CVSS-based Estimation and Prioritization for Security Risks

Roman Wirtz and Maritta Heisel
[1]*Working Group Software Engineering, University of Duisburg-Essen, Oststr. 99, Duisburg, Germany*

Keywords: Security Risk, Risk Management, Risk Estimation, CVSS, Pattern, Requirements Engineering.

Abstract: During software development, it is of essential importance to consider security threats. The number of reported incidents and the harm for organizations due to such incidents highly increased during the last few years. The efforts for treating threats need to be spent in an effective manner. A prioritization can be derived from the risk level of a threat, which is defined as the likelihood of occurence and the consequence for an asset. In this paper, we propose a risk estimation and evaluation method for information security based on the Common Vulnerability Scoring System (CVSS). Our method can be applied during requirements engineering. The application in one of the earliest stages of a software development lifecycle enables security engineers to focus on the most servere risks right from the beginning. As initial input, we make use of a pattern-based description of relevant threats to the software. When estimating the risk level of those threats, we consider three perspectives: (1) software providers, (2) data owner, and (3) third parties for which a potential harm may exist, too. Our method combines attributes of the pattern and the different perspectives to estimate and prioritize risks. The pattern-based description allows a semi-automatic application of our method, which ends with a ranking of risks according to their priority as final outcome.

1 INTRODUCTION

Security is one key factor for building successful software. The number of reported security incidents and the resulting harm for organizations and private stakeholders highly increased during the last few years. The effort for protecting against such threats increases, the later one considers them during software development. Therefore, we advocate the concept of security-by-design, which means to consider security aspect as early as possible. Prioritizing threats helps security engineers to spend the necessary effort in an effective manner. A prioritization can be performed based on risks. A risk is defined as the combination of the likelihood of the occurrence of a threat and its possible consequence for some asset.

A risk management process describes a set of coordinated activities to identify, estimate and treat risks. In this paper, we provide a method that assists security engineers in estimating and evaluating the previously identified risks during requirements engineering. Our method provides novel contributions for evaluating risks: First, we identify stakeholders for which consequences may exist and make their consequences explicit when estimating the risks. Second, we make use of existing pattern-based threat knowledge from which we automatically derive consequence values.

The initial input of our method is a set of identified threats which are described using an attribute-based pattern (Wirtz and Heisel, 2019). The pattern is based on the Common Vulnerability Scoring System (CVSS), which not only provides attributes to describe vulnerabilities, but also metrics to calculate the severity of vulnerabilities. In our method, we combine the threat description with the CVSS metrics to estimate risks. For the estimation, we consider three different types of stakeholders: (1) software provider, (2) data owner, and (3) third parties for which a potential harm may exist. Our method starts with identifying those stakeholders, and we make the value of the assets explicit for all of these stakeholders. To calculate the severity of a threat, we combine the values to derive an overall severity. Using risk matrices, we first evaluate the risks with regard to their acceptability and finally, we prioritize unacceptable risks.

The final outcome of our method is a list of risks which are ordered according to their risk level.

The paper is structured as follows: We introduce relevant background knowledge in Section 2. In Section 3, we describe our method, which is the main contribution of the paper. To exemplify our method,

Wirtz, R. and Heisel, M.
CVSS-based Estimation and Prioritization for Security Risks.
DOI: 10.5220/0007709902970306
In *Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2019)*, pages 297-306
ISBN: 978-989-758-375-9

we provide a case study in Section 4. We discuss different aspects and limitations of our approach in Section 5, and we discuss related work in Section 6. Finally, we conclude our work in Section 7 with a summary and some future research directions.

2 BACKGROUND

In this paper, we make use of the *Common Vulnerability Scoring System (CVSS)* (FIRST.org, 2015) to estimate security risks. It consists of different metrics to calculate the severity of vulnerabilities. In the following, we describe the *Base Metric Group* and parts of the *Modified Base Metrics* which we will adapt for our method. For each metric, there is a predefined qualitative scale. To each value of the scale, the CVSS assigns a numerical value. Those numerical values are then used in formulas to calculate the severity.

(Wirtz and Heisel, 2019) relate the CVSS to the description of threats for an application during requirements engineering. The authors propose a pattern which describes a threat based on the base metrics of the CVSS. Table 1 shows the relevant excerpt of a pattern instance for the threat *Injection*. In the following, we explain the different metrics and corresponding values.

Threat Description A threat might be accidental or deliberate (*Threat Type*). A *Threat Agent* realizes a threat and can be *human*, *technical* or *natural*. The *Threat Vector* (attack vector in CVSS) describes possible ways how to realize a threat. There are four different values: (1) *network*, which means access from an external network; (2) *adjacent*, which means a local network; (3) *local*, which means direct access to the computer; and (4) *physical*, which describes access to the hardware.

