Methods, Methodologies, and Tools for Threat Modeling with Case Study

Amina Hajrić, Tarik Smaka, Sabina Baraković, and Jasmina Baraković Husić, *Member, IEEE*

Abstract — **The security of each system is essential for its use. In order to make this process as successful as possible, it is advisable to develop a threat model for the system under consideration at the design stage. The purpose of the threat model is to enable the identification of security threats, by whose further analysis we can conclude which are the greatest vulnerabilities of the system and which pose the greatest risk. There exist many different approaches to threat modeling in terms of methods, methodologies, and tools. In this paper, we give an overview of those approaches and apply one of them, i.e., the most represented and mature to a specific system. A STRIDE-based methodology, software-centric method, and Microsoft Threat Modeling Tool (MTMT) mixture has been used to threat model the Web of Things (WoT)-based temperature management system which is in the design phase.**

Keywords — **attack, countermeasures, ICT, methodologies, modeling, security, threat, tools, WoT.**

I. Introduction

Security problems of computer systems have increased with the increased development of those systems. Rise in the level of knowledge and capabilities led to an increase in the level of abuse of the system. Connected devices with special-purpose have a significant number of potential interaction surfaces and interaction patterns, all of which must be considered as a framework for providing digital access to these devices. All this is also revealed by cybersecurity challenges and issues statistics [1] which are characterized to be on the rise on a day-to-day basis. When the system is in a design stage, it is of great importance to understand possible threats and vulnerabilities to it in order to apply the appropriate defense measures [2]. In order to perform that, one can do the threat modeling process. Threat modeling is an engineering technique that can be used to identify threats, attacks, vulnerabilities, and appropriate countermeasures in the context of a particular application [3]. It is not a one-off process and it is closely linked and intertwined with the design and development stages of an application or system. Threat modeling helps to find problems in the initial design phase. Getting rid of threats in the beginning phase is much easier than adding countermeasures, testing them, and ensuring they stay up to date.

There are many methodologies, methods, and tools used to perform threat modeling of a system. This paper aims to contribute to the readership by providing a brief overview of those approaches, which beginners in this field can find very useful. Also, the additional goal of the paper is to illustrate the process on a concrete system, i.e., to apply the most represented approach to a specific system in its design stage and identify possible threats for it. Therefore, this paper additionally contributes by conducting a STRIDE-based software-centric threat modeling approach to the abstract Web of Things (WoT)-based temperature management system by using Microsoft Threat Modeling Tool (MTMT). The given combination of method, methodology, and tool has been used due to the nature of the specific system and the fact that the STRIDE is the most mature threat model approach and MTMT the most mature tool as it will be seen from the analysis.

The structure of the paper is as follows: after having introduced the threat modeling as a notion, Section II explains the steps of the process. Section III describes threat modeling methods, while Section IV provides a description of threat modeling methodologies. Section V gives an explanation of the threat modelling tools. Further on, Section VI provides a case study of threat modeling of an abstract information and communication technology (ICT) system, i.e., WoT system by using a STRIDE-based software-centric principle. Analysis of the MTMT obtained threat modeling results and their discussion are given in the same section. Section VII concludes the paper.

II. Threat Modeling Process

Threat modeling has two different meanings in computer security that are interconnected. The first is to describe the safety issues that designers need to pay attention. The second defines threat modeling as a specific set of possible attacks to consider for a particular part of a program or computer system [4].

As already mentioned, threat modeling is an iterative process that starts in the early stages of application development and lasts throughout the application lifecycle. There are two main reasons for this approach. For starters, it is impossible to identify all existing threats in one pass. On the other hand, the threat modeling process needs to be repeated along with the development of the application because applications are rarely static, so they need to be constantly adapted and enhanced to meet business requirements [4], [5]. The threat modeling process itself, consisting of six phases and is described in the following text.

