Securing Software Systems - A Survey

Yong Weixiong [1], Kohei Dozono [1], Robin Lee [1], Alvin Kon Soon Seng [1], and Fatima tuz Zahra [2]

[1]Affiliation not available
[2]Taylors University

October 30, 2023

Abstract

This paper aims to discuss the standard guidelines of the development process of secure software and will give justification on different types and ways of the software development processes. Additionally, a survey is conducted, the aim of which is to observe user behavior towards software system usage, user attitude in terms of privacy and policy awareness, security and privacy concerns. This is followed by discussion on how to secure software systems in development stage.

Securing Software Systems: A Survey

1st Yong Weixiong
School of Computer Science & Engineering
Taylor's University
Selangor, Malaysia
yongweixiong000321@gmail.com

2nd Kohei Dozono
School of Computer Science & Engineering
Taylor's University
Selangor, Malaysia
koheidozono01@gmail.com

3rd Robin Lee
School of Computer Science & Engineering
Taylor's University
Selangor, Malaysia
leerobin22@gmail.com

4th Alvin Kon Soon Seng
School of Computer Science & Engineering
Taylor's University
Selangor, Malaysia
ahkonkon@gmail.com

5th Fatima-tuz-Zahra
School of Computer Science & Engineering
Taylor's University
Selangor, Malaysia
fatemah.tuz.zahra@gmail.com

Abstract—**This paper aims to discuss the standard guidelines of the development process of secure software by researching related available professional research papers, which will mainly give justification on different types and ways of the software development processes. Besides, the importance of secure software, as well as the risks of not following the security procedures, will also be covered in this paper. It is a fact that the technology industry is a boom in the world today, and known as one of the drastic growing industries. The technology involves both hardware and software; if one of them becomes vulnerable, the entire system will go down. Attackers aim to exploit software in order to abduct important or profitable data, cause ransom or threat to industries, officials, or a person. Despite the fact that there are many methods for attackers to exploit the system, a single successful attack will cause the company to lose almost everything, therefore, it is essential for the software developers to do whatever they can to keep the software away from the attackers. Hence, this paper aims to review various software development procedures and security risks along with surveys conducted to observe people behaviour in terms of software system usage.**

Keywords— Software system modelling, secure software systems, SDLC, Scrum, threat modelling

I. INTRODUCTION

Computing industry had been growing ever since the first digital computer was invented in the 20th century called the ENIAC, which was created by the first computer company, the Electronic Controls Company [1]. Despite the fact that the previous computers were invented even before the 20th century, the first-ever so-called "the computer" was the machine that was used for calculations but non-programmable [2]. Ever since institutes and corporations like the International Business Machines (IBM) and the Institute of Electrical and Electronics Engineers (IEEE) are putting much more effort into computer-related projects and research to further evolve the machines. In 1956, GM-NAA I/O, it is the first OS that was ever designed by one of the groups in IBM. Then, the UNIX system, Microsoft, Apple were later founded coming with fresh and innovative Operating Systems. This is where Software Industries started to rise, growing, emerging, advancing and evolving.

Technology has evolved in a very short period with many complexes yet advanced features, being completely different from 20-30 years ago. From the perspective of "computers" being a luxury item back in the past, in this modern world, computers are everywhere. As the popularity of computers grows, so do engineers, especially in the software aspects. We already have more than 23 million software developers in 2018, and it is still the Top 1 Highest Demanded Jobs in the world [3] [4]. More and more software is developed in this world, applications, systems, web applications, databases, etc. are widely and commonly used in almost every country. However, great innovations always attract great challenges, once technology involves business and properties, some people will put on all their strengths to obtain that data by going through the backdoors. Which is also known as one of the major limitations of technology, the Security and Privacy Concerns.

Beside software bugs or functionalities, the security of software is one of the most concerned problems in the world today, in fact, software bugs or malfunctioning features also may lead to a security breach or so-called vulnerabilities to the actual software. The importance of security in software is frequently neglected by both users and engineers. In the world today, about 99% of a company's data are stored and recorded in digitally, which made those data the assets or the properties of the company, losing them will be a complete tragedy. Attackers or Hackers created an algorithm in many forms and plant them into the network, also known as Malware, which has different forms and functionalities. Malware is used to steal, sniff, spy, phishing and many more different kinds of attack that anyone could think of, it exists. Ever since the viruses and attacks including Melissa Virus 1999, ILOVEYOU Virus 2000, Mydoom Worm 2004, Shamoon Malware 2012 (largest computer hack in history), and last but not least, WannaCry Ransomware 2017-2018 (most affected property) [5]. These attacks had spread widely also causing billions of affected properties. Humans learned from mistakes and experiences, we have finally realized how important was security. Technology has been evolving, so do the techniques of security, many software developers follow the software quality standards and protocols, making sure that the developed software has robust security with at least 99% free of bugs and vulnerabilities. Also the perfection of the crucial functions that prevent the software from being exploited by hackers.

II. Literature Review

Nowadays, software has become a critical aspect of our life. Many organizations use software to facilitate doing something. These lead to that software contains private, sensitive as well as important information. According to Haralambos, software engineers have to consider a lot of aspects of security. They should be such as non-functional requirements, performance, and reliability. Also, they have to care about after they designed the software. If those engineers ignore those security aspects during the development process, it could lead to a serious problem [6]. In addition, it was argued that security needs to be addressed from the early stages of the development process in order to create more secure software systems, and designers of software systems must evaluate not only the system but also its environment. It is so important since all operations of software systems within an environment as well as the various elements such as stakeholders and users who might impact the system's security aspects [6]

Furthermore, the lack of secure software development may lead to computer crime or part of the cybercrime. These days, it is rising fast rather than other crimes and cause serious harm to the economic, political and social sector. If the software leak sensitive data such as usernames, passwords, credit card information using vulnerability by hackers, it will become a serious crime and the company which makes the software may go out of business. According to Shafinah, 80% of all breaches are application-related, the traditional perimeter defenses [7]. For example, they are the antivirus systems, firewalls, and intrusion detection systems. Therefore, we cannot rely on security software to prevent the attack. We have to be concerned about the security of software from the beginning of the SDLC. Some companies conduct secure software practices but they only focus on the requirement engineering phase. Additionally, they may still use conventional software practices even if a hacker is creating a new attack method and malware [7]. Thereby, at this time, we are going to describe some studies and collaborate with each of them.

