A risk-centric defensive architecture for threat modelling in e-government application

Maheshwari Venkatasen* and Prasanna Mani

School of Information Technology and Engineering, Vellore Institute of Technology University, Vellore, Tamilnadu, 632014, India
Email: maheshwarivenkat24@gmail.com
Email: prasanna.m@vit.ac.in
*Corresponding author

Abstract: To improve the security of an e-government, software engineering plays a vital role. During the application development for an e-government, there exist several risks. To analyse those risks, threat modelling methodology which is defined as the process to understand and address the threats of an application. Threat modelling is used to determine security controls and countermeasures for the targeting attacks. This paper describes an approach to identify how far the attack penetrates in risk layers and how the model defends from an attacker in e-government systems. The relevant attacks are retrieved from the attack pattern information is gathered from MITRE’s common attack pattern enumeration and classification (CAPEC) security source. This architecture dynamically identifies the risk severity and prioritises the risk in a single step. An attack pattern applied to a risk-centric defensive architecture model to identify threat severity and also it is prioritised based on its impact. We validate risk-centric defensive architecture model by implementing it in a tool based on data flow diagrams (DFDs), from the Microsoft security development methodology.

Keywords: threat model; Microsoft STRIDE; attack pattern; CAPEC; common attack pattern enumeration and classification; SDLC; software development life cycle; e-government.


Biographical notes: Maheshwari Venkatasen is an MS (by research) candidate at the Vellore Institute of Technology University. She is currently working on threat modelling, security attacks, and software testing.

Prasanna Mani is an Associate Professor at the School of Information Technology and Engineering Department of Vellore Institute of Technology University. His research area of interest includes software testing, object-oriented analysis and design and software engineering.
1 Introduction

Due to the rapid changes of information and communication technologies (ICTs), an electronic government (e-government) becomes more in evolution but less satisfactory in its permanent security, because of the rise in new threats and vulnerabilities. Electronic government is defined as providing e-services to the citizen with the use of electronic communication devices, are embedded with internet enabled applications for digital interactions which are potential in anywhere, anytime and anyplace.

1.1 Secure software development

Many software engineering approaches efficiently used for design and development of software that includes waterfall model, agile model, spiral model and iterative model. But security is not to consider in above-mentioned approaches, so special considerations have to take place for security from beginning phase of the life cycle to make secure software engineering. Due to increases in the growth of internet and network, software is exploited by vulnerabilities and threats in software systems. Developers should be updated in security requirements iteratively and to identify security threats. Most of the organisations process their information system, personal data, and privacy data using software system. But it was stolen by the hackers because of vulnerabilities and threats in a software system process. Therefore, software security is a major issue and it has to be practised in the early stage of software life cycle (SLC).

The main actions to be taken in secure SLC model are security requirement and analysis, security design and implementation, and security testing and deployment.

Security requirements and analysis

The requirements are collected from the clients through a business analyst. It was mainly categorised into functional security requirement, non-functional security requirement, and derived security requirement.

Security design and implementation

In this phase, a threat model is designed by the developer with the details of requirement analysis. A threat model is a technique which identifies threats, vulnerabilities, and countermeasures. With the detailed information of security requirement and data flow diagram (DFD), it will be easy to trace the entry and exit points of all possible threats that an attacker can exploit these points. Attack trees methodology describes the clear sketch of possible attacks on the software system. In Implementation security risk and its countermeasures are listed.

Security testing and deployment

Tester role is to identify the perspective view of the attacker in a different way and also functional testing, risk-based testing has to be done to check whether software requirements are met or not. Testing techniques such as penetration testing, fuzzy testing identifies the vulnerabilities in a software application.
1.2 What, why threat model?

What is a threat model?

Threat modelling is the art of identifying, assessing and mitigating risk. Secure software is developed only by thinking about security which focused on about threats against software and attackers goal.

Why threat model?