A. Identifying resources

This step involves identifying the resources that need to be protected. These resources cover a wide range of data, ranging

Paper received March 11, 2020; revised July 12, 2020; accepted July 14, 2020. Date of publication July 31, 2020. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Miroslav Lutovac.

Amina Hajrić is with Infobip BH, Tešanjska 24a, Sarajevo, Bosnia and Herzegovina; (e-mail: amina_hajric@hotmail.com).

Tarik Smaka is with AS Holding, Ekonomija bb, Tešanj, Bosnia and Herzegovina; (e-mail: tariksmaka@hotmail.com).

Corresponding author Sabina Baraković, is with the Ministry of Security of Bosnia and Herzegovina, Trg BiH 1, Sarajevo, Bosnia and Herzegovina; (e-mail: barakovic.sabina@gmail.com).

Jasmina Baraković Husić is with the University of Sarajevo, Zmaja od Bosne, Sarajevo, Bosnia and Herzegovina; (e-mail: jbarakovic@etf.unsa.ba).

from confidential information to website availability. Confidential information includes personal information, intellectual property information, credit card number information and passwords [3].

B. Documenting architecture

The primary goal of this step is to document the function of the application, its architecture and physical appearance, and the technologies that make the application work. This phase involves performing the following tasks [3]: i. Identification of the application function; ii. Creation of an architecture diagram; and iii. Identification of technologies.

C. App parsing

Application breakup involves disassembling the application and creating a security profile for the application based on vulnerability. The following tasks should be performed [3], [4]: i. Identifying trust boundaries; ii. Identifying data flow; iii. Identifying entry points; iv. Privilege code recognition; v. Security profile documentation.

D. Identification of threats

This step consists of identifying threats that can affect the system and endanger resources. Two approaches can be used to classify threats [3], [4]: i. STRIDE - a goal-based approach that addresses the targets of the attacker; and ii. categorized threat lists - an approach which starts with a list of common threats sorted into networks, host, and category applications. STRIDE is a classification scheme for characterizing known threats by the type of exploitation for which it is used or by the motivation of the attacker. It is an acronym composed of the first letters of each of the six threat categories [3], [6]: i. Spoofing - trying to access the system using a false identity; ii. Tampering - tampering with data; iii. Repudiation - the user can dispute transactions with no audits and activity repositories; iv. Information disclosure - unwanted reading of private information; v. Denial of Service (DoS) - action by disabling the service; vi. Elevation of privilege - the less privileged user assumes the identity of the privileged user. All actions must be enclosed with an authorization matrix to ensure that only privileged users can access a privileged functionality.

E. Documenting threats

A template that contains several threat attributes is used to document the threat. The most important attributes are the threat description and the target of the threat. Attribute scan emphasizes exploited vulnerabilities, while countermeasure attributes are needed to address threats. The risk attribute remains blank at this stage and is filled in at the last stage of the threat modeling process [3].

F. Threat assessment

By this step of the process, a list of threats to the observed application has been compiled. Threats are evaluated on the basis of the risks they bring. On this basis, a list of threats is formed at the top of which are threats that carry the greatest risks [3].

There are several ways to perform rating of threats, and the one that is used mostly is DREAD model [7], which considers the following items: i. Damage – how bad would an attack be?; ii. Reproducibility – how easy is it to reproduce the attack?; iii. Exploitability – how much work is it to launch the attack?; iv. Affected users – how many people will be impacted?; v. Discoverability – how easy is it to discover the threat? The threat is assessed by answering the questions above and assigned values for each item (high, medium, low). Rating values represent severity and are expressed in numbers (3-high, 2-medium, 1-small) [7]. After that, all ratings are summed up and a final rating for a specific threat is obtained, ranging from 5 to 15. The scale determines the risk: i. High risk - 12-15; ii. Medium risk - 8-11; iii. Low risk - 5-7.