The *Complexity* of a threat is defined by two possible values: *low* and *high*. A high effort is required when a threat agent needs some preparation to realize the threat and that the threat cannot be repeated an arbitrary number of times.

To state whether privileges are required to successful realize the threat, we make use of the corresponding attribute. There are three possible values: (1) *None*; (2) *Low*, e.g. a user account; and (3) *High*, administrative rights.

A threat realization may require some *User Interaction*, for example by confirming the installation of malicious software.

The *Threat Scope* may change when a threat agent uses a component to reach other parts of the software.

The impact on confidentiality, integrity and availability is measured using qualitative scales. The used scale consists of three values: *None*, *Low* and *High*.

The pattern provides two different attributes which are not covered by the CVSS base metrics, but which supports the estimation of the likelihood of the occurrence of a threat.

We first distinguish the *Threat Type* which might be *Accidental* or *Deliberate*. A *Threat Agent* which realizes the threat can be *Human*, *Technical* or *Natural*.

3 RISK ESTIMATION AND EVALUATION METHOD

Figure 1 provides an overview of the structure of the method, which consists of seven steps. Steps that are marked in gray can be carried out automatically. From the second to sixth step, the method is carried out iteratively for all assets. In the following, we describe the different steps in detail.

3.1 Initial Input

To carry out our method, we require the following initial input:

1. **Assets:** Since our method focusses on information security, we consider an asset as a piece of information that shall be protected with regard to confidentiality, integrity or availability. The definition of assets is not within the scope of the paper.
2. **Identified Threats:** In previous steps of the risk management process, we identified threats that might lead to a harm for at least one asset. Those threats are represented as instances of the pattern we introduced in Section 2.

3.2 Step 1: Likelihood Estimation

For each threat that has been identified, it is necessary to estimate its likelihood of occurrence. We define that likelihood by its frequency of occurrence per year.

In contrast to the consequence of a threat, the occurrence of it is independent of any asset. Therefore, we estimate the values in the beginning.

Currently, our method does not support an automated estimation of likelihoods. Therefore, the step requires the expertise of security engineers and domain experts. Nevertheless, the threat description may support the estimation. For example, in companies where computer scientists are working, the like-

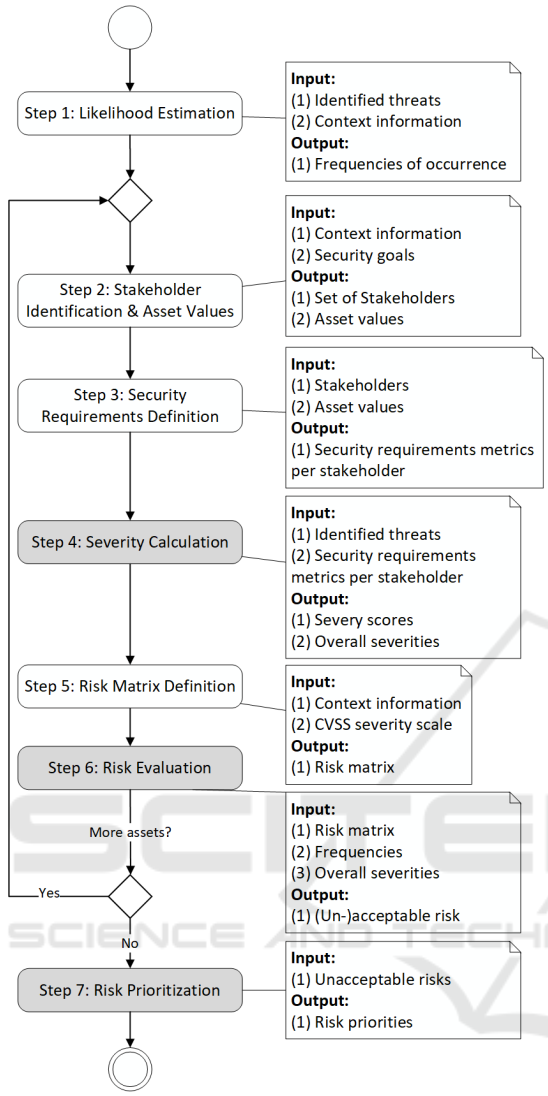

Figure 1: Risk Estimation Method.

lihood of accidental threats for software may be lower than in other companies.

3.3 Step 2: Stakeholder Identification & Asset Values

The specific value of an asset may differ for each stakeholder. For example, harming the integrity of health records may lead to death for a patient, whereas for a hospital, it leads to a loss of reputation. In existing risk management processes, e.g. CORAS (Lund et al., 2010), security engineers define a single point of view to estimate risks. Doing so, consequences for some stakeholders may be omitted, which leads to incomplete risk estimations.