Rather than generating threat models to all components in the system, threat models can be created for a whole application (Maheshwari and Prasanna, 2015, 2016). Previously software engineering has two major parts to the design of a program i.e., functional specification and design specification. Apart from that, testing specification has been focused on the Microsoft organisation team which is really more difficult to test the components. It results that threat model is another major specification to identify threats in the program. The threat model is a key part of SD³ i.e., secure by design, secure by default and secure by deployment. It analyses the components of the program and it sees how it will react to an attacker. The threat model has been forced to design the program based on the attacker’s point of view, a large set of failures cases, snooping communication, corrupting keys and files etc.

For example, if we want to build a threat model, the developer has to concentrate on assets, entry points, trust boundaries, trust levels, data flow.

1.3 Steps process to ensure security: threat modelling (TM)

Threat modelling begins with the definition of the objectives and the formulation of specific targets to be achieved in application security design. It is essentially a repeatable process to find and address the threats, such as to identify assets, create an architecture overview, decompose the application, identify the threats, document the threats, and rate the threats. It is one of the best effective methods to deal with facts and solutions as they find a way to solve security problems.

In general, the following steps are involved will be involved in security design.

Step 1: Creating threat model for a system

A developer needs to understand the system, where to identify key processes and identify threats to those processes. A DFD is drawn which includes an external entity, process, multi-process, data store, data flow and trust boundaries. They need to understand the view of the system as an enemy, distinguish the system and find out the threats.

Step 2: Identify threats

The threat agents are interested in targeting an application especially on its data, assets, attack methods, attack surface exposed data flows (input and output misuse), authentication, authorisation, cryptography, open data, logging data etc. Some methods such as attack trees, threat libraries, the checklist includes OWASP application security verification standard (ASVS), OWASP proactive controls 2016, use case or misuse case, Cornucopia, STRIDE (Microsoft SDL threat modelling), process for attack simulation and threat analysis (PASTA) are involved in making threats visible easier.
The functional identification of threats is input and data validation, authentication, authorisation, configuration management, sensitive data, session management, cryptography, parameter manipulation, exception management, auditing, and logging.

**Step 3: Determine mitigation and risk**

The mitigation measures are done by threat attack tree where security risk is calculated using DREAD model. The scale contains rate from 1 to 10, where 1 is the least damage potential mentioned to each threat category. The final result is generated as a high, low, and medium rating. Where risk management is calculated by Bug bar (critical, important, moderate, low), Fair (factor analysis information risk) and risk rating (high, medium, low). The main two factors of risk are ease of exploitation and business impact.

**Step 4: Threat modelling report**

The threat reports are documented which includes threats, threat tree, vulnerabilities, and mitigation made in the aforementioned steps for identifying vulnerable areas of the system.

### 1.4 Threat models in e-government system

Information security is the significant factor of e-government in ICTs. In software development life cycle (SDLC) design phase, the security vulnerabilities are identified by establishing the design requirements, analysing attack surface and by using threat modelling. The majority of existing threat modelling work focuses mainly on the attack trees. Attacks are more common in the organisation; software developers should have knowledge of current situation and trends to defend a system from attackers. The most critical risk in e-government systems is to prioritise testing activities. For example, the most critical risk has been witnessed in safety systems, security systems, and military operations. Many kinds of threats occur due to malfunction, insider threats, outside threat agents and hackers that cause harm like a physical loss, financial loss, reputation or brand of an organisation. The key factors of threats are an environmental factor, attack motives, vulnerabilities and targets in business information. On the other hand, a risk analysis, which ensures that risk re-assessment is made to all phases of the test process and reduces the risk of the software system. Security threats of a software component are deeply analysed by the tester according to attacker’s perspective (Anwer et al., 2017). Various kinds of threat types have been described for each phase of SDLC, which includes training, requirements, design, implementation, code, quality assurance, and production. Many effective threat modelling techniques have come into existence like STRIDE, DREAD, OWASP, and CLASP and some of them are effective in finding certain types of faults.

