute the basic requirement for the technology to be developed from the users' perspective. (2) The next area to be addressed already in the early stages of development pertains to the system's *reliability and stability:* possibilities to guarantee for the basic functionalities even in the worst case of losing the connection to the cloud and/or a blackout must be provided, as well as *mechanisms preventing manipulation* by third parties. (3) Finally, it is crucial to attend to the *users' autonomy.* The technologies and services to be developed should be kept as *flexible and adaptable* from the beginning on, to avoid the problem that people were forced to adapt to the technologies in order to use them. Furthermore, possibilities have to be provided, allowing to *override or disable the smart systems* at any time to counter the threat of a perceived loss of control by the user, who, under any circumstances, still wants "to stay master in one's own house"(SH-06:51).

18.5 Résumé and Future Work

Within the previous sections we have dealt with the objectives, the design and the results of the exploratory qualitative acceptance study. This section concludes the article with resuming our findings and giving a short outlook about future sociological work to be realized within or spinning off the SencorCloud project. In detail, after a roundup of the exploratory study in the context of our research questions, this section comprises a short outlining of the participant observation of further stages of the innovation process itself, the discussion of the future workshop method, and, finally, a brief pledge for a further needed quantitative survey.

18.5.1 Central Findings

As formulated in the introduction, our aim with this article was to contribute to an understanding of one of the central non-technical factors influencing the adoption and diffusion of cloud technologies within the context of smart devices and environments—the question of social acceptance against the background of end-users' perceptions, visions, and ideas. Concerning our first question for end-users'

awareness of the central concepts, we have found that the concepts of both Cloud Computing and smart home are rather unknown among potential users of cloud-based smart home applications. Especially for Cloud Computing we may state that the ideas connected with the concept are rather vague. Albeit many of our interlocutors use cloud services on a daily basis (mainly for communication purposes), their awareness of the cloud pervading their everyday lives is quite low. While the idea of a smart home seems to be more comprehensible than the concept of Cloud Computing, actual experiences with such intelligent environments and appliances are still extremely rare.

Regarding our second question—the users' visions for the application of cloud-connected smart artifacts—, four main fields could be identified. These comprise safety and security applications, improving the household's energy use and efficiency, as well as using respective technologies for the sake of achieving a higher degree of convenience in the family home in general. Whereas all the interviewees agree upon the first three topics as desirable, the latter is perceived somehow controversially as a (technologically produced) gain in comfort partially is expected to be linked to a loss of quality of life in other dimensions like social relations or individual problem solving capacities.

With respect to the third question dealing with the expected perils that might serve as acceptance barriers, our study shows that for the case of cloud-based smart technologies all areas of concern apply which are linked to either the idea of Cloud Computing or the vision of autonomous artifacts when considered separately. Here, we are able to identify three main strands of concern: the first crucial factor mainly stems from the Cloud Computing background and considers privacy concerns and such of data security. Second, worries about autonomy losses—rooted in the notion of autonomous artifacts—are strongly prevalent. Finally, the system's reliability and stability is of particular importance.

From these insights, some fundamental requirements for the development of cloud-based smart environments and devices from an end-user perspective can be derived: Regarding the problem of privacy and data security, it is not only vital to respect those needs already in early stages of the development process. It is likewise important to transparently communicate the system's features and procedures concerning this context to potential users in order to support their decision-making and risk assessment. To prevent an anticipated loss of autonomy and control on the user side, smart artifacts' development needs to include mechanisms allowing the user to regulate the components' level of autonomous action and to override decisions made by the technology. At last, with cloud-connected smart appliances being involved in a wide range of everyday family activities and thus inducing a certain degree of dependence, failure of one or more of the fundamental infrastructures must be foreseen and addressed by technology development. This might result for example in including fallback mechanisms which allow for keeping up basic functions in case of cloud connection failure or for a manual operation of the components in a worst-case-scenario like an energy grid black-out.

18.5.2 Further Considerations

After having summarized the central findings of our exploratory approach to users' acceptance of cloud-connected smart environments, we conclude with giving an outlook on potential future work.

In the sense of a broader elaboration of our participatory design approach, different interest groups like end-users, service providers, experts, etc. should be empowered to become active elements of technology development in the SensorCloud project as early as possible. Therefore, the participatory *future workshop* method [2, 25] will be applied, being regarded as one method of choice considering such an early integration of different stakeholders perspectives into the development process [16]. In order to individually address end-users (B2C) as well as companies (B2B), two separate future workshops with experts have been conceptualized to be held within further project progress, one of which has already been realized at the time of this publication. On the one hand, this allows for a deepened understanding of the exploratory results of the qualitative study, e.g. the concretion of details which have not been further explored in the interviews, yet. On the other hand, the two workshops will be used as means for gathering new insights about rational choice and frame selection-based decision-making [9, 10] concerning implementation of cloud-based services in the B2C and B2B areas of application, respectively. The future workshop already realized was settled within the B2C field and focused on the topics of data security, user autonomy and the technology's usability. The second workshop is planned to deal with cloud-based control and surveillance of industrial appliances.

Additionally, observing the SensorCloud *innovation process* refers to a complex innovational setting. As stated elsewhere [8], technological innovations do not arise in a vacuum. Instead, complex innovations like the projects conducted within the Trusted Cloud program result from joint efforts of heterogeneous actor-networks from science, (corporate) research, political actors, and intermediaries, all of them trying to pursue their own individual goals in developing and implementing such a project. Therefore, further analysis will critically examine how the actors balance their respective interests in order to come to mutually shared understandings of process-related problems, decisions, and common solutions. Such a sociological perspective of technology development may obtain evidence on the chances for innovation under the condition of conflicting political, legal, social and economic interests and requirements.

Finally, as our exploratory interviews showed, security, privacy, and user autonomy are declared to be major areas of concern and impediments of cloud adoption from an end-users' perspective. Whereas there are just first results of surveying consumers' privacy beliefs and expectations towards cloud storage services [23], *quantitative surveys* might be conducted to follow-up our qualitative in-depth interviews and future workshops. We consider such an additional approach especially useful for deepening the understanding of both users' privacy and autonomy attitudes and beliefs regarding their use of Cloud Computing and smart artifacts exceeding the scope of our current SensorCloud project.