III. Threat Modeling Methods

In order to improve the security of ICT systems, the first step is to get an overview of the vulnerabilities in the system or organization and the potential attacks where those vulnerabilities are exploited. In order to find these threat scenarios, there are several common techniques, and an individual or team looking for threat scenarios in any given situation usually focuses on the specific actors and features of the ICT systems they are modeling. There are four methods to threat modeling [7], [8], which are explained in the following sections.

A. Attacker – centric method

As the name implies, this type of threat modeling method focuses on the potential attacker. A list of potential threat agents is used by the analyst as a basis for finding threat scenarios. That list can be created by the analyst or be part of a collection of attacker lists. The analyst will then look at the list of attackers and get to know each attacker individually to understand how they think, behave and act. When an analyst learns more about possible attackers, he can see and understand what their goals are and thus see relevant threat scenarios.

B. Software – centric method

A software-centric method often referred to as system-oriented is centered around software models. If the system is large and complex, it is perfectly acceptable and normal to add more diagrams that show certain details that do not fit into the overview diagram of the system. Then, to help find where things can go wrong, it is helpful to add confidence limits to the diagram. These are the boundaries that separate the different parts of the system according to the confidence that each module or part has in each other. The reason for this is because vulnerabilities are often located around such boundaries, and so defining the system can help the analyst to get a clearer picture of the system and where to focus attention to find as many vulnerabilities and threats as possible.

C. Asset – centric method

An asset-centric method is based on describing the resources of the information technology (IT) system or organization the analyst wants to protect. Generally, the term resource can be categorized into the following ways: i. things the attackers want, ii. things you want to protect, iii. stepping stones to either of these. Because the list of resources can become quite large, it is important for the analyst to decide what resources are actually relevant in the terms of information security. Once the funds are defined, the analyst selects a resource from the list, one at a time, and focuses on that specific resource. The analyst then describes threat scenarios that could affect a particular resource. After the relevant asset evaluation, the result is a description of threat scenarios that could affect the resource of the system or organization.

D. *Defense – centric method*

In a defense – centric method, threat modeling is performed based on the assessed weaknesses in security surveillance [4].

IV. Threat Modeling Methodologies

Threat modeling methodologies are used to create a system abstraction, profiles of potential attackers, including their goals and methods, and a catalog of potential threats that may arise [9]. There are many threat modeling methodologies focusing on abstraction, people, risks or privacy. The basic threat modeling methodologies are given in the following sections.

A. *STRIDE*

STRIDE is currently the most mature threat modeling methodology. It has evolved over time to include new threats and variant-specific tables, and one differentiates between STRIDE-by-element and STRIDE-by-interaction [5], [9]. However, a precondition for STRIDE is the existence of a good model of the system. Model diagrams need to be drawn so that it is easy to spot and place lines around confidence boundaries. Data Flow Diagram (DFD) is a type of diagram commonly used to describe how network architectural systems work.

B. *PASTA*

The Process for Attack Simulation and Threat Analysis (PASTA) is a risk-focused threat modeling methodology. This seven-stage process comprises the following activities [9], [10]: i. Definition of goals; ii. Definition of the technical framework; iii. Decomposition; iv. Threat analysis; v. Vulnerability and vulnerability analysis; v. Modeling attacks; vi. Impact and risk analysis. PASTA's goal is to link business goals and technical requirements through various design and execution tools at different stages. This method elevates the threat modeling process to a strategic level by involving key decision-makers and requiring security from operations, management, architecture, and development. Broadly viewed as a risk-focused framework, PASTA uses an attacker – centric method to produce asset-centric output in the form of threat and scoring [9], [10].

C. *LINDDUN*

LINDDUN is a methodology that focuses on privacy issues and it can be used for data security. It is an acronym made out of the initial letters of the following words: linkability, identifiability, non-repudiation, detectability, disclosure of information, unawareness, non-compliance. Also, LINDDUN consists of six steps and thereby provides a systematic approach to privacy assessment [9], [11]: creation of system's DFD with mapping of threat categories to the system parts and identification of cases in which the subject threats can occur. Afterward, those threats are prioritized and solutions and mitigation strategies are found.