One of the applications of e-government is radio frequency identification (RFID) based E-toll plaza management system. It is used to collect the payment of the vehicles passes through the tolls electronically using RFID tags. Since payments are made electronically through the internet, the toll systems are vulnerable to threats. Here threat model plays a major role in identifying the vulnerabilities and security threats of RFID tags.

In this paper, we address the problem of the security software design models required for e-toll plaza management systems. A risk-centric defensive architecture model describes the prioritisation for the threats and analyses how far the risk penetrates into the
risk levels in critical based testing applications. The contribution of this paper represents are as follows:

- the finding of risk-centric defensive architecture model has been achieved by using severity stages of risk.
- risk-centric defensive model has applied in real time case study such as RFID technology to show that this method can be widely applied to all kinds of e-government applications to identify the attacks.

The rest of this paper is organised such that, Section 2 describes related work, Section 3 defines the methodology of risk-centric defensive architecture model represents risk severity and priorities. An experimental result is shown in Section 4. In Section 5 and 6, the experimental result of proposed method and comparison of Microsoft threat modelling STRIDE was discussed and finally, Section 7 draws a conclusion.

2 Related work

Security is a process; it integrates with SDLC to increase the growth of security models in a critical web application. The most known methodologies for integrating security within SDLC are Microsoft development life cycle (MS DLC), the building security in maturity model (BSIMM) and the OWASP software assurance maturity model (SAMM) (Scandariato et al., 2015). These methodologies are criticised because of their respective time and resource consumptions. Nowadays many web applications are complex systems and a point of security failure exists because of integrating a number of technologies in a system. So web application prioritisation is necessary to overcome this problem. The software is modelled by understanding threat agent, asset, attack, attack surface, attack goal and security control. Most security problems are caused by software flaws and bugs.

“Threat modelling at the design phase is really the only way to bake security into the SDLC.” (Sportsman, 2011)

An approach to threat modelling is assets-centric, attacker-centric and software-centric. The mitigation is discovered for defined threats but some threats may be hidden if they are not defined in a model. To enhance security and quality, OWASP risk modelling, Microsoft threat modelling process, Trike, AS/NZS 4360:2004, Risk management, CVSS, and OCTAVE are critically important. STRIDE threat modelling is a structured approach to identify, quantify and address the security risk associated with the software. The threat modelling process can be decomposed into three steps: Decompose the application, determine threats according to those rank threats and determine countermeasures and mitigation (UcedaVelez and Morana, 2015; Scandariato et al., 2015). The author (Scandariato et al., 2015) presented a descriptive study evaluating threat modelling Stride about valid threats, invalid threats, and overlooked threats. The weakness in security requirement is identified but the quantity of cost and effectiveness has not been carried out yet. The experiment carried by Sindre and Opdahl (2005) and Opdahl and Sindre (2009) did not assess the correctness of produced threats. They compared misuse case to attacks trees with out-of-box-thinking and by security experts but there is no effective difference that is identified. Karpati et al. (2012) compared
misuse case with misuse maps and considerably identified more mitigation. Meland et al. (2010) proposed to structure the analysis process by adopting a checklist. According to Stride category, they conducted an experiment for efficiency with two reusable threat models i.e., misuse case diagram and misuse case stubs. But the author failed in his experiment because there is no difference in discovering a number of threats. Kong et al. (2010) have proposed a threat driven approach for securing the functional model. The integration of attack trees and state charts helps to analyse the behaviour of the model. Ray and Mohapatra (2013) proposed state-based risk assessment methodology, initially, the risk is proposed for component scenario and finally grouped together for risk assessment. This approach achieves test efficiency more when compared to component based risk assessment approach. The risk is estimated based on SCOTEM (State collaboration test model) and transition probabilities. In future, the risk estimation can be carried out for requirement stage. Xu et al. (2012) provide a threat model which is built for the real-world system i.e., Magento and FileZilla server by Predicate/Transition nets and also by STRIDE tool for examining the security risk using test code. The vulnerability in the system is detected by creating the mutants for web applications. Few number of security mutants are killed because of the less number of scanner and analyser being used for testing. The advantage of security test is reproducible and repeatable. In the future, security test to be implemented in dynamic testing has to remain the same. Masood and Java (2015) discusses OWASP top most ten vulnerabilities for web services and discovered the vulnerabilities by static code analysis techniques. Most public facing vulnerabilities are shell-shocked and heart bled. The author proposes an adapter that is implemented in web services like SOAP and REST to improve the understanding of attacks and vulnerabilities in security development lifecycles.

