Security of E-Commerce Software Systems

Esmiralda Moradian

Department of Software and Computer Systems,
KTH Royal Institute of Technology,
Forum 120, 164 40 Kista, Sweden
moradian@kth.se

Abstract. Cybercrime is costly both for businesses and consumers. Criminals can have different purposes, such as financial winnings, defacement and disruption, which not only cause financial loss but also damage organization's reputation and image. To prevent a number of cybercrimes and simple mistakes, such as not insuring that all traffic into and out of a network pass through firewall, security of e-commerce systems should be considered from the very beginning, i.e. early stage of the e-commerce software development. This is due to software vulnerabilities are a huge security problem. Therefore, to enhance security of e-commerce software, we propose the use of multi-agent system. The research in this paper is focused mainly on the design of agents that provide support to engineers during development process. Moreover, the multi-agent system, presented in this research, supports implementation of patterns and extraction of security information, and provides traceability of security requirements in the engineering process.

Keywords: E-commerce, software system, security, multi-agent system, decision support.

1 Introduction

Software systems do not exist in a vacuum and must, during their lifetime, interact with the users, other systems (both software and hardware), and the environment. Regardless, a huge number of software systems that are used in e-commerce are developed without security consideration. E-commerce software systems can induce security weaknesses and defects, if security is not considered during the software development process.

Ignoring security, commonly, result in vulnerabilities that can be exploited by criminal and malicious users. Malicious users, which often belong to criminal organizations', exercise attacks that target specific selected software of which most provide access to sensitive information. Weaknesses in software may be exploited to gain access to and control of the system, steal sensitive information via the system, and use the system against the owners and users [14]. It is possible due to vulnerabilities that software possesses. Hence, it is essential to reduce vulnerabilities, which are usually the result of defective requirements and design specifications, but also implementation and testing. Software bugs are unknowingly or consciously injected into software by developers [7]. Security issues in software systems, nowadays, have been given attention due to the severe impacts software systems have on the human society, including e-commerce, governmental, military, financial, health, telecom, and transport sectors. Since security is not considered during development process, a large number of these systems, unfortunately, contain vulnerabilities, and are not resistant to attacks.

Nowadays, agent technologies are used in information and communication systems in order to provide management, search, monitoring, etc. Multi-agent systems have, also, been used for monitoring and logging purposes as well as for network security. To consider security during development and to be able to build secure software systems, engineers need support provided not only by security specialists but also by automated tools. However, to our best knowledge, such support does not exist. In this paper, we propose multi-agent system that can support engineers during the development process. Presence of semi-automated multi-agent system is necessary due to several reasons, such as: monitor decisions and activities; search for security measures and mechanisms; perform checks; and provide advices and feedbacks.

In this research, we focused mainly on the design of agents that provide support to engineers during development process. Moreover, the multi-agent system provides monitoring of the software development process, traceability of security requirements, and gives security advices in a form of checklists. Further, the proposed multi-agent system supports implementation of ontology-based patterns by extracting security information.

2 Related Work

Jennings et al. [5] in the paper entitled "Autonomous Agents for Business process management" suggest using intelligent agents to manage business processes. The authors describe in their paper the motivation, conceptualization, design and implementation of an agent-based business process management system. In the proposed system, responsibility for enacting various components of the business process is delegated to a number of autonomous problem-solving agents. To enact their role, these agents typically interact and negotiate with other agents to coordinate their actions and to buy in the services they require.

Fasli [3] discusses the use of intelligent agents in e-commerce and highlights risks that emanate from stealing information.

Mařík and McFarlane [6] discuss the need of agent-based solutions for enterprise system to provide intelligent decision-making, manufacturing control, monitoring, and transportation purposes. The authors state that intelligent agent-based technology has been applied to solve different problems like distributed order pre-processing, production planning processes and financial management and billing [6].

In our research, we propose the use of agents in multi-agent system to support engineers during software development process in order to facilitate development of more secure software for e-commerce use.

3 Essence of Software Security Engineering

Systems are often built, operated, and maintained by different groups or organizations. "Software flaws and defects can cause software to behave incorrectly and unpredictably, even when it is used purely as its designers intended" [4]. Vulnerabilities in software that are introduced by mistake or poor practices are a serious problem today. Since software controls the organizations systems, the loss of information, as well as financial loss, depends to a large extent, on how insecure software is [4, 7, 16]. Software development process offers opportunities to insert malicious code and to unintentionally design and build software with exploitable weaknesses.