References

1. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I.: A view of cloud computing. Communications of the ACM **53**(4), 50–58 (2010)
2. Bell, W.: Foundations of futures studies: History, purposes, and knowledge, *Human Science for a New Era*, vol. 1. Transaction Publishers, New Brunswick, London (1997)
3. Cerulo, K.: Nonhumans in social interaction. Annual Review of Sociology **35**, 531–552 (2009)
4. Cervone, H.F.: An overview of virtual and cloud computing. OCLC Systems & Services **26**(3), 162–165 (2010)
5. Dahm, M.: Grundlagen der Mensch-Computer-Interaktion. Pearson, New York (2006)
6. Doelitzscher, F., Sulistio, A., Reich, C., Kuijs, H., Wolf, D.: Private cloud for collaboration and e-learning services: from IaaS to SaaS. Computing **91**(1), 23–42 (2011)
7. Dwivedi, Y.K., Mustafee, N.: It's unwritten in the cloud: the technology enablers for realising the promise of cloud computing. Journal of Enterprise Information Management **23**(6), 673–679 (2010)
8. Eggert, M., Häußling, R., Henze, M., Hermerschmidt, L., Hummen, R., Kerpen, D., Pérez, A.N., Rumpe, B., Thißen, D., Wehrle, K.: SensorCloud: Towards the interdisciplinary development of a trustworthy platform for globally interconnected sensors and actuators. Tech. rep., RWTH Aachen (2013)
9. Esser, H.: Die Rationalität des Alltagshandelns. Zeitschrift für Soziologie **20**(6), 430–445 (1991)
10. Esser, H.: Rationality and commitment: the model of frame selection and the explanation of normative action. In: Raymond Boudon: A life in Sociology, pp. 207–230. The Bardwell Press, Oxford (2009)
11. Ferscha, A.: Implicit interaction. In: J. Pitt (ed.) This Pervasive Day: The Potential and Perils of Pervasive Computing, pp. 17–36. Imperial College Press, London (2012)
12. Forschungsverbund AACCrisk: Kooperative Bewertung und Kommunikation der systemischen Risiken ubiquitärer Informations- und Kommunikationstechnologien. Available online at `http://www.ecolog-institut.de/fileadmin/user_upload/Publikationen/T_U_Publ/AACCrisk_Schlussbericht_2010-10_2.pdf` – last checked: 2013-06-20 (2010)
13. Fox, A.: Cloud computing-what's in it for me as a scientist. Science **331**(6016), 406–407 (2011)
14. Graham, M.: Time machines and virtual portals. the spatialities of the digital divide. Progress in Development Studies **11**(3), 211–227 (2011)
15. Guizzo, E.: Robots with their heads in the clouds. Spectrum, IEEE **48**(3), 16–18 (2011)
16. Hallberg, N., Pilemalm, S., Jägare, S., Irestig, M., Timpka, T.: Quality function deployment (QFD) extended future workshop: An approach for effective and enjoyable user participation. In: Proceedings of the Participatory Design Conference (PDC), New York, USA (2000)
17. Häußling, R.: Design als soziotechnische Relation. Neue Herausforderungen der Gestaltung inter- und transaktiver Technik am Beispiel humanoider Robotik. In: S. Moebius, S. Prinz (eds.) Das Design der Gesellschaft. Zur Kultursoziologie des Designs, pp. 263–288. transcript, Bielefeld (2012)
18. Herzog, O., Schildhauer, T. (eds.): Intelligente Objekte. Technische Gestaltung – Wirtschaftliche Verwertung – Gesellschaftliche Wirkung. Springer, Berlin, Heidelberg (2009)
19. Hilbert, M.: The end justifies the definition: The manifold outlooks on the digital divide and their practical usefulness for policy-making. Telecommunications Policy **35**(8), 715–736 (2011)
20. Hochstein, L., Schott, B., Graybill, R.B.: Computational engineering in the cloud: Benefits and challenges. Journal of Organizational and End User Computing (JOEUC) **23**(4), 31–50 (2011)
21. Huang, C., Hsieh, C.: Sociology view on cloud computing value: Actor network theory perspective. In: Proceedings 1st Conference on Cloud Computing GRIDs and Virtualization, Lisbon, Portugal, pp. 145–149 (2010)

22. Hughes, T.P.: The seamless web: technology, science, etcetera, etcetera. Social Studies of Science **16**(2), 281–292 (1986)
23. Ion, I., Sachdeva, N., Kumaraguru, P., Čapkun, S.: Home is safer than the cloud! Privacy concerns for consumer cloud storage. In: Proceedings of the Seventh Symposium on Usable Privacy and Security, SOUPS '11, pp. 13:1–13:20. ACM, New York, NY, USA (2011). DOI 10.1145/2078827.2078845. URL http://doi.acm.org/10.1145/2078827.2078845
24. Joint, A., Baker, E., Eccles, E.: Hey, you, get off of that cloud! Computer Law & Security Review **25**(3), 270–274 (2009)
25. Jungk, R., Müllert, N.: Future Workshops. Institute for Social Inventions, London (1987)
26. Kaufman, L.M.: Data security in the world of cloud computing. Security & Privacy, IEEE **7**(4), 61–64 (2009)
27. Kollmuss, A., Agyeman, J.: Mind the gap: Why do people act environmentally and what are the barriers to pro-environmental behavior? Environmental Education Research **8**(3), 239–260 (2002)
28. Latour, B.: Über technische Vermittlung. In: W. Rammert (ed.) Technik und Sozialtheorie, pp. 29–82. Campus, Frankfurt/Main (1998)
29. Latour, B.: Reassembling the Social: An Introduction to Actor-Network-Theory. Oxford University Press, New York (2005)
30. Leimeister, S., Böhm, M., Riedl, C., Krcmar, H.: The business perspective of cloud computing: Actors, roles and value networks. In: 18th European Conference on Information Systems (ECIS 2010) (2010)
31. Li, Z., Chen, C., Wang, K.: Cloud computing for agent-based urban transportation systems. Intelligent Systems, IEEE **26**(1), 73–79 (2011)
32. Loebbecke, C., Thomas, B., Ullrich, T.: Assessing cloud readiness: Introducing the magic matrices method used by Continental AG. In: M. Nüttgens, A. Gadatsch, K. Kautz, I. Schirmer, N. Blinn (eds.) Governance and Sustainability in Information Systems. Managing the Transfer and Diffusion of IT, pp. 270–281. Springer, Heidelberg, Dordrecht, London, New York (2011)
33. Marston, S., Li, Z., Bandyopadhyay, S., Zhang, J., Ghalsasi, A.: Cloud computing—the business perspective. Decision Support Systems **51**(1), 176–189 (2011)
34. Mayring, P.: Qualitative Inhaltsanalyse: Grundlagen und Techniken, 11th edn. Beltz, Weinheim (2010)
35. Mell, P., Grance, T.: The NIST definition of cloud computing. Communications of the ACM **53**(6), 50 (2010)
36. Miller, K.W.: Ethical analysis in the cloud. IT Professional **12**(6), 7–9 (2010)
37. Nelson, M.R.: Building an open cloud. Science **324**(5935), 1656 (2009)
38. Norris, P.: Digital Divide: Civic Engagement, Information Poverty and the Internet in Democratic Societies. Cambridge University Press, Cambridge, MA (2001)
39. Organisation for Economic Co-operation and Development (OECD): OECD Communicaions Outlook 2011. OECD Publishing, Paris (2011)
40. Pandey, S., Voorsluys, W., Niu, S., Khandoker, A., Buyya, R.: An autonomic cloud environment for hosting ECG data analysis services. Future Generation Computer Systems **28**(1), 147–154 (2012)
41. Petruch, K., Stantchev, V., Tamm, G.: A survey on IT-governance aspects of cloud computing. International Journal of Web and Grid Services **7**(3), 268–303 (2011)
42. Rammert, W.: Hybride Handlungsträgerschaft: Ein soziotechnisches Modell verteilten Handelns. In: O. Herzog, T. Schildhauer (eds.) Intelligente Objekte. Technische Gestaltung – Wirtschaftliche Verwertung – Gesellschaftliche Wirkung, pp. 23–33. Springer, Berlin, Heidelberg (2009)
43. Rammert, W., Schulz-Schaeffer, I.: Technik und Handeln: Wenn soziales Handeln sich auf menschliches Verhalten und technische Abläufe verteilt. In: W. Rammert, I. Schulz-Schaeffer (eds.) Können Maschinen handeln? Soziologische Beiträge zum Verhältnis von Mensch und Technik, chap. 6, pp. 11–64. VS, Frankfurt/Main, New York (2002)
44. Ryan, M.D.: Cloud computing privacy concerns on our doorstep. Communications of the ACM **54**(1), 36–38 (2011)