A. *Methods Used for Improvement in Software Reliability and Security*

Software design and development is a cumbersome task, especially when security of software systems is concerned. In order for a system to be reliable, it must fulfil the standard security criteria. This is a area of active research due to the fact that individual and organizational businesses and other entities vastly rely on software systems. The process of software engineering plays a huge role in the computer engineering domain and is of current research interest. Developing a software from scratch is comparable to reinventing the wheel by many, hence solutions are offered to secure existing software systems. Some to name are Component-Based Software Engineering (CBSE), use of authentication schemes, systematic frameworks, graphical authentication models for mobile devices and applications [8], specialized attack detection techniques [9], etc. CBSE helps in minimizing the efforts of building a software system by reusing the components, "component dependency and component interaction" [10]. To measure the reliability of CBSE, [10] have proposed a model which uses fuzzy logic and PSO to estimate the reliability. To minimize the risk of attacks, use of two factor authentication is also common and prominent where smart card is used as the second factor of authentication in systems like online banking, online trading, etc. [11]. [12] have gone a step further by proposing three-level security (3LS) to secure systems from ransomware. Machine learning is widely being used in efforts to address the issue of security in terms of detection of intrusion. Reviews on the ML-based solutions offered to address security issues at different levels of software systems have been performed in parallel to evaluate as to which methods are effective and offer efficient solutions. For example, [13] have reviewed intrusion detection systems developed for systems which use datasets. Similarly, [14] have conducted empirical study to find out how effective the deep learning and ensemble methods are when used for developing network intrusion detection systems.

Trust among team members involved in the development of a reliable software system is a key to successful product development. The organizations which use global software development (GSD) approach like multinational firms, in particular, require the trust factor for strong bonding within the team to avoid any risks. [15] have proposed a conceptual research model to observe the relationship among four factors which determine the success of GSD. Knowledge sharing is one of them [16]. It is likely for a system to be better in terms of reliability and security if team members of respective knowledge domains share their knowledge and implement it in development of the system.

Along with development of strategies, techniques and lifecycles involved in the design and development of software systems, it is also necessary to evaluate them based on factors like the size and type of the project, development environment [17] and target audience. [18] have reviewed five software development lifecycle models, i.e., linear sequential model, iterative model, prototype model, spiral model and v-shaped model and observed linear sequential model to be used in database related projects, prototype models for design and development of online systems, spiral models for long term projects involving high risk factor, and V-shaped model to be used in small projects where requirements are clearly known. Similarly, [19] have reviewed and analysed twenty one available software development methodologies, highlighting their positive and negative aspects, steps involved, in order to provide a guide as to which method is suitable in certain environments. Next sections covers the phases of software development life cycle to understand how it can be used to construct software systems.

B. *SDLC Methodology*

Software development life cycle or secure software life cycle as known as SDLC. That is a way to help a developer who wishes to develop secure software. Developers need to follow seven steps that are identified, plan, design, build, test, deploy and maintain [20].

Figure 1: SDLC Methodology [21]

First, identity is you need to know what kind of software you want to develop. You may get the information from your stakeholders, leaders or company. Sometimes you may also need to help them to decide what software they want to develop, because some of the clients may not know what exactly they want. This step is a critical step since requirements gathering acts as the building block of the whole system/software, is to get all the needs of the stakeholders or clients which then will be put together forming into one system.

Second, is planning how to do the software that you want to develop. You have to plan the time to develop your software and the total cost. Planning or analysis is performed to the gathered information or requirements regarding the system, this step helps to define more on the features or capabilities of the final system so it will solve the problems faced or fulfil the needs of the stakeholders or clients.

Next is designing the structure of the system. Design is a huge process in software development, you may need to take more time and use the diagrams to show the structure of the system and the flow of how the system will operate and work. In traditional SDLC, security rarely being prioritized and taken into consideration especially in the design phase this will cause the design to not have any security measures that keep the system protected and will be very vulnerable once it is deployed [22]. A good secure design of the system's architecture will have better results and reduce the possibilities of any loopholes in the system.

Then is the build phase or development phase. This is mainly handled by developers or programmers. The development stage is where programmers will start coding the system based on the system's architecture. Programmers or developers must not take security for granted, must have security knowledge and must follow the secure coding standards to have optimal results [22].

The testing phase is done after the system is completed, the main objective of testing phase it to check whether all the features or functionality of the system works as it is intended to. But most often in traditional SDLC, developers or testers did not test for the security of the system, so the system is very prone to many threats that may harm the system [22] [23].

After the testing phase will become the deployment phase. Deployment is deploying the system or software so that the system or software can be used by the client. The deployment phase is a highly automated phase. Because the software is instantly deployed when it is ready, so this phase is almost invisible.

Last but not least, the maintenance phase is the last step to complete a software life cycle. We knew that every software is cannot be able 100% bug free. The maintenance phase is to ensure the bugs are not affected by the user experience. When there is reported that the bug comes out, develop team has to solve the bug as soon as possible.

All of the above are part of the secure software lifecycle procedures. Also, there are many other ways to implement with the Secure SDLC, such as Scrum, Waterfall, Lean, Iterative, Spiral, and DevOps, etc. Secure development can be implemented to each of the classic SDLC methodologies as part of the development process by integrating security requirements along with threat modeling to the traditional process.

C. Secure Scrum

Scrum is a very popular framework that helps teams develop software. It is an empirical approach is based on process control theory to involve productivity, flexibility, and adaptability during the development process. These advantages of the approach are transparency, inspection, and adaptation [24]. However, this agile approach doesn't have any security approach parts. Secure Scrum applies those approaches and enables teams to identify security-relevant parts of the project. This approach is to enhance the level of security of software.

The aim of Secure Scrum is to achieve an appropriate security level for a given software project. On software development, resources such as money and time are important aspects. Therefore, the developers have to make sure the appropriate security level for efficient software development.

The developers need to implement measures to solve potential security risks. However, the team members may come up with some choices of methods to deal with them. Secure Scrum is to help developers to determine appropriate security testing for security-relevant parts of a software project.

In addition, for some security issues, the developer may use external resources such as security consultants. Secure Scrum enables to involve of outside resources without breaking the Scrum development approach [25].

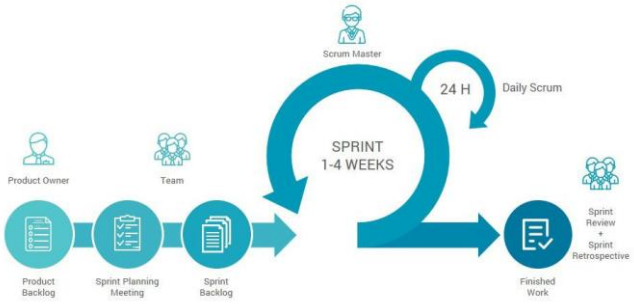

Figure 2: Scrum Methodology [26]

To satisfy those things, Secure Scrum has to contain things as can see below [27].

