Importance of Secure Software Development for the Software Development at Different SDLC Phases

Faris Mohamed Ahmed Hassan¹, Shampa Rani Das¹, and Manzoor Hussain¹

¹Affiliation not available

August 15, 2023

Abstract

The advancement of technology has made the development of software applications become unstoppable. The wide use of software applications has increased the threat to cyber security. The recent pandemic required the organization to adapt and manage new threats and cyberattacks due to the rising number of cybercrime activities all around the digital ecosystem. This situation has led to the importance of ensuring that the software is safe to use. Therefore, software development that emphasizes security aspects in every phase of the software development life cycle (SDLC) should be prioritized and practised to minimize cybersecurity problems. In this study, a document survey be conducted to achieve an understanding of secure software development processes and activities. The source of information is retrieved from different reliable resources of scientific research databases such as IEEE, Science Direct and Google Scholar. Moreover, trusted web resources also be referenced to support the argument in the literature study. Findings show that there are several steps of security measures for every phase of SDLC that should be conducted to improve the security performance of the software developed. The author also suggests solutions for dealing with current issues in secure software development which include educating and training the development team on secure coding practices, utilizing automated tools for software testing and implementing continuous automated scanning of threats and vulnerabilities in the system environment.
Importance of Secure Software Development for the Software Development at Different SDLC Phases

Faris Mohamed Ahmed Hassan¹, Shampa Rani Das¹ and Manzoor Hussain²

¹ School of Computer Science, Taylor’s University, Malaysia
² Faculty of Computing & Information Technology Indus University

farismohd949@gmail.com; shampa.swe@gmail.com; manzoor.hussain@indus.edu.pk

Abstract

The advancement of technology has made the development of software applications become unstoppable. The wide use of software applications has increased the threat to cyber security. The recent pandemic required the organization to adapt and manage new threats and cyberattacks due to the rising number of cybercrime activities all around the digital ecosystem. This situation has led to the importance of ensuring that the software is safe to use. Therefore, software development that emphasizes security aspects in every phase of the software development life cycle (SDLC) should be prioritized and practised to minimize cybersecurity problems. In this study, a document survey be conducted to achieve an understanding of secure software development processes and activities. The source of information is retrieved from different reliable resources of scientific research databases such as IEEE, Science Direct and Google Scholar. Moreover, trusted web resources also be referenced to support the argument in the literature study. Findings show that there are several steps of security measures for every phase of SDLC that should be conducted to improve the security performance of the software developed. The author also suggests solutions for dealing with current issues in secure software development which include educating and training the development team on secure coding practices, utilizing automated tools for software testing and implementing continuous automated scanning of threats and vulnerabilities in the system environment.

Keyword: Secure Software Development Lifecycle, cybersecurity threat and vulnerability

1.0 Introduction

Advancement of technology has made the development of software application become unstoppable which sooner will become the foundation of computer world. However, recent pandemic that has make organization need to adapt and manage new threat and cyberattack due to the arising number of cybercrime activities in the internet. Moreover, transition to remote working has resulted significant implication for cybersecurity [1]. This situation has cause unexpected cloud migration and rapid IT product and service demand and acquisition in order to fit in remote working environment. Many organizations pushed or neglected conventional security procedures in order to keep company operations continuing, without realizing exposing them to unprecedented level of vulnerability and risk throughout all industries.

The demand for cloud-based services and infrastructure has surged due to COVID-19 which make people to work remotely. This trends will continuously evolve since more organization implementing cloud-hosted service to adapt on current situation and keep operation running [1] [2]. Despite the benefit of cloud services such as flexibility, efficiency and cost reduction, they remain a major target for cybercriminals. This will required organization to evaluate the cloud computing security implementation and check current infrastructure for any weakness.
Misconfigured cloud setting, for example were the primary source of data breaches in 2020 and has cost of $4.41 million [3].

Ransomware has become the most significant risk for data security of organization and it became as top cybersecurity concern [4]. Ransomware attacks has cost more compared to attack of average data breaches. The complexity of the strategies used by cybercriminals is also increasing. Extortion attacks has become more frequent, in which hackers take a company’s data then encrypt it so they cannot access it. Cybercriminals will then threat to expose its confidential information unless a ransom is paid. The consequences of this cyberattack are significant, with sensitive data at stake and the financial consequences of paying the ransom.