45. Sinus-Sociovision: The Sinus-Milieus® in Germany 2011. http://www.sinus-institut.de/en – last checked: 2013-06-11
46. Sultan, N.A.: Reaching for the "cloud": How SMEs can manage. International Journal of Information Management **31**(3), 272–278 (2011)
47. Truong, D.: How cloud computing enhances competitive advantages: A research model for small businesses. The Business Review, Cambridge **15**(1), 59–65 (2010)
48. Venters, W., Whitley, E.A.: A critical review of cloud computing: Research desires and realities. Journal of Information Technology **27**, 179–197 (2012)
49. Walker, E., Brisken, W., Romney, J.: To lease or not to lease from storage clouds. Computer **43**(4), 44–50 (2010)
50. Wang, C., Ren, K., Lou, W., Li, J.: Toward publicly auditable secure cloud data storage services. Network, IEEE **24**(4), 19–24 (2010)
51. Ward, B.T., Sipior, J.C.: The internet jurisdiction risk of cloud computing. Information Systems Management **27**(4), 334–339 (2010)
52. Weiser, M.: The computer for the 21st century. Scientific american **265**(3), 94–104 (1991)
53. Willcocks, L.P., Venters, W., Whitley, E.A., Hindle, J.: Shifting to cloud services: current challenges and future opportunities. In: M.C. Lacity, L.P. Willcocks (eds.) Advanced Outsourcing Practices: Rethinking ITO, BPO, and Cloud Services, pp. 169–196. Palgrave Macmillan, Basingstoke (2012)
54. Yang, H., Tate, M.: A descriptive literature review and classification of cloud computing research. Communications of the Association for Information Systems **31**, 35–60 (2012)
55. Yeo, C.S., Venugopal, S., Chu, X., Buyya, R.: Autonomic metered pricing for a utility computing service. Future Generation Computer Systems **26**(8), 1368–1380 (2010)
56. Zhang, Q., Cheng, L., Boutaba, R.: Cloud computing: state-of-the-art and research challenges. Journal of Internet Services and Applications **1**(1), 7–18 (2010)
57. Zissis, D., Lekkas, D.: Securing e-government and e-voting with an open cloud computing architecture. Government Information Quarterly **28**(2), 239–251 (2011)

Chapter 19
Cutting Through the Jungle of Cloud Computing Whitepapers: Development of an Evaluation Model

Pascal Grochol, Stephan Schneider, and Ali Sunyaev

Abstract Cloud computing is hyped as a revolutionary form of information technology (IT) resource provision, attracting the attention of numerous researchers and practitioners. This hype results in a disparate collection of whitepapers that act as guidelines or best practices to counter the prevailing challenges in the domain. But the variety of cloud computing whitepapers differs substantially in its quality and relevance for specific contexts. Based on the analytic hierarchy process (AHP), we propose an evaluation model that enables readers to assess the quality and relevance of cloud computing whitepapers in order to select appropriate whitepapers that support their current needs. Furthermore, the evaluation model supports authors to create whitepapers for a specific target audience and accordingly to provide a ready-for-use taxonomy to structure adequate whitepapers. The target audience of this article are potential cloud adopters referring to the business perspective with a demand of high quality whitepapers for their own requirements.

19.1 Introduction

Ongoing trends can lead to innovative information technologies, which many companies feel obligated to adopt for their business processes in a competitive market. But not every hype brings a sustainable result: Apple's Newton and IBM's PCjr are examples for unfulfilled high expectations in information technologies [16]. Just recently, cloud computing emerged as new, on-demand, service-based deployment model for IT resources, bearing high expectations to transform a large part of the IT industry [4]. However, cloud computing environments are characterized by uncertainty and lack of transparency [32]. Hence, researchers and practitioners have published numerous cloud computing whitepapers, resulting in a disparate collection

Pascal Grochol · Stephan Schneider · Ali Sunyaev
Department of Information Systems, Pohligstr. 1, 50969 Cologne, Germany
e-mail: {grochol|schneider|sunyaev}@wiso.uni-koeln.de

H. Krcmar et al. (eds.), *Trusted Cloud Computing*,
DOI 10.1007/978-3-319-12718-7_19

of contributions from research and practice [28]. Whitepapers represent a general compendium of contributions such as guidelines or best practices to address particular challenges of a specific domain. The variety and quantity of cloud computing whitepapers available complicates selection of adequate whitepapers for each context. Therefore, the objective of this paper is to develop an evaluation model for cloud computing whitepapers that enables readers to assess the quality and relevance of existing whitepapers for their specific context and supports authors in creating whitepapers for a specific target audience. This article shall thereby provide guidance to potential cloud adopters that have concrete questions (e.g., concerning security, procurement, contracting, monitoring) to assess the quality of adequate whitepapers for their practical applications. Furthermore, creators of cloud computing whitepapers can use this model to assess requirements or to review their whitepapers. Thus, our model provides a taxonomy to structure existing whitepapers and accordingly to serve as guideline for authors of recent whitepapers.

The remainder of this paper is structured as follows. First, we provide definitions of the key aspects in this paper, that are evaluation model, cloud computing, and whitepaper. We then provide an overview of evaluation models and derive the dimensions of our evaluation model for cloud computing whitepapers. Next, we present the findings of a survey where experts assess our derived dimensions. Finally, we provide a recommendation of an evaluation model for cloud computing whitepapers.

19.2 Background

There is no consistent definition of evaluation models, which allows following a schematic approach. The literature provides various meanings: valuation model, for example, is mostly used in human resource management or investment decisions [14, 33], whereas assessment model is used as a suffix for different specializations, for instance, cost assessment model or performance assessment model [7, 18]. Based on a keyword search of selected synonyms, we aim at finding generic characteristics of evaluation models (cf. Sect. 19.3.1).

CC is an IT deployment model, based on virtualization, that provides on-demand network access to a shared pool of managed and scalable IT resources on a pay-per-use basis [21]. These IT resources refer to hardware (Infrastructure as a Service, IaaS), development platforms (Platform as a Service, PaaS), and applications (Software as a Service, SaaS) and ”can be rapidly provisioned and released with minimal management effort or service provider interaction” [21]. Despite promising opportunities related to cloud computing [4], numerous adoption success stories [23], and auspicious market predictions [13], there is still a high level of uncertainty concerning the adoption of cloud computing [12].

In this paper, cloud computing whitepapers are defined as online documents that focus on a particular topic related to cloud computing and that ideally provide a benefit. Consequently, a whitepaper can provide suggestions, solutions, and refer-

ences for different challenges in the context of cloud computing, such as security, integration, or legal aspects. Since a whitepaper can act as an advertising media, the quality and relevance of whitepapers varies.