A distinguishing feature of our method is that we make all stakeholders explicit. For each identified stakeholder, we estimate the value of an asset with regard to confidentiality, integrity and availability independently. Those values are defined independently of any threat. We consider three types of stakeholders:

Software Provider. Stakeholder or company that is responsible for the software, e.g. development and maintenance. Since all assets are related to the same piece of software, the software provider is the same for all assets.

Data Owner. Stakeholder to which the asset belongs, e.g. a patient is the data owner for his/her health record. The data owner may also be a company, for example when protecting business information.

Third Parties. Set of other stakeholders for which consequences might exist. We investigate each relevant third party independently of each other.

Using a detailed description of the context, in which the application shall be deployed, we identify data owner and relevant third parties for each asset. For each so identified stakeholder, we estimate the impact for each security property using the same unit, e.g. in terms of monetary impact. The monetary impact can also later be used to evaluate the costs of selected controls.

We use a table as shown in Table 4 to document the stakeholders and the asset values. There is one table per asset. Currently, our method does not require a common format for the context description. Therefore, the step has to be carried out manually. Using context patterns, e.g. (Beckers, 2015), security engineers can be assisted in identifying relevant stakeholders.

3.4 Step 3: Security Requirements Definition

Each threat description states the maximum impact on confidentiality, integrity and availability independently of the concrete context. Therefore, the descriptions also do not consider the specific impact for a stakeholder. In contrast to existing methods, we put a special focus on stakeholders and make them explicit during severity calculation. Therefore, it is necessary to adjust the importance of impact metrics accordingly.

To reweight the importance of impacts, the CVSS contains metrics to define security requirements. The metric is defined as a qualitative scale with the following values: *Not defined*, *Low*, *Medium* and *High*. *Not defined* means that the asset has no value for the

stakeholder. Using those metrics, we take the different stakeholders into account. We define security requirements for each stakeholder to reflect his/her specific value of the asset with regard to confidentiality, integrity or availability.

Since we defined monetary asset values in the previous step, security engineers need to derive the qualitative metric values manually. We make use of a table, such as shown in Table 5, to document the security requirements for each asset. Since the value of an asset does not depend on any threat, we do not need to consider threats in this step.

3.5 Step 4: Severity Calculation

Our method provides an easy and precise way to calculate the severity of a threat with regard to a specific asset and the different stakeholders. We consider the pattern instances of the identified threats and the security requirements metrics of the stakeholder as input.

The severity needs to be calculated for each threat that might harm the asset under investigation. Since the impact differs per stakeholder, we calculate the severity for each stakeholder using the security requirements metrics and the pattern instances. The CVSS defines formulas for that calculation (FIRST.org, 2015), which we will use for that task.

If a security requirement for a security property has been set to *not defined*, it is necessary to adjust the related impact metric. Not defined means that harming the security property will not lead to an value loss for the stakeholder. We then define a modified base metric for that property and set its value to *None*. During the severity calculation, a modified base metric overwrites the base metric provided by the threat description.

The calculation yields a set of severities per threat. Next, we combine the values of the set to derive the overall severity of a threat for an asset. For this, we propose two different approaches: (1) Taking the maximal value of the set or (2) calculating the average of all values. In case that software provider and data owner are the same for the asset, we consider the corresponding severity only once. The corresponding impact for the stakeholder can also happen only once. For later prioritizing risks, the values have to be comparable for all assets. Therefore, the same approach has to be taken for all calculations.

Since we defined the metrics for the calculation in previous steps of our method, the calculation can be automated by implementing the formulas and providing the threat description and metrics in a machine-readable way.

3.6 Step 5: Risk Matrix Definition

When treating risks, it is important to focus on the most important ones. In a first step, it is necessary to define risk levels that are considered as acceptable or unacceptable. Later on, only unacceptable risks need further inspection and hence, only those risks need to be prioritized. We make use of risk matrices to evaluate risks. That kind of matrix has already been used in other risk methods, for example in CORAS (Lund et al., 2010).

Define Scales. Prior to the definition of the risk matrix, it is necessary to define its scales. The CVSS score describes the severity of a threat. The severity is derived from conditions under which a threat can be successfully realized and its corresponding impact on an asset. The second dimension is the likelihood of the occurrence of a threat as mentioned in the first step of the method. For creating a risk matrix, we define intervals for the occurrence which we use to define a qualitative scale.

The likelihood scale is the same for all risk matrices and hence, for all assets. Therefore, it is only necessary to define it once during method execution.

The severity calculation leads to values between 0 and 10. In the CVSS specification document (FIRST.org, 2015), there is an interval-based qualitative scale. It consists of the following values: *None*, *Low*, *Medium*, *High* and *Critical*. We make use of that scale in our risk matrices.