D. *CVSS*

The Common Vulnerability Scoring System (CVSS) captures the main features of vulnerability and provides a numerical assessment of severity. It is developed by the National Institute of Standards and Technology (NIST) and is hosted by the Forum of Incident Response and Security Team (FIRST) with support and input from CVSS's Special Interest Group (SIG). CVSS provides users with a common and standardized scoring system across various cyber and cyber-physical platforms. The CVSS score can be calculated using a calculator available online [9], [12]. CVSS consists of three metric groups (i.e., base, time, and environmental), each with a set of metrics. In order to give an idea of each of these groups, we can describe them as follows [13]: i. The base represents vulnerability characteristics that are constant over time in the user environment; ii. The time represents vulnerability characteristic that changes over time but does not mention user environments; iii. The environment presents vulnerability characteristics that are relevant and/or unique to a particular user environment. Once each of these basic metrics has been assigned a value, the basic equation calculates a score that ranges from 0 to 10.

E. *Trike*

Trike treats threat modeling from a risk management and defense perspective [4]. It begins with the definition of the system. The analyst builds a demanding model by enumerating and understanding the system actors, resources, intended actions, and rules. This step creates a matrix of activity and resource actors in which columns represent resources and rows represent actors. Each matrix cell is divided into four sections, one for each creates, read, update, and delete (CRUD) action. In these cells, the analyst assigns one of three values: allowed action, illegal action, or rule action. The rule tree is attached to each cell [9]. After defining the request, a DFD is built. Each element is mapped to select participants and assets. Repeating through DFD, the analyst identifies threats that fall into one of two categories: elevation of privilege or denial of service. Each detected threat becomes a root node in the attack tree [8], [14], [15].

F. *VAST*

The methodology for modeling Visual, Agile, and Simple Threats (VAST) is based on the automated ThreatModeler [16] threat modeling platform. Its scalability and usability allows it to be embraced across large organizations across the infrastructure to produce effective and reliable results for a variety of stakeholders [9]. Recognizing business differences and concerns between development and infrastructure teams, VAST requires the creation of two types of models, i.e., application threat models and operational threat models. Application threat models use process flowcharts, which represent an architectural point of view. Operational threat models are created from the point of view of attackers based on DFDs [9].

G. *OCTAVE*

The Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE) is a risk-based strategic assessment and planning methodology for cybersecurity. OCTAVE focuses on assessing organizational risks and does not address technology risks. Its main aspects are operational risk, security practices, and technology [9]. OCTAVE has three phases: i. Build resource-based threat profiles (this is an organizational assessment); ii. Recognize infrastructure vulnerability (this is an information infrastructure assessment); iii. Develop a security strategy and plans (this is a risk identification for the organization's critical resources and decision making) [9], [17].

In addition to the methodologies described above, there are attack trees, persona non grata, security cards, the Hybrid Threat Modeling Method (HTMM), and quantitative threat modeling methodology.

V. Threat Modeling Tools

There are many commercial and open source tools that can be used to automate the threat modeling process. One of the most prominent commercial tools is ThreatModeler [18], while the open source ones, such as MTMT [19], Threat Dragon [20], CORAS [21], Trike [22], IriusRisk [23], etc. are briefly described in the following text.

A. MTMT

The MTMT is probably the best-known freeware tool for analyzing and modeling security threats. It allows professionals to access known information, such as business requirements and application architecture. They are then used to build a feature-rich threat model. In addition to automatically detecting threats, the tool also offers valuable security artifacts [19].

B. Threat Dragon

Threat Dragon is a free, open-source, cross-platform threat modelling application including system diagramming and a threat rule engine to auto-generate threats/mitigations. The focus of the project is on great user experience (UX), a powerful rule engine and integration with other development lifecycle tools. The application comes in two variants: i. web application where the model's files are stored in GitHub, and ii. desktop application where models are stored on the local file system. End user help is available for both variants. This repository contains the files for the web application variant [20].

