International Journal of Information Security (2021) 20:553–570
https://doi.org/10.1007/s10207-020-00518-3

REGULAR CONTRIBUTION

DeTRACT: a decentralized, transparent, immutable and open PKI certificate framework

Thomas Sermpinis[1] · George Vlahavas[1] · Konstantinos Karasavvas[1] · Athena Vakali[1]

Published online: 27 August 2020

Abstract
Public key infrastructure (PKI) is widely used over the Internet to secure and to encrypt communication among parties. PKI involves digital certificates which are managed by certificate authorities (CAs) that authenticate users identity, in order to establish encrypted communication channels. The centralized operation model of CAs has already caused several targeted attacks due to the distribution of rogue certificates. Users remain vulnerable since it is too challenging to detect and revoke such certificates, but also to speed up the user update process when a certificate is revoked. To address such issues, a decentralized PKI alternative approach, targeting Domain Validated certificates, is proposed. In the proposed approach, which is based on blockchain technologies (such as Bitcoin and Ethereum), the transparency, immutability and decentralization aspects of these technologies have been leveraged. Comparisons among the proposed approach, the conventional PKI and other decentralized approaches have been implemented to showcase the impact and the potential of the proposed approach.

Keywords Decentralized PKI · Blockchain technologies · Certificate management

1 Introduction

A secure model is necessary to safeguard the intense and evolving Internet users interactions, offering trusted user roles' certification. Public key infrastructure (PKI) empowers entities to link their physical identity with the digital one, so that everyone acknowledges and trusts communication channels, which are used among users. PKI binds the physical identity of a user with certificates, which are issued, verified and revoked by a set of centralized entities, called certificate authorities (CAs). These certificates (or public keys) are the actual digital identities which correspond to individuals, and CAs verify the link between the users digital and the physical identity [1]. In public key cryptography, which is PKI's approach to encrypt the communication between a server and the end user of a Web service, a "key pair" is used. This key pair consists of a "private key" which is known only to the owner of the key pair, and the "public key" which may be disseminated widely. The public key is actually derived from the private key by applying special "one-way" cryptographic functions, such that everyone can encrypt a message with the public key, but only the owner of the corresponding private key can decrypt this message [2]. The RSA algorithm [3] is typically used for that purpose, although it is possible to use other algorithms as well.

CAs are trusted parties which are built into any device and software that requires a secure communication between a client and a server. PKI follows a hierarchical structure, with two types of CAs: root CAs and intermediate CAs. CAs that are higher in the hierarchy have the authority to issue certificates to other lower hierarchy CAs, in order to enable the lower hierarchy CAs to issue certificates to users. CAs are commonly chosen by the vendor of the application or by the operating system. The PKI system consists of several components including CAs, a registration authority, a directory to store and index keys and certificates, as well as a certificate management system [4].

The PKI model that is commonly used today follows a centralized model of operation and it is prone to single point of failure vulnerabilities. This is due to the fact that CAs might issue rogue certificates with no validations guaranteed. Several cases where such rogue certificate were issued have been reported with negative implications and attacks targeting many users [5–8]. In most of these cases, it is a matter of which CA is considered as "trusted"

✉ George Vlahavas
gvlahavas@csd.auth.gr

[1] School of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece

by the other CAs and how the verification can be safeguarded.

This work places emphasis on resolving the single point of failure problem which is dominant in typical PKI systems due to its demand for trust in centralized CAs. The novelty of the proposed work is on its blockchain technology exploitation to leverage the concept of decentralization. Blockchain technologies have become popular with their use in Bitcoin, a peer-to-peer electronic cash system [9]. They maintain an immutable chain of interlinked blocks, implemented as a simple database that stores the transactions of an underlying network of users who communicate and interact. Apart from Bitcoin, blockchain technologies have been proven beneficial to many other application domains, like education [10] and medicine [11], which leverage its non-centralized nature and its private–public key pair requirement for transaction verification. This verification is employed by a cryptography-based approach involving all other networked entities, called "nodes." The nodes in a blockchain network refer to clients that support the network's trust by having a copy of all the blocks and transactions that happen on this network. These characteristics of blockchain technologies provide them with the capability to support a trustless PKI certificate issuing process.

To address the PKI single point of failure problem, a decentralization framework, *DeTRACT*, is introduced to enable *de*centralized, *tra*nsparent, immutable and open PKI *c*ertificate revoca*t*ion. The proposed *DeTRACT* framework enables certificates creation, updating and revocation by exploiting blockchain based technologies (such as the popular Bitcoin and Ethereum [12]). Unlike other implementations [13–16], the proposed approach aims to decentralize the process through public blockchains, mainly by improving the certificate revocation aspect. Under *DeTRACT*, the user (from now on stated as the "Domain Owner") creates self-signed certificates and stores them on either of the two of the most secure blockchain networks, namely Bitcoin and Ethereum, to fully exploit the blockchain characteristics. Thus, *DeTRACT* certificates will be tamper resistant (due to the decentralized aspect of blockchains), in contrast to the current conventional centralized approach. *DeTRACT* only targets Domain Validated (DV) certificates that are used in Web site communication encryption between a client and a server, and where only proof of control over a DNS domain is needed. The certificates that are issued with *DeTRACT* are self-signed. This is due to the fact that the domain owner signs their own certificates, because there is no central authority to manage the issuance of keys and certificates. To validate the identity of the owner, "uPort" [17], a decentralized identification system based on blockchain technologies is used.

DeTRACT's main contributions are due to its blockchain-driven PKI trusted certificate solutions which are characterized by the following properties:

- *Decentralized* since a blockchain stores data across a network, serious risks of having data managed by a central entity are eliminated. Using distributed networking, there are no central points of reference, so there is no single point of failure. In blockchains, every node has a copy of the entire transaction history. This results in the assurance that data have not been tampered with or changed, due to massive database replication methods as well as computational trust [18]. No central trusted CA and no CA hierarchies are involved, so *DeTRACT* overcomes questionable transfer of trust;
- *Transparent* since all transactions on public blockchains can be viewed by anyone, transparency will be safeguarded by establishing a network which will verify that no counterfeit certificate is produced, thus preventing rogue certificate issuing by domain owners. In *DeTRACT*, there will be no trusted parties to validate the certificates, but transparency will allow network users to verify certificates and domain owners to prevent any undesirable action, such as fake identities or certificates;
- *Immutability of data* since once a transaction for a certificate has been recorded in the blockchain by a node, no one can alter or remove it. This core difference between a blockchain and a conventional database system will enable a trusted and non-editable certificate issuing approach;
- *Process integrity* since users can be certain that transactions will follow the rules set by the community supporting the proposed blockchain network. That way, the need for a trusted centralized CA in the *DeTRACT* framework, for the purpose of setting rules, is eliminated;
- *Openness* since public blockchains are open source and permissionless technologies, thus, everyone can use a public blockchain, participate in the certificate issuing network and even fork the source code to create their own variation of the blockchain.

To implement the *DeTRACT* framework, two different approaches were developed, each one exploiting different capabilities of the blockchain technologies:

- *DeTRACT-T*, i.e., a transaction-based approach, which exploits the capability of storing metadata alongside blockchain transactions, using the scripting language of Bitcoin [19]. Also, the Ethereum blockchain, an alternative blockchain-based cryptocurrency can be used, so that the different blockchain characteristics that it has to offer, like faster transactions and easier storage capabilities, can be exploited by the *DeTRACT-T* approach.
- *DeTRACT-SC*, i.e., smart contracts approach in the Ethereum blockchain, which are self-executing contracts written in a turing-complete programming language that are stored in the blockchain and are decentralized. Smart

contracts are used in order to decentralize the process of certificate creation, updating and revocation of the *DeTRACT* framework and back it up with the performance and security enhancements of smart contracts.

These approaches will be compared and evaluated with respect to their characteristics.

In order to address the single point of failure problem with PKI, several proposals appear in the literature. Most of them still retain the use of a centralized component like a CA, while some are based on blockchain technologies. For a more detailed description of related projects, see Sect. 2. Unlike these implementations, the proposed framework aims to decentralize the process through public blockchains, by improving the certificate revocation aspect, where many of them are lagging behind.

The structure of this paper is as follows: In the next section, related work is presented, along with a comparison between the different solutions. In Sect. 3, the *DeTRACT* framework is outlined with emphasis on the proposed decentralized PKI approach and with details about the specific decentralized technologies that have been adopted. In Sect. 4, the proposed *DeTRACT* approach is evaluated with a specific cost analysis, which allows for a comparison among the different blockchain technologies used here and other decentralized PKI alternatives. Finally, conclusions are summarized in the last section and extensions to the proposed solution are indicated.

2 Related work

In this section, solutions that have been proposed in the literature, to improve on the existing PKI practices, some of whom are actually used in practice, are presented and compared according to their characteristics.

Several implementations of PKI exist, with SSL/TLS [29] being one of the most widely used. Also, many alternatives have been implemented using blockchain technologies, with some of them retaining CAs in their process [13–16,30,31]. In addition, some approaches try to completely decentralize the process, either by creating a new blockchain implementation [25], by using preexisting blockchains [28] or by utilizing a web-of-trust peer-to-peer network [15,27].

The web-of-trust concept is used by some solutions [15,27], in order to decentralize PKI. The web-of-trust is a decentralized concept which aims to bind a physical identity with a public key. It is a decentralized alternative to the PKI model, which uses self-signed certificates. In this concept, all the network entities act as agents of trust, where in PKI only the CA is trusted. The trust here is "bi-directional," and everyone in the network contributes to a network of trust, or simply a web-of-trust. Instead of trusting a CA, in web-of-trust, users are responsible to verify the identity corresponding to a public key they want to communicate with, or trust other people that have already validated the identity of a public key. After that, the public key is imported to the keyring of the user for future use [32].