Define Risk Matrices. The acceptance threshold for risk highly depends on the importance of an asset. Therefore, it is necessary to provide a risk matrix for each asset. We annotate the severity scale horizontally and the likelihood scale vertically. For each cell of the matrix, it is necessary to define whether the risk level is acceptable or not. In a graphical representation, we mark acceptable values in green and unacceptable values in red. An example of such a risk matrix is shown in Table 9. Other categories of risks may be added, as well, e.g. for risks that do not need to be treated but which shall be monitored.

The likelihood scale might be reused from other software projects, but the definition of acceptance needs some manual interaction. Therefore, we do consider this step currently as non-automatable.

3.7 Step 6: Risk Evaluation

For each asset, we evaluate the acceptance of identified risks. The risk of a threat for an asset is composed by its corresponding likelihoods (Step 1) and the severity for the asset (Step 4). Using these values, we fill the risk matrix. Those risks which are consid-

ered as unacceptable are prioritized in the next step. Acceptable risks do not need any further investigation, and hence will be omitted.

3.8 Step 7: Risk Prioritization

A well known concept for calculating the risk level is to multiply the likelihood and the severity (Stoneburner et al., 2007). For prioritizing risks, we consider the numerical values for likelihood and severity, which enhances the precision.

We calculate the risk level for all unacceptable risks. The resulting value states the priority of the risk. The higher the value, the higher the priority.

The final step of our method takes all risks into account. The final outcome of our method is therefore a list of all unacceptable risks which are ordered according to their priority. The list ensures that risks can be treated in an effective manner by considering their priority.

Using the calculated values and the results of risk evaluation, the step can be automated.

4 CASE STUDY

To illustrate our risk estimation and evaluation method, we make use of a smart home scenario. We first describe the scenario and the initial input, and then we execute the different steps of our method.

4.1 Scenario & Input

Our scenario is a smart grid which enables the energy supplier to measure a customer's power consumption remotely. The invoices are calculated automatically based on the measured values. The gateway at the customer's home is called *Communication Hub*, for which the software shall be developed. It is the bridge between energy supplier and measuring units. Customers can connect to the communication hub using a mobile app in the local area network to check the invoices or to change their personal data. The invoices are calculated based on the customer's tariff parameters, which are stored at the communication hub, as well as the personal data and the measured values. As assets, we consider the customer's tariff parameters, which shall be protected with regard to integrity, and customer's personal data, which shall be protected with regard to confidentiality.

In the following, we will focus on the functional requirement for changing personal data for which we identified two threats using the method for risk identification as described by (Wirtz and Heisel, 2019).

Table 1: Description of Injection (Wirtz and Heisel, 2019).

Threat Information	
Threat Type	☐ Accidental ☑ Deliberate
Threat Agent	☑ Human ☐ Technical ☐ Natural
Threat Vector	☐ Network ☑ Adjacent ☐ Local ☐ Physical
Complexity	☑ Low ☐ High
Privileges Required	☐ None ☑ Low ☐ High
User Interaction	☑ None ☐ Required
Threat Scope	☐ Unchanged ☑ Changed
Confidentiality Impact	☐ None ☐ Low ☑ High
Integrity Impact	☐ None ☐ Low ☑ High
Availability Impact	☐ None ☐ Low ☑ High

Injection and *Inception* are examples of threats for which we provide pattern instances in Tables 1 and 2. We will use those threats as the initial input of our method.

4.1.1 Injection

For an injection (see Table 1), an attacker may take the role of a user and uses the connection to the gateway via the app to insert malicious database queries and updates. The functional requirement only considers changing the customer's personal data. Since the tariff parameters are stored in the same database, it is possible to harm the integrity of the asset using malicious updates. The threat agent is defined as deliberate and human. Since the app can be used in the local area network, the threat vector is defined as *adjacent*. The complexity of injecting malicious queries is considered as low. The threat agent only needs user privileges to realize the threat, which leads to a low privilege value. There is no additional user interaction and the threat scope is changed, because the threat agent uses the software to manipulate the database. In general, an injection has possibly a high impact for all three security properties.

4.1.2 Interception

The threat description for interception is given in Table 2. An attacker may also intercept the local network connection (adjacent) to disclose transmitted data. The threat is relevant for the functional requirement of our scenario, because it describes the transmission of personal data. Hence, the threat leads to a harm of the confidentiality of personal data. The description of the threat states a human deliberate threat agent. The threat has a high complexity (and requires

Table 2: Description of Interception.