- Security must be visible and tangible
- The product owner plays a key role
- Requirements and priorities come from stakeholders
- Security practices need to emerge, not be prescribed

To follow components as you can see, Secure Scrum includes four components.

- Identification component
- Implementation component
- Verification component
- Definition of Done component

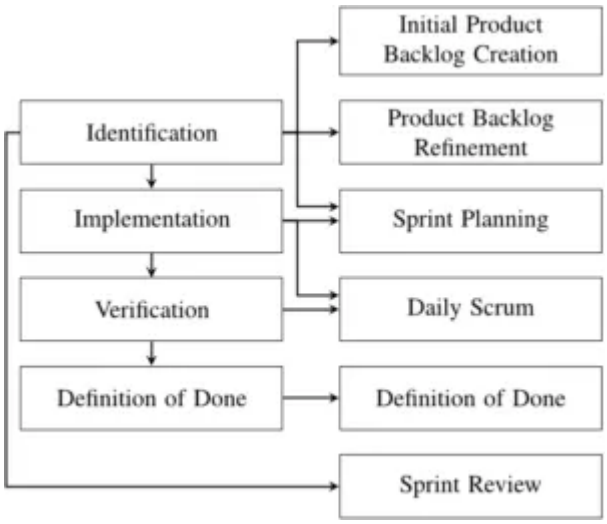

Figure 3: Secure Scrum Components [25]

The identification component is used during initial Product Backlog creation, during Sprint Planning and Product Backlog Refinement. First of all, the Product Owner and team members order the rank of different user stories according to their loss value. The loss value doesn't have the meaning of the cost of development. It is the loss that may get damages from attacks. Then the team and stakeholders evaluate those cases and rank them by their risk. After those processes, the Product Owner, team members, and stakeholders make sure security risks in the Product Backlog. To write a document for understanding in the Product Backlog, Secure Scrum uses S-Tag as can see below figure 4.

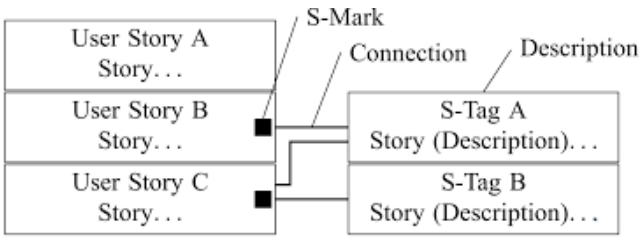

Figure 4: Secure Scrum User Story [25]

S-Tag contains one or more S-Marks, Product Backlog and a connection between Product Backlog and one or more S-Tags. S-Mark is the marker to identify security relevance. This makes them visible and tangible at all times. In addition, S-Tags has a detailed description of the security issue helps the team to make sure the security concern. This can have a user story, abuse story, misuse story, or whatever a team decides to use as description technology.

The implementation component is to make team members understand which topics of the project are important. It means they need to decide the top priority. The requirements on Product Backlogs must be ensured by daily work which must be present in the Sprint Backlog. During Sprint Backlog, some user stories are broken down into tasks. This approach regulates that the interconnection will make team members be aware of the security issues.

Also, S-Tag can be used for the verification of the emerging software. First, S-Tag determines parts of the emerging software that need security verification. Second, S-Tag helps them estimate the effort for verification.

The definition of done can ensure that the connection between S-Mark as well as its corresponding S-Mark manages existing all the time of the project. It enables the loss of security concerns or stays untested.

In addition, when the Scrum team concerns security, they may need to take special security knowledge from external resources such as security consultants or security specialists because security components are quite complicated such as implementing and testing cryptographic algorithms. Security Scrum is able to involve external resources. As you can see, there are three functions of external resources:

1. Enhance knowledge
2. Solve challenges
3. Provide external view

III. Traditional SDLC Model Versus Scrum Methodology

Since there is more than one method to complete secure software. Those methodologies can be grouped into two groups which are traditional SDLC and Scrum methodology. Traditional SDLC has included the methods which mention in SDLC methodology which are Waterfall, Lean, Iterative, Spiral, and DevOps. Scrum Methodology also knew as Agile Methodology.

Let's take an example from traditional SDLC. The waterfall methodology is a well-known method to make secure software. It will only follow the sequential order. Because it follows only the sequential order, so the development team is not allowed to change anything based upon customer requirements. Then this will lead to face some problems like a customer not get satisfaction, cost more time, or requirements will be pending.

Waterfall has to follow each step in a linear path. Overlapping is not allowed in every phase of development proceeds which means it has to complete in a specific time period. Furthermore, if there is any requirement that is not clear, then the project will be not easy to complete.

Advantages of the Waterfall Model:

- Easy to implement
- A minimal amount of resources is required.
- Proper documentation is followed for the quality of the development.

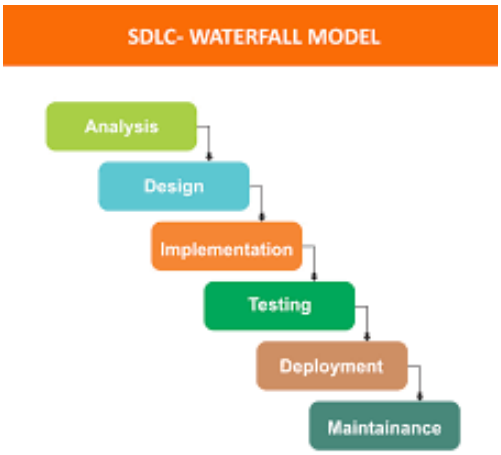

Figure 5: SDLC - Waterfall Model [28]

Disadvantages of Waterfall Model:

- Problems can be led by anyone phase which is never solved completely.
- Requirements cannot be changed
- Freezes scope - Customer requirements contract
- Delivery time is intended

Scrum methodology has more freedom to negotiate with the progress. So the development teams who use Scrum methodology are able to deal with customer's requirements in a better manner. Additionally, because Scrum methodology can rearrange the priority of the requirements, so there is time flexibility.