The architectural model provides to abstract and captures the different aspects of information system in diagrammatic descriptions. Threat behaviours are modelled with an architectural model to driven the security of the system. A UML sequence diagram describes an interaction behaviour, realisation, sequence of the message, objects, classes, components and subsystem of a threat scenario. First, the developer should have knowledge in security policy and potential threats to build a threat model and also to know attacker view, how to attempt the threat model (Wang et al., 2007). The effectiveness and performance of middleware architecture and frameworks for e-government is discussed (Mecca et al., 2016). An architecture metamodels and security risk can be inferred from metamodel instantiations. From instantiations, the probabilistic dependency model is generated to calculate the risk associated with these metamodel instantiations. This model contains classes, attributes and class relationship (Sommestad et al., 2010). A graphical descriptor provides for assessing security risk in the model. The disadvantage of CORAS doesn’t provide data needed to quantify security risk or risk based on the model.

2.1 Software web application design modelling

There are several approaches used to represent security at the design phase. Our risk-centric defensive architecture model is aimed at automating the widely used threat modelling step process of identification of threats and risk ranking which uses DFDs to
represent software architecture. Web Server is the lowest level that defines the specifications of the web application. The browser gives a request to a web server which considered web applications consist of providing secure configuration, handling expectations, validating input, and authorising users. Application Server manages auditing and logging activity, transactions, authenticating and authorisation upstream identities. Database Server Protects and encrypts sensitive data. The general threats and countermeasure of this layer include, Web server and application server serves the main role in securing application such an input validation, authentication, authorisation, configuration management, sensitive data, session management, cryptography, parameter manipulation, exception management, auditing, and logging. The security functions provide for securing the host, such as update patches and services, protocols, accounts, files and directories, ports, registry, auditing, and logging. In a database server, the main three components: router, firewall, and switch help to secure a network (Meier et al., 2007).

2.2 Types of threats

The common threats available in today’s environment are web threats, cloud threats, cloud provider threats, and mobile threats.

Web threats

A website specific attack particularly happens to cause harm in SQL forms and JavaScript URL. In order to mitigate the threats some patterns, libraries, framework, testing tools are used to find unique threats which pose vulnerable to the website. A standard form of DFD is drawn by understanding the requirements from the human user. The best way to analyse the threats is to ensure that dependencies security issues are met and also DFD is to be done for both Server and client side.

Web browser threats

To identify threats in a web browser, one should be expert in security issues, security goals and to ensure how those securities help in detecting threats. The most deployed plugin in a web browser is Java and Flash which are the most on-going security problems. Building block STRIDE is applied to a web browser to find threats. A browser security model and privacy model element help in creating a browser plugin.

Cloud threats

The internal threat is by developers, cloud admins, users where external threats are done by hackers, governments i.e., Stuxnet, Operation Aurora. Most of the threats occur in Cloud service provider, internet/psn, Software-as-a-service (Saas), Platform-as-a-service (Paas), Infrastructure-as-a-service (Iaas), and Testing-as-a-service (Taas).