Threat Information	
Threat Type	☐ Accidental ☑ Deliberate
Threat Agent	☑ Human ☐ Technical ☐ Natural
Threat Vector	☐ Network ☑ Adjacent ☐ Local ☐ Physical
Complexity	☐ Low ☑ High
Privileges Required	☐ None ☐ Low ☑ High
User Interaction	☑ None ☐ Required
Threat Scope	☑ Unchanged ☐ Changed
Confidentiality Impact	☐ None ☐ Low ☑ High
Integrity Impact	☑ None ☐ Low ☐ High
Availability Impact	☑ None ☐ Low ☐ High

high privileges. Since the attacker intercepts the connection and only discloses data on that level, the scope remains unchanged. There is a high impact on confidentiality but no impact on integrity or availability.

In the following, we describe the application of our method for each step in detail. The first step is applied for both threats, and the following steps are performed for both assets, tariff parameters and personal data.

4.2 Step 1: Likelihood Estimation

For our example, we estimate the likelihood for the occurrence of the threat injection as *25 times a year*. That is, 25 times a year an attacker tries to inject malicious code to manipulate tariff parameters.

The likelihood that an attacker tries to intercept a local network is considered as *25 times a year*, too.

4.3 Step 2: Stakeholder Identification & Asset Values

The software provider in our scenario is the energy supplier, which is the same for all assets.

Tariff Parameters. The first asset are *tariff parameters*, which shall be protected with regard to integrity. Since tariff parameters are defined by the energy supplier, we consider that stakeholder as the data owner. We define an asset value per customer of 200 €, because invoices are generated automatically based on the tariff parameters, and not each invoice is checked for its correctness by the energy supplier. A lower invoice amount will lead to a loss of money, whereas a higher amount might harm the reputation of the company and also produces effort to correct the incorrect invoices manually. In case that the tariff parameters are not manipulated by customers themselves, we consider customers as a relevant third party. Manipulated tariff parameters lead to an incorrect invoice and may request the customer to pay more money than necessary. Since most customers check their invoices, we estimate an impact only at 50 €. Customers who check their invoices and find errors still have to spend some effort to get it sorted. Table 3 summarizes the results for the asset tariff parameters. *SP* stands for software provider, *DO* for data owner and *TP* for third party. We only define values for integrity, because the scenario only requires that security property.

Table 3: Stakeholders and Values for Tariff Parameters.

Stakeholder	Conf.	Integr.	Avail.
(SP) Energy Supplier	–	200€	–
(DO) Energy Supplier	–	200€	–
(TP) Customer	–	50€	–

Table 4: Stakeholders and Values for Personal Data.

Stakeholder	Conf.	Integr.	Avail.
(SP) Energy Supplier	400€	–	–
(DO) Customer	50€	–	–

Personal Data. The second asset is the personal data of the customer who is the data owner. There are no other third parties. When personal data is disclosed, the software provider may be liable for damages. Therefore, we estimate a value of 400€ per customer. The personal data only consists of the customer's address to provide the invoice, which may also be accessible via the phone book. Therefore, we do not consider address data as highly sensitive information, and we estimate a relatively low value of 50€ for the data owner.

4.4 Step 3: Security Requirements Definition

Tariff Parameters. For the asset tariff parameters, there is no impact on confidentiality and availability for both stakeholder. The security requirement metric for both security properties is set to *not defined (–)*. Using the previously defined asset values, we define the impact on integrity for the energy supplier as *medium* and for customers as *low*. The values are documented in Table 5.

Personal Data. For the second asset, there is only an impact on confidentiality. For the customer, we define the impact as *low*, whereas for the software provider, the impact is *medium*. The results are documented in Table 6.

Table 5: Security Requirements Metrics for Tariff Parameters.

Stakeholder	Conf.	Integr.	Avail.
(SP) Energy Supplier	–	Medium	–
(DO) Energy Supplier	–	Medium	–
(TP) Customer	–	Low	–

Table 6: Security Requirements Metrics for Personal Data.

Stakeholder	Conf.	Integr.	Avail.
(SP) Energy Supplier	Medium	–	–
(DO) Customer	Low	–	–

4.5 Step 4: Severity Calculation

In our example, there is one threat per asset for which we need to calculate its severity.

Tariff Parameters. For the asset *tariff parameters*, we identified the threat *Injection*. The formulas provided by the CVSS specification document (FIRST.org, 2015) are filled with the base metrics contained in the instance of the threat pattern. In the third step of our method, we defined security requirements metrics. Since the metrics for confidentiality and availability have been set to *not defined*, we define corresponding modified base metrics which are set to *none*.

The severity needs to be calculated for the energy supplier and the customer independently. For the calculation, we use a web-based tool[1] which takes the metric values as an input and calculates the severity automatically based on the defined formulas. The results of the calculation are summarized in Table 7. There is one column per threat in which we state the corresponding severity for the energy supplier and the costumer, the maximum severity and the average severity. In our example, software provider and data owner are the same. Therefore, that stakeholder counts only once for calculating the average.

