TASHKENT UNIVERSITY OF
INFORMATION TECHNOLOGIES
NAMED AFTER MUHAMMAD AL-KHWARIZMI

MUHAMMAD AL-XORAZMIY NOMIDAGI
TOSHKENT AXBOROT TEXNOLOGIYALARI
UNIVERSITETI

BULLETIN OF TUIT:

MANAGEMENT AND COMMUNICATION TECHNOLOGIES

ISSN:2181-1083 DOI:
Rustamov Doniyor № 3s(34)2024

Methods and models for identifying threats at the design stage of information systems

Rustamov Doniyor,
Researcher,
Tashkent University of information technologies named after Muhammad al-Khwarizmi
Tashkent, Uzbekistan

***Abstract*— This article examines cybersecurity issues that affect the efficient and fully functional operation of information systems which have rapidly integrated into human activities today. It analyzes approaches to identifying and eliminating cybersecurity threats not only during the creation of information systems but also at the stage of their architecture formation. The application of STRIDE, CWE, and OWASP methodologies and their shortcomings are highlighted. It should be noted that a comprehensive analysis and description of the information system's services in threat modeling leads to the maximum formalization of potential threats. It is known that the software development lifecycle is recognized as a standard for all programmers. However, cybersecurity experts acknowledge that even in this scheme, there are insufficient stages that account for all cybersecurity threats. To address this, it is proposed to identify cybersecurity threats at the initial stages and define measures to eliminate them by introducing additional processes into the lifecycle of SDLC information system development. In this regard, it is important to develop a metamodel of information system elements and formalize the forms of activity.**

***Keywords*—** *information systems, cybersecurity, threats, STRIDE, Alloy, CAPEC.*

1. INTRODUCTION

The use of information systems in many areas has made cybersecurity an important factor in ensuring the quality of service. The transformation process from traditional computer systems to IoT, Cloud computing, and wireless communication systems requires the development of secure and reliable systems in their place. According to professional programmers around the world, cybersecurity problems are not only identified in the program code, they must be identified at the initial stages of the information system development process. These stages include the stages of processing, transmitting, and storing information at higher levels, starting from the formation of the architecture of the information system.

2. RELATED WORKS

Currently, the life cycle of information system development also focuses on cybersecurity issues and offers solutions to existing problems. For example, Microsoft's Security Development Lifecycle (SDLC) [1] can address some cybersecurity issues and even provide modeling of harmful situations and threats during the design process. Additionally, methodologies and techniques such as STRIDE [1], Common Attack Pattern Enumeration and Classification (CAPEC) [2], and Common Weakness Enumeration (CWE) [3] are designed to help visualize potential threats to information systems. The role of these tools is crucial for security analysts, as they allow for the classification, isolation, and consolidation of threats to system assets during the information system development process.

Analysis of information system architecture is highly beneficial in identifying cybersecurity threats during the initial stages of development. Analyzing cybersecurity threats at the information system architecture level serves as a supplement to assessing the risk of the system architecture, as these threats emerge at the detailed level of the information system architecture. Consequently, the threats identified during the risk assessment process are re-evaluated and can be addressed accordingly. However, the complexity of designing a secure system necessitates the support of automated tools, where precise identification and classification of threats become essential requirements.

The fundamental concept of modeling related to security requirements and information system architecture is widely known. For instance, threat modeling [4, 5] is employed to precisely identify potential risks, threats, and impact levels to determine system security requirements and to comprehend possible attack scenarios and vulnerabilities. The evolution of threat modeling methodologies aids in the development of secure system designs and architectures. Model Driven Engineering (MDE) can make a highly valuable contribution to the design and analysis of information systems [6]. The importance of models and MDE in security engineering has been emphasized and elucidated in numerous studies [7, 8]. Consequently, threat modeling and analysis should be considered at a specific point in the development processes of model-based systems.

Over the past decade, threat models and security features have been studied for the verification and validation of system models. Among the attempts to formalize existing threats, the main components of threat modeling techniques using VDM++ include STRIDE, DREAD, and the identification of mechanisms for privacy, integrity, usability, authentication, authorization, and non-repudiation. While the research does not aim to cover the entire approach, it presents key ideas and provides a foundation for adopting formal methods in threat modeling and secure system development. In this context, A. Mana and G. Pujollar [9] utilized logic-based representations to describe abstract security properties in their research, which were implemented and verified using Coq [10]. Furthermore, C.L. Heitmeyer [11] employed Software Cost Reduction (SCR) tables in their study to identify and analyze

security properties that could be encoded from system requirements.