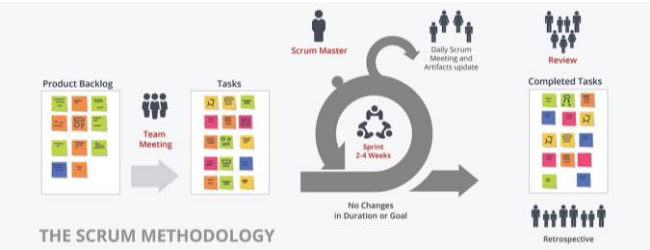

Figure 6: Scrum Methodology [29]

Scrum is a series of processes of the sprint. Each sprint will have its own task to complete. Also, Scrum encourages team works on a less documentation environment. Every scope in each sprint is adjustable, because there is a cycle that allows the task which has not finished bring forward to the next sprint.

Advantages of Scrum Methodology:

- Provides customer satisfaction by optimizing turnaround time and responsiveness to requests
- High quality
- Requirements are changeable
- Spending less time to create estimates
- Freeze's schedule - Short sprint by short sprint
- Features prioritize

Disadvantages of Scrum Methodology:

- Less documentation (difficult to check the previous work)
- Team members dedication is very important
- Teamwork is highly essential
- The project may fail if team members do not cooperate well

So, for using which methodology or model to develop a project really depends on the team. As we can see both sides have their advantages and weak points as well. For those development teams who usually work together, the Scrum methodology will bring more benefits to them. On the other side, the teams which just build up and have no experience of developing a project or working with other teams, the traditional methodology is suitable for them [27].

IV. Threat Modelling

Threat modeling is a form of risk assessment method that represents a model aspect of an attack and defense sides of a specific logical entity, such as an application, a system, a piece of data or information, and an environment [30]. Threats on its own can be people or persons, vulnerability, malware, and espionage which are harmful and undesired [31]. A threat model is created to help provide detailed information on potential threats that might harm the business/software assets and its possible mitigation to help reduce the impacts of a threat and the probability of the attack.

Threat modeling becoming a more common part of the security development of software or system. This is because when using threat modeling in the early stage of the development, the chances are it helps decrease the total cost of the project. As the threats are discovered early in the development, fixes, and preventions will be done to solve any design flaws and issues, this minimizes the need to re-design and refactor the whole system which will increase the cost and time of the project.

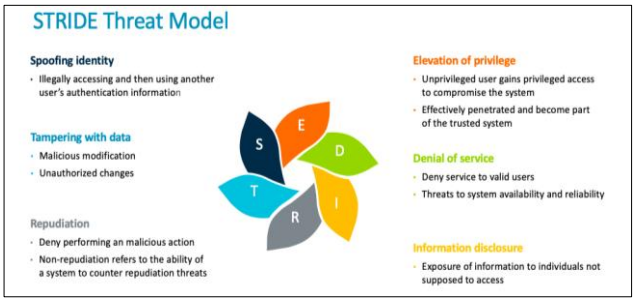

Figure 7: STRIDE Threat Modeling [32]

There are multiple threat modeling methodologies and tools that are already developed, which are distinguished by the logical entity that is being modeled, the phase of the entity's lifecycle or system lifecycle, and also the main goal from the threat modeling [31]. Generally, threat modeling has several phases, first is to understand the logical entity that is being modeled, second is to figure out possible threats that may threaten the logical entity, and lastly is to provide

mitigation strategies to overcome the threats and keep the logical entity safe. 4 category of threat modeling:

1) Asset and Impact-centric Modelling:

Asset-oriented modeling focuses on identifying critical assets and the impacts on them. A list of valuable assets will be identified from the system or software then threat scenarios that may impact each of the assets are described and prioritized accordingly. Mitigation strategies will be done according to the priority of each asset.

2) Attacker and Threat-centric Modelling:

Attacker-oriented modeling focuses on identifying potential attackers, their characteristics, intentions, capabilities, and behavior. This is to provide an understanding of what does the attacker desire to gain while trying to attack the system or software and how will the attacker approach attack the system or software, so tactics, techniques, and procedures can be implemented to mitigate this type of attack.

3) Software and System-centric Threat Modelling:

Software-centric threat modeling is carried out during the software design phase and development phase to reduce the vulnerabilities of the software, while system-centric threat modeling is focusing on the operating system to improve the overall security of the software [30]. A data flow diagram is commonly used to model the system and identify the threats that are relevant to each component in the system.

4) Data-centric Threat Modelling:

Data-centric threat modeling focuses on identifying critical types of data within a system instead of the applications or hosts. The critical data are identified and the emphasis is given to the characteristics of the location where the data will be stored, the flow of data transmitted, how data is being inputted and output within the system and the adding permission of authorized user that is able to access the data [30].

V. Data Collection

In order to get further information to prove the importance of security and privacy in developing software, we have conducted a survey to gather user's perspective and awareness on software security and privacy in general as our part of our research. Out of the four general primary data gathering methods, Observations, Questionnaires/Surveys, Interviews, and Focus Groups, we will carry on with our research with observations on a similar range of research online and distribute specialized online surveys on this specific topic. The reason we have chosen to conduct online survey is because surveys provide are usually an easy approach for the respondents and are helpful in terms of providing anonymity on personal information [33]. Also, it will be comparatively convenient and straightforward since the questions are constructed by the team, therefore, we will be gathering all of the necessary information that helps the research.

Table 1: Threat Modeling Methods [34]

Threat Modelling Methods	Usage
STRIDE	- Stands for **Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege** - Intended to analyze bugs and vulnerabilities in a software - Help identify mitigation techniques - Most mature method - Easy to follow/use and good documentation but time-consuming
PASTA	- Stands for **Process for Attack Simulation and Threat Analysis** - Has an attacker-centric approach and produces an asset-centric results - Integrated prioritization of threat mitigations - Has rich documentation but time and labor-intensive
Attack Trees	- One of the oldest and most used threat modeling methods - Diagrams that illustrate an attack in a tree form - The root of the tree represents the goal of the attack while the leaves represent the possible ways to achieve the goal - Easy to adopt and implement but a thorough understanding of the system is required
LINDDUN	- Stands for **Linkability, Non-Repudiation, Detectability, Disclosure of Information, Unawareness, Non-Compliance** - Focuses on privacy concerns and data security - Integrated prioritization of threat mitigations - Provide good privacy knowledge base and documentation but is time-consuming and labor-intensive