Mobile threats

Security vulnerabilities, data exposure at rest, data exposure to other apps, data exposure in transit, unauthorised data collection, mobile malware propagation, mobile ‘Tap and Pay’ attacks, Ransomware, Botnet call home, Hacking as a service, Adware, Trojan-SMS etc.
To perform the modelling, our implementation is based on the common attack pattern enumeration and classification (CAPEC) information and required a few specific additions to the original risk-centric layers by placing attributes of attacks. Attributes are nothing; it is a resource of an attack is distributed in the layers based on the CAPEC attack pattern severity. For example, a buffer flow attack pattern might be valued as high. This is based on the possible harm that would result in the resource compromised.

3 Case study

One of the characteristics of e-government systems is automation of e-government services. In this criterion, RFID plays a major role by placing it in an object and collecting information from it. RFID technology is widely used in all environments since it is low cost and high reliability. In the Automated e-toll plaza management system, RFID technology plays a high impact by reducing the traffic in tollgate of the highway. The RFID tag and RFID reader are contained in RFID technology. RFID tag is placed on vehicle and RFID reader is placed in tollgate. If a vehicle passes through the toll gate the RFID module in the reader receives the signal from RFID tag and passes the data to the system. RFID tags and RFID readers work in an unsafe noisy environment. So RFID network is highly vulnerable and contains a high range of malicious attacks.

The threat model is an effective method to manage attacks of RFID in e-toll plaza management system. The possible attacks in RFID networks: relay attacks, cloning, spoofing, impersonation, eavesdropping, buffer overflow, malicious code injection, denial of service (DOS), crypto attacks, desynchronisation attacks, forward secrecy.

4 Methodology

The aim of our model is to automate the threat model process of threat identification and ranking. As shown in Figure 1, the software developers identify the threats during requirements and design phase and it describes how to automate the threat process throughout the life cycle. The risk-centric defensive model has 7 layers, 3 inner layers are strong layers which are named as $L_1$, $L_2$, $L_3$ and 4 outer layers are weak layers which are named as $L_4$, $L_5$, $L_6$, $L_7$. The strong and weak layers are considered based on the components used for e-toll plaza management systems. For our case study, the security architecture is framed as follows: the strong layers contain the components like a Database server, web server, and Application server. In this, the database server is high impact component, so it is placed in the innermost layer $L_1$. SQL query and response action takes place. The application server is placed in the continuous next layer $L_2$, which holds IBM web sphere, BEA Web logic, Servlets, EJB. In strong zone $L_3$ layer, the web server component is placed which contains IIS windows server, Apache HTTP Server, PHP/JSP/ASP. The weak zone layers are $L_4$, $L_5$, $L_6$, and $L_7$. Layer $L_4$ holds technologies, Layer $L_5$ contains a web browser, $L_6$ represents architecture and $L_7$ is the user or e-toll plaza management systems.

From CAPEC security source, the severity value of web attacks is placed in the hierarchical layer has high, low and medium. The MITRE’s CAPEC and CWE (common weakness enumeration) information are integrated into the risk-centric defensive model to find efficient threat attacks and weakness in security features. Firstly, the model
accepts relevant network traffic attacks of RFID Technology and set as input and extracts the attack type of each input. An attack class is generated and split into four categories as DOS, probe, the user to root (U2R) and root to local (R2L). The main benefit of this approach is to defence attack in depth and to find the maximum effectiveness of attacks. It will rank the attack types by the relative level of attack attributes associated with each attack. The final ranking of risk is done through a risk-centric defensive model approach. The following algorithm describes the risk-centric defensive model approach for risk prioritisation:

Step 1: Every attack type request for a resource with a determined priority and also a priority of every attack type is compared with each other attack types. The last four outer layers are weaker zones and the innermost three layers are strong.

Step 2: The dynamic priorities are calculated during the penetration of attack types in the innermost core level layers.

- Calculate best resources for each attack and place in the layer according to its CAPEC attack pattern (AP) information for prioritisation of risk.
- Determine the resource (R) that is adequate to achieve its attacks such as DOS, Probe, U2R, and R2L. If there is no resource to achieve the target of the attack, then penetration is excluded from the layers for next resources i.e., the resources are killed.
- Based on the risk-centric defensive model, the resources (R) rearrange themselves.