Personal Data. The severity of the threat *Interception* for the asset *personal data* is calculated in the same manner. We state the corresponding results in Table 8. Here, we do not have any third party. The average severity is only calculated based on software provider and data owner.

4.6 Step 5: Risk Matrix Definition

Likelihood Scale. We define a qualitative likelihood scale for the frequency of occurrences per year with the following values: *Never*, *Seldom* (up to 20 times a year), *Frequently* (up to 50 times a year) and *Often* (more than 50 times a year).

[1]https://www.first.org/cvss/calculator/3.0 - CVSS Calculator v3 (last accessed on 26 November 2018)

Table 7: Severity for Tariff Parameters.

Stakeholder	Severity of Injection
(SP) Energy Supplier	6.8
(DO) Energy Supplier	6.8
(TP) Customer	4.5
Average	5.65
Maximum	6.8

Table 8: Severity for Personal Data.

Stakeholder	Severity of Interception
(SP) Energy Supplier	4.2
(DO) Customer	2.4
Average	3.3
Maximum	4.2

Risk Matrix. We define a risk matrix to evaluate whether a risk is acceptable or unacceptable. On the vertical axis, we annotate the previously defined likelihood scale and on the horizontal axis we annotate the CVSS severity score. The resulting matrix is shown in Table 9. Acceptable risks are shown in green and unacceptable risks are shown in red. In the present example, we use the same matrix for both assets.

Table 9: Risk Matrix.

	None 0.0	Low 0.1–3.9	Med. 4.0–6.9	High 7.0–8.9	Critical 9.0–10.0
Never 0 times					
Seldom $\leq$ 20 *times*					
Freq. $\leq$ 50 *times*		$R2_{Avg}$	$R1_{Avg}$ $R1_{Max}$ $R2_{Max}$		
Often $>$ 50 *times*					

Table 10: Calculated Risk Levels.

Risk	Likelihood	Maximal Severity	Risk Level
R1	25	6.8	170
R2	25	4.2	105

4.7 Step 6: Risk Evaluation

To evaluate the risks, we make use of the risk matrix shown in Table 9.

Tariff Parameters. For the asset *tariff parameters*, there is one risk concerning the threat *Injection*. In Table 9, we use **R1** as an abbreviation for the corresponding risk. *Max* indicates the risk level when using the maximal value of all severities, and *avg* indicates the average value. Both approaches lead to an unacceptable risk which is indicated by a red cell.

Personal Data. For the asset *personal data*, there is a risk for the threat *Interception*. In Table 9, we use **R2** as an abbreviation for the corresponding risk. Using the risk matrix, we consider the risk as acceptable for using the average value for the severity, which is indicated by the green cell. The risk is unacceptable for using the maximum of all severities. Hence, further inspection of the threat is necessary for the asset *personal data* when taking the maximum severity.

4.8 Step 7: Risk Prioritization

To prioritize risks, we multiply likelihood and severity for each unacceptable risk. The higher the calculated value, the higher the priority of the risk.

Using the average of the severities, we only identified one unacceptable risk in the sixth step. Therefore, a prioritization is not necessary.

Using the maximum of the severities, there are two risks that need to be prioritized: **R1**, risk of injection for the asset tariff parameters; and **R2**, risk of interception for the asset personal data. The results of the calculation are summarized in Table 10. The risk of injection has a higher level, and hence will have priority during risk treatment.

5 DISCUSSION

Based on the description of our method in Section 3 and the application for the case study in Section 4, we discuss benefits and limitations of our method.

5.1 Usability

Threats that have been identified during risk identification are described in a pattern format. Limiting the effort for security engineers, the pattern allows to calculate the severity without collecting additional information about the threat. For each step, we explicitly state input, output and procedure which assists engineers in applying our method. Additionally, we provide a structured documentation for the outcome of each step in form of tables. We designed our method in such a way that it can be easily integrated into a tool (see future research directions in Section 7). To limit the effort for applying our method, we describe how steps can be automated.

Nevertheless, the security engineers need some specific expertise, for example in estimating the likelihood for a threat. Using the table-based documentation, we aim to assist security engineers in collecting and documenting the results in an effective and structured way.

5.2 Scalability

The complexity of our method mainly depends on the number of assets, threats and identified stakeholders. The complexity of the first step which deals with the likelihood estimation of identified threats cannot be improved. It is always necessary to estimate the likelihood of a threat depending on the concrete context.

Since we identify different stakeholders for estimating the severity of a threat, we increase the complexity of some steps. When omitting the stakeholders, we will improve the scalability of our method but the estimated risk levels will be less precise. Therefore, it is necessary to find a compromise between both limitations. We automated all steps as much as possible to limit the manual effort for engineers to perform those steps. The required calculation is simple enough to ensure a good scalability. The CVSS provides a format to store the values of the different metrics in a vector string. Such a simple format supports the scalability for larger applications, since it is easy to use and does not require much storage or complex calculations.