Lewison and Corella, sponsored by the US Department of Homeland Security, presented a semi-decentralized way of a blockchain PKI implementation [13]. Issuance of private keys and certificates was taken care of by a bank, which aimed to use this system to remotely prove the identity of its users. In this implementation, the Ethereum blockchain was used, with its ability to store and manage data, to store the key-value pair. To revoke this certificate, the bank places the same key-value pair in another blockchain store that it controls. That way, users can use this certificate to login to the banks' online services. The bank can then certify that the private key used is the one that corresponds to this user, by looking it up in the blockchain. It is a three-factor authentication, using biometrics beyond everything else.

In a revocation-oriented implementation of a blockchain PKI that was presented in 2017 [14], CAs continue to be the trusted part of the process, and they are the only entities authorized to issue certificates. Therefore, this solution is also not completely decentralized. It still is a traditional PKI model, but it mitigates one of the most important problems of a PKI, the reliability and security of the certificate revocation information, with the use of blockchain technologies.

SCPKI [15] is a smart contract implementation that is trying to fix one of the most important weaknesses of the traditional PKI: the centralized schema where every CA has the ability to issue rogue certificates for any entity. SCPKI tries to completely decentralize the process, using the Ethereum platform with a smart contract that allows users to publish attributes, signatures and revocations. It also uses the IPFS storage layer to allow users to store attribute data like certificates and PGP keys, with the peer-to-peer data distribution protocol, where nodes of the network form a distributed file system. A working prototype of SCPKI has been developed in solidity with a Python client script. For identification purposes, SCPKI uses web-of-trust principals, where users of the system can sign each other's public key.

A custom blockchain implementation was presented in 2017 [16] to bypass some of the limitations present in some of the traditional public blockchains. One of the problems it is trying to address is storage requirements. The solution that it presents is pruning, in which old transactions are discarded and expired accounts in the account tree are pruned. Also, in this implementation CAs still exist and they are responsible for signing CA proofs. A custom Proof of Stake protocol is used in order to add new blocks in the blockchain, which requires trust in an amount of stakeholders, unlike the trustless schema of Bitcoin's blockchain. In this implementation, CAs issues certificates which are signed both from them

and from the domain owners signing key. The revocation of the certificates is automatic and happens once the domain owner changes its signing key, so all its certificates signatures become invalid and the certificates get revoked.

DCSP [20] is a low-latency approach that provides up-to-date and accurate certificate revocation information. In order to store that information, it uses the Domain Name System (DNS) infrastructure. DCSP is able to achieve increased performance compared to traditional OCSP, while also preserving the user's browsing history privacy.

CCSP [21] is a proposal that improves on DSCP and emphasizes performance gains, through aggressive time- and space-based compression. The number of signatures operations required by CAs and OCSP servers is reduced by as much as six orders of magnitude. By mitigating this overhead, man-in-the-middle attacks can be more effectively prevented.

Certificate Transparency [22] is an effort to provide an open auditing and monitoring system that let any domain owner or CA determine whether their certificates have been mistakenly issued or maliciously used. Certificate Transparency dictates the creation of a system of public logs and advocates that all valid certificates should be publicly and widely known through these public logs. The goal of Certificate Transparency is to seek to eventually record all certificates issued by publicly trusted certificate authorities, allowing efficient identification of mistakenly or maliciously issued certificates.

DNS-based Authentication of Named Entities (DANE) [23] offers the option to use the DNSSEC infrastructure to store and sign keys and certificates that are used by TLS. This is achieved by introducing a new type of DNS record, where the whole certificate of the domain is stored. It then uses DNSSEC to validate the integrity of certificates. That way, it is able to completely replace CAs. Unfortunately, it is subject to MITM attacks [21] and also imposes a significant extra burden on the DNS infrastructure.

Instant-Karma PKI (IKP) [24] is a blockchain-based PKI that builds upon Google's Certificate Transparency project. IKP provides automatic response to CA misbehavior in order to solve one of the most serious problems of the traditional PKI. It also offers incentives for users that help the system detect misbehavior. IKP extends the traditional TLS architecture with the use of Ethereum smart contracts. The IKP contract takes the certificate as an input and checks it against a Domain Certificate Policy. This policy contains the list of CAs that are allowed to issue certificates. If the check results in a CA that is not authorized, a "Reaction Policy" takes place which transfers Ether from the CA to the user and the user that reported the violating CA. IKP is one more PKI implementation that exploits blockchain as a technology to fix a certain aspect of the traditional schema, but not to completely decentralize it.

Blockstack [28] is another PKI implementation that supports a DNS implementation in the blockchain, similar to Namecoin. Namecoin [33] was the first project to implement a decentralized DNS through the use of a blockchain. Blockstack initially ran on top of the Namecoin blockchain, but now uses Bitcoin [34]. An "OP_RETURN" transaction[1] is used in order to write the hash of a "zonefile" in the blockchain. This zonefile is a file that contains the public key of the user and a pointer to the profile of the user. This profile is located at a JSON document which contains identification information for the user and it can be stored in any publicly accessible storage. Blockstack also has its own distributed filesystem implementation, maintained by Blockstack nodes that store the user's zonefiles. Blockstack tries to build a completely decentralized PKI with the inclusion of an identity system in a blockchain-agnostic way, which means that it can be implemented in other blockchains too.

Certcoin was presented by MIT in 2014 [25]. Certcoin is a completely decentralized PKI that uses existing blockchain-based technologies in the same manner as Namecoin [35], to provide stronger identity retention, compared to traditional PKI implementations. In Certcoin, users own two key pairs. An "online" is used to authenticate messages to and from the server hosting the Web site, and an "offline" is used to sign or revoke new keys in security incidents. Signature information about these two keys are contained in the transaction that the user generates in order to register their domain. Certcoin uses its own blockchain implementation and network.

PB-PKI [26] builds upon Certcoin to provide a privacy aware solution. It is possible to achieve total anonymity with PB-PKI, but at the cost of security: Members of the network may be able to tamper with the public keys of other members. Security can be improved, by allowing a slightly lower level of privacy. In order to achieve the desired levels of privacy, the public identity of the user and their public key are not publicly linked and may be revealed upon concensus of a network majority. Effective revocation would require that link to be available.

Keychains [27] uses the PGP web-of-trust model to provide a truly decentralized public key infrastructure. Certificate revocation is possible, but a malicious peer has the ability to cache and use revoked keys. Web-of-trust systems have been successfully operating for decades, providing decentralized solutions and removing any central points of failure. However, it also has a few shortcomings. It is difficult for a new user to join an existing network, as they would have to have their public key signed by an existing member. The existing member would have to trust the new member and

[1] OP_RETURN is a script word (opcode) in the Bitcoin scripting language, which is the standard way to include extra data in a Bitcoin transaction. OP_RETURN marks the transaction as unspendable when used [19].

endorse them with their support, which is usually done only after they meet in person. Also, within a web-of-trust, there is no way to deal with key recovery. A "designated recoverer" may be specified by a key owner to have the permission to revoke the key owner's key, in case the key owner loses their own private key, also losing the ability to revoke their own public key.

An overview of the related work presented here is shown in Table 1. In most of the implementations of proposed solutions, centralized processes continue to exist, such as centralized identification, storage or even lack of CA removal. Others do not consider ways of effective revocation of certificates. In contrast with traditional PKI and the related work described here, the *DeTRACT* framework accomplishes the complete decentralization of the process, with no central points of failure, in both the transaction-based *DeTRACT-T* approach, and the smart contract-based *DeTRACT-SC* approach, as will be presented in detail in the next section.

3 The *DeTRACT* framework

In this section, the *DeTRACT* framework is outlined. Key technologies used by *DeTRACT* are presented, describing how they are used to overcome different problems. The two approaches of *DeTRACT*, the transaction-based *DeTRACT-T*, and the smart contract-based *DeTRACT-SC*, are described in detail. Finally, a description of how the framework would be utilized by client-side processes is outlined.

The proposed decentralized PKI framework allows domain owners to retain a digital identity while they are enabled to create, update and revoke certificates, by exploiting the decentralization aspect of blockchain technologies. The generated certificates follow the X.509 standard,[2] due to its popularity by original PKI implementations. As a result, easier future integration by client applications (i.e., browsers) and easier development of a practical implementations are possible. The *DeTRACT* framework also exploits the data storage capability of blockchain technologies that is described in Sect. 1.

DeTRACT was initially developed using the Bitcoin blockchain, but the flexibility of the design allowed for an implementation using the Ethereum blockchain as well. That way, some of Ethereum's technical advantages over Bitcoin are exploited. The most important of these are faster transactions and easier storage capabilities. Moreover, a smart contract approach is presented in Sect. 3.3. That way, several approaches of *DeTRACT* can be implemented, which can serve different needs of domain owners, as will be examined in Sect. 4. As outlined in Fig. 1, a number of steps are fol-

[2] X.509 is a format used in public key certificates, used in many protocols, like TLS/SSL. https://tools.ietf.org/html/rfc5280.