Step 3: Allocate the calculated resources to the attack types following the order of the priority list. Most common resources are in the outer layer and less common resources are in the inner layer.

Step 4: The layer moves in a clockwise direction, the resource also should move in the same direction but not in the opposite direction. So the head off attacks can be minimised i.e., the number of resources in the layer will be less. The attacker has to repeat the attack against less experienced attributes now so that it can move to the next layer.

Step 5: Assign each attack type and finally receive a risk prioritisation ranking from low, medium and high.

4.1 Algorithm: generating risk priority from risk-centric-architecture model design approach

Input: An attack type of denial of service, network probe, remote to local and user to root is given as input to implement the prioritisation of attacks in an e-toll plaza management systems.

Output: Risk architecture-centric priorities are calculated for dynamically changing attributes in e-toll plaza management systems.

Declare: \( A = \{A_1, A_2, A_3, A_4 \ldots A_n\} \) be a set of attack types that request a resource for the security flaws in a web application; \( R = \{R_1, R_2, R_3 \ldots R_n\} \) be a set of resources (i.e., attributes) available at web application; Risk layers \( L = \{L_1, L_2, L_3, L_4, L_5, L_6, L_7\} \); Flag = true or false, Priority \( P_i \) = high, low, medium where high = \( L_7, L_6, L_5 \); medium = \( L_3, L_4 \); low = \( L_1, L_2 \); Attack pattern info = \( R_{Des} > R_{Ass} \).
1. Begin
2. Attack type \( (A) \) ← node of initial marking in \( L_7 \)
3. Attack Queue ← \{Attack type\}
4. While Attack Queue ≠ \( \emptyset \)
5. Current Layer ← First Attack type in Attack Queue;
6. For each attack type \( A_i \in A \), do
7. If \( A_i < \text{Attack pattern info} \)
8. Create resources accordingly in risk layers \( L_4, L_5, L_6 \) and \( L_7 \) i.e., Weak zone
9. Else
10. Place resources in \( L_1, L_2, \) and \( L_3 \) i.e., Strong zone
11. End if
12. End
//the dynamic priorities are calculated during the penetration of attack types in the innermost core level layers.
13. Begin
14. Front = rear = -1;
15. If Attack pattern then
16. Label X:
17. While \( A_i < \text{Attack pattern} \) do
18. If \( R_i == L_i \)
19. Place \( R_i \) in \( L_i \).
20. \( P_i = L_i \)
21. \( Attack = A_i \).
22. Else
23. Go to X//the attacker has to repeat the attack against less experienced attributes now so that it can move to the next layer.
24. While A
25. End if
26. End
//based on the risk-centric defensive model, the resources \( (R) \) rearrange themselves.
27. While
28. \( j < L.length \)
29. Do
30. Index = index of \((L, R[j])\);
31. \( L = \text{remove} (L, \text{index}) \);
32. While
33. \( i < L.length \);
34. \( L[i]. \text{getcalltype} == \text{null} \);
35. \{ 
36. Flag = false;
37. Else
38. Return flag;
39. \} 
40. Return true;
41. do
42. result \[i\] = \( L[i-1] \);
43. End While
44. Result \[0\] = \( R[j] \);
45. \( L = \text{result} \);
46. End While
5 Experimental result

In this section, we implemented our model in web-based RFID Technology for specifying what network traffic attack is best to mitigate. The resources are identified in the web application by typical execution flow of components. Random network traffic consists of checking the web application to see how many resources are currently available to detect the attacks. These resource details are fetched from CAPEC source (Enumeration, 2017) and implemented in risk-centric defensive architecture model in order to find out the risk priority for e-government systems.