5.3 Precision

To calculate the severity of a threat with regard to a specific asset, we use the CVSS. The defined metrics are widely accepted by the community and many industrial partners to estimate the severity of vulnerabilities. Based on the threat description pattern (cf. Section 2), we adapted the scoring system to estimate the severity of threats. The corresponding formulas to calculate the score have been defined by security experts based on real vulnerabilities. Although the metrics and formulas have been defined on sound expertise, there are limitations in their precision. The instances of the threat pattern do not consider the concrete context of the application, and the values for the metrics are qualitative. As mentioned above, predefined scales have the benefit of a better usability. We try to address the issue with a context-independent de-

scription by an explicit identification of stakeholders and by adjusting the base metrics.

To evaluate risks, we make use of qualitative scales in risk matrices. Those scales are only used to define intervals for risk acceptance. The scales we use in this paper can be easily replaced by arbitrary ones with a more fine-grained resolution. For prioritizing risks, we use the numerical values of frequency and severity which leads to a higher precision when calculating the priority for a risk.

6 RELATED WORK

To identify related work, we performed a simplified literature review using Scopus[2]. We used the built-in search engine to identify relevant publications that either describe a risk estimation or risk evaluation method. Those methods should be applicable during requirements engineering and should put a special focus on security.

Argyropoulos et al. (Argyropoulos et al., 2018) suggest to use the analytic hierarchy process (AHP) (Saaty, 1988) in the context of security. The AHP method allows to prioritize risks relatively to each other. The approach has a high precision but requires an overhead in terms of effort because all risks need to be compared pair-wise.

(Llansó et al., 2015) considers the level of effort to realize a cyber attack as an important factor to determine the likelihood of the attack. The authors propose a model-based algorithm to estimate such effort. In our approach, we consider the level of effort by some attributes of the CVSS, e.g. threat vector. The proposed algorithm may improve the precision of our method.

Using Bayesian Networks and agent-based simulation, other authors aim to provide a probabilistic approach to support risk analysis (Tundis et al., 2017b; Tundis et al., 2017a). There, a risk level is defined as the percentage of failure for a functionality. Using the mentioned approaches, it is possible to analyze the propagation of risks throughout the system's components. There is no prioritization of risks, but the approaches may extend our method to analyze the dependencies between different risks.

ArgueSecure (Ionita et al., 2017) is a method for argument-based risk assessment that does not rely on any quantitative estimation of risks. The proposed framework relies on a qualitative method that is performed in brainstorming sessions. The results are documented in a tree structure. The proposed graphical notation is designed for an application by non-experts, but there are no explicit risk levels, which makes it hard to evaluate the identified risks. CORAS (Lund et al., 2010) combines a graphical notation for risk identification in brainstorming sessions with a semi-quantitative risk evaluation.

SERA (Abeywardana et al., 2016) is a risk analysis framework with a special focus on social engineering attacks. The importance of human factors is also mentioned in other publications (e.g. (Rajbhandari, 2013)). Currently, neither the CVSS nor our method supports the consideration of social engineering, which is a limitation.

Islam et al. (Islam et al., 2016) propose an attribute-based estimation of risks which is based on the Common Criteria (Common Criteria, 2017). The attributes to define the likelihood are comparable to the CVSS, whereas the impact is not measured with regard to a specific security property. The values for the attributes need to be set manually, whereas we make use of existing pattern-based threat knowledge.

Elahi et al. (Elahi et al., 2010) make use of a qualitative method to analyze goal models. The i*-notation has been extended to model attacks. The authors mention that the qualitative evaluation makes an application easier, but it is less precise.

Another approach is to combine threat trees with Monte Carlo models (Pardue et al., 2009). The risk is defined by a set of parameters, such as complexity and motivation of an attacker. The assigned values are used for a Monte Carlo simulation to estimate risk values. The method does not rely on existing threat knowledge and does not allow to prioritize risks.

Labunets et al. have carried out a study to compare graphical and tabular representations for security risk assessment (Labunets et al., 2017). The results of the study revealed that there is no significant difference between both representations with regard to the perceived efficacy. Our method relies on tabular descriptions for the results and does not contain any graphical notation. Since the study shows the equivalence of both notations, there is no need to add such a notation.

In contrast to our method, none of the mentioned methods makes different stakeholders' perspectives explicit for estimating risks. To the best of our knowledge, it is a novelty in our method.

7 CONCLUSION

In this paper, we proposed a semi-automatic method to estimate and evaluate security risks. Our method

[2]www.scopus.com - Scopus (last accessed on 4 December 2018)

has been designed for an application during requirements engineering, which enables security engineers in focusing on the most severe risks right from the beginning of a software development process. The distinguishing features of our method are:

(1) We make the impact for different stakeholders explicit. The different perspectives improve the precision of the risk estimation.