C. CORAS

CORAS is a method for conducting security risk analysis and provides a customized language for threat and risk modelling. It is model-based. The Unified Modeling Language (UML) is typically used to model the target of the analysis. For documenting intermediate results, and for presenting the overall conclusions one uses special CORAS diagrams which are inspired by UML. In this tool a security risk analysis is conducted in eight steps: i. preparations for the analysis, ii. customer presentation of the target, iii. refining the target description using asset diagrams, iv. approval of the target description, v. risk identification using threat diagrams, vi. risk estimation using threat diagrams, vii. risk evaluation using risk diagrams, viii. risk treatment using treatment diagrams [21].

D. Trike

Trike is an open source threat modeling methodology (as mentioned in previous section) and tool. The project began in 2006 as an attempt to improve the efficiency and effectiveness of existing threat modeling methodologies, and is being actively used and developed [22].

E. IriusRisk

IriusRisk is a threat modeling tool with an adaptive questionnaire driven by an expert system which guides the user through straight forward questions about the technical architecture, the planned features, and security context of the application [23].

VI. A Case Study: Threat mModeling of the WoT System

For the purpose of explaining and illustrating the threat modeling process, we will apply the process on the abstract temperature management system as one WoT [24] based system. This system is still in the design phase, i.e., it is abstract, and its idea architecture is given in Fig. 1. In order to conduct the process, we used a computer tool MTMT [19], which is a STRIDE-based tool.

A. Threat Modeling of WoT system – Step by Step

Firstly, we created a diagram of the system. A prerequisite for that is to be familiar with all elements and relations of the system. Based on system architecture, MTMT generates a report where one can find all potential threats of the system. The final step is to analyze all threats and consult with the team.

In order to explain the idea architecture of the WoT-based temperature management system, we divided it into five zones. All zones are interconnected by different technologies and protocols. The first zone is the user zone. The second zone is the device and browser zone, which includes devices, such as desktops, laptops, tablets, smartphones, wearable devices (e.g., smartwatches), etc. The Wireless Fidelity (Wi-Fi) router represents the third zone and is also the interface between the user zone, the device zone, the browser and the WoT zone of the device. The router enables communication to the Internet through different protocols depending on the device used. The fourth zone is the zone of WoT devices and it includes WoT devices, such as WoT server or WoT Application Programming Interface (API). In the observed system, the WoT is an air conditioner. Inside the air conditioners a temperature sensor is located and connected by the wire. This represents the fifth zone of the observed system.

In accordance with the given procedure of threat modeling, we proceed with the process on a concrete system as follows.

I: Identifying System Resources – In this step, it is necessary to identify all those resources that are relevant to the particular system and environment in which it is deployed. Resources that are important for the temperature management system are: i. Hardware - temperature sensor, air conditioning; ii. Software - web application, web server; iii. Data - user data (i.e., user personal data, codes, usernames, etc.), data stored on a web server.

II: Architecture Documentation – Architecture documentation involves the presentation of the architecture with identification of the basic functionality of the system, creation of high-level architecture models, and identification of technologies that ensure the functionality of the system. The basic function of this system is to allow remote control of the room temperature. The user accesses the system through various devices and web browsers, which poses an additional security challenge.

III: Layering Architecture and App Parsing - At this stage, it is necessary to break down the complete architecture into smaller sections so that system elements of vulnerabilities can be better understood and analyzed. This involves identifying confidence limits, identifying data flows between subsystems, identifying system entry points, etc. The architecture shows the physical arrangement of the individual system, as well as their interconnection. The architecture is divided into several zones, which together form one whole.

The user zone is the zone from which the user sends requests to access the system. In order to be able to do this, it needs to establish the Internet connection with a Wi-Fi router in this case. It accomplishes this through a user's browser. The router zone allows users to connect to the Internet, so that they can access the web server's data. Also, it should be noted that the Internet connection can be made via the third generation (3G) network.