Table 1 Related work comparison table

	CA removal	Decentralized identification	Decentralized storage	Effective revocation	Practical implementation
Lewison and Corella [13]	No	No	No	Yes	Yes
Baldi [14]	No	No	No	No	No
SCPKI [15]	No	No	Yes	Yes	Yes
Fredriksson [16]	No	No	Yes	Yes	No
DCSP [20]	No	No	No	Yes	No
CCSP [21]	No	No	No	Yes	Yes
Certificate Transparency [22]	No	No	No	Yes	Yes
DANE [23]	Yes	No	No	Yes	Yes
IKP [24]	No	No	Yes	Yes	Yes
Certcoin [25]	Yes	Yes	Yes	No	No
PB-PKI [26]	Yes	Yes	Yes	No	No
Keychains [27]	Yes	Yes	Yes	No	No
Blockstack [28]	Yes	Yes	Yes	Yes	Yes
DeTRACT-T	Yes	Yes	Yes	Yes	Yes
DeTRACT-SC	Yes	Yes	Yes	Yes	Yes

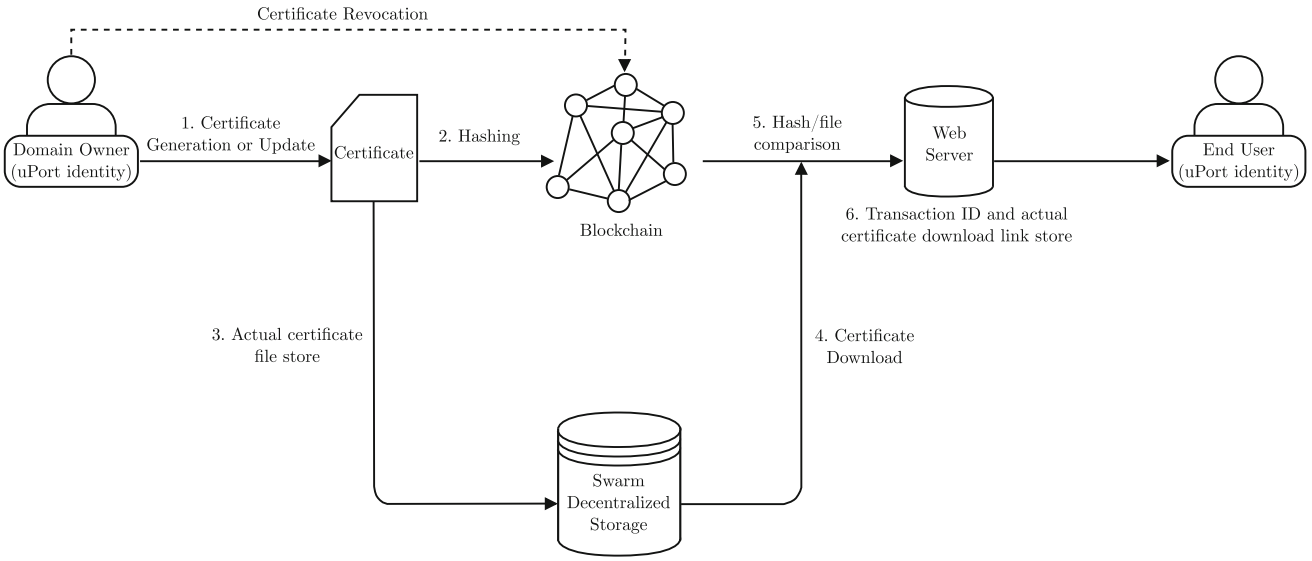

Fig. 1 *DeTRACT* framework

lowed in our approach which are analyzed in detail in the following subsections.

3.1 Building blocks

The following technologies are key parts of the *DeTRACT* framework, as it was designed and implemented. A brief introduction to these technologies follows:

Bitcoin is a decentralized digital currency, introduced in 2008 [9]. Bitcoin is the first application of blockchain technology and first successful cryptocurrency. It uses peer-to-peer technology to operate with no central authority: Transaction management and currency issuance are carried out collectively by the network. It enables instant payments to anyone, anywhere in the world. Bitcoin includes its own scripting language that helps to automate more complex transactions and implement simple smart contracts. This scripting language, among others, includes "OP_RETURN," a command that enables creating special transactions, whose only purpose is to store data on the blockchain.

Ethereum is an alternative public and open source cryptocurrency that comes with many changes and additions to the traditional Bitcoin blockchain. Ethereum was founded in 2013 in order to improve the scripting, and other, restrictions of Bitcoin and other cryptocurrencies. Its main aspect is the addition of a decentralized Turing-complete virtual machine, the Ethereum Virtual Machine (EVM), where anyone can develop and execute code, called "smart contracts," using the Solidity programming language. Smart contracts in Ethereum can carry arbitrary state and can perform any arbitrary computations. Code included in smart contracts is executed by each node as part of the block creation process [12].

uPort is an Ethereum-based, decentralized identification platform that links physical identities with Ethereum addresses. uPort's open identity system allows users to register their own identity on the Ethereum blockchain send and request credentials, sign transactions, and securely manage keys and data. uPort is described as a "self-sovereign identity and user-centric data platform." It is self-sovereign, since it provides users the ability to make statements about themselves, without relying on centralized authorities or platforms [36]. Additionally, it is user-centric as it allows users (or even applications) to exchange data privately, using the Ethereum blockchain [17].

Swarm is a decentralized storage platform and a content distribution service available to the Ethereum blockchain. It lets users store and distribute DApp (Distributed Application) code or data using its peer-to-peer data sharing network, where files are addressed by the hash value of their content. Swarm users are able to upload data similarly to how it is traditionally done in the World Wide Web. The difference is that with Swarm, the uploads are not stored in a specific server. Instead, the data are distributed and uploaded to network nodes in a redundant manner; all data stored in Swarm are guaranteed to be replicated across multiple nodes at any given time. This characterizes Swarm as secure, censorship resistant, fault tolerant and with zero downtime [37].

DeTRACT could potentially be implemented using almost any other alternative blockchain network, but the Bitcoin and Ethereum networks are by far the most secure blockchain

networks today [38]. Since security is most probably the most sought-after characteristic of a decentralized PKI solution, the use of these two major blockchain networks was chosen over any other.

Using the combination of these technologies allows *DeTRACT* to overcome problems faced in other solutions, including:

- Replacement of Certificate Authorities, since public blockchains with no intermediaries to manage certificates are used.
- Effective revocation models, without decentralization violation, as many alternative "decentralized" PKI implementations still make use of CAs in order to manage identities and perform certificate revocation actions.
- Use of uPort for decentralized identification, instead of CAs that manages the identity verification. That way, anyone can verify the identity of any entity in the network.
- Use of Swarm for decentralized storage, in order to avoid centralized server vulnerabilities that can result in tampering of certificates and information altering, which could affect the safety and privacy of the end users.
- Proof of concept implementations, in order to prove that the *DeTRACT* framework works in practice.

As a result, both approaches of the *DeTRACT* framework address the single point of failure problem, while at the same time offering effective revocation for issued certificates. As a side effect, the use of public blockchains makes both approaches of the *DeTRACT* framework to also constitute cheaper solutions, when compared to traditional methods, as will be detailed in Sect. 4.1.

3.2 The transaction based *DeTRACT-T* approach overview

In this subsection, the *DeTRACT-T* approach is presented. This is based on transactions on either one of the Bitcoin or the Ethereum blockchains. The way that domain owners can create a digital identity and map it with their physical one (Sect. 3.2.1) being initially outlined. The details for the process of certificate creation and update (Sect. 3.2.2 which covers the steps 1–5 of Fig. 1) are followed. Finally, the certificate revocation process of *DeTRACT-T* is described, to cover the dashed lines of Fig. 1.

3.2.1 Identity creation

Domain owners using the *DeTRACT* framework have to create a digital identity in order to participate in the network. This digital identity is actually mapped to the domain owner's physical identity. Identities in the blockchain are pseudo-anonymous, i.e., a digital identity exists but there is no actual link to the physical one. For that reason, domain owners have to use uPort which is a decentralized identification method that maps a physical identity with an Ethereum address through the use of a smart contract. This approach enables domain owners to include one or more Bitcoin addresses in their digital uPort identity. These addresses will be mapped with their physical identity and will be used in the *DeTRACT* framework to create, update and revoke certificates.

Domain owners, under the transaction based approach *DeTRACT-T*, have to create several addresses in order to follow the proposed process of the *DeTRACT* framework. The following three types of addresses will be mapped to the domain owner's uPort identity (Fig. 2):

- "Generation" address. This is required in order to support domain owners managing the currency units needed for the certificate creation and update.
- "Certificate" address. This is used to keep the currency units transferred-in from the "Generation" address. By having available currency units in this address, it is stated by the domain owner that their certificate, which is mapped to this address, is still active. It also manages the revocation process, if needed. That way, in cases that the certificate has to be revoked, the currency units are spent. This, in turn, states that the certificate address has no active certificate.
- "Revocation" address. This manages the revocation process in extreme security conditions, such as when the domain owner has lost ownership of both of the previously mentioned addresses. The "Revocation" address remains secret from end users until the need for an extreme revocation scenario, as described in Sect. 3.2.3.

3.2.2 Certificate creation and update

Several steps have to be followed to create a certificate and store it in the blockchain. The certificate generation and update process is initiated by an actual certificate creation to include information related to the domain owner. Then, this certificate gets hashed and is stored in the blockchain, by using a decentralized storage technology called Swarm. That way, the end users can access this certificate in a decentralized manner. They can then compare it with the hash that is stored in a transaction on the actual blockchain, in order to be certain about the integrity of the certificate. As a result, end users will be able to securely connect to the Web application that corresponds to this certificate.

In summary, domain owners need to follow the next steps in order to create a new certificate, or update an existing one:

1. A client-side script will prompt the domain owner to fill the information needed to generate the X.509 certificate.

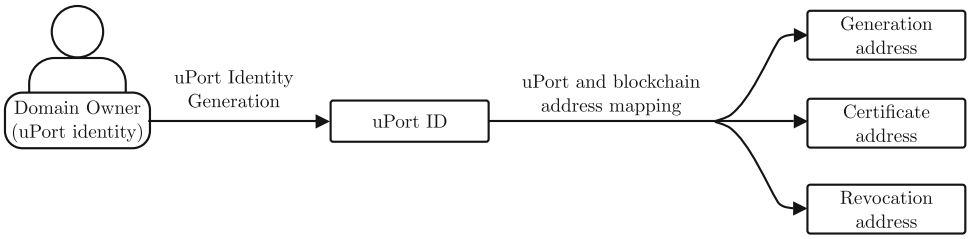

Fig. 2 Identity creation and mapping

This will result in a certificate file (of type .crt) that will include the following:

- The domain owner's information.[3]
- The domain name.
- The address of the uPort identity of the domain owner.
- Signatures of all the three addresses used for the identity. These signatures result from the private keys of the addresses used to sign the uPort address of the user. That way, end users will have a reason to believe that the certificate owner is also the owner of these addresses.
- The expiration date of the certificate.