By using firewalls and/or Intrusion Detection Systems (IDS), e-commerce organizations are often trying to protect themselves from attackers but are unaware that their assets are exposed even through firewalls and IDS's. E-commerce software can be exposed in many ways; therefore, developers of e-commerce systems need to decide how the software should react in different situations in a defined environment. However, it is difficult to perceive progress and traceability in software development [17]. Therefore, security-enhanced processes and practices—and the skilled people to perform them—are required to build software that can be trusted not to increase risk exposure [4].

“Software security has as its primary goals three aspects, the preservation of the confidentiality, integrity, and availability of the information assets and resources that the software creates, stores, processes, or transmits including the executing programs themselves” [2]. Security criteria should be included in each SDLC phase's input and output checkpoints [4].

Unfortunately, security solutions are often isolated from the system functionality, and can be inadequate to the stakeholders' requirements. Hence, multi-agent system that is able to provide support for developers and enhance traceability of security requirements throughout the development lifecycle is highly needed.

Requirements traceability is necessary in order to ensure that requirements are not lost in the design or implementation phases [4]. Pfleeger [15] states that capturing requirements is one of the critical parts of development process, which affects system development during all other phases. We argue that threats, attacks, and vulnerabilities are imperative factors from which security requirements are derived. Therefore, traceability between these factors in relationship with stakeholders' requirements, laws and regulations should result in definition of the security requirements.

We propose the multi-agent system that supports stakeholders, developers and managers during the engineering lifecycle. The purpose of the multi-agent system is to provide monitoring over the development lifecycle, verify and validate security requirements, and provide advices regarding security activities and security controls in a form of checklists.

4 Multi-Agent System

The proposed multi-agent system is a web-based system that can operate both internally, e.g. within the organization's network, and externally, in distributed networks, since data and information can reside in distributed environment.

The environment that agents can work within is cooperative, accessible, episodic, deterministic, dynamic and discrete. In the cooperative environment, communication between the agents takes place. Accessible environment implies that agents have access to the information and knowledge needed to perform a task. The environment is divided into atomic episodes, where each episode has an agent that performs a single task. Deterministic means that next state of the environment is determined by the current state and the action that is being executed by an agent. Dynamic environment refers to the environment that can change. Discrete environment can have a finite number of states; it also can have a discrete set of perceptions and actions. Dynamic environment refers to the environment that can change. [9,10]

The agents are communicative, mobile, cooperative, goal-oriented, autonomous, adaptive, and reactive. The agents are mobile, and can move between different locations over the networks while searching for components and services [11].

The system concerns with how agents cooperate to achieve goals, i.e., requests from users, and what is required of each individual agent in order to accomplish the goals.

Multi-agent system consists of different modules, such as, interface and authentication module, search module, and match and check module. Two types of agents are used: meta-agents and software agents. The agents in multi-agent system are organized in hierarchy, which consists of meta-agents and ground level software agents. Meta-agents operate at macro-level, while software agents operate at micro-level. The meta-agents are autonomous. It means that the meta-agents are able to take decision to satisfy their objectives [19]. Meta-agent makes a choice of best alternative regarding request from the user [8]. Implemented alternative leads to outputs and results that should satisfy the predefined goals [18]. Before the choice can be made, each alternative must be evaluated in terms of the extent to which they satisfy the objectives, i.e. defined goals.

In the proposed system some of the concepts, such as hierarchy, roles, responsibilities, and permissions are adopted from Gaia methodology [1].

Each role, that the agent possesses, is associated with a service (function). Each service has an input, an output, pre-condition, and post-condition [14]. For example, as input 'GetDocument' function can take keywords (for example, asset pattern, threat pattern) and a security level value and compare input tags to document tags. For instance, pre-condition and post-condition for 'GetDocument' function is that the knowledge base must not be empty (knowledge base empty=false) [14].

A role is defined by following attributes: responsibilities, permissions, and activities. A role also has the ability to generate information, to monitor and log events. Responsibilities are divided into two properties, such as satisfaction and security. Satisfaction defines states where an agent fulfills the goal. Satisfaction expression are activities that define an action the agent can perform, for example, SearchAgent = search [14]. The agents work with one task at time. To increase efficiency and shorten search time, software agents, in our work, execute in parallel, i.e., the group of software agents can perform one or more tasks to one or more destinations [9].