The frontend zone allows data to be displayed. The WoT zone represents the zone where the web server is located and the sensor is wired to the WoT device (which in this case represents the web server). All data collected by the sensor is displayed in the frontend section through the web server. An administrator is a person who has the ability to access the web server and perform certain administrative tasks or settings such as filtering certain data (Fig. 1.). The used technologies are: i. HyperText Transfer Protocol Secure (HTTPS) application layer communication protocol, and ii. Wi-Fi technology for data transfer between users and systems (in addition, 3G and 4G technology can be used).

IV and V: Threat Identification and Documentation - For the temperature management system, 93 threats have been identified using MTMT [19]. A generated MTMT report returns a list of all possible threats to the system architecture set. The report provides the threat names (which briefly describes the threat itself), the threat category according to the STRIDE model, and a more detailed description of the threat. In a detailed description, one can gain better understanding of the very nature of the threat, as well as identify possible security measures. Also, some of the protection tips are provided in the description itself. Threats are divided into previously grouped parts of the system. One type of threat can occur in different places within the system, but it is clear that it will not do the same damage everywhere, and it will not be equally dangerous for every part of the system.

VI: Threat Assessment - Once all threats to a given system have been identified, it is necessary to evaluate them. The evaluation is done primarily so that we can know which of these threats are the most dangerous because they are our highest priority and it is desirable to protect the system against them. Previously described DREAD [7] method was applied for the threat assessment. In this system there are 13 different relations. MTMT identifies the threats per every single relation. All threats were evaluated individually using DREAD.

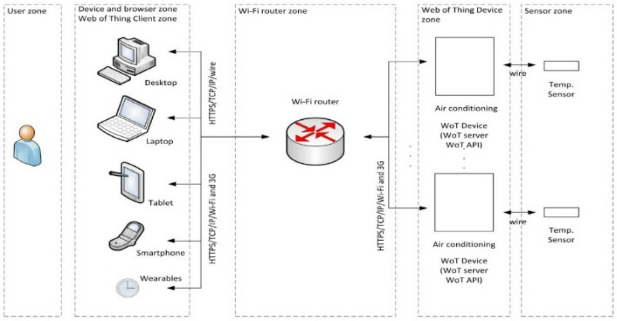

Fig. 1. Idea architecture of the abstract WoT-based temperature management system.

B. Analysis and Discussion of Obtained Results

As already stated, 93 system threats were identified using the STRIDE-based software-centric approach with MTMT tool. They are all categorized based on the STRIDE methodology and the number of threats by category is shown in Fig. 2.

The percentage representation of threat categories is given in Fig. 3. Based on the obtained results, we can notice that in the analysis of the WoT temperature management systems, DoS attacks have the highest probability of occurrence (31%), wherein many ways users are prevented from using the system, which certainly threatens the security of both the data and the environment in which the sensor is placed. The second place is taken by Elevation of privilege threats (27%), while Spoofing (17%), Repudiation (14%), Tampering (8%), and Information disclosure (3%) are ordered from the third to the sixth place, respectively. Table I shows the number of occurrences of a given attack level within the STRIDE attack classification. When it comes to high-risk threats, we can see that most of these threats fall into the category of Elevation of privilege (although Information disclosure is not negligible, as only three threats in total from that category are identified and all three are high-risk). Then, most medium-risk threats fall into the DoS category (this is the category from which most threats are identified, but they are not high-risk). Finally, when it comes to low-risk threats, the highest is also in the DoS category.

As indicated, the DREAD method was used to evaluate the identified threats, where each threat was evaluated individually and as a result, a final risk assessment was carried out (high, medium, low). The results indicate that 75.27% of threats are medium-risk ones, while there is approximately the same number of high (11.83%) and low-risk (12.9%) ones.