CVSS	- Stands for **Common Vulnerability Scoring System** - Identifying a vulnerability and scoring it based on its severity - Consistent outcomes/results on repeated test - Integrated prioritization of threat mitigations
Security Cards	- Threat brainstorming toolkit - Identify and explore possible threats and improve better security mindset - Good for education and creativity but less consistent results
Persona con Grata	- Focuses on the motivation and the capability of an attacker - Has an attacker-centric approach - Consistent outcomes/results on repeated test - Only discover some subsets of the threats
Quantitative TMM	- Quantitative Threat Modeling Method is a hybrid method which consists of STRIDE, Attack trees, and CVSS - Focus on building an attack tree for the 5 threat category of STRIDE, then CVSS applied to calculate the scores for each component - Consistent outcomes/results on repeated test - Integrated prioritization of threat mitigations
Trike	- Threat modeling from risk management and defensive perspective - Integrated prioritization of threat mitigations - Automated components - Vague and lack of documentation
Vast Modeling	- An automated threat modeling platform - Designed to be scalable - Integrated prioritization of threat mitigations - Consistent outcomes/results on repeated test
Octave	- Stands for **Operationally Critical Threat, Asset, and Vulnerability Evaluation** - Risk-based assessment and security practices - Consistent outcomes/results on repeated test - Integrated prioritization of threat mitigations - Time-consuming and doesn't address technological risks

In the survey, excluding Demography, we will split the questions into 3 main sections, "User Behaviour", "Policies and Procedures" and "General Security & Privacy Concerns". The list of questions will be provided in Table 2.

Table 2: Threat Modeling Methods [34]

User Behaviour	
Question	**Answering Method**
You will save your sensitive information (e.g. Credit Card Number) in an application.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
You will check the software ratings/comments before installing it.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
You will read/scan through the application's Terms & Conditions.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
You will allow all permissions (e.g. Microphone, Camera, Files) that the application will access.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
You will disable an antivirus/firewall when the application requested.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
You will keep your software up-to-date.	Rating (Strongly Disagree 1 - 5 Strongly Agree)

These questions in the survey (Table 3 and 4) provide awareness, knowledge, and feedback for/from the user or even a person that has IT-based knowledge. Sections like the "User Behaviour", help us to understand how a user behaves generally while using a software or web application.

Table 3: Policies and Procedures

Policies and Procedures	
Question	Answering Method
Are you aware of the standard policies and procedures to use software securely?	Yes, No, Maybe
Do you think that companies/organizations should have strict policies and procedures?	Yes, No, Maybe
Do you think that software should be used accordingly with appropriate policies?	Yes, No, Maybe
Do you think that a personal/company network should have limited access (blocking certain websites)?	Yes, No, Maybe

As sometimes as it proves, user behaviors do affect the overall security of software, which opens gates for a hacker/attacker to gain access to exploit the system or steals important information from the user that would cause critical consequences. Next, "Policies and Procedures", this section will be focusing on the user's awareness and opinions regarding the use of standard and specialized policies. For instance, policies and procedures for companies to follow in order to keep the company's software safe from attacks. Lastly, the "General Security & Privacy Concerns", the main idea of this section would be the general concept of software should behave based on the end user's perspectives. Gathering ideas from the end-users on how should companies and software handle their information appropriately.

Table 4: General Security and Privacy Concerns

General Security & Privacy Concerns	
Question	**Answering Method**
The software should be fully tested before deployment.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
The software should be vulnerable.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
The software should contain almost or absolutely no bugs.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
The software should have strong account authentication management.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
The software should store all of the user's sensitive information.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
Software stored information should be fully encrypted.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
Software developers should keep a copy of the user's information.	Rating (Strongly Disagree 1 - 5 Strongly Agree)
The software should expose sensitive information to the public without any permission.	Rating (Strongly Disagree 1 - 5 Strongly Agree)

VI. Discussion

The survey conducted has received a total of 48 unbiased responses, each of the individuals that had submitted the survey contributed to answering the questions with their honest thoughts regarding the specific statements. Firstly, we will be analyzing the findings result from the survey, which will be divided into 4 different sections "Demography", "User Behaviour", "Policies and Procedures" and "General Security & Privacy Concerns".

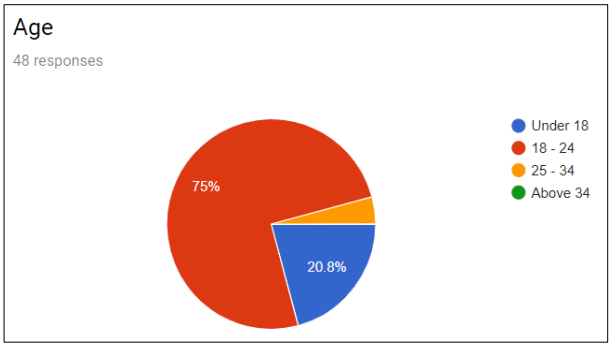

Figure 8: Demography (Age)

Under this section, the only demographical information that is needed and related was age and whether people have general knowledge in the IT field or not. As shown in figures 11 and 12, the majority of our respondents are mostly young (46 under age 25).

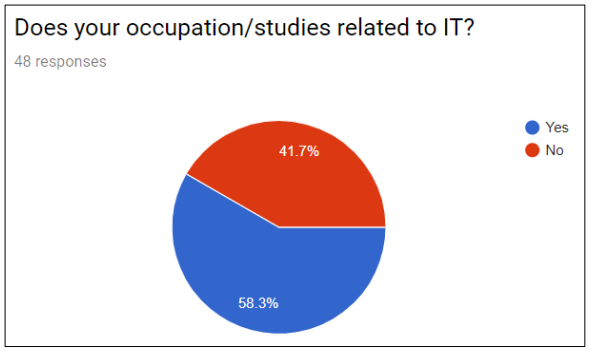

Figure 9: Demography (IT)

58.3% (28 out of 48) of them do have either occupation or studies related to the IT field. However, even though the majority are IT related, we should not be assuming all of them have the knowledge on software security including those who have experience in IT, therefore the following questions will be carried on.

A) User Behavior

In this section the questions will be asked as statements as if it is mentioning the respondents, then the respondents will be answering with 1 to 5 ratings to show whether the statement is likely or unlikely to be relatable.

According to the data in Table 5, there are several questions that are highly positive or highly negative, or even causes a certain dilemma that the results may be quite uncertain to positive or negative. Starting from the first

question, the majority disagrees on the saving their sensitive information in any application, but there are still about 20.9% will save their sensitive information, which is an important factor that may affect the risk of user's information getting exposed to the hackers.