19.3 Identifying Dimensions and Criteria for Cloud Computing Whitepapers

19.3.1 Evaluation Models

We conducted a systematic literature review following the guidelines of Webster and Watson [35]. Based on an initial literature search, we identified a list of keywords representing synonyms of the terms *evaluation* and *model*. We constructed search terms from *assessment*, *evaluation*, *rating*, *validation* and *valuation*, and *model*, *paradigm* and *pattern* resulting in a combination of 15 search terms. From now on, we use the term evaluation model as synonym for assessment, rating, validation, and valuation model. We searched the 20 top-ranked journals from the AIS journal ranking [2]. Table 19.1 lists the journals that were included in our keyword research.

# AIS	Journal
1	MIS Quarterly
2	Information Systems Research
3	Communications of the ACM
4	Management Science
5	Journal of Management Information Systems
6	Artificial Intelligence
7	Decision Sciences
8	Harvard Business Review
9	IEEE Transactions (various)
10	AI Magazine
11	European Journal of Information Systems
12	Decision Support Systems
13	IEEE Software
14	Information & Management
15	ACM Transactions on Database Systems
16	IEEE Transactions on Software Engineering
17	ACM Transactions (various)
18	Journal of Computer and System Sciences
19	Sloan Management Review
20	Communications of the AIS

Table 19.1: Journal list

We used the database services of EBSCOhost and ProQuest because most of the journals in Table 19.1 are listed in these databases. Journals that are not listed

in one of the databases were searched manually by using the journal's own online database. For each search, the exact search term has to appear in the metadata of the paper (title, keywords, abstract). We conducted the search in August 2012, which resulted in 51 articles.

The resulting literature shows that evaluation models are used for a variety of contexts. For instance, results matching on the search term assessment model are used in the context of cost assessments [7], impact assessments [34], performance assessments [18], or risk assessments [31]. Besides the wide differences in our 51 literature results, we identified seven papers that represent noticeable similarities within their structure. We divide these similarities into four attributes:

Characteristic
: A characteristic represents the general purpose of the evaluation model that states what should be fundamentally evaluated.

Significance
: The evaluation model addresses itself to a target audience. This given audience turns the characteristic to a significance. The evaluation of a performance, for example, could be a general purpose of the model. But only a target audience decides about the specific kind of performance and how to evaluate.

Partition
: The evaluating object, that is given by its significance, is divided into partitions. If, for example, the efficiency of an organization is the significance of the model, then a partition of business processes is possible.

Method
: Measure and evaluation are based on scientific methods (e.g., methods of a decision theory).

Table 19.2 presents the attributes of the seven similar evaluation models identified in the literature. Here, the different models demonstrate a wide variety of scientific methods. In addition, the comparison shows that the authors' decision relating to the partition and chosen method results in subjective models. On the one hand, this subjectivity is shown additionally by statements like "the structure of the model we developed was predicated on our desire to assist administrators in making decisions" [17] and "we believe that the proposed model could facilitate the design of QAS" [25]. But on the other hand, the targets in these statements are clearly defined with regard to get an adequate evaluation model. Therefore, the comparison in Table 19.2 confirms a differentiation in evaluation models but also illustrates that similarities at a conceptual level exist. These findings are the basis to develop an evaluation model for cloud computing whitepapers.

An evaluation of cloud computing whitepapers has to measure the quality of whitepapers for a specific target audience. The audience is represented by companies that are interested in adopting cloud computing. However, depending on which type of service a company is planning to adopt, the level of integration within the internal IT, existing management structures, and business processes within the company, different information needs have to be satisfied by cloud computing whitepapers. These two aspects indicate that the significance has to be a combination of two

Model	Characteristic	Significance	Partition	Method
A [19]	Quality and trustworthiness of SaaS	Reputation, quality and cost	Parameters that measure behavior of services (e.g., uptime, response time)	Generic automation with different mathematical functions (depending on each parameter)
B [25]	Satisfaction of question answering systems	Reliability for end users	Ease of use, usefulness, qualities	Likert scale
C [30]	Evaluation of data retrieval systems	Work load and performance during data retrievals	Parameters of measurement (e.g., disk rotations, response time)	Stochastic model (Zipf distribution)
D [17]	Accessibility for mentally retarded (relating to MIS)	Efficiency of resource allocation	Different kinds of constraints (of mentally retarded)	Linear programming model
E [10]	Evaluation of outsourcing provider for SMEs	Performance and capacity	Capacities and demands of SMEs	Delphi method, analytic hierarchy process
F [7]	Cost assessment for enterprises	Cost estimation (with goal of cost reduction)	Categories of cost (e.g., planning cost, implementation cost)	Infrastructure-development methods (e.g., waterfall)
G [18]	Assessment of e-commerce performance	Efficiency of using e-commerce	Business processes (e.g., sales area, collaboration with partners)	Methods of operation research (e.g., DEA, TOPSIS)

Table 19.2: Attributes of evaluation models found in the literature

dimensions. At first, the significance has to contain a relevancy factor for the companies, representing the information needs regarding cloud computing. In addition to the relevance, the overall quality of a whitepaper has to be assessed. In a nutshell, significance consists of relevance and quality.

19.3.2 Relevancy Dimensions

The evaluation model should be a response to the issue of interests in cloud computing. Therefore, the relevance to companies represents one significance of this model, so an answer of following question should be given: what are the topics of cloud computing that companies are interested in? To answer this question, a method originating from decision theory should act as a fundamental method. Chang et al. [10] developed a model that is similar to the relevance part of our evaluation model. On the one hand, the evaluation of IT outsourcing providers is near to the topic of cloud computing. On the other hand, they use an empirical approach with the analytic hierarchy process (AHP), a method originating from decision theory. The AHP was developed by Saaty [26] and provides an opportunity to prioritize alternatives

within hierarchical structures. This method of prioritization has been applied to different domains, from investment decisions [8] to evaluation of software suites [6]. Consequently, the analytic hierarchy process seems qualified for applying to our evaluation model with regard to the relevance part.

The AHP requires a selection of elements that are arranged to a hierarchical structure. To evaluate the interests of cloud computing, an adequate selection of cloud computing topics is important. Therefore, we have to break down the generic term to its sub-categories that fits the required structure [27]. To categorize cloud computing, we use an iterative method for taxonomy development [24]. The objective is to create a useful taxonomy with regard to the relevance for companies and that simultaneously fits the AHP's needs. Thus, we determine following ending conditions to terminate the iterative process of the taxonomy development method: no more than four dimensions (direct sub-categories of cloud computing) with no more than five characteristics (direct sub-categories of a dimension) in each case. In addition, a total number of 15 characteristics must not be exceeded. These ending conditions prevent a high quantity of AHP comparisons afterwards.

The sources of research are focused on literature that, first, gives an extensive overview of cloud computing [3], and second, refers to the business processes of cloud computing [20]. After five iterations of the taxonomy development method we reached our defined ending conditions with 4 dimensions and a total number of 14 characteristics that are explained in Table 19.3. The converted hierarchy structure is shown in Fig. 19.1.