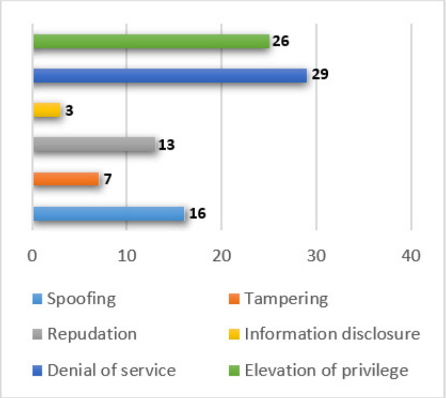

Fig. 2. Threats classified by STRIDE categories.

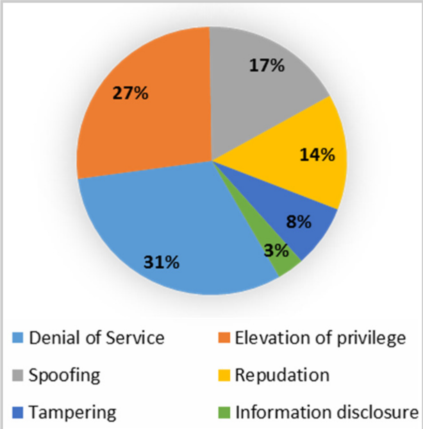

Fig. 3. The percentage representation of threat categories.

Threat modeling in general can be performed by using developed methods, methodologies, and tools, or by security experts brainstorming. Since the latter is time and people resource consuming, it is much easier to use the former. How realistic and reliable the results are, depends on the used approach. In this paper, we have used the software, i.e., system-oriented method of threat modeling due to nature of the system. In terms of methodology, we have used the most mature one – STRIDE, while the used tool is the most used open-source one

– MTMT. In general, the reliability of each tool depends on the update of the mechanisms and databases on which it relies on. If the background mechanisms are regularly updated with the newest threats and vulnerabilities, but also with new ICT elements, then their reliability is higher. However, the only true way one can perform the quality evaluation of threat modeling method, methodology, or tool is to perform evaluation by human [5].

TABLE I RISK LEVEL PER STRIDE CATEGORY.

Threat	***High risk***	***Medium risk***	***Low risk***
Spoofing	3	13	0
Tempering	1	6	0
Repudiation	0	11	2
Information disclosure	3	0	0
Denial of service	1	21	7
Elevation of privilege	4	19	2

The obtained results in terms of identified threats and their ratings for abstract WoT-based temperature management system addressed in this paper will help this system's developers to find the potential problems in the initial design phase of the system. The described process of threat modeling provided a good basis for specifying security requirements for the system during the development. It helped the developers in many ways: from authenticating application architecture, identifying and assessing threats, finding countermeasures, etc. Also, it needs to be pointed out that threat modeling is an iterative process that starts in the early stages of application development and lasts throughout the system lifecycle.

VII. CONCLUSION

The security of each system is essential for its use and the most versatile application. In order to make this process as successful as possible, it is advisable to develop a threat model for the system under consideration at the design stage. The purpose of this model is to enable the identification of security threats. In other words, threat modeling is a process by which system threats are identified, evaluated, and applied. There are many approaches to conduct threat modeling in terms of methods, methodologies, and tools.

This paper aimed to give a brief overview of the existing ones and to apply the selected to the abstract system in the design phase in order to illustrate the process. Also, we have illustrated the process of threat modeling on an abstract WoT-based temperature management system in the design phase by using the STRIDE-based software-centric approach for threat identification, DREAD for threat evaluation, and MTMT tool for the entire process.

MTMT identified 93 different threats for the observed system architecture. After summarizing the results and analyzing them, we conclude that the greatest number of threats (29) falls into the category of DoS, wherein many ways the use and access to the system are prevented. The DoS threats are at the same time the ones characterized as medium and low-risk ones, while the most represented high-risk threats come from the Elevation of privilege class of threats.