Every activity corresponds to a security property. Security property states that the system is monitored and security properties of the multi-agent system, such as confidentiality, integrity, availability, accountability, and non-repudiation, are satisfied.

Agents are assigned permissions due to responsibilities. The principle of least privilege is applied [9, 11], which implies that an agent must be able to access, i.e., read, write, execute, and/or generate only the resources (information resources or knowledge the agent possess) that are necessary for its legitimate purpose, i.e., execution and fulfillment of the assigned task (goal). Communication pathways define communication between agents, which can be unidirectional (a→b) or bidirectional (c ↔ d). The process is initiated by user request.

Each agent has allocated specific roles, which for meta-agents can be any of following: InterfaceAgent, AuthenticatorAgent, Management and Coordination Agent, ControlAgent, and MatchAgent. Ground level software agent is assigned a SearchAgent role. The IntefaceAgent processes requests from developers where a search request can contain terminology, as well as classes and/or properties. The AuthenticatorAgent authenticates the user and checks access rights. The ManagementAgent assigns tasks, coordinates the search, and manages software agents. Software agents can retrieve ontology-based patterns from local knowledge repository, as well as data from databases and repositories, for example, vulnerability databases and control catalogues. The ManagementAgent merges the results from the SoftwareAgents, which means that a meta-agent can compare and map different ontologies [13]. The process is as follows: InterfaceAgent receives a request from the user. The request (message) is validated if all constraints satisfied and passed to the ManagementAgent. ManagementAgent manages and coordinates the activities of SoftwareAgents. ManagementAgent also assigns the task to the Software Agents, to search and retrieve ontology patterns from the repository. The user in MAS is defined as a human agent (HA). Human agent roll can be assigned to stakeholders, managers, and engineers, i.e., security specialists, requirements engineers, architects, designers, programmers, and testers.

To the human agent, Interface agent is acting as an interface. A Role Schema of the InterfaceAgent is presented in Table 1:

Table 1. Role Schema Interface Agent. Source [14]

Role Schema: InterfaceAgent

Description

Receives request from the user (HA), passes request to ManagementCoordinationAgent, and returns response.

Function and Activities:

CheckForRequest, AuthenticateUser, CheckAccessRights, PassRequest, InformUser.

Permissions:

read, execute	supplied userInformation	//login information
generates	checklist	

Responsibilities:

Satisfaction InterfaceAgent = (IdentifyUser. PassRequest. GenerateChecklist)

IdentifyUser = (AuthenticateUser. CheckAccessRights)

GenerateChecklist = (ProduceChecklist. InformUser)

Security:

- UserInformation = bad ⇒ login =nil
- LoginAttempt (monitorEvent, logEvent)

In table 1, role, functions and activities of the InterfaceAgent are illustrated. The activities are as follows: AuthenticateUser, CheckAccessRights, PassRequest, and InformUser. The agent possesses following permissions: read and execute the login information, provided by the user, and generate a checklists. Satisfaction expressions are activities that define actions the InterfaceAgent can perform. The activities involve:

- IdentifyUser, which include AuthenticateUser and CheckAccessRights.
- PassRequest.
- GenerateChecklist, which involve ProduceChecklist. InformUser

Security constraints defined are as follows:

- IF input, i.e., the user login information is incorrect THEN access shall be denied
- Login attempts shall be monitored and logged

To demonstrate message exchange between the agents in the multi-agent system, Unified Modeling Language (UML) is utilized. An example of message exchange between the agents is depicted in Figure 1.