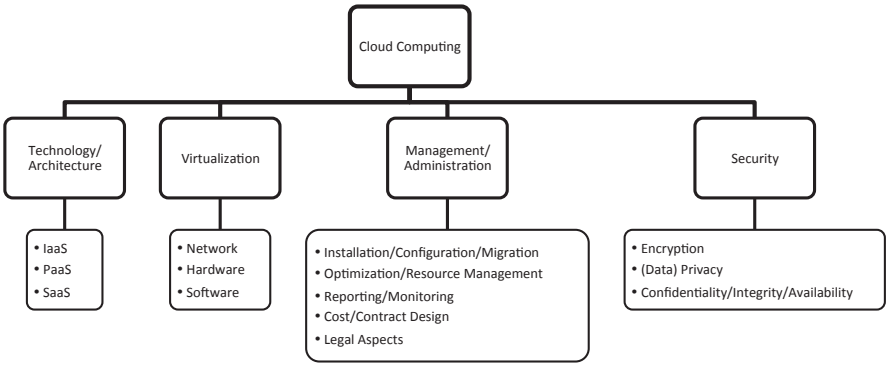


Fig. 19.1: Hierarchy structure of cloud computing

Dimension	Characteristic	Explanation
Technology/ Architecture	IaaS	Services that provide infrastructure on demand (this includes virtualized hardware resources such as CPU, storage, networks)
	PaaS	Services that provide development/runtime environments on demand
	SaaS	Services that provide (software) applications on demand
Virtualization	Network	Virtual Local Area Networks, Virtual Private Networks
	Hardware	Virtualization of desktops, servers and storage
	Software	Virtualization of software applications and operating systems
Management/ Administration	Installation/Configuration/ Migration	Administrative tasks during introduction of or transition to cloud computing (e.g., changes in organizational processes)
	Optimization/Resource Management	Allocation of (hardware) resources (e.g., to improve scalability or to ensure interoperability)
	Reporting/Monitoring	Quality assurance, bug fixing/debugging, performance control
	Cost/Contract Design	Cost calculation/analysis/reduction, types of contracts
	Legal Aspects	E.g., licensing, legal aspects at an international level
Security	Encryption	Instructions, pros and cons of encryption methods
	(Data) Privacy	Protecting of corporate and personal data, and ensuring of anonymity/privacy
	Confidentiality/Integrity/ Availability	Represent the core themes of information security

Table 19.3: Explanation of cloud computing characteristics

19.3.3 Quality Dimensions

Having identified the relevance of a whitepaper for a company, the actual quality needs to be assessed. A relevant whitepaper does not automatically imply a good one in terms of quality. Therefore, another part of significance in our evaluation model is represented by the quality of whitepapers. From previous research on evaluation models, we have identified that quality could be abstracted in the context of its meaning [19]. Here, Limam and Boutaba [19] determined that the definition of quality dimensions (regarding to Quality of Service) is a critical issue. As a result, they abstract the Quality of Service to common web services, so a comparison and adaption of quality dimensions (with regard to web services) is offered.

To the best of our knowledge, there is no literature that provides quality dimensions with regard to whitepapers. Thus, the abstraction approach is adopted to trans-

fer whitepapers to a more general definition. Whitepapers are documents without subjecting to any standards. Because of a free design and their public availability, they can be broadly considered as an online document. With this definition, literature exists that provides quality dimensions [11]. Eayrs [11] argues that online sources are not checked by any reviews, so they need a classification of quality criteria. He refers to the quality dimensions of Beck [5] and adapted them to publications of the World Wide Web. Beck herself updated her dimensions *accuracy*, *authority*, *objectivity*, *currency* and *coverage* with regard to online documents as well. They are presented in the form of questions so it could be answered if an online source seems to be reliable. These quality dimensions are similar to the 10 dimensions of information quality [22] that were used and adapted to evaluate Question Answering Systems [25]. The adaption varies between multiple papers, so, for example, the dimensions are also used as a subset of other quality dimensions to evaluate the quality of digital libraries [15] or as a slight modification to evaluate web-located health information [29]. The following questions should be answered to assess the value of each dimension:

Accuracy
: Is the document error-free with regards to content? Is the information reliable?

Authority
: Where does the document come from? Who is the author? Is the author qualified?

Objectivity
: Does the document show a bias? Is the information trying to influence an opinion? Is the document free of ads?

Currency
: Is the content up-to-date? Does the document provide any news value?

Coverage
: Is the range of topics adequate? Is the range of topics relevant? Is the range of topics explored in reasonable depth?

These quality dimensions are also suited for application within the AHP, hence a converted hierarchy structure is shown in Fig. 19.2.

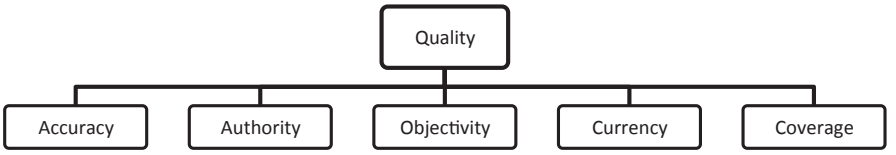


Fig. 19.2: Hierarchy structure of quality dimensions

19.4 Research Method

In order to assess the relative importance of the quality dimensions, a survey was conducted. The main part of the survey uses a layout that conforms to the analytic hierarchy process. Participants of the survey have to assess different terms in paired comparisons [26]. These terms comply with our developed hierarchies in Fig. 19.1 and Fig. 19.2. Relating to Fig. 19.1, the participants first have to compare the relative importance of all characteristics within their corresponding dimensions (e.g., relative importance of encryption compared to privacy in the dimension security). Second, they have to assess the relative importance of all dimensions with regard to cloud computing (e.g., relative importance of virtualization compared to security). Relating to the quality dimensions in Fig. 19.2, the same procedure is applied analogously.

Scale	Definition
1	Equal importance
3	Weak importance
5	Essential or strong importance
7	Demonstrated importance
9	Absolute importance
2, 4, 6, 8	Intermediate values

Table 19.4: Assessment scale

To make the comparisons, Saaty [26] suggested a numerical scale with a range from 1 to 9. Via this scale, a participant of the survey is able to assess the relative importance of an attribute X versus the importance of another attribute Y in a range from ”equal importance” (1) to ”absolute importance” (9). Table 19.4 illustrates the whole assessment range with corresponding definitions.

The assessment of two attributes is designed with a 17-digit scale that combines the alternatives of relative importance of attribute X over Y or attribute Y over X. As illustrated in Fig. 19.3, the left attribute is considered to be 6 times more important than the right attribute or, in other words, virtualization has an essential importance over security in this example.

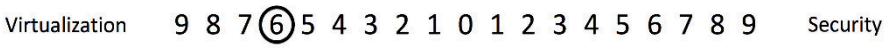


Fig. 19.3: Example of the survey’s 17-digit scale

To obtain participants for our survey, we used a business social network. With the focus on professionals, we posted our survey in different discussion groups. Therefore, we select discussion groups that are interested in general IT topics or are dedicated to specific cloud computing topics. The survey was online for three weeks

and 174 individuals followed the link, of which 66 were valid participants finished the survey. Among these valid participants, 40 (60 %) possess a professional IT knowledge for more than ten years. 58 (88 %) of participants are using cloud computing or have gained experience in this topic, and only 2 (3 %) participants never have gained any personal experience in cloud computing. Nearly three quarters of valid participants (74 %) stated to be executives, business managers, IT managers, or consultants.