2. After the certificate file is generated, it will be signed by the private RSA key[4] that will correspond to this certificate from now on. The RSA key will then be able to be generated directly from the client-side script.
3. The certificate file will then get hashed and stored on the blockchain with a single data transaction (Fig. 3). In order to differentiate between certificate generation and update, a prefix is used before the certificate's hash. The two available prefixes are:
   - "CC:" which refers to certificate creation.
   - "UC:" which refers to certificate update.
4. Currency units are included in the data transaction which includes the certificate hash, and as long as the "Certificate" address has available currency units, the certificate, the hash of which can be found in the data field of the transaction is considered active. This process is different for the two alternative blockchains used:
   - In the Bitcoin blockchain, the domain owner has to make two transactions because of a limitation with the OP_RETURN script code. In the first transaction, currency units get sent from the "Generation" to the "Certificate" address. In the second transaction, the hash that was generated earlier gets stored with an OP_RETURN transaction from the "Certificate" address.
   - In the Ethereum blockchain, the domain owner only has to make a single transaction from the "Generation" to the "Certificate" address. This will include both the currency units and the certificate hash.

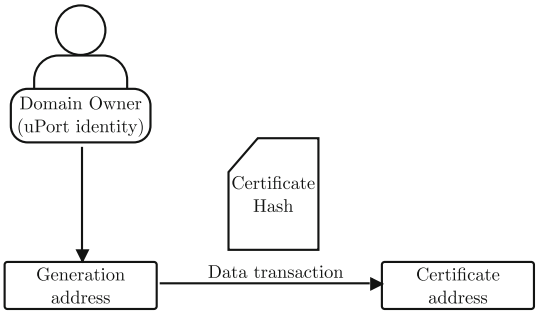

Fig. 3 Certificate creation and update process

5. The actual certificate file gets uploaded to Swarm decentralized storage.
6. A TXT file will be generated which will include the certificate hash transaction ID and the Swarm link for the actual certificate download. This file will be signed by the uPort ID address and will be stored in the "well-known" directory of the Web server that corresponds to the domain of the certificate, as typically used for DV certificate validation [39].

The certificate update process is almost identical to the certificate generation process. There are only a few differences. During the generation of the X.509 certificate (step 1 as described above), the domain owner may either reuse existing private keys to generate the respective signatures, or create new signatures by adding newly generated private keys. The latter would be the more secure option in any case. If they choose to generate new private keys, these should also be added to their uPort identity. Additionally, as indicated in step 3, before hashing the certificate, the proper prefix for certificate update should be used.

[3] Domain owners information can include the information specified by the available fields in the X.509 standard, like name, address and others.

[4] RSA is a public-key cryptosystem, which is widely used for secure data transmission. Most traditional PKI approaches use RSA to secure the connection between end users and the web application server.

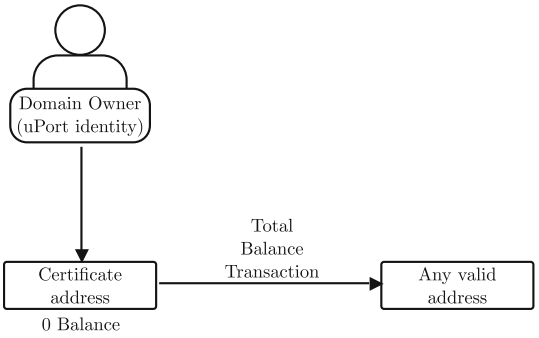

Fig. 4 Simple revocation process

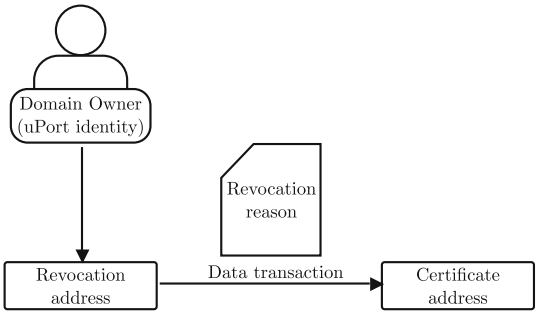

Fig. 5 Extreme revocation process

3.2.3 Certificate revocation

The ability to revoke a certificate is one of the most important aspects of the *DeTRACT* framework, as a tampered certificate can result in serious security implications, as shown in Sect. 1. There are two revocation methods in the *DeTRACT* framework: simple revocation and extreme revocation. The simple revocation method is used for reasons that are commonly stated in CRL profiles [40], such as loss of ownership of the Generation address, the loss or compromise of the RSA key and compromise of the CA. This method requires the "Certificate" address to have zero balance. To perform a simple revocation, domain owners have to transfer all the available balance from the "Certificate" address, so that end users (or simply the client application) know that the certificate is revoked because the "Certificate" address has zero balance (Fig. 4). If the balance of the certificate address is not zero, it means that no revocation has taken place. Therefore, the certificate is considered active.

Finally, there is the extreme revocation method that is used in extreme cases of compromised credentials. These cases include not having access to either the "Certificate" address or to both the "Certificate" and "Generation" addresses. In such cases, domain owners will not be able to perform a simple revocation and reduce the balance of the "Certificate" address to zero, so the "Revocation" address is used to revoke the certificate. It does so by issuing a transaction from the "Revocation" address to the "Certificate" address, stating the revocation reason in the data field[5] of the transaction (Fig. 5). Once the transaction goes through and becomes part of the blockchain, end users (or simply the user client application) know that the "Revocation" address is active. They can then validate the revocation by comparing the signature included in the original certificate file, with the public key that generated the revocation. Also, by stating the revocation reason, the client application will be able to know the reason and be able to present it to the end user.

[5] The process is different in the Bitcoin and Ethereum blockchains, because of the OP_RETURN limitation as shown in Sect. 3.2.3.

A coding scheme, similar to the one in certificate generation and update, is followed here too, in order to differentiate the different revocation reasons. The revocation codes with the corresponding reasons are listed below:

- *R1* The domain owner still has access to all of the addresses but needs to revoke the certificate for a specific reason. The reason is stated by the domain owner and passed with the transaction to the blockchain.
- *R2* The domain owner loses access to the generation address, but retains access to the other addresses. In this case, the message passed with the transaction to the blockchain is "R2: No access to Generation address."
- *ER1* The domain owner retains access only to revocation address, and the message passed with the transaction to the blockchain is "ER1: No Access to Generation and Certificate address."
- *ER2* The domain owner loses access only to the certificate address, but retains access to the other addresses. The message passed with the transaction to the blockchain in this case is "ER2: No Access to Certificate address".

3.3 *DeTRACT-SC*: smart contract-based approach

A smart contract-based approach of the *DeTRACT* framework has been developed in order to reduce the complexity of the transaction-based *DeTRACT* approach, by exploiting capabilities of smart contracts. In this approach, there is no need for a number of different addresses. All actions are carried out using a smart contract that is hosted in the Ethereum blockchain (Fig. 6). The identification part is once again being taken care of by the uPort platform. The *DeTRACT-SC* approach consists of two primary components:

- The smart contract, carrying out the certificate management in the blockchain.
- A Web UI that helps the domain owner as well as the end users to communicate with the smart contract.

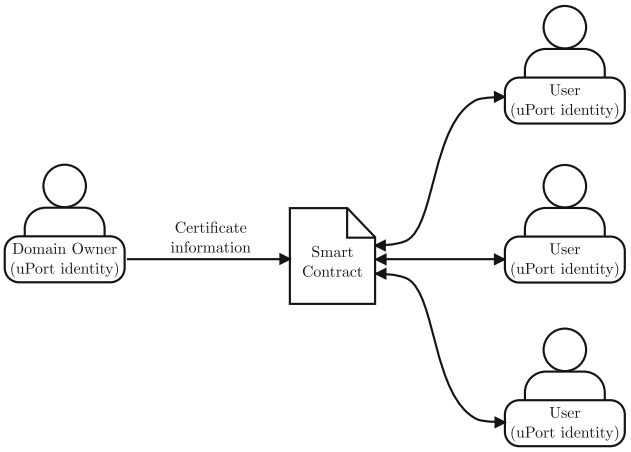

Fig. 6 Process of certificate generation in the smart contract approach

In the smart contract approach of the *DeTRACT* framework, the domain owner only needs a single Ethereum address, which is mapped to their identity, using the uPort platform. The initial process that the domain owner has to follow in this approach is the following:

1. Generate the identity with uPort, using the actual Ethereum wallet address of the domain owner.
2. Generate the certificate file, using the same process as with the *DeTRACT-T* approach.
3. Upload the actual certificate file to Swarm.
4. Store the following information in the blockchain using the web UI that communicates with the smart contract:
   - The domain name that corresponds to the certificate of the domain owner.
   - The domain owner's uPort address.
   - The hash of the newly generated certificate.
   - The date that the certificate will expire.
   - The download link for the newly generated certificate.

Domain owners have the ability to update their certificate following a similar process as with generating one. The main purpose of a certificate update is to update the expiration date of the certificate. For that reason, domain owners may only select to update the expiration date field and leave all the other information unchanged, in order to reduce transaction costs related to data transactions.

Following that, the certificates that a domain owner is uploading through the smart contract are linked to their Ethereum address. That way, only the domain owner has the authority to update or even revoke the certificate. In order to revoke a certificate, the owner has to prove that their private key is the one that matches the signature that signed the certificate generation transaction, from which arises that this user is the owner of this certificate. Thus, users will be notified with the revocation reason and the certificate will not be trusted anymore.