Another approach to formulating a model of threats to information systems involves modeling system vulnerabilities as threat scenarios. The main objective of this approach is to describe how attackers can exploit existing weaknesses in the system architecture, and the application of STRIDE, CAPEC, and CWE methodologies proves highly effective in this regard. These methodologies informally describe a set of threat scenarios, with each threat scenario having a signature. The signature identifies the conditions under which a threat may arise and thus identifies threats according to specified scenarios. However, threat scenarios are described informally, so their application can sometimes lead to numerous shortcomings and be time-consuming. Therefore, it is advisable to use the Object Constraint Language model to automatically identify threats in an architectural model by defining the signatures of threat scenarios. This approach examines four threat scenarios: Denial of Service, Falsification, Injection, and Man-in-the-Middle.

Typically, the information system architecture is modeled in UML and is referred to as the System Description Model (SDM). This SDM model is compared with the security specification model, which provides cybersecurity mechanisms for information systems.

Another approach to developing security threat identification is proposed through extending it to misuse cases [12] in the context of security-oriented engineering requirements. The aim of this approach is to describe features that the system should not permit, identify security requirements, and define constraints on assets.

Approaches to threat identification have been presented in the works of many scientists, one of which is called FATHoM (FormAlizing THreat Models), which describes the structure as a relational model. In this approach, the identification of threats is carried out based on the state of logical specifications of cybersecurity features. The goal of the FATHoM approach is to ensure the detection of inconsistencies in threat models, where it is particularly well-suited for the domain of virtualized systems.

Furthermore, a partially automated approach to analyzing the architecture of information systems was proposed by M. AbiAntoun and J.M. Barnes [13], called SECORIA (Security Conformance of Object-Oriented Runtime Views of Architecture). This approach is used to verify the compliance of object-oriented programs in the field of cybersecurity. It utilizes the Acme description language to define the security architecture and formulates constraints using first-order logic. The constraints primarily focus on vulnerabilities in global information flow, with particular emphasis on threats of spoofing, tampering, and information disclosure in accordance with the STRIDE classification. The security model is constructed based on Data Flow Diagrams (DFD).

In the approach to detecting errors at the program code level, the program code is first transformed into STRIDE DFD using static analysis. Subsequently, the processes of automatic threat identification and determination of mitigation measures are carried out using threat templates based on "best practices" compiled by professionals in this field. The identified measures are attached to the DFD as annotations and can be applied to eliminate threats or reduce their impact level.

Overall, the outcome of the aforementioned approaches is a set of recommendations and guidelines for identifying, assessing, and eliminating potential threats or reducing their impact level. This facilitates the determination of security requirements. However, a global closed-loop approach was not applied to enable the possibility of iteration in determining the full set of requirements. It should be noted that most of the conducted research has limitations in formalizing threats (considering only a small set of threat categories), their reuse and expansion, and the automation of the verification process.

3. METHODS

First, we will present the architecture of the information system using the example of the OWASP (Open Worldwide Application Security Project) web application [14]. Figure 1 shows a general description of the web application architecture using a UML (Unified Modeling Language) class diagram. The information system architecture typically consists of several software components, including a client, a web server, and a database server.

It is common for an information system to offer a range of services to users, as this is considered convenient for covering all types of threats in identifying and mitigating cybersecurity risks.

Users can access the information system and utilize various services. System administrators will be able to add services and maintain statistics. Describing the architecture of the information system using UML class diagrams allows for the representation of software components as classes and the relationships between components in the form of associations.

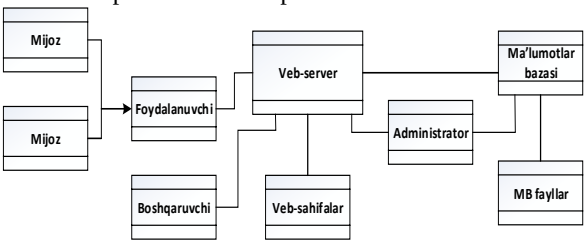

Figure 1. Description of the architecture of an information system in the UML language of a high level.