(2) Our method makes use of pattern instances based on the CVSS for describing identified threats. The pattern format simplifies the risk estimation.

(3) We provide guidance for each step by defining input and output and describing its execution in detail. Since our method is semi-automatic, we reduce the manual effort for security engineers in applying it.

Based on our method, we plan to assist security engineers in selecting and evaluating controls. To do so, we will adapt our method to suggest a combination of controls that provides a sufficient risk reduction. The selection will be based on the risk priorities and the effort for applying a control.

Currently, we only take security for software-based systems into account. In future work, we plan to investigate how our method can improve the evaluation of privacy and safety risks. Depending on the context, we will elaborate whether it is possible to combine the process for security, privacy and safety.

As mentioned in Section 5, we will develop a tool for our method. The tool will be designed in form of a workflow that asks the engineers for inserting the required data, documents the results in a usable way and finally provides a list of risks with the assigned priority.

REFERENCES

Abeywardana, K., Pfluegel, E., and Tunnicliffe, M. (2016). A layered defense mechanism for a social engineering aware perimeter. pages 1054–1062.

Argyropoulos, N., Angelopoulos, K., Mouratidis, H., and Fish, A. (2018). Risk-aware decision support with constrained goal models. *Information and Computer Security*, 26(4):472–490.

Beckers, K. (2015). *Pattern and Security Requirements - Engineering-Based Establishment of Security Standards*. Springer.

Common Criteria (2017). Common Criteria for Information Technology Security Evaluation v3.1. Release 5. Standard.

Elahi, G., Yu, E., and Zannone, N. (2010). A vulnerability-centric requirements engineering framework: analyzing security attacks, countermeasures, and requirements based on vulnerabilities. *Requirements Engineering*, 15(1):41–62.

FIRST.org (2015). Common Vulnerability Scoring System v3.0: Specification Document. https://www.first.org/cvss/cvss-v30-specification-v1.8.pdf.

Ionita, D., Kegel, R., Baltuta, A., and Wieringa, R. (2017). Arguesecure: Out-of-the-box security risk assessment. pages 74–79.

Islam, M. M., Lautenbach, A., Sandberg, C., and Olovsson, T. (2016). A risk assessment framework for automotive embedded systems. In *Proceedings of the 2Nd ACM International Workshop on Cyber-Physical System Security*, CPSS '16, pages 3–14, New York, NY, USA. ACM.

Labunets, K., Massacci, F., and Paci, F. (2017). On the equivalence between graphical and tabular representations for security risk assessment. *Lecture Notes in Computer Science*, 10153 LNCS:191–208.

Llansó, T., Dwivedi, A., and Smeltzer, M. (2015). An approach for estimating cyber attack level of effort. *2015 Annual IEEE Systems Conference (SysCon) Proceedings*, pages 14–19.

Lund, M. S., Solhaug, B., and Stølen, K. (2010). *Model-Driven Risk Analysis. The CORAS Approach*. Springer.

Pardue, H., Landry, J., and Yasinsac, A. (2009). A risk assessment model for voting systems using threat trees and monte carlo simulation. In *2009 First International Workshop on Requirements Engineering for e-Voting Systems*, pages 55–60.

Rajbhandari, L. (2013). Consideration of opportunity and human factor: Required paradigm shift for information security risk management. In *2013 European Intelligence and Security Informatics Conference*, pages 147–150.

Saaty, T. L. (1988). What is the analytic hierarchy process? In Mitra, G., Greenberg, H. J., Lootsma, F. A., Rijkaert, M. J., and Zimmermann, H. J., editors, *Mathematical Models for Decision Support*, pages 109–121, Berlin, Heidelberg. Springer Berlin Heidelberg.

Stonerburner, G., Goguen, A., and Feringe, A. (2007). Risk management guide for information technology systems, 2002 (nist special publication 800-30).

Tundis, A., Mühlhäuser, M., Gallo, T., Garro, A., Saccá, D., Citrigno, S., and Graziano, S. (2017a). Systemic risk analysis through se methods and techniques. volume 2010, pages 101–104. cited By 0.

Tundis, A., Mühlhäuser, M., Garro, A., Gallo, T., Saccá, D., Citrigno, S., and Graziano, S. (2017b). Systemic risk modeling & evaluation through simulation & bayesian networks. volume Part F130521. cited By 0.

Wirtz, R. and Heisel, M. (2019). A systematic method to describe and identify security threats based on functional requirements. In Zemmari, A., Mosbah, M., Cuppens-Boulahia, N., and Cuppens, F., editors, *Risks and Security of Internet and Systems*, pages 205–221, Cham. Springer International Publishing.