In this approach, there is no extreme revocation. In cases where the owner has lost access to their Ethereum address, the revocation method of the uPort identification platform to revoke this identity is used. That way, if end users check a certificate with a revoked uPort identity, the certificate is immediately marked as invalid by the client application.

3.4 Client-side process analysis

Until now, only the process on the domain owners side has been described, as it was implemented for both *DeTRACT* approaches. The client-side application would take the form of a Web browser plug-in. The process that could be followed by the client, would be the following:

- Users will download and install a browser add-on, similar to the Namecoin add-on, which will connect to the blockchain used by the implementation, using lightweight technologies,[6] such as SPV (Simplified Payment Verification) [41] or Electrum [42,43], in order to avoid big storage requirements.
- Users will navigate to a domain that uses the *DeTRACT* framework, and the add-on will trigger, by the existence of the TXT file that contains the certificate hash transaction and the download link of the actual certificate, as seen in step 1 of Fig. 7.
- The actual certificate will be downloaded, hashed and the generated hash will be compared with the hash stored in the blockchain, as shown in steps 2 and 3 of Fig. 7.
- Once the certificate is identified as trusted, it will be added in the browser with the RSA keys, and the communication between the client and the server will be secure, as shown in step 4 of Fig. 7.

Also, every time that the site is browsed by the user, the browser add-on will check for possible updates or revocations by the following means:

- To check for updates, the add-on checks the "Certificate" address that corresponds to the domain, and if there are any new transactions with the "UC:" code in the hash, the new certificate is downloaded and checked against the same process as the initial download described above.
- To check for possible revocations, every time the domain is browsed, the following cases are checked:
  - If the balance of the "Certificate" address is equal to zero, the certificate corresponding to this domain is

[6] technologies that allow transactions without the need of the full blockchain existing in the client application [41].

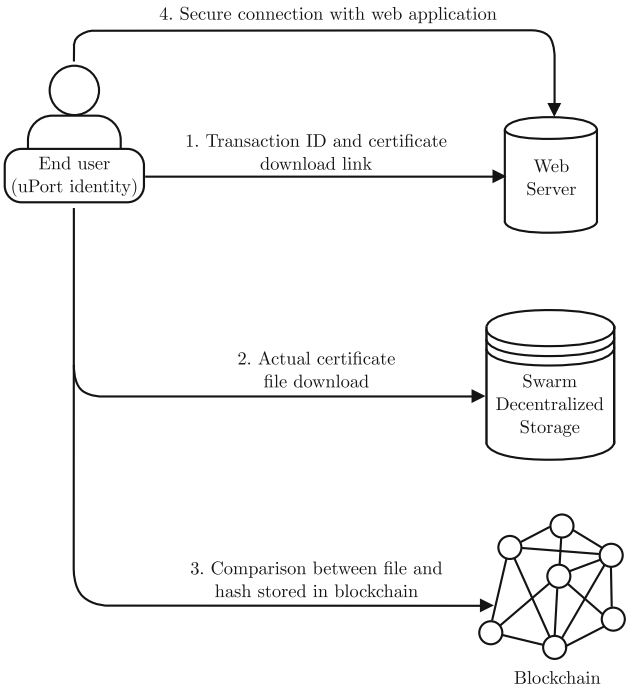

Fig. 7 Client-side process

immediately marked as invalid and the connection as insecure.

– The add-on also checks for any transactions that have been recorded in the blockchain, in which the "Certificate" address is specified as an output. It then takes the public key that corresponds to the address of any of the transaction inputs and compares it with the signature of the "Revocation" address, which is stored in the certificate file. In case the address that generated this transaction is indeed the "Revocation" address, the certificate is immediately marked as invalid and the connection is henceforth deemed insecure.

It has to be noted that lightweight blockchain clients require only a small initial download (currently around 40MB in the case of Bitcoin) that includes only block headers, instead of the entire block contents. Each block header weighs only 80 bytes, so it is possible even for constrained devices, or in this case a browser add-on, to stay current with blockchain updates [9]. Even when using a lightweight client, such as one that implements the SPV protocol, it is practically impossible for any malicious nodes to convince the client to accept an ill-formed block or transaction [44]. All lightweight clients verify the proof-of-work for all blocks, using only the information that is stored on the block headers. Additionally, the inclusion of transactions of interest in the respective blocks is verified by calculating the hash of the merkle path of these transactions and cross-referencing it with the one received by the connected full node [9]. Malicious nodes can only hide valid transactions from an inquiring lightweight node. However, this attack is commonly mitigated by connecting to several random full nodes in the network and determining whether they all provide the same information.

4 Experimentation—validation

This section includes the evaluation of the *DeTRACT* framework. A cost analysis for issuing and revoking certificates is performed for the different approaches of *DeTRACT* as well as for issuing Domain Validated (DV) certificates through traditional means. The different approaches of *DeTRACT* are also compared between them with respect to several characteristics, such as complexity, volume and speed. The proof-of-concept implementations that have been developed are presented, and finally, the limitations of the *DeTRACT* framework are presented.

4.1 Cost analysis

Three implementations were developed based on the *DeTRACT* framework. Cost analysis was held in July 2019 (1 BTC = \$12,400 and 1 ETH = \$313 at the time), when the fastest and cheapest transaction fee in the Bitcoin blockchain was 60 satoshis/byte (1 satoshi = 10^{-8} BTC) and 68 gas/byte (20 Gwei/gas, 1 Gwei = 10^{9} ETH) in the Ethereum blockchain. Due to the volatility of the cryptocurrency market prices, these will probably differ for different timeframes.

Storage costs have to be taken into account as well. However, since Swarm is still in early access stage and it has not been implemented to the main network yet, there is no way to know how much it will cost to store certificates on the blockchain. The storage cost, judging from similar implementations like IPFS, is, however, expected to be insignificant. Therefore, storage costs are not included in the calculations.

Starting with the Bitcoin blockchain *DeTRACT-T* approach, there is a certificate hash that has to be included in the OP_RETURN transaction, a simple transaction fee for the currency transaction from the "Generation" to the "Certificate" address as well as the storage cost for the actual certificate. An average bitcoin transaction consists of approximately 250 bytes and a hash with the statements for generation or update consists of 67 bytes. This means that for the two transactions, approximately 634 bytes, which translate to 76,080 satoshis or approximately \$4.72, are needed. A revocation or extreme revocation transaction is by all means a standard bitcoin transaction, which has an average size of 250 bytes, resulting in a revocation cost of 15,000 satoshis, or approximately \$1.86.

On the other hand, in the Ethereum *DeTRACT-T* approach there is only a single transaction, which costs 21,000 gas. It also contains the certificate hash of 67 bytes. For nonzero bytes of data ("Gtxdatanonzero" transaction [45]) in the blockchain, a fee of 68 gas/byte is necessary, so an additional amount of 4556 gas has to be paid, in order to add the certificate hash to the transaction. With a total of 25,556 gas to pay for the transaction, and for the fastest transaction speed price of 20 Gwei/Gas, the final transaction fee is 0.0005111 ETH, or approximately $0.16. This is significantly lower than the Bitcoin *DeTRACT-T* approach. A revocation or extreme revocation transaction is also a standard Ethereum transaction in this case too. With a cost of 21,000 gas, the revocation cost amounts to 420,000 Gwei, or approximately $0.13.

Finally, for the smart contract *DeTRACT-SC* approach, non-fixed size variables are used, like the domain name and the download link that will be added to the blockchain. Taking the fixed values and adding an average value for the non-fixed ones, there is a total of 200 bytes of data to be transferred to the blockchain. That way, a total of 34,600 gas is needed. This translates to 0.000692 ETH, or approximately $0.22, corresponding to the fastest transaction speed price of 20 Gwei/Gas. Even if the non-fixed variables increase considerably in size, the resulting price will not have significant variations. The smart contract code for certificate revocation is estimated to require 20,604 gas. This is equivalent to 0.000412 ETH, or approximately $0.13.

Our decentralized PKI approaches target domain validation (DV) certificates of the traditional PKI implementations that with the average market price cost around $10 per year. In Table 2, a complete price comparison is displayed, with our Ethereum blockchain *DeTRACT-T* framework being significantly cheaper than the centralized alternative. Comparison with implementations from the literature review was not carried out, because these implementations either used new proof-of-concept blockchains, so actual prices cannot be calculated, or the prices are similar to ours due to use of the Ethereum or Bitcoin blockchains.

In this price analysis and comparison, some really important aspects have to be kept in mind:

- Cryptocurrency Price Volatility: Due to many factors, the exchange price of Bitcoin and Ethereum is very volatile and unstable. The information above concerns only a specific time. For future reference, the necessary price changes in each cryptocurrency have to be implemented, in order for the true cost of each implementation to be revealed.
- Certificate Prices: Prices for actual SSL/TLS certificates are not standard and heavily depend on the provider. Providers are usually companies, and prices depend on the size of the company and on the number of customers. On our analysis, the numbers shown is the average of three of the biggest certificate providers in the market, Comodo, Thawte and GeoTrust.
- Contract Price: In the smart contract-based *DeTRACT-SC* approach, the price for the contract publishing was not included in the analysis. That is because this is an one-time cost, and it is not defined by whom it will be paid, as it does not concern us in the context of this work.

4.2 Approach comparison

Because of the development in different blockchains and technologies, it is necessary to perform a comparison between them in order for users to be able to judge the implementations that suits them best, or even to help future research and development. To compare these implementations, the following categories were used:

- *Cost* Difference in cost, when using different blockchains and technologies
- *Transaction speed* The speed which is needed for a certificate to be generated, while referring to the transaction speed of each blockchain used.
- *Data volume* The amount of data that needs to be stored in the blockchain.
- *Usage complexity* The complexity by the certificate creators and users side.
- *Development complexity* Development complexity for each approach.
- *Information availability* The amount of information that is stored in the blockchain and is available for use instantly, without having to query another information source, such as a distributed data storage. In the transaction-based *DeTRACT-T*, it was possible to store only the hash of the certificate and be able to access it instantly, in contrast to *DeTRACT-SC*, where the amount of data stored, which is available immediately is significantly larger.
- *Security* The security of the data stored on the blockchain, with respect to the cost of a sustained attack against it.