Table 5: Results of Survey Regarding User Behavior

Question	Rating (1-5) Percentage (%)					Summary
	Strongly Disagree	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**	
Saving Sensitive Information	**37.5**	29.2	12.5	18.8	2.1	Negative
Check software reviews	4.2	2.1	16.7	29.2	**47.9**	Positive
T&C Review	**37.5**	27.1	20.8	2.1	12.5	Negative
Allow Permission	2.1	20.8	**37.5**	27.1	12.5	Neutrally Positive
Disable security	**31.3**	27.1	29.2	6.3	6.3	Negative
Software Update	-	-	8.3	29.2	**62.5**	Positive

Secondly, most of the respondents (77.1%) will check the software's ratings or reviews before installing it, this is also an important element that could prevent certain "known" threats by reading the comments from the community.

Moreover, about 64.6% of the respondents do not read or scan through the application's Terms and Conditions (T&C), T&C is a critical component for both companies and users to understand their position and rights towards using the application, as well as the information that the software will keep or reuse, if the application uses anything beyond, shall be penalized.

The fourth question is also extremely critical, 37.5% of the respondents are neutral and 39.6% are positive towards allowing all the permissions that the application will access.

Despite the well-known applications are too using these features, it is unsafe to grant permissions for applications to access certain features such as location, files, microphone, etc.

Fifth, 58.4% disagreed and 29.2% neutral on disabling anti-virus software or firewall when the application requested, which is a good sign that the majority do realize the danger of disabling the assist and filter from both anti-virus software and firewall. Both of them help to prevent the known threats, therefore will be able to keep out from virus attacks that are coming from malicious software.

Lastly, 91.7% of the respondents agreed and 8.3% neutral on keeping the software up-to-date, which is also a good sign that almost all of the respondents recognize the importance of up-to-date software. Not just for the up-to-date user interface or features, but also for the bug fixes and lessen the vulnerabilities of the software.

B) Policies and Procedures

In this section the information gathering process is performed regarding user's ideas on Companies or Personal Policies and Procedures while using any software. The following questions will be answered by a Yes or No answer.

Table 6: Results of Survey Regarding Policies and Procedures

Question	Rating (Yes, No, Maybe) Percentage (%)			Summary
	Yes	**No**	**Maybe**	
Standard Policies and Procedures Awareness	**47.9**	16.7	35.4	Mostly Yes
Strict Policies and Procedures	**81.3**	2.1	16.7	Yes
Software used with appropriate policies	**92.8**	-	6.3	Yes
Limited network access	37.5	12.5	**50**	Uncertainly Yes

Firstly, the standard policies and procedures to use software securely should be given by the software distributors as part of the guide in order to help the users to protect themselves. 47.9%, which is the majority do acknowledge and aware of the policies and procedures, and 35.4% are unsure or may be aware of it.

For both questions, 81.3% and 92.8% of respondents agreed on the company should have strict policies and software should be used accordingly with appropriate policies respectively. It is a crucial matter that the companies should have strict policies and followed accordingly by the employees. Not just to make sure the sequences of workflow, but also to prevent the use of the company's software or system is safe from external attacks.

Finally, the majority (50%) is uncertain of a company network should have limited access such as blocking certain websites or any access. The rest of the half, 37.5% think that it is important to have limited access. Taking a banking company as an example, with very important data in possession, in order to keep the customer's data safe from hackers, the banking company should have policies and procedures for the employees to follow to prevent every attack possibilities.

C) General Security and Privacy Concerns

In this section, information is gathered regarding the user's awareness of general security & privacy concerns while using any software. The criterion to answer the questions is Strongly Disagree to Strongly Agree (Rate 1-5).

Table 7: Results of Survey Regarding General Security and Privacy Concerns

Question	Rating (1-5) Percentage (%)					Summary
	Strongly Disagree	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**	
Tested before deploy	-	-	2.1	27.1	**70.8**	Positive
Vulnerable	**33.3**	25	27.1	10.4	4.2	Negative
Bug-less	-	12.5	10.4	33.3	**43.8**	Positive
Strong authentication Management	4.2	-	2.1	27.1	**66.7**	Positive
Storing user's sensitive information	10.4	22.9	**33.3**	12.5	20.8	Neutral
Stored information encryption	-	2.1	6.3	25	**66.7**	Positive
Keeping a copy of the user's information	20.8	20.8	**33.3**	18.8	6.3	Neutrally Negative
Expose sensitive information	**95.8**	-	2.1	2.1	-	Negative

By referring to the data collected from the survey, 97.9% had agreed on software should be fully tested before deploying it to the market. A proper software testing procedure should be implemented, including functionality tests, using tools for code reviews or manually, unit tests and checks, penetration tests, and so on. Although it may not be 100% bug-less after tested, however, it may eliminate most of the bugs and vulnerabilities of the software.

Secondly, both of the similar questions, the vulnerability of software and bugs in the software, 58.3% of the respondents disagreed on software should be vulnerable and 77.1% agreed on software should contain almost or absolutely no bugs. These factors are crucial in order to minimize the threats towards the company and most importantly the user's data. Imagine if an e-commerce website is full of vulnerabilities and bugs, users' sensitive information can be easily exposed, including credit card number, password, or location.

Furthermore, 93.8% of the respondents think that having a strong account authentication management is extremely important for software. There are many possible ways to authenticate a user, username and passwords are most commonly used. However it is easy to crack passwords in the world today, therefore, a mix of different authentication methods can be used together to make accounts much safer. For example, One Time Password (OTP) authentication, duo authentication from a mobile phone, email verification, or answering pre-registered questions.

Lastly, will be all questions regarding user's information security and privacy. "Software should store all of the user's sensitive information", the response to this question is mostly neutral (33.3%), and both positive and negative have the same number of percentage, so most of the respondents may think that software may or may not store the user's sensitive information up onto a certain level. Besides, 91.7% out of 48 respondents think that the stored information should be fully encrypted, regardless of the encryption techniques. Additionally, 41.6% disagree and 33.3% neutral for the software developers keeping a copy of the user's information, 33.3% of the respondents may think certain information that the developers could keep but 41.6% disagree with it. Correspondingly, almost everyone (95.8%) feels that software should not expose sensitive information to the public without granting permissions from them.

D) Proposing a Unique Solution

As mentioned in the previous survey findings, most of the customers want to have security software and aware of the importance of having one, but most do not know how and why risks could happen. As software developers, we should know how and where do attacks and risks are caused, which should be mainly from vulnerabilities and bugs from the miscoded software. There are many existing ways and practices to develop secure software, or keeping your software safe and prevent the attacks. One of the most commonly used practices are the Secured SDLC Methodology, but using just one method would not be enough, understanding the incoming threats, prepare a recovery, policies, and procedures are also needed to be implemented for keeping the risk to the minimum.