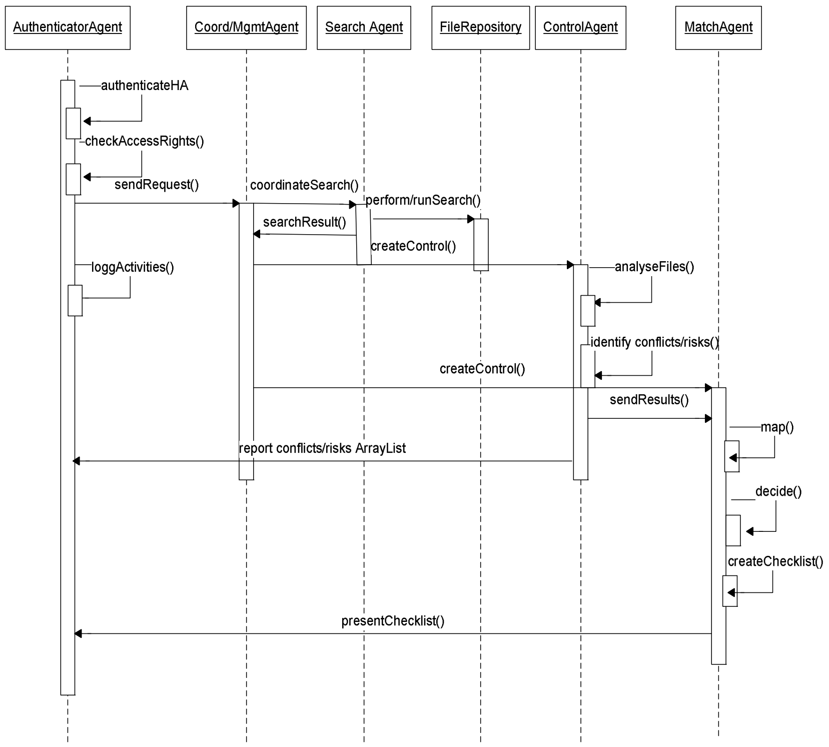

Fig. 1. Message exchange protocol

When AuthenticatorAgent receives a request from HA, the request is validated and if all constraints are satisfied, then request is passed to CoordinationAndManagementAgent. Coordination/Management agent creates control and match agents by cloning itself. Software agents verify links between documents and retrieve documents to map.

Control agent is responsible for analyzing goals and verification of requirements. Security requirements are derived from stakeholders' requirements, threats, and vulnerabilities. Meta agents analyze requirements to identify errors. In response to errors a problem report is generated and the process is halted until new request or task is received. If no errors are found the analyses result is send to match agent that provides mapping. The meta-agents provide mapping between security requirements and threats, as well as threats and countermeasures. Mapping to relevant expert knowledge stored in knowledge base performed in order to retrieve success/failure scenarios.

Meta-agents perform content analysis and mapping in order to create more complete solutions, while SoftwareAgents perform searching for documents and security ontologies.

The ontologies, for example ´Threat´ ontology and ´SecurityControl´ ontology, are retrieved from a local knowledge base [13]. The meta-agent uses a knowledge base, which contains facts and rules, to compare the ontologies, and an interpreter to match and execute rules. Rules can be constructed by terms or rules with several different alternatives, such as synonyms. The interpreter browses through the ontology with knowledge from the knowledge base. The ontologies are checked against the knowledge base with tags like *owl:Ontology, owl:Class, rdfs:subClassOf,* and *owl:onProperty*. Ontologies must contain related parts in order to make the mapping possible [13].

The search request can contain terminology, as well as classes and/or properties. The search and filtering the right documents are performed according to some criteria. The ManagementAgent merges the search result from the SoftwareAgents and passes it to the InterfaceAgent. Thus, the meta-agent perform mapping by comparing different ontologies. If there is a direct match the meta-agent can continue to work with the next part of the ontologies. Meta-agent is, also, able to combine ontologies in order to produce more complete result.

Security patterns contain documented knowledge of security professionals where a specific problem is addressed and solution described. Hence, security patterns can provide developers with the important information. Therefore, security patterns are included in the security ontology and are designed by using the ontology-based techniques. The ontology-based technique provides reusable and structured security information. Due to the OWL representation, the security patterns are available in a machine readable format and can be utilized in the multi-agent system [12].

Conclusively, multi-agent system can enable retrieval of knowledge from security ontologies and patterns and, hence, provide developers with the information about a specific threat or vulnerability, the impact of the threat on the particular security property, as well as information about possible solutions in order to minimize or prevent a particular security issue.

5 Conclusion and Further Work

In this paper, we have emphasized some crucial problems in development process in order to elucidate the essence of security issues. Security impacts core business processes in every organization. To prevent a number of cybercrimes and simple mistakes, security of e-commerce systems should be considered from the very beginning, i.e. early stage of the e-commerce software development. We have discussed the significance of the software security and necessity of traceability of security requirements.

To enable implementation of automated controls, support developers during software development and to provide visibility over the development process, multi-agent system that can fulfil these tasks, have been presented.