For the comparison, a three-point scale consisting of the values "Low," "Medium" and "High" was used. The *DeTRACT-T* solution based on Ethereum seems preferable to the Bitcoin-based one in most aspects. However, although both networks are generally considered very safe, the Bitcoin network is probably the safest of the two. At the time of writing (July 2019), the theoretical cost of a 51% attack on the Bitcoin network, sustained for 1 h, would be approximately $1 million, while that for the Ethereum network, only about $160,000 [38]. A 51% attack is a potential attack on a blockchain network, where a single entity or organization controls the majority of the hashing power, allowing them to

Table 2 Certificate generation price comparison between *DeTRACT-T*, *DeTRACT-SC* and the conventional certificate type, for single domain certificates (July 2019)

	DeTRACT-T (Bitcoin)	*DeTRACT-T* (Ethereum)	*DeTRACT-SC*	Domain validation (DV) certificates
Certificate generation	76,080 satoshi	0.000511 ETH	0.000692 ETH	Approx. $10/year
	($4.72)	($0.16)	($0.22)	
Certificate revocation	15,000	0.000420 ETH	0.000412 ETH	$0
	($1.86)	($0.13)	($0.13)	
Extreme revocation	15,000	0.000420 ETH	N/A	N/A
	($1.86)	($0.13)		

N/A: Not Applicable

Table 3 Comparison between the three different *DeTRACT* blockchain implementations

	DeTRACT-T (Bitcoin)	*DeTRACT-T* (Ethereum)	*DeTRACT-SC*
Cost	Medium	Low	Low
Transaction speed	Medium	High	High
Data volume	Low	Low	High
Usage complexity	High	High	Low
Development complexity	Medium	Medium	Low
Information availability	Low	Low	Medium
Security	High	Medium	Medium

disrupt the network, including reversing of transactions that have previously been included in the blockchain [46].

The comparison results, as shown in Table 3, show that it is not possible to conclude toward a single solution, and every blockchain and technology used may offer different characteristics to the end user. The three different implementations of *DeTRACT* can result in the following advantages for the end user:

- *DeTRACT-T* (Bitcoin): In this implementation, it can be seen that, apart from security, there are no categories where it stands out from the other two. Nevertheless, it was chosen because the Bitcoin blockchain is stable, secure and with one of the biggest communities to support it.
- *DeTRACT-T* (Ethereum): The Ethereum blockchain has also been chosen because of its stability and support, but it has other aspects to offer too. The transaction speed is fast, in the order of a few seconds, the cost is the lowest of the three implementations and the development complexity is relatively low. This gives an advantage over the three implementations in cost and development complexity.
- *DeTRACT-SC*: The smart contract approach is the most complex implementation of the three in terms of development. However, it has relatively low cost, really fast transaction speed (resulting in really fast certificate generation), it can store the biggest data volume of the three, which increases the security and immutability aspects, and it is the less complex of the three in terms of usage.

4.3 Implementations

Proof-of-concept implementations of the *DeTRACT* framework have been developed, using different blockchains and technologies. The working prototypes are available as open source projects, under an MIT license. The implementations consist of the following:

4.3.1 DeTRACT-T_Bitcoin

This is a client script implementation for transaction-based *DeTRACT-T* using the Bitcoin blockchain.[7] The client script is implemented using the Python programming language.

The script provides the user with an interactive session that allows them to perform all actions that pertain to:

1. Identity creation, including the creation of the "Generation," "Certificate" and "Revocation" addresses, as previously described in Sect. 3.2.1.
2. Certificate creation and signing, as described in Sect. 3.2.2.
3. Certificate update, where the user is able to update any information included in the certificate. This includes the domain owner's information, the domain name, the

[7] https://github.com/Cr0wTom/DeTRACT-T_Bitcoin.

address of the uPort identity, the signatures of all three addresses as well as the certificate's expiration date.
4. Certificate revocation, including the ability to execute a simple revocation or an extreme revocation process, by sending the appropriate transactions to the Bitcoin blockchain network, as described in Sect. 3.2.3.
5. Uploading of the generated or updated certificates to the Bitcoin blockchain network by creating and sending the respective transactions.

4.3.2 DeTRACT-T_Ethereum

This is a client script implementation for transaction-based *DeTRACT-T* using the Ethereum blockchain.[8] It is also written in the Python programming language. This also provides exactly the same functionality as its DeTRACT-T_Bitcoin counterpart does, with the only difference being that it uses the Ethereum blockchain network instead of Bitcoin.

4.3.3 DeTRACT-SC

The smart contract *DeTRACT-SC* approach[9] consists of a proof-of-concept client application script, a smart contract and a Web-based user interface.

The client script is implemented using the Python programming language and allows users to create their identity and generate the respective certificate files, in a similar manner as with the *DeTRACT-T* approach. Additionally, it allows users to upload the generated certificate files to Swarm, where they may be accessible by anyone.

The smart contract is written in the Solidity programming language [47], currently the most common language for developing smart contracts for the Ethereum blockchain. It functions as the backend software of our smart contract-based solution. Through the provided application binary interface (ABI), it provides access to its smart contract methods that allow users to interact with the Ethereum blockchain network. This includes the ability to upload certificates to it, as well as retrieve information about certificates stored in the blockchain.

The Web-based user interface (Fig. 8) provides a simple proof-of-concept implementation of how an end user would be able to interact with the smart contract. It is implemented using plain HTML5 and the Javascript programming language, without the use of any Web development framework. The Javascript code uses the web3.js collection of libraries [48] to interact with nodes that are part of the Ethereum network and the uport-connect library [49] to interface with uPort identities.

[8] https://github.com/Cr0wTom/DeTRACT-T_Ethereum.

[9] https://github.com/Cr0wTom/DeTRACT-SC.

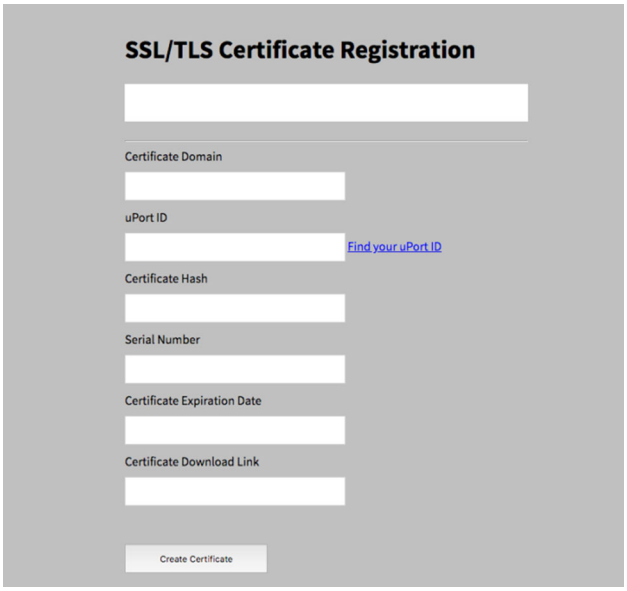

Fig. 8 Screenshot of the proof-of-concept Web user interface

These components are in fact parts of a decentralized application (DApp). DApps do not have to be hosted on a specific Web site, like most Web applications do. The backend part of the applications is running exclusively on decentralized peer-to-peer networks, such as the Ethereum network, as smart contracts. The frontend part of the DApp can be hosted anywhere; there can be multiple instances of the frontend code running on different servers at the same time, as individual applications on mobile phones or PCs, or even be hosted in a decentralized manner on decentralized storage such as Swarm and IPFS.

4.4 Limitations and security considerations

As previously mentioned, the *DeTRACT* framework only targets Domain Validated (DV) certificates. For these kinds of certificates, no identity information with respect to the organization or company issuing the certificate is vetted by anyone. This is the simplest type of certificate; however, it is the most common type found online. For other types of certificates, such as Extender Validation (EV) certificates and Organization Validated (OV) certificates, which require at least some vetting to establish that the identity of the organization or company is indeed the one specified by the certificate, an external validator, such as a certificate authority, needs to be included in the process. For that reason, a completely decentralized solution for EV and OV certificates is not possible.

It also has to be noted that this proposal does not emphasize on performance issues at all. In fact, existing centralized solutions are more performant than *DeTRACT*. For example, CRLite [50] and OCSP Stapling [51] only require a single

connection to fetch a website and determine authenticity of its certificate. *DeTRACT* would require three connections to accomplish the same task. Intuitively, that would mean that DeTRACT would be at least three times slower than existing technologies that are performance oriented. However, we believe that this is an acceptable trade-off between performance and decentralization.

The size of the Bitcoin and Ethereum networks might be an important factor with respect to *DeTRACT*'s adoption. The number of existing nodes in Bitcoin or Ethereum networks might not be sufficient to cope with the increased traffic that *DeTRACT* will present. Both networks are comprised of a few thousand active nodes at this time. However, scaling any of these blockchain networks is not a problem. Any interested parties, such as ISPs, or even individuals, can always choose to run their own node and share the load that is generated by the increased number of users.

If a browser plug-in for *DeTRACT* connects to a random node in the Bitcoin or Ethereum networks, in order to establish that certificates are valid, it may reveal browsing history to those nodes. Solutions to mitigate this problem, such as the Simplified Payment Verification (SPV) protocol exist, but may still leak some information [41]. However, for users with privacy concerns, it is possible to run their own nodes, so that no information leaking to external actors takes place.