Due to potential inconsistencies[1] during the assessment process, a consistency test is necessary to prepare an analysis with the analytic hierarchy process [27]. We set a consistency ratio (CR) of 0.2 as the maximum tolerable value. This acts as a compromise between Saaty's [27] suggestion and a higher loss of empirical data [1, 9]. For the relevancy dimensions, 45 out of 66 valid participants have reached a CR $\leq$ 0.2, where 29 belong to small and medium-sized enterprises (SMEs) and only 16 belong to large enterprises. For the quality dimensions, 46 have reached our maximum CR value: 29 from SMEs and 17 from large enterprises. Participants with CR $>$ 0.2 were dropped in the respective dimension. We used Expert Choice 11.5 to assess the consistencies and to analyze the prioritization of the dimensions in our hierarchy structures of cloud computing and whitepaper quality.

19.5 Results

At the relevance part of our survey, we are able to analyze the data of 45 respondents. Figure 19.4 illustrates the results of the structured cloud computing hierarchy and represents a first prioritization of all valid participants within the defined consistency ratio ($\leq$ 0.2). However, a separation of the 29 small and medium-sized enterprises and the 16 large enterprises results in significant differences of prioritization, and therefore the firms' interests. For instance, for SMEs, security is nearly twice as important as management topics (cf. Fig. 19.5). But for large enterprises, management topics are rated considerably higher in importance than for SMEs (cf. Fig. 19.6). In addition, this has an impact on the characteristics (on the left of Fig. 19.5 and Fig. 19.6). For example, software virtualization is considered as the most important kind of virtualization for SMEs, whereas hardware is the top characteristic of virtualization from the perspective of large enterprises. Hence, a separate view of enterprise sizes seems essential.

The hierarchy of quality dimensions is also suited for using with the AHP, so the participants of the survey have to assess these dimensions. At this part of our survey we have reached an overall quantity of 46 respondents with a CR $\leq$ 0.2. Figure 19.7 to 19.9 are ordered by priority and demonstrate a similar phenomenon of different points of view. Accordingly, the separation of all valid participants (Fig. 19.7) to small and medium (Fig. 19.8) and to large enterprises (Fig. 19.9) illustrates a change

[1] Caused by intransitive relations during pairwise comparisons (e.g., a relation is transitive whenever an attribute X is more important than Y, and Y is more important than Z, then also X has to be more important than Z).