More businesses are using multi-factor authentication (MFA) as a supplemental layer of protection against data breaches and attacks [5] [6]. For user to access their personal data, they are required to prove their identity with more than one device. While MFA has become critical for security precaution, Microsoft has advised users to avoid phone-based MFA (where one-time passcode is supplied to phone through SMS text) because to current phone network security flaws [7]. Because SMS-based messages are not encrypted, attackers may retrieve them in plain text. Instead, organization should use more robust MFA approaches, such as application-based MFA such as Google Authenticator or Microsoft Authenticator.

Artificial intelligence and machine learning are becoming more sophisticated and capable, thus make organization utilize these technologies as part of their security infrastructure [8] [9]. AI is increasingly being applied to enhance automated security system that substitute human interaction, allowing more quick analysis for large volume of risk data. This is advantageous for both big businesses dealing with vast volumes of data and small and mid-sized businesses with under-resourced security teams. While AI provides a significant potential for organizations to improve their threat detection, the innovation and greater application of this technology has both positive and negative implications. Criminal networks are using AI to automate their attacks, and they’re using strategies like data poisoning and model theft to do it.

2.0 Literature Review

This section will discuss and summarize on previous studies on challenges and issues in software development, current vulnerabilities in software development, practice of secure software development software as well as secure software development life cycle.

2.1 Secure Software Development Life Cycle

Development and advancement of technology has make all daily transaction and online based activities to rely on online application. The wide use of software application has also increase the threat of software security [10]. This situation has led to the importance of ensuring that the software is safe to use in order to maintain trust of users in services provided. Therefore, software development that emphasizes security aspects should be prioritized and practiced in software development life cycle (SDLC) to minimize cybersecurity problems.

Secure Software Development Life Cycle (S-SDLC) is the integration of best practices aim to increase security in every phases of standard SDLC [11]. Execution of secure SDLC involves
committed work at each phases, from requirement gathering, development to maintenance. This practices requires commitment from development team that aim to instil and empower them to build a secure application as a norm instead of just focusing solely on functionality. With determination and hard work, security issues can be address well before deployment and production.

SDLC is the foundation of software development models. From SDLC, variety model has being derived such as Waterfall, Agile, Prototyping, Spiral and Rapid Application Development (RAD). All these models have the same basic phases with different execution mechanism and objectives [12]. The basic stages including 1) requirement, 2) design, 3) development, 4) testing and 5) deployment. Figure 1 shows SDLC with the security measures that need to be carried out in each phases.

![Software Development Life Cycle](image)

Figure 1: Software Development Life Cycle [11]

Implementation of equally distributed security measures in every phases of software development phases are crucial and has significantly reduce vulnerability of the system as well as reducing the cost and time consume to develop the system. This is because installing a patching software will be much more expensive than solving the issues in real time during the SDLC phases [13]. Indirectly, this has improved software quality as well as development productivity and efficiency. [14] [15].

Integration of security practices from the design stages allow security related requirements are recorded and formalized before the development start. This make both management and development teams are informed of procedures of development, the software security risks and threats and not being diverted by irrelevant ones [27]. As a result, the development strategy can be improve to ensure the code developed as the SDLC progresses.

Studies shows that integration of security approaches on software development life cycle more emphasize on development phase where coding writing were carried out. Static and dynamic analysis are the most commonly used methods for performing security measures throughout the coding phase [16]. Test case prioritization is a technical techniques to rearrange the execution of test cases for minimizing regression testing expenses. This method also support
that coding phase contribute to the highest percentage to the needed of test case prioritization for cost reducing purpose [17]. Moreover, an automated technique, fuzz testing, can be used to identify implementation flaws and security vulnerabilities by inputting data that modified and malformed [18]. This testing has been proven to be cost effective and also has high efficiency be used in coding phase. These indicate that security practices in coding phases has a vital role in determining the quality and cost of software development.

In order to decrease the number of software failures, [19] has suggested on autonomic advisor which will provide recommendations to developer as well as performing risk assessment throughout SDLC processes. This alternative can enhance the efficiency of SDLC and quality of developed software. Risk assessment throughout the SDLC cycle will ensure the software delivered achieve quality with high level of security performance [20].

2.2 Current Security Vulnerabilities in Software Development

Throughout pandemic, there are rising number of organizations experiencing an increasing in cyberattacks [21]. Due to this scenario, Runtime Application Self Protection (RASP) is now needed as an extra layer of protection in the latest security and privacy framework standard [22]. RASP solutions provide extensive application security, including vulnerability prevention, while consuming resources optimally and contributing minor delay to an application. It monitors the programme