To search for patterns, to analyze the content and combine the ontologies in order to create more complete solutions according to the user request, agents in multi-agent system are applied. Moreover, ontology mapping performed by agents can save time and reduce human errors, which may help to increase security of the intended system. The multi-agent system, presented in this research, enables traceability of security requirements. The design of multi-agent system concerns with how agents cooperate to achieve goals, and what is required of each individual agent in order to accomplish the goals.

References

1. Cernuzzi, L., Juan, T., Sterling, L., Zambonelli, F.: The Gaia Methodology: Basic Concepts and Extensions 11, Part II, 69–88 (2004), doi:10.1007/1-4020-8058-1_6
2. Davis, N., Howard, M., Humphrey, W., McGraw, G., Redwine, S., Zibulski, G., Graettinger, C.: Processes to Produce Secure Software. In: Redwine Jr., S.T., Davis, N. (eds.) Software Process Subgroup of the Task Force on Security across the Software Development Lifecycle, vol. 1 (March 2004)
3. Fasli, M.: On agent technology for e-commerce: trust, security and legal issues. The Knowledge Engineering Review 22(1), 3–35 (2007)
4. Goertzel, M.K., Winograd, T.: Enhancing the Development Lifecycle to Produce Secure Software. A Reference Guidebook on Software Assurance, Technical Report, DACS (October 2008)
5. Jennings, N.R., Norman, T.J., Faratin, P., O'Brian, P., Odgers, B.: Autonomous Agents for Business Process Management, pp. 145–189. Taylor & Francis (2000) 0883-9514/00
6. Marik, V., McFarlane, D.: Industrial adoption of agent-based technologies. IEEE Intelligent Systems 20(1), 27–35 (2005), doi: `http://dx.doi.org/10.1109/MIS.2005.11`
7. McGraw, G.: Software Security Building Security. Addison-Wesley Pearson Ed. (2006) ISBN 0-321-35670-5
8. Moradian, E.: Secure transmission and processing of information in organisations systems. International Journal of Intelligent Defence Support Systems 2(1), 58–71 (2009)
9. Moradian, E., Håkansson, A., Andersson, J.-O.: Multi-Agent System Supporting Security Requirements Engineering. In: SERP 2010 - The 2010 International Conference on Software Engineering Research and Practice (WorldComp 2010), vol. 2, pp. 459–465. CSREA Press, USA (2010)
10. Moradian, E., Håkansson, A.: Controlling Security of Software Development with Multi-agent System. In: Setchi, R., Jordanov, I., Howlett, R.J., Jain, L.C. (eds.) KES 2010, Part IV. LNCS, vol. 6279, pp. 98–107. Springer, Heidelberg (2010)
11. Moradian, E., Håkansson, A.: Software Security Engineering Monitoring and Control. In: SAM 2011 The 2011 International Conference on Security and Management (WorldComp 2011). CSREA Press, USA (2011)
12. Moradian, E., Håkansson, A., Andersson, J.-O.: Security Patterns for Software Security Engineering. Accepted at the 16th International Conference, KES, San-Sebastian, Spain, September 10-12 (2012)
13. Moradian, E., Håkansson, A.: Ontology Design and Mapping for Building Secure E-Commerce Software. Accepted at the 8th International Conference on Web Information Systems and Technologies, Porto, Portugal, April 18-21 (2012)
14. Moradian, E.: Integrating Security in Software Engineering Process: The CSEP Methodology, KTH Royal Institute of Technology (2012)
15. Pfleeger, S.L.: Software Engineering Theory an Practice, 2nd edn. Prentice-Hall, Inc. (2001) ISBN 0-13-029049-1
16. Rice, D.: Geekonomics The Real Cost of Insecure Software. Pearson Ed. Inc. (2008) ISBN 0-321-47789-8
17. Van Vliet, H.: Software Engineering Principles and Practice, 2nd edn. John Wiley and Sons (2004) ISBN 0-471-97508-7
18. Van Gigch, J.P.: Applied General Systems Theory, 2nd edn. Harper & Row Publishers, New York (1978) ISBN 0-06-046776-2; Copyright 1978 by Van Gigch, J.P.
19. Wooldridge, M.J.: Introduction To Multi-Agent Systems. John Wiley and Sons Ltd. (2002) ISBN 9780471496915