Since the analyzed abstract system is still in the design phase, the obtained results will contribute to the implementation of the principle security and privacy by design, thereby increasing the security of the system as a final product. For future research, it is necessary to consider in more detail the architecture of the considered WoT-based temperature management system, i.e., to build a higher-level architecture model compared to one observed in this paper.

REFERENCES

[1] R. Sobers. (2020). *Must-Know Cybersecurity Statistics for 2020.* Available: https://www.varonis.com/blog/cybersecurity-statistics/.

[2] P. Meadows. (2018). *Internet of things (IoT) architecture*. Available: https://docs.microsoft.com/en-us/azure/iot-fundamentals/iot-security-architecture.

[3] J. D. Meier, A. Mackman, B. Wastell, P. Bansode, J. Taylor, R. Araujo. (2010). *Threat Modeling Web Applications*. Available: https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff648006(v=pandp.10)?redirectedfrom=MSDN.

[4] Centar informacijske sigurnosti (CIS). (2012). *Modeliranje sigurnosnih prijetnji (Threat modelling).* Available: https://www.cis.hr/files/dokumenti/CIS-DOC-2012-05-049.pdf.

[5] A. Shostack. *Threat Modeling: Designing for Security.* John Wiley & Sons, Inc., 2014.

[6] F. Swiderski, W. Snyder, *Threat Modeling*, Microsoft Press 2004.

[7] M. Haider. (2017). Application Threat Modeling using DREAD and STRIDE. Available: https://haiderm.com/application-threat-modeling-using-dread-and-stride/.

[8] D. Eng, "Integrated Threat Modelling," MSc Thesis, University Oslo, 2017, Available: https://www.duo.uio.no/bitstream/handle/10852/55699/dae-thesis.pdf?sequence=1&isAllowed=y.

[9] N. Shevchenko, T. A. Chick, P. O'Riordan, T. Scanlon, C. Woody. (2018). *Threat modeling: A summary of available methods*. Available: https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=524448.

[10] Versprite. (2020). *Application threat modelling.* Available: https://versprite.com/security-offerings/appsec/application-threat-modeling/.

[11] LINDDUN. (2020). Available: https://www.linddun.org/.

[12] Common Vulnerability Scoring System Version 3.0. Calculator, (2020). Available: https://www.first.org/cvss/calculator/3.0.

[13] P. S. Kadhirvelan, A. Söderberg-Rivkin, "Threat Modelling and Risk Assessment Within Vehicular Systems,", MSc Thesis, Chalmers University of Technology & University of Gothenburg, August 2014, Available: http://publications.lib.chalmers.se/records/fulltext/202917/202917.pdf.

[14] P. Saitta, B. Larcom, M. Eddington. (2005). Trike *v.1 Methodology Document [Draft]*. Available: http://www.octotrike.org/papers/ Trike_v1_Methodology_Document-draft.pdf.

[15] Octotrike. (2020). Available: http://www.octotrike.org/.

[16] Threatmodeler. (2020). Available: https://threatmodeler.com/.

[17] C. J. Alberts, A. J. Dorofee, J. F. Stevens, C. Woody, *Introduction to the OCTAVE Approach.* Software Engineering Institute, Carnegie Mellon University, 2003.

[18] ThreatModeler. (2020). Available: https://threatmodeler.com/.

[19] Microsoft Threat Modeling Tool. (2020). Available: https://www.microsoft.com/en-us/download/details.aspx?id=49168.

[20] Threat Dragon. (2020). Available: https://threatdragon.org/login.

[21] M. S. Lund, B. Solhaug, K. Stolen. (2015). Model-Driven Risk Analysis. Available: http://coras.sourceforge.net/.

[22] OctoTrike. (2012). Available: http://www.octotrike.org/.

[23] IriusRisk. (2013). Available: https://iriusrisk.com/threat-modeling-at-scale/.

[24] W3C. (2020). Web of Things at W3C.