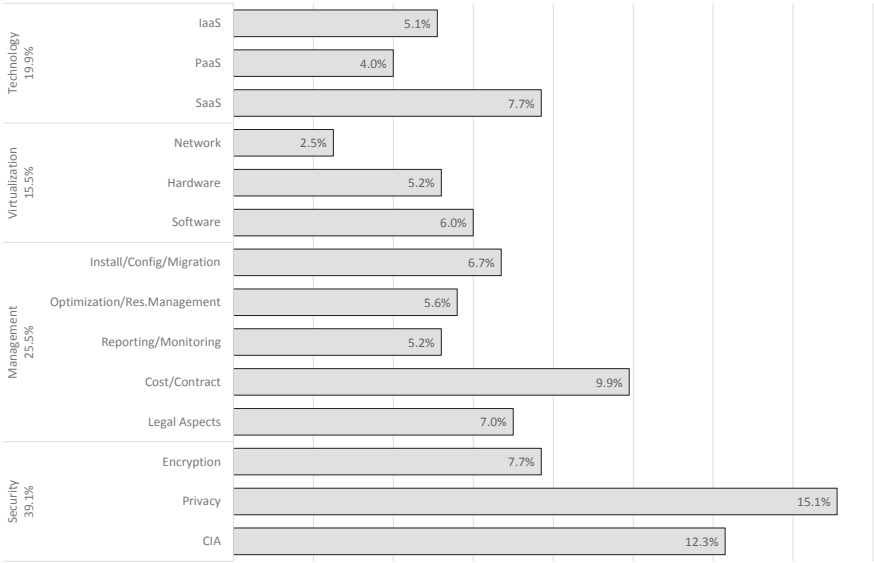


Fig. 19.4: Relevance of all valid participants with CR ≤ 0.2

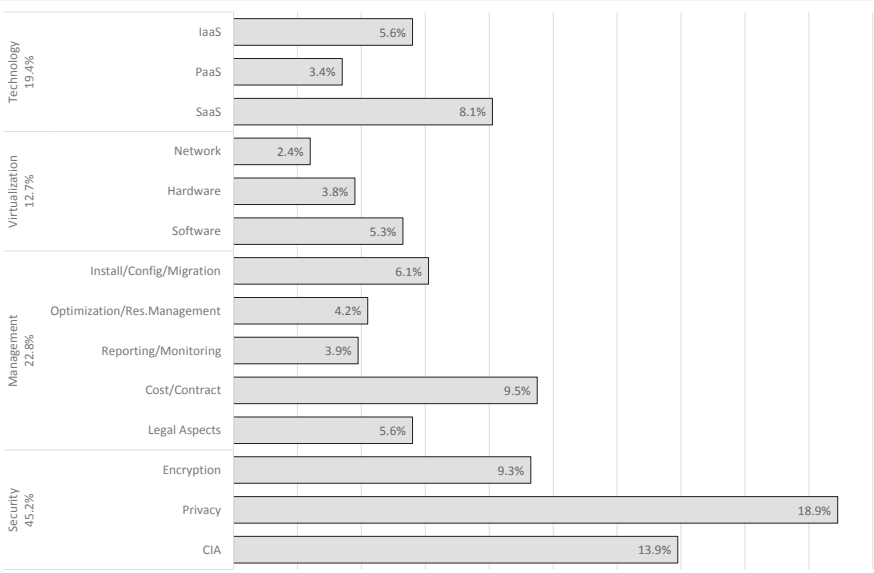


Fig. 19.5: Relevance of SME participants with CR ≤ 0.2

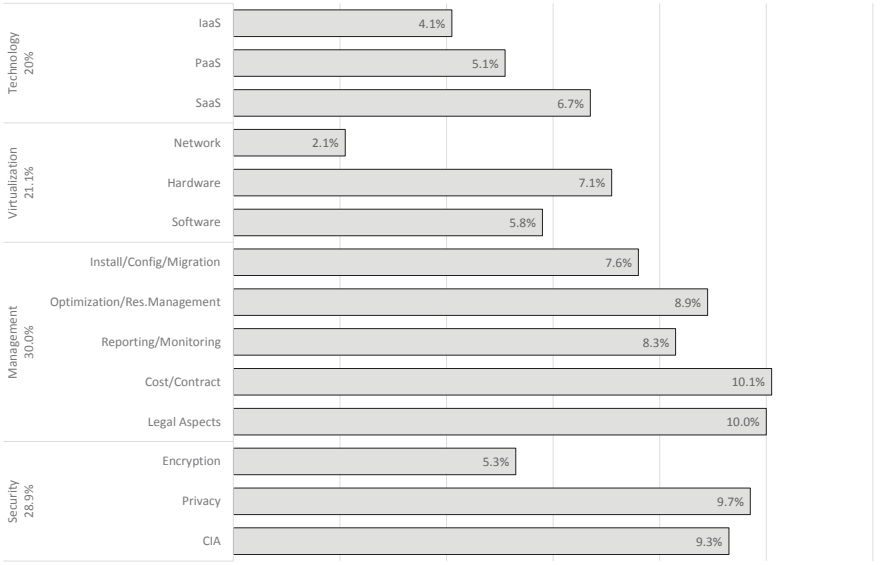


Fig. 19.6: Relevance of large enterprise participants with CR ≤ 0.2

of prioritization. As a result, the accuracy of a document is the most important quality dimension, independent of the enterprise size. But currency is a dimension that makes the key difference. An up-to-date document seems to be nearly as important as a given objectivity for SMEs, whereas currency is much less important for large enterprises, so it brings up the rear with coverage and authority. This shift of the currency dimension (and a maximum priority tolerance of 3 %) reveals two different patterns at the analysis graphs of quality dimensions:

SMEs
accuracy $>$ currency and objectivity $>$ coverage and authority
LEs
accuracy $>$ objectivity $>$ currency and coverage and authority

Because of a limited number of participants from large enterprises, two separate approaches of an evaluation model will not be comparable with each other. For the relevance dimension, as is shown by the comparison between Fig. 19.4 (all valid participants) and Fig. 19.5 (only SMEs), the order and values of prioritization only varies in few cases. This similarity is caused by the majority of SMEs in the data set, so a direct comparison with the few data of large enterprises would not be fair.

Therefore, we focus on the empirical data of SMEs. For the quality dimension, the limited number of 17 large enterprise participants compared to 29 participants from SMEs lead to the conclusion as previously mentioned, so that a focus on small and medium-sized enterprises is necessary.

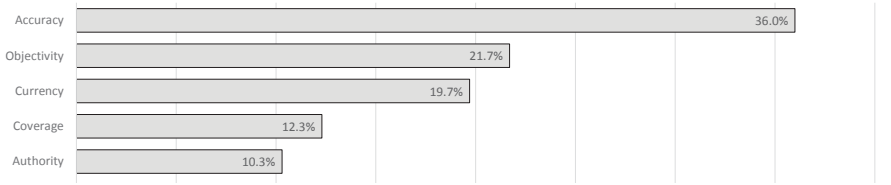


Fig. 19.7: Quality dimensions of all valid participants with CR $\leq$ 0.2

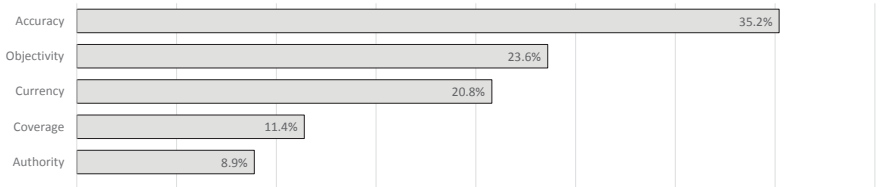


Fig. 19.8: Quality dimensions of SME participants with CR $\leq$ 0.2

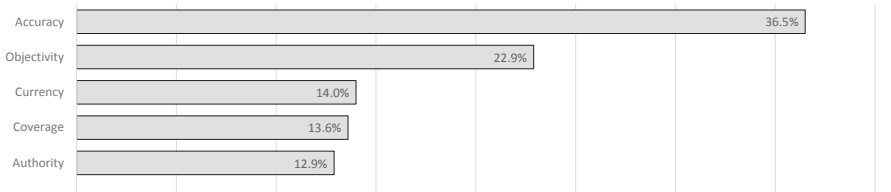


Fig. 19.9: Quality dimensions of large enterprise participants with CR $\leq$ 0.2

19.6 Evaluation Model for Cloud Computing Whitepapers

In order to integrate our findings from the literature analysis and empirical study into an applicable evaluation model, we follow recent research and introduce a scoring system [10]. The basis for the scoring system is our development of relevance and quality dimensions, and the results of the prioritization of our survey analysis. Table 19.5 shows the scoring system for the relevance part of our evaluation model. The scores are based on the percentage values of each characteristic, with the total score set at 100 points. Relating to the quality dimensions, the same procedure is applied analogously in Table 19.6. The scoring system represents a recommendation for the evaluation of cloud computing whitepapers and provides an instrument to evaluate the relevance and quality of these documents. Using this scoring system, whitepapers can be classified and compared to each other.

Dimension	Characteristic	Points
Technology	IaaS	6
	PaaS	3
	SaaS	8
Virtualization	Network	2
	Hardware	4
	Software	5
Management	Install/Configuration/Migration	6
	Optimization/Resource Management	4
	Reporting/Monitoring	4
	Cost/Contract Design	10
	Legal Aspects	6
Security	Encryption	9
	(Data) Privacy	19
	Confidentiality/Integrity/Availability	14
Total		100

Table 19.5: Scoring system (relevance part)

Quality dimension	Points
Accuracy	35
Authority	9
Objectivity	21
Currency	24
Coverage	11
Total	100

Table 19.6: Scoring system (quality part)

19.7 Conclusion

This paper contributes to research and practice. The contribution to research is twofold. First, with the comparison of different approaches and the corresponding similarities of evaluation models, we add to the body of knowledge on research on evaluation models. As a result, a basis to develop an individual evaluation model is defined. Second, we manage an assessment of quality of online documents and demonstrate the academic value by adapting these quality dimensions in the context of whitepapers. The contribution to practice is twofold. First, a graduation on the basis of a scoring system enables a methodical orientation. Thus, for example, a direct comparison of whitepapers is possible by calculating their overall scores, or a general orientation for a single whitepaper is given by means of each value from Table 19.5 and 19.6. Second, our scoring system represents a template to design new whitepapers. Hence, on the one hand, it provides a template that paves the way for relevant cloud computing topics, and on the other hand it provides a template that assists to create whitepapers of high quality and classify existing whitepapers, for instance, according to the relevance hierarchy structure.

As with any research, this paper has its limitations. First, the number of experts who participated in the survey only reached a small number. But previous research has shown that this amount seems adequate for a AHP-based survey [10]. Second, because of this limited number of participants we focus on SMEs. Further research should conduct a more elaborate survey for large enterprises or focus on a specific subtopic (e.g., SaaS whitepapers). Third, a precise evaluation of whitepapers is difficult to realize, because whitepapers are not defined by any guidelines. Therefore, a careless application of the scoring system could result in inaccurate results. So, for example, whitepapers with a general roundup of cloud computing would be preferred over specific ones due to the quantity of covered cloud computing topics. Here, a certain amount of experience is essential to be capable of specify the key issues before applying the scoring system to this kind of whitepapers. The scoring system represents a template that is disposed to get customized as may be required. Fourth, the interest in cloud computing topics may vary over time and across different organizations (e.g., an increasing interest in data privacy that occurs during political discussions in consequence of global security concerns). Fifth, the assessment with our dimensions often implies a qualitative analysis (especially with our quality dimensions), which is why certain subjectivity exists.