A typical execution flow for a web application system and their web components were explained. By using these components the identical resources are identified. From the identified resource, the possible attacks categories and attack types are fetched from the CAPEC security source information. The main components are a web browser, web server, servlet, and database. Then apply attack types in the risk-centric defensive architecture model for the penetration of security attacks in the risk layers. Once the attack has been identified in the layers, we can define risk priority to mitigate these attacks as shown in Figure 2. After identifying attacks, prioritise the security attacks considering various resources affected in our model as discussed in Table 1.
A risk-centric defensive architecture for threat modelling

Figure 2 Identification of risk priority according to risk architecture-centric model (see online version for colours)

Table 1 Risk priority for the toll plaza-RFID technology by risk-centric defensive architecture model result

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Attack class</th>
<th>Attack type</th>
<th>Risk priority</th>
<th>Zone type</th>
<th>Risk attack layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DOS</td>
<td>Land</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_3$</td>
</tr>
<tr>
<td>2</td>
<td>DOS</td>
<td>Back</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_2$</td>
</tr>
<tr>
<td>3</td>
<td>DOS</td>
<td>Neptune</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_2$</td>
</tr>
<tr>
<td>4</td>
<td>DOS</td>
<td>Pod</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_3$</td>
</tr>
<tr>
<td>5</td>
<td>DOS</td>
<td>Smurf</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_3$</td>
</tr>
<tr>
<td>6</td>
<td>DOS</td>
<td>Teardrop</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_3$</td>
</tr>
<tr>
<td>7</td>
<td>DOS</td>
<td>Apache2</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_4$</td>
</tr>
<tr>
<td>8</td>
<td>DOS</td>
<td>Udpstorm</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_4$</td>
</tr>
<tr>
<td>9</td>
<td>DOS</td>
<td>Processtable</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_5$</td>
</tr>
<tr>
<td>10</td>
<td>DOS</td>
<td>Worm</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_5$</td>
</tr>
<tr>
<td>11</td>
<td>Network probe</td>
<td>Satan</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_3$</td>
</tr>
<tr>
<td>12</td>
<td>Network probe</td>
<td>Ipsweep</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_3$</td>
</tr>
<tr>
<td>13</td>
<td>Network probe</td>
<td>Nmap</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_3$</td>
</tr>
<tr>
<td>14</td>
<td>Network probe</td>
<td>Portsweep</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_3$</td>
</tr>
<tr>
<td>15</td>
<td>Network probe</td>
<td>Mscan</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_5$</td>
</tr>
</tbody>
</table>
Table 1  Risk priority for the toll plaza-RFID technology by risk-centric architecture model result (continued)