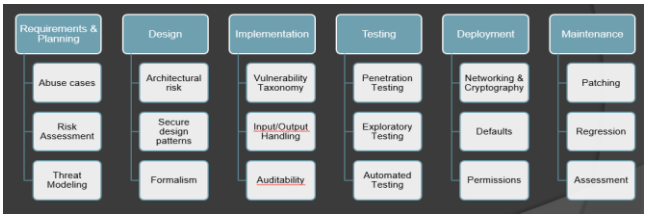

Figure 10: Secure SDLC [35]

Firstly, the SDLC Methodology will be used with Agile/Scrum approach, agile in software development is well-known as an ultimate methodology for enhancing and perfecting development process, saving both time and cost along with better software quality and assurance, most importantly, it reduces production risk [36]. Scrum helps in making the traditional process of SDLC (Waterfall Approach) much detailed on each of the processes, Gathering Requirements/Identifying, Design, Implementation, Testing,

Deployment, and Maintenance. As previously mentioned about Secure Scrum and Scrum Methodology, it divides the workload into several sprints, then combines all of the tasks together in one piece.

Next, would be knowing the possible incoming threats by implementing threat modeling or risk management, such as Open Web Application Security Project (OWASP) Top 10 Vulnerabilities, STRIDE, PASTA, and so on. Threat modeling like the STRIDE and PASTA is widely used for software development, which is also previously mentioned under the Threat Modeling topic. Besides, the OWASP Top 10 Vulnerabilities list can also help from preventing the known threats, similar to the STRIDE or PASTA threat model, but this is mainly aiming for web applications and most of the risks on the web. The top 10 list is [37]:

- "SQL Injections"
- "Broken Authentication and Session Management"
- "Cross-Site Scripting (XSS)"
- "Referencing Malicious Object Directly"
- "Misconfiguration of Security"
- "Sensitive Information Exposure"
- "Inappropriate Access Control"
- "Cross-Site Request Forgery (CSRF)"
- "Components with Known Vulnerabilities"
- "Invalidated Redirects and Forwards" [37]

Knowing the threats will help the developers to predict and prevent the known vulnerabilities and perform an appropriate recovery.

With the collected findings in previous sections, it is shown that companies should have proper policies and procedures as detailed instructions to cultivate the best practice for both employees and users. Essentially, it helps to reduce the risks by accidentally opening gates for hackers to create exploits or stealing information. A standard procedure to follow will bring everyone into the same page where it will not cause unnecessary exploits. Besides, raising awareness also can help for preventing threats, such as helping users to understand how things can be hacked or how to further enhance the security of the user's account and assets.

In a worst-case scenario, attacks succeeded, companies should have a recovery plan to deal with the attack aftermath then bounce back with a professional response. Different types of attacks will be a different kind of response. Besides, there are many types of attacks and using different types of methods, thus companies should be always ready to have a recovery plan to minimize the damage. Given the process or steps to deal with attacks [38]:

1. Identify the type of attack
2. Run through investigation to find out the source of the attack
3. Damage Analysis
4. Fixing vulnerabilities source and repair damage
5. Update related employees and affected users
6. Improve system/application by learning from experiences [38]

A planned recovery makes a company be prepared with proper recovery, in fact, the process above is a general idea of a recovery plan, and companies should have detailed plans for a data breach or cyber-attacks recovery plan, especially companies that have a large user base and network.

VII. Conclusion

There may not be an ultimate solution for handling the threats, but a good development process will be capable of dealing with the problem. A good development process like previously mentioned about the Secure SDLC with Agile or Scrum methodology, a good architectural design along with secure code review, will be the best solution for making the software security in the development stage. While the software is published in the market, the company should have a response team for encountering all of the feedback from the active users, then inform the developer team to fix the bugs or possible vulnerabilities of the software as soon as possible. Meanwhile, also perform upgrades and security checks from time-to-time to make sure the security of the software is still on track.

VIII. Future Work

In this paper, the focus has been on reviewing software systems and ways to secure them while they are in development stage. These systems are vulnerable to internal and external attacks, therefore, they need to be secured using efficient security tools embedded in them during the building stage. Some surveys have also been presented in regards with people perception of security in usage of some of the systems. In future, the aim is to implement the secure software system and evaluate it in terms of the security that it offers. The comparative methodology will be used where an unsecure form of a system will be presented versus its secure replica to find out the difference it makes in terms of trust and reliability among people.

References

[1] "When was the first computer invented?," *Computer Hope*, 02-Aug-2019. [Online]. Available: https://www.computerhope.com/issues/ch000984.htm.[Accessed November 13, 2019].

[2] W. Harris, "Who Invented the Computer?," *HowStuffWorks Science*, 30-Jan-2020. [Online]. Available: https://science.howstuffworks.com/innovation/inventions/who-invented-the-computer1.htm. [Accessed November 13, 2019].

[3] M. Chand, "How Many Software Developers Are There In The World?," *C# Corner*. [Online]. Available: https://www.c-sharpcorner.com/article/how-many-software-developers-are-there-in-the-world/. [Accessed November 13, 2019].

[4] J. Zambas, "The 10 Most In-Demand Jobs in the World (2018)," *CareerAddict*, 22-Apr-2019. [Online]. Available: https://www.careeraddict.com/most-demand-jobs. [Accessed November 13, 2019].

[5] L. Rochford, "The Worst Computer Viruses in History," *CEO Today*. [Online]. Available: https://www.ceotodaymagazine.com/2019/06/the-worst-computer-viruses-in-history/. [Accessed November 13, 2019].

[6] H. Mouratidis, P. Giorgini, and G. Manson, "When security meets software engineering: a case of modelling secure information systems," *Information Systems*, vol. 30, no. 8, pp. 609–629, 2005.

[7] S. F. P. Mohamed, F. Baharom, A. Deraman, J. Yahya, and H. Mohd, "Secure software practices among Malaysian software practitioners: An exploratory study," 2016.

[8] Teoh Joo Fong, Azween Abdullah, NZ Jhanjhi, Mahadevan Supramaniam, "The Coin Passcode – A Shoulder-Surfing Proof Graphical Password Authentication Model for Mobile Devices", in

International Journal of Advanced Computer Science and Applications (IJACSA), Vol 10, No, 1, pp. 302-308, 2019

[9] Kok, S.H.; Abdullah, A.; Jhanjhi, N.; Supramaniam, M. Prevention of Crypto-Ransomware Using a Pre-Encryption Detection Algorithm. *Computers* 2019, *8*, 79.