However, this paper is a first endeavor to shed light on the magnitude of whitepapers that exist in the context of cloud computing and provides an instrument that is ready-to-use and highly customizable at once.

References

1. Aguarón, J., Moreno-Jiménez, J.M.: The geometric consistency index: Approximated thresholds. European Journal of Operational Research **147**(1), 137–145 (2003). DOI 10.1016/S0377-2217(02)00255-2
2. AIS: Senior scholars' basket of journals (2011). URL `http://home.aisnet.org/displaycommon.cfm?an=1\&subarticlenbr=346`
3. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., Zaharia, M.: Above the clouds: A berkeley view of cloud computing: Tech. report (2009). URL `http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html`
4. Armbrust, M., Stoica, I., Zaharia, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A.: A view of cloud computing. Communications of the ACM **53**(4), 50 (2010). DOI 10.1145/1721654.1721672
5. Beck, S.: The good, the bad & the ugly: or, why it's a good idea to evaluate web sources (1997). URL `http://lib.nmsu.edu/instruction/evalcrit.html`
6. Benlian, A.: Is traditional, open-source, or on-demand first choice? developing an ahp-based framework for the comparison of different software models in office suites selection. European Journal of Information Systems **20**(5), 542–559 (2011). DOI 10.1057/ejis.2011.14
7. Bhattacharjee, S., Ramesh, R.: Enterprise computing environments and cost assessment. Communications of the ACM **43**(10), 74–82 (2000). DOI 10.1145/352183.352208
8. Bodin, L.D., Gordon, L.A., Loeb, M.P.: Evaluating information security investments using the analytic hierarchy process. Communications of the ACM **48**(2), 78–83 (2005). DOI 10.1145/1042091.1042094
9. Byun, D.H.: The ahp approach for selecting an automobile purchase model. Information & Management **38**(5), 289–297 (2001). DOI 10.1016/S0378-7206(00)00071-9

10. Chang, S.I., Yen, D.C., Ng, C.S.P., Chang, W.T.: An analysis of it/is outsourcing provider selection for small- and medium-sized enterprises in taiwan. Information & Management **49**(5), 199–209 (2012). DOI 10.1016/j.im.2012.03.001
11. Eayrs, M.: A principled methodology for information retrieval on the web. Issues in Informing Science and Information Technology **3**(1), 211–218 (2006)
12. European Commission: Cloud computing: Public consultation report (2011). URL `http://ec.europa.eu/information_society/activities/cloudcomputing/docs/ccconsultationfinalreport.pdf`
13. Gartner Research: Gartner says worldwide cloud services market to surpass $109 billion in 2012 (2012). URL `http://www.gartner.com/it/page.jsp?id=2163616`
14. Gillespie, J.F., Leininger, W.E., Kahalas, H.: A human resource planning and valuation model. Academy of Management Journal **19**(4), 650–656 (1976). DOI 10.2307/255798
15. Gonçalves, M.A., Moreira, B.L., Fox, E.A., Watson, L.T.: "what is a good digital library?" – a quality model for digital libraries. Information Processing & Management **43**(5), 1416–1437 (2007). DOI 10.1016/j.ipm.2006.11.010
16. Haskin, D.: Don't believe the hype: The 21 biggest technology flops (2007). URL `http://www.computerworld.com/s/article/9012345/Don_t_Believe_the_Hype_The_21_Biggest_Technology_Flops`
17. Heiner, K., Wallace, W.A., Young, K.: A resource allocation and evaluation model for providing services to the mentally retarded. Management Science **27**(7), 769–784 (1981). DOI 10.1287/mnsc.27.7.769
18. Huang, J., Jiang, X., Tang, Q.: An e-commerce performance assessment model: Its development and an initial test on e-commerce applications in the retail sector of china. Information & Management **46**(2), 100–108 (2009). DOI 10.1016/j.im.2008.12.003
19. Limam, N., Boutaba, R.: Assessing software service quality and trustworthiness at selection time. IEEE Transactions on Software Engineering **36**(4), 559–574 (2010). DOI 10.1109/TSE.2010.2
20. Marston, S., Li, Z., Bandyopadhyay, S., Zhang, J., Ghalsasi, A.: Cloud computing — the business perspective. Decision Support Systems **51**(1), 176–189 (2011). DOI 10.1016/j.dss.2010.12.006
21. Mell, P., Grance, T.: The nist definition of cloud computing: Recommendations of the national institute of standards and technology. URL `http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf`
22. Miller, H.: The multiple dimensions of information quality. Information Systems Management **13**(2), 79–82 (1996). DOI 10.1080/10580539608906992
23. Narasimhan, B., Nichols, R.: State of cloud applications and platforms: The cloud adopters' view. Computer **44**(3), 24–28 (2011). DOI 10.1109/MC.2011.66
24. Nickerson, R.C., Varshney, U., Muntermann, J.: A method for taxonomy development and its application in information systems. European Journal of Information Systems (2012). DOI 10.1057/ejis.2012.26
25. Ong, C.S., Day, M.Y., Hsu, W.L.: The measurement of user satisfaction with question answering systems. Information & Management **46**(7), 397–403 (2009). DOI 10.1016/j.im.2009.07.004
26. Saaty, T.L.: A scaling method for priorities in hierarchical structures. Journal of Mathematical Psychology **15**(3), 234–281 (1977). DOI 10.1016/0022-2496(77)90033-5
27. Saaty, T.L.: How to make a decision: The analytic hierarchy process. European Journal of Operational Research **48**(1), 9–26 (1990). DOI 10.1016/0377-2217(90)90057-I
28. Schneider, S., Lansing, J., Gao, F., Sunyaev, A.: A taxonomic perspective on certification schemes: Development of a taxonomy for cloud service certification criteria. In: Proceedings of the 47th Hawaii International Conference on System Sciences (HICSS 2014), pp. 4998–5007. Big Island, Hawaii, USA (2014). DOI 10.1109/HICSS.2014.614
29. Sellitto, C.: Towards a weighted average framework for evaluating the quality of web-located health information. Journal of Information Science **31**(4), 260–272 (2005). DOI 10.1177/0165551505054168

30. Siler, K.F.: A stochastic evaluation model for database organizations in data retrieval systems. Communications of the ACM **19**(2), 84–95 (1976). DOI 10.1145/359997.360010
31. Sun, L., Ivastava, R.P., Mock, T.J.: An information systems security risk assessment model under the dempster-shafer theory of belief functions. Journal of Management Information Systems **22**(4), 109–142 (2006). DOI 10.2753/MIS0742-1222220405
32. Sunyaev, A., Schneider, S.: Cloud services certification. Communications of the ACM **56**, 33–36 (2013)
33. Tam, K.Y.: The impact of information technology investments on firm performance and evaluation: Evidence from newly industrialized economies. Information Systems Research **9**(1), 85–98 (1998). DOI 10.1287/isre.9.1.85
34. Weber, R.: Computer technology and jobs: an impact assessment model. Communications of the ACM **31**(1), 68–77 (1988). DOI 10.1145/35043.35049
35. Webster, J., Watson, R.T.: Analyzing the past to prepare for the future: Writing a literature review. MIS Quarterly **26**(2), xiii–xxiii (2002)