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Attack class</th>
<th>Attack type</th>
<th>Risk priority</th>
<th>Zone type</th>
<th>Risk attack layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>Network probe</td>
<td>Saint</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_5$</td>
</tr>
<tr>
<td>17</td>
<td>R2L</td>
<td>Guesspw</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_5$</td>
</tr>
<tr>
<td>18</td>
<td>R2L</td>
<td>Ftpwriter</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_2$</td>
</tr>
<tr>
<td>19</td>
<td>R2L</td>
<td>Imap</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_2$</td>
</tr>
<tr>
<td>20</td>
<td>R2L</td>
<td>Phf</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_4$</td>
</tr>
<tr>
<td>21</td>
<td>R2L</td>
<td>Multihop</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_4$</td>
</tr>
<tr>
<td>22</td>
<td>R2L</td>
<td>Warezmaster</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_2$</td>
</tr>
<tr>
<td>23</td>
<td>R2L</td>
<td>Warezclient</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_2$</td>
</tr>
<tr>
<td>24</td>
<td>R2L</td>
<td>Spy</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_2$</td>
</tr>
<tr>
<td>25</td>
<td>R2L</td>
<td>Xeock</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_5$</td>
</tr>
<tr>
<td>26</td>
<td>R2L</td>
<td>Xsnoop</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_5$</td>
</tr>
<tr>
<td>27</td>
<td>R2L</td>
<td>Snmpgen</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_5$</td>
</tr>
<tr>
<td>28</td>
<td>R2L</td>
<td>Snmpgetattack</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_5$</td>
</tr>
<tr>
<td>29</td>
<td>R2L</td>
<td>Http tunnel</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_3$</td>
</tr>
<tr>
<td>30</td>
<td>R2L</td>
<td>Sendmail</td>
<td>High</td>
<td>Strong</td>
<td>$L_7 - L_3$</td>
</tr>
<tr>
<td>31</td>
<td>R2L</td>
<td>Named</td>
<td>Medium</td>
<td>Weak</td>
<td>$L_7 - L_3$</td>
</tr>
<tr>
<td>32</td>
<td>U2R</td>
<td>Buffer overflow</td>
<td>Low</td>
<td>Weak</td>
<td>$L_7 - L_6$</td>
</tr>
<tr>
<td>33</td>
<td>U2R</td>
<td>Load module</td>
<td>Low</td>
<td>Weak</td>
<td>$L_7$</td>
</tr>
<tr>
<td>34</td>
<td>U2R</td>
<td>Root kit</td>
<td>Low</td>
<td>Weak</td>
<td>$L_7$</td>
</tr>
<tr>
<td>35</td>
<td>U2R</td>
<td>Perl</td>
<td>Low</td>
<td>Weak</td>
<td>$L_7 - L_6$</td>
</tr>
<tr>
<td>36</td>
<td>U2R</td>
<td>S2lattack</td>
<td>Low</td>
<td>Weak</td>
<td>$L_7 - L_6$</td>
</tr>
<tr>
<td>37</td>
<td>U2R</td>
<td>Xterm</td>
<td>Low</td>
<td>Weak</td>
<td>$L_7$</td>
</tr>
<tr>
<td>38</td>
<td>U2R</td>
<td>Ps</td>
<td>Low</td>
<td>Weak</td>
<td>$L_7$</td>
</tr>
</tbody>
</table>

6  Comparison with Microsoft threat modelling – STRIDE

The RFID technology in e-toll plaza management system is modelled by means of DFD through data flows, external entities, processing nodes and data storage points. Next, the DFD elements are mapped to the threat categories. In STRIDE, the threats are structured according to six categories: Spoofing (S), Tampering (T), Repudiation (R), Information disclosure (I), Denial of service (D), Elevation of privilege (E). Each element type of the DFD is vulnerable to one or more of the threat categories. The threats are drawn out from the checklist and finally, the document is prepared for threats which describe prevention of misuse cases to identify the security requirements. STRIDE has provided a good coverage of threat types and the system functions but the threat models may not represent all potential security attacks. Our novel approach risk-centric defensive architecture model identifies maximum-security attacks and risk prioritisation is effective which is
shown in Figure 3. The obtained result is compared with STRIDE threat modelling tool to show testing effectiveness from the case study and also evaluates vulnerability detection as a risk priority, which is shown in Figure 4.

**Figure 3**  Risk-centric defensive architecture model priority for attack types (see online version for colours)

**Figure 4**  Threat modelling STRIDE-priority (see online version for colours)
7 Conclusion

The work presented in this paper sketches a risk-centric defensive architecture model for the risk prioritisation and security of e-toll plaza management systems. A risk-centric defensive architecture model is based on defence formation in depth and supports several classes of attack for maximum effectiveness. The attack can be originated from different resources ex. Attack relates to the resources of the e-toll plaza management systems security testing instance is a part of or attack related to the denial of service on which the instance is running. Attacks of different classes are layered on the top of each other using CAPEC information resource, making it possible to add or remove attack layers at runtime. We believe, that the described idea offers a quite sophisticated approach to architecture design and security of e-toll plaza management systems. Our model is compared with STRIDE threat modelling tool and analyses the risk priority. The defined threats prioritisation differs and provides maximum effectiveness in the risk-centric defensive architecture model. Future works include further evaluating countermeasure for attacks and also to test in all kind of environment.

References


A risk-centric defensive architecture for threat modelling