[10] Waheed Sara, Hamid Bushra, NZ Jhanjhi, Humayun Mamoona, Nazir A Malik, Improving Knowledge Sharing in Distributed Software Development, In International Journal of Advanced Computer Science and Applications(IJACSA), Vol.10, issue 6, pp. 434-443

[11] C. Diwaker, P. Tomar, A. Solanki, A. Nayyar, NZ Jhanjhi, et al., "A New Model for Predicting Component-Based Software Reliability Using Soft Computing," in IEEE Access, vol. 7, pp. 147191-147203, 2019. doi: 10.1109/ACCESS.2019.2946862

[12] K Hussain, NZ Jhanjhi, H Mati-ur-Rahman, J Hussain, MH Islam, Using a systematic framework to critically analyze proposed smart card based two factor authentication schemes, Journal of King Saud University-Computer and Information Sciences

[13] SH Kok, Azween Abdullah, NZ Jhanjhi and Mahadevan Supramaniam,(2019). Ransomware, Threat and Detection Techniques: A Review, IJCSNS International Journal of Computer Science and Network Security, 19 (2), pp. 136-146

[14] S.H. Kok, A. Abdullah, NZ. Jhanjhi and M. Supramaniam, "A Review of Intrusion Detection System using Machine Learning Approach", International Journal of Engineering Research and Technology 12 (1), 8-15.

[15] Adeyemo Victor Elijah, Azween Abdullah, NZ JhanJhi, Mahadevan Supramaniam and Balogun Abdullateef O, "Ensemble and Deep-Learning Methods for Two-Class and Multi-Attack Anomaly Intrusion Detection: An Empirical Study" International Journal of Advanced Computer Science and Applications (IJACSA), 10(9), 2019.

[16] Mamoona Humayun, NZ Jhanjhi., "Exploring The Relationship Between GSD, Knowledge Management, Trust And Collaboration", in Journal of Engineering Science and Technology (JESTEC), April 2019, Vol. 14 No. 2 pp. 820-843

[17] Soobia Saeed, NZ Jhanjhi, Mehmood Naqvi, and Mamoona Humayun, "Analysis of Software Development Methodologies", in International Journal of Computing and Digital Systems, vol 08, issue 05, pp. 445-460

[18] Seema, Suresh & Kute, Seema & Surabhi, Deependra & Thorat,. (2014). A Review on Various Software Development Life Cycle (SDLC) Models. 3. 2320-5156.

[19] M. Mahalakshmi, M. Sundararajan, "Traditional SDLC Vs Scrum Methodology A Comparative Study," *International Journal of Emerging Technology and Advanced Engineering*, 3(6), Jun 2013.

[20] "News," *Miles YCE Bamboo House*. [Online]. Available: http://boholmilesbamboohouse.com/news/. [Accessed November 13, 2019].

[21] H. Assal, S. Chiasson, "Security in the Software Development Lifecycle," SOUPS @ USENIX Security Symposium, 2018.

[22] Synopses, "Software Integrity Blog", Secure SDLC 101, Available at: https://www.synopsys.com/blogs/software-security/secure-sdlc/, 2016.

[23] "Best Scrum Software in 2019: Agile Project Management," *Cert Learners Info*, 07-Jul-2019. [Online]. Available: http://www.certlearners.com/best-scrum-software-in-2019-agile-project-management/. [Accessed November 13, 2019].

[24] F. Anwer, S. Aftab, S.S. Muhammad, U. Waheed, "Comparative Analysis of Two Popular Agile Process Models: Extreme Programming and Scrum," *International Journal of Computer Science and Telecommunications*, 8. 1-7, 2017.

[25] C. Pohl, H. Hof, "Secure Scrum: Development of Secure Software with Scrum," *ArXiv*, abs/1507.02992,2015.

[26] S. Türpe, A. Poller, "Managing Security Work in Scrum: Tensions and Challenges," 2017.

[27] E. Billah, "SDLC Waterfall," *Medium*, 24-Sep-2019. [Online]. Available: https://medium.com/@ersandibillah03/sdlc-waterfall-3a3c893be77b. [Accessed November 13, 2019].

[28] *Systems Valley*. [Online]. Available: https://www.systemsvalley.com/insights/. [Accessed November 13, 2019].

[29] S. Kumar, P. Dubey, "Software Development Life Cycle (Sdlc) Analytical Comparison And Survey On Traditional And Agile Methodology," Abhinav National Monthly Refereed Journal Of Research In Science & Technology, 2. 22-30, 2013.

[30] "Threat Modeling," *Avotis*. [Online]. Available: https://avotis.com.sg/threat-modelling/. [Accessed November 13, 2019].

[31] M. Souppaya, K. Scarfone, "Guide to Data -Centric System Threat Modeling," March 2016.

[32] S. Juuso, "Evaluation of Threat Modeling Methodologies," May 2019.

[33] N. Shevchenko, T. A. Chick, P. O'Riordan, T. P. Scanlon, C. Woody, "Threat Modeling: A Summary of Available Methods," Jul. 2018.

[34] S. Puckett, "Data Collection Techniques," *Fulcrum*. [Online]. Available: https://www.fulcrumapp.com/blog/field-data-collection-methods/. [Accessed November 13, 2019].

[35] "Engineering Secure Software. A Ubiquitous Concern You can make a security mistake at every step of the development lifecycle Requirements that allow. - ppt download," *SlidePlayer*. [Online]. Available: https://slideplayer.com/slide/10706063/. [Accessed November 13, 2019].

[36] M. Miller, "5 Biggest Benefits of Scrum - ClearlyAgile - Agile Transformation, Certified Training, DevOps, and Agile Software Development," *ClearlyAgile*, 18-Sep-2018. [Online]. Available: https://www.clearlyagileinc.com/agile-blog/5-biggest-benefits-of-scrum. [Accessed November 13, 2019].

[37] *OWASP Top Ten*. [Online]. Available: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project#tab=OWASP_Top_10_for_2013. [Accessed November 13, 2019].

[38] S. Hewitt, "Six-step plan for dealing with a cyber security breach," *Airmic*. [Online]. Available: https://www.airmic.com/news/six-step-plan-dealing-cyber-security-breach. [Accessed November 13, 2019].