Figure 2 illustrates a class diagram with a set of security measures added to address a set of threats identified by applying STRIDE classification. However, security controls may be subject to certain changes and/or adaptations (e.g., new security solutions, removals, implementation of modifications), testing (e.g., formal, empirical), and reuse (e.g., within or across the same domain). The structure of the underlying information system architecture may be preserved. To help select appropriate security controls and increase flexibility, it is useful to include a set of modeled system properties to constrain the use of the system in protecting against appropriate threats. Each property represents a security requirement that programmers, primarily architects, want to easily express in software modeling and analysis languages. The focus is on finding a solution based on the reuse of predefined threat and requirement libraries in this threat. The study intends to use formal and MDE methods to solve the problems described above.

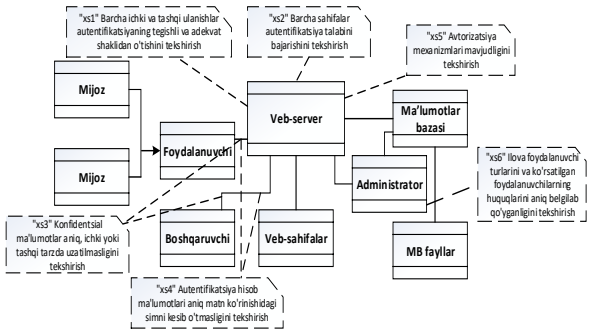

Figure 2. Information system structure supplemented with security measures.

Nowadays, the threat identification process requires the use of modern threat modeling methodologies. Therefore, the purpose of this study is to formalize the system security requirements identified through threat modeling and risk analysis as necessary features of the system architecture, validate the required features, and propose new security requirements to improve the system design and address any identified security vulnerabilities.

Figure 3 below shows a diagram of the general programming process.

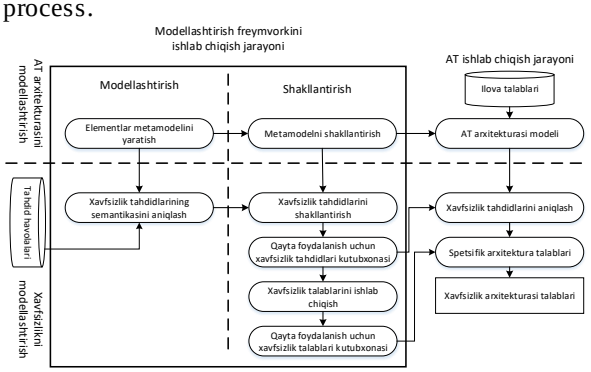

Figure 3. Scheme of the process of programming an information system.

To define an information system architecture that supports the specification and testing of reusable security solutions, it is necessary to consider modeling and formulation activities from a system architecture and security perspective.

4. RESULTS

Modeling. First, it is necessary to develop an element metamodel as a high-level concept of the information system architecture. In this, using the STRIDE threat classification [1], an abstract system computational model of distributed systems is created, and the semantics of a set of security threats are modeled. In particular, the semantics of a security threat are defined as the properties of the modeled system in a technology-independent specification language using first-order and modal logic. This semantics allows us to capture the basic communication primitives of the element metamodel, as well as the actions of legitimate and illegitimate message providers and consumers. As part of this abstract and technology-independent configuration, threat classes based on the STRIDE threat classification can be defined, which can be formalized to test the necessary properties of the elements used to define the set. Also, in this process, security requirements for designing a secure software architecture for distributed systems should be taken into account.

Formalization. After the formation of the metamodel of elements and semantics of security threats, a model is developed using a suitable language based on the verification of the model and an automated tool for it [14]. Thus, for each category of STRIDE threats, we define a metamodel of system architecture elements and a formal specification of some security threats. As a result, it defines a set of reusable threats that can identify security threats in a specific information system architecture and ensure the development of security solution specifications. This approach is the basis for developing a set of abstract formal modules of easily reusable security solutions.

Currently, many cybersecurity experts are conducting research using ALLOY (the language of first-level logic modeling) to make threats formal. This stage is necessary to support the software and existing tools used in system modeling, allowing for the easy creation and verification of models to support the identification of the corresponding set of security requirements to eliminate identified threats.

By identifying and modeling threats, it will be possible to create an architecture that can cover all possibilities by transferring the abstract architecture metamodel to a specific model for a specific information system. Then, using the application-specific system security requirements, security threat libraries are used to identify potential threats in the system model and check its compliance with the desired functions that reflect the security requirements. If the system model determines that it does not meet the security requirements, the developed libraries are used to implement protection solutions against these threats in the information system architecture model. It is then checked whether the security requirements are met in the updated information system architecture model. In this way, the requirements specification and the architectural design can be revised to revise and improve the set of security requirements that help eliminate the threats. As a result, a complete set of system security requirements is formed.