Furthermore, there are some security considerations deriving from the use of lightweight technologies such as SPV by client applications. A malicious node in the network cannot possibly lie about the contents of a transaction, but it is possible to completely hide a transaction from inquiring lightweight nodes [52]. Therefore, a malicious node could potentially hide a revocation transaction from the client. This would lead the client to believe that the certificate is still valid. A way to possibly mitigate this issue would be to contact several random network nodes at the same time and determine if they all provide the same information. This is what most Bitcoin and Ethereum lightweight clients such as wallet applications for mobile phones already do. However, this would pose an additional performance penalty and the system would still be susceptible to network partitioning attacks. This issue can be completely eliminated if one runs their own network node, or otherwise communicates with a node that they know and trust, and their client applications communicate solely with that node.

Another limitation which derives from the use of blockchain technologies is the inherent limit to the rate of transactions that can be issued. The Bitcoin network currently has a limit of about seven transactions per second imposed by its maximum block size [53]. The Ethereum network currently has a limit of about 15 transactions per second, imposed by a limit to the maximum number of computations that can be performed with each block [54]. Several proposals for effectively scaling the capacity of blockchain networks to perform transactions exist, e.g., [53,55–59] and it is expected that their adoption will provide a solution in the immediate future. However, the issue still exists.

uPort is an important part of the DeTRACT framework, providing user identification functionality. uPort identities are simple smart contracts that are controlled by a controller contract and contain key recovery and access control logic. These identities are tightly connected to the user's smartphone, where their personal keys are stored securely. Therefore, the loss of one's smartphone might impose security risks. In that case, the attacker would also need to have acquired the user's credentials along with their smartphone. While this is unlikely to happen, it is still possible. A uPort identity is in essence just an Ethereum smart contract, which is represented in the Ethereum network by an address on the blockchain, just as any other Ethereum address. As such, a popular and effective way to mitigate these types of attacks would be for the users to employ the use of multisignature addresses [60]. These can be trivially implemented using Ethereum smart contracts. In a multisignature scheme, M-of-N ($M \leq N$) signatures are needed in order to authorize a transaction on the blockchain. Typical examples include 2-of-3, 3-of-3, 4-of-5 schemes, etc. The higher the number of keys M that is required to authorize a transaction, the more difficult it is for a potential attacker to compromise an identity. So, in this case, the attacker would need to have acquired at least M smartphone devices, possibly from different people and locations, simultaneously, along with their respective credentials, making it substantially more difficult to compromise an identity.

One of the most important points of *DeTRACT* is the effective revocation of certificates. In the Ethereum network, new blocks are added to the blockchain every few 10–15 s. So, a revocation transaction would not normally take more than half a minute to propagate through the entire network. In the Bitcoin network, however, new blocks are generated on average every 10 min. Block propagation time is negligible compared to that, as it only takes a few seconds for a new block to reach almost every node in the network [61]. So, it could take up to 10–20 min for a revocation transaction to appear in all network nodes. This is still better than the hourly or daily revocation times for certificate revocation lists (CRL), but not as fast as that of other solutions such as OCSP stapling, which exhibits response times that may be down to a few seconds. During this time, the client might be vulnerable.

In order for a revocation scenario to be executed, the domain owner should lose access to the "Generation" address, by means of losing or disclosing the respective private key. In this case, a simple revocation would be performed using the "Certificate" address as described in Sect. 3.2.3. If access to the "Certificate" address is lost as well, then an extreme revocation procedure can be performed by issuing

a transaction from the "Revocation" address to the "Certificate" address. The "Revocation" address should be kept secret at all times and only be used when it is needed. This would make it extremely difficult for any attacker to acquire the respective private keys. Private keys for these "Generation", "Certificate" and "Revocation" addresses should be stored securely and separately from each other. That way, if an attack occurs, the attacker will not be able to acquire all of them at the same time, and it would have to be a three-step process. In any case, best practices indicate that private keys should be only stored on devices that are not connected to any network, on hardware wallets, on paper, or other offline mediums, in encrypted form. For additional security, all three of these addresses could also be protected under a multisignature scheme, a practice which is very common in Bitcoin and Ethereum. Even in the unlikely event that the domain owner loses access to private keys for all three addresses, there would still be a way to counter an attack. This could be accomplished by simply disabling the connected uPort identity. That would indicate that all certificates issued with that identity become invalid. The user would then have to issue a new identity and repeat the certificate creation process.

Finally, another disadvantage of the *DeTRACT* framework is that users are forced to use self-signed certificates. This is not a technical problem, and it can be mitigated only in a social level, if users learn to use decentralized identification methods, such as web-of-trust. That way, they will be able to trust an identity that was verified in a decentralized way, and with that, trust for approaches similar to ours will come.

5 Conclusions and future work

In this section, the general outcome of this work is briefly described. Our goal was to prove that a completely decentralized PKI system is viable and certificate authorities can be replaced, using blockchain technologies. In similar systems to ours, the system is not completely decentralized and CAs continue to be the central point of failure. In addition, systems that tried to fix the centralized aspect and replace CAs fall short in other aspects, like effective revocation, use of decentralized storage technologies and proof-of-concept implementations.

In our decentralized PKI approaches, a completely decentralized alternative has been developed, bypassing and fixing the central point of failure that comes with traditional PKI. We developed different approaches of our proposed framework, using different blockchain technologies and compared these approaches with respect to their individual characteristics. As a result, implementations for different user needs have been developed, each with their own advantages and disadvantages.

Although the use of blockchain technology as a replacement to certificate authorities and a means of decentralization of a PKI system seems tempting, it is not without drawbacks. Users have to be well educated and well informed in order to use a completely decentralized technology, as there is no central management. As a result, implications that may arise will be difficult to resolve. Also, for the system to be completely decentralized and secure, there is a storage requirement aspect for nodes. This may be an important consideration for the use of this technology. Success of our proposal largely depends on the adoption speed and the user behavior toward the system.

As it turns out, the objectives of this work have been achieved, but a lot of improvements could be done to improve our system. We have already set some long-term goals, some of which are the following:

- Lighter smart contract implementation. Despite the efficiency of the smart contract *DeTRACT-SC* approach, a better and lighter implementation has to be developed in order to manage a smaller amount of data, which will result in faster and cheaper transactions.
- Client-side application and browser integration. Despite the viability of our approaches, there has to be an easy and not complicated way for the users to be able to use them. This can happen either by a really well-implemented client-side application, or by building the functionality into browser applications much like the traditional PKI.
- The approaches of this work were built with reliability in mind. That is the reason why two of the most secure blockchain technologies were used. In order for the system to acquire different features, ones that the technologies used here cannot offer, other potential blockchains may be explored.

Nevertheless, the different approaches to decentralizing PKI that have been developed are effective and cost-efficient. Leveraging the transparency, immutability and decentralization aspects of blockchain technologies, *DeTRACT* showcases that it is a good building block for a completely decentralized system, which could replace the traditional PKI system, removing the need for centralized CAs.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants performed by any of the authors.