This approach can be easily applied to define cybersecurity requirements for different phases of the SDLC [14] systems development life cycle. Figure 4 suggests the use of this approach as an addition to the SDLC.

As shown in Figure 4, the following are added to the existing phase. For example, the Requirements Specification has been expanded with activities on "Formalizing Security Threats" and "Developing Security Requirements." These steps are relevant to the modeling framework development process and should only be performed when it is necessary to add a new threat or security requirement to the framework, that is, if the required threat or security requirement is not identified in the libraries. The architectural design is extended with the activities “Model the Software Architecture” and “Define Security Threats”. The purpose of these activities is to ensure that the information system architecture model meets the required characteristics of the system being developed. Finally, if it is determined that the system design does not meet the required characteristics, actions are taken to revise it. This iterative approach ensures that security considerations are seamlessly integrated into each phase of the development lifecycle

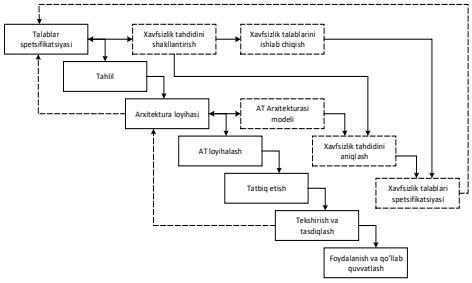

Figure 4. Scheme of implementation of the proposed approach to SDLC.

5. CONCLUSION

In the early stages of the operation of an information system developed according to this scheme, it is possible to identify data theft, modification, disclosure, destruction, loss of usability, use by an unauthorized party, and other threats. In addition, in the early stages of the information system development process, it is possible to formulate plans for cybersecurity requirements and measures to eliminate threats.

REFERENCES

[1] Microsoft, the STRIDE Threat Model, Microsoft Corporation, 2009.
[2] MITRE Corporation, Common attack pattern enumeration and classification (capec), 2018, http://capec.mitre.org/.
[3] MITRE Corporation, Common weakness enumeration (cwe), 2018, https://cwe.mitre.org/.
[4] A. Shostack, Threat Modeling: Designing for Security, John Wiley & Sons, 2014.
[5] T. Uceda Vélez, M.M. Morana, Risk Centric Threat Modeling: Process for Attack Simulation and Threat Analysis, first ed., John Wiley & Sons, 2015.
[6] B. Selic, The pragmatics of model-driven development, IEEE Softw. 20 (5) (2003) 19–25.
[7] J. Jensen, M.G. Jaatun, Security in model driven development: A survey, in: Proceedings of the 2011 Sixth International Conference on Availability, Reliability and Security. ARES '11, IEEE Computer Society, 2011, pp. 704–709.
[8] L. Lucio, Q. Zhang, P.H. Nguyen, M. Amrani, J. Klein, H. Vangheluwe, Y.L. Traon, Advances in model-driven security, Adv. Comput. 93 (2014) 103–152.
[9] S. Hussain, H. Erwin, P. Dunne, Threat modeling using formal methods: A new approach to develop secure web applications, in: Proceedigns of the 7th International Conference on Emerging Technologies, 2011, pp. 1–5.
[10] A. Mana, G. Pujol, Towards formal specification of abstract security properties, in: Proceedings of the Third International Conference on Availability, Reliability and Security, 2008, pp. 80–87.
[1] Y. Bertot, P. Castéran, Interactive Theorem Proving and Program Development: Coq'Art: The Calculus of Inductive Constructions, in: Texts in Theoretical Computer Science An EATCS Series, Springer Berlin/Heidelberg, 2004.
[11] OMG, Object constraint language (OCL), version 2.2, 2010, http://www.omg.org/spec/OCL/2.2
[12] D. Sgandurra, E. Karafili, E. Lupu, Formalizing threat models for virtualized systems, in: S. Ranise, V. Swarup (Eds.), Proceedings of Data and Applications Security and Privacy XXX, in: Lecture Notes in Computer Science, vol. 9766, Springer International Publishing, 2016, pp. 251–267.
[13] OWASP, Application threat modeling, 2017, https://www.owasp.org/index.php/ Application_Threat_Modeling.