References

1. Adams, C., Lloyd, S.: Understanding PKI: Concepts, Standards, and Deployment Considerations. Addison-Wesley Professional, Boston (2003)
2. Nechvatal,J.: Public-key cryptography. Tech. Rep. NIST Special Publication 800-2, National Institute of Standards & Technology, Research Information Center, Gaithersburg, MD 20899 (1991)
3. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM **21**(2), 120 (1978). https://doi.org/10.1145/359340.359342
4. Vacca, J.R.: Public Key Infrastructure: Building Trusted Applications and Web Services. Auerbach Publications, New York (2004)
5. Adkins, H.: Google online security blog: an update on attempted man-in-the-middle attacks. https://security.googleblog.com/2011/08/update-on-attempted-man-in-middle.html (2011). Accessed 20 Oct 2018
6. Ducklin, P.: The TURKTRUST SSL certificate fiasco—what really happened, and what happens next? https://nakedsecurity.sophos.com/2013/01/08/the-turktrust-ssl-certificate-fiasco-what-happened-and-what-happens-next/ (2013). Accessed 5 Oct 2018
7. Constantin, L.: Trustwave admits issuing man-in-the-middle digital certificate, Mozilla debates punishment. https://www.cio.com/article/2399630/internet/trustwave-admits-issuing-man-in-the-middle-digital-certificate-mozilla-debates-punishment.html (2012). Accessed 10 Oct 2018
8. Keizer, G.: Solo Iranian hacker takes credit for comodo certificate attack. https://www.computerworld.com/article/2507258/security0/solo-iranian-hacker-takes-credit-for-comodo-certificate-attack.html (2011). Accessed 20 Oct 2018
9. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008)
10. Devine, P.: Blockchain learning can crypto-currency methods be appropriated to enhance online learning? http://oro.open.ac.uk/44966/ (2015). Accessed 18 Oct 2018
11. Hoy, M.B.: An introduction to the Blockchain and its implications for libraries and medicine. Med. Ref. Serv. Q. **36**, 273–279 (2017)
12. Buterin, V.: A next-generation smart contract and decentralized application platform-Ethereum whitepaper, 2014. https://www.weusecoins.com/assets/pdf/library/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf (2014). Accessed 20 Oct 2018
13. Lewison, K., Corella, F.: Backing rich credentials with a blockchain PKI. https://pomcor.com/techreports/BlockchainPKI.pdf (2016). Accessed 18 Oct 2018
14. Baldi, M., Chiaraluce, F., Frontoni, E., Gottardi, G., Sciarroni, D., Spalazzi, L.: Certificate validation through public ledgers and blockchains. In: Proceedings of the First Italian Conference on Cybersecurity, Venice, Italy, pp. 156–165 (2017)
15. Al-Bassam, M.: SCPKI: a smart contract-based PKI and identity system. In: Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts, New York, NY, USA, 2017, BCC '17, pp. 35–40. https://doi.org/10.1145/3055518.3055530. http://doi.acm.org/10.1145/3055518.3055530
16. Fredriksson, B.: A distributed public key infrastructure for the web backed by a blockchain. https://helix.stormhub.org/data/exjobb/Compact.pdf (2017). Accessed 8 Oct 2018
17. Lundkvist, C., Heck, R., Torstensson, J., Mitton, Z., Sena, M.: Uport: a platform for self-sovereign identity. https://whitepaper.uport.me/uPort_whitepaper_DRAFT20170221.pdf (2017). Accessed 20 Oct 2018
18. Boersma, J., Bulters, M.: Blockchain technology: 9 benefits & 7 challenges. https://blog.deloitte.com.ng/blockchain-technology-benefits-challenges/ (2017). Accessed 20 Oct 2018
19. Bartoletti, M., Pompianu, L.: An analysis of Bitcoin OP_RETURN metadata. In: Brenner, M., Rohloff, K., Bonneau, J., Miller, A., Ryan, P.Y., Teague, V., Bracciali, A., Sala, M., Pintore, F., Jakobsson, M. (eds.) Financial Cryptography and Data Security, pp. 218–230. Springer, Cham (2017)
20. Chariton, A.A., Degkleri, E., Papadopoulos, P., Ilia, P., Markatos, E.P.: DCSP: performant certificate revocation a DNS-based approach. In: Proceedings of the 9th European Workshop on System Security, pp. 1–6. (2016). https://doi.org/10.1145/2905760.2905767
21. Chariton, A.A., Degkleri, E., Papadopoulos, P., Ilia, P., Markatos, E.P.: CCSP: a compressed certificate status protocol. In: IEEE INFOCOM 2017—IEEE Conference on Computer Communications, pp. 1–9 (2017). https://doi.org/10.1109/INFOCOM.2017.8057065
22. Internet Engineering Task Force (IETF). Rfc6962: certificate transparency. https://tools.ietf.org/html/rfc6962 (2013). Accessed 7 July 2019
23. Internet Engineering Task Force (IETF). Rfc6698: The DNS-based authentication of named entities (DANE) transport layer security (TLS) protocol: TLSA. https://tools.ietf.org/html/rfc6698 (2012). Accessed 7 July 2019
24. Matsumoto, S., Reischuk, R.M.: IKP: turning a PKI around with blockchains. IACR Cryptol. ePrint Arch., 1018 (2016)
25. Fromknecht, C., Velicanu, D., Yakoubov, S.: A decentralized public key infrastructure with identity retention. IACR Cryptol. ePrint Arch., 803 (2014)
26. Axon, L., Goldsmith, M.: PB-PKI: A Privacy-aware Blockchain-based PKI. In: Proceedings of the 14th International Joint Conference on e-Business and Telecommunications, vol 4, pp. 311–318 (2016). https://doi.org/10.5220/0006419203110318
27. Morselli, R., Bhattacharjee, B., Katz, J., Marsh, M., Keychains: A decentralized public-key infrastructure. Tech. rep., Department of Computer Science, University of Maryland. https://drum.lib.umd.edu/bitstream/handle/1903/3332/0.pdf (2006). Accessed 8 July 2019
28. Ali, M., Nelson, J., Shea, R., Freedman, M.J.: Blockstack: a global naming and storage system secured by blockchains. In: 2016 USENIX Annual Technical Conference (USENIX ATC 16), pp. 181–194. USENIX Association, Denver (2016). https://www.usenix.org/conference/atc16/technical-sessions/presentation/ali
29. Rescorla, E.: SSL and TLS: Designing and Building Secure Systems. Addison-Wesley Reading, Boston (2001)
30. Dykcik, L., Chuat, L., Szalachowski, P., Perrig, A.: BlockPKI: an automated, resilient, and transparent public-key infrastructure (2018). arXiv preprint arXiv:1809.09544
31. Singla, A., Bertino, E.: in *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)* (IEEE, 2018), pp. 9–15
32. Caronni, G.: Walking the web of trust. In: Proceedings IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE 2000), pp. 153–158 (2000). https://doi.org/10.1109/ENABL.2000.883720
33. Namecoin. https://www.namecoin.org/ (2019). Accessed 11 Sept 2019
34. Blockstack naming service (BNS). https://docs.blockstack.org/core/naming/introduction.html (2019). Accessed 11 Sept 2019
35. Kalodner, H.A., Carlsten, M., Ellenbogen, P., Bonneau, J., Narayanan, A.: An empirical study of namecoin and lessons for decentralized namespace design. In: WEIS. Citeseer, Princeton (2015)
36. Mühle, A., Grüner, A., Gayvoronskaya, T., Meinel, C.: A survey on essential components of a self-sovereign identity. Comput. Sci. Rev. **30**, 80–86 (2018). https://doi.org/10.1016/j.cosrev.2018.10.002. http://www.sciencedirect.com/science/article/pii/S1574013718301217

37. Dhillon, V., Metcalf, D., Hooper, M.: Unpacking Ethereum, pp. 25–45. Apress, Berkeley (2017). https://doi.org/10.1007/978-1-4842-3081-7_4
38. Pow 51% attack cost. https://www.crypto51.app/
39. Internet Engineering Task Force (IETF). Rfc8615: Well-known uniform resource identifiers (URIS). https://tools.ietf.org/html/rfc8615 (2019). Accessed 7 July 2019
40. Internet x.509 public key infrastructure certificate and certificate revocation list (CRL) profile. 5.3.1 reason code. https://tools.ietf.org/html/rfc5280#section-5.3.1 (2008). Accessed 18 Jul 2019
41. Gervais, A., Capkun, S. Karame, G.O., Gruber, D.: On the privacy provisions of bloom filters in lightweight bitcoin clients. In: Proceedings of the 30th Annual Computer Security Applications Conference, pp. 326–335. ACM (2014)
42. Electrum protocol. https://electrumx.readthedocs.io/en/latest/protocol.html (2020). Accessed 08 May 2020
43. Cadena, R.V.: Cryptographic characterization of bitcoin software electrum. https://www.researchgate.net/profile/Romel_Vera/publication/301958666_CRYPTOGRAPHIC_CHARACTERIZATION_OF_BITCOIN_SOFTWARE_ELECTRUM/links/572cd2e508aeb1c73d11b219/CRYPTOGRAPHIC-CHARACTERIZATION-OF-BITCOIN-SOFTWARE-ELECTRUM (2016). Accessed 21 May 2019
44. Karame, G.O., Androulaki, E.: Bitcoin and Blockchain Security. Artech House, Norwood (2016)
45. Ethereum yellow paper. https://ethereum.github.io/yellowpaper/paper.pdf (2019). Accessed 06 Jul 2019
46. Li, X., Jiang, P., Chen, T., Luo, X., Wen, Q.: A survey on the security of blockchain systems. Future Gener. Comput. Syst. (2017). https://doi.org/10.1016/j.future.2017.08.020. http://www.sciencedirect.com/science/article/pii/S0167739X17318332
47. Solidity. https://solidity.readthedocs.io (2020). Accessed 08 May 2020
48. web3.js - Ethereum Javascript API. https://web3js.readthedocs.io (2020). Accessed 08 May 2020
49. uport connect. https://github.com/uport-project/uport-connect (2020). Accessed 8 May 2020
50. Larisch, J., Choffnes, D., Levin, D., Maggs, B.M., Mislove, A., Wilson, C.: CRLite: a scalable system for pushing all TLS revocations to all browsers. In: 2017 IEEE Symposium on Security and Privacy (SP), pp. 539–556 (2017). https://doi.org/10.1109/SP.2017.17
51. Internet Engineering Task Force (IETF). Rfc6066: transport layer security (TLS) extensions: extension definitions. https://tools.ietf.org/html/rfc6066 (2011). Accessed 7 July 2019
52. Bitcoin's security model: A deep dive. https://www.coindesk.com/bitcoins-security-model-deep-dive (2016). Accessed 10 May 2020
53. Li, C., Li, P., Zhou, D., Xu, W., Long, F., Yao, A.: Scaling Nakamoto consensus to thousands of transactions per second (2018). arXiv e-prints arXiv:1805.03870
54. How will ethereum scale? https://www.coindesk.com/information/will-ethereum-scale. Accessed 11 July 2019
55. Sompolinsky, Y., Zohar, A.: Accelerating bitcoin's transaction processing. Fast money grows on trees, not chains. Cryptology ePrint Archive, Report 2013/881. https://eprint.iacr.org/2013/881 (2013)
56. Croman, K., Decker, C., Eyal, I., Gencer, A.E., Juels, A., Kosba, A., Miller, A., Saxena, P., Shi, E., Gün Sirer, E., Song, D., Wattenhofer, R.: On scaling decentralized blockchains. In: Clark, J., Meiklejohn, S., Ryan, P.Y., Wallach, D., Brenner, M., Rohloff, K. (eds.) Financial Cryptography and Data Security, pp. 106–125. Springer, Berlin (2016)
57. Zamani, M., Movahedi, M., Raykova, M.: RapidChain: scaling blockchain via full sharding. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18), pp. 931–948. ACM, New York (2018). https://doi.org/10.1145/3243734.3243853. http://doi.acm.org/10.1145/3243734.3243853
58. Xin, W., Zhang, T., Hu, C., Tang, C., Liu, C., Chen, Z.: On scaling and accelerating decentralized private blockchains. In: 2017 IEEE 3rd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), pp. 267–271 (2017). https://doi.org/10.1109/BigDataSecurity.2017.25
59. Poon, J., Dryja, T.: The bitcoin lightning network: Scalable off-chain instant payments. https://lightning.network/lightning-network-paper.pdf (2016). Accessed 11 July 2019
60. Bellare, M., Neven, G.: Identity-Based Multi-signatures from RSA. In: Abe, M. (ed.) Topics in Cryptology—CT-RSA 2007, pp. 145–162. Springer, Berlin (2006)
61. Decker, C., Wattenhofer, R.: Information propagation in the bitcoin network. In: IEEE P2P 2013 Proceedings, pp. 1–10. https://doi.org/10.1109/P2P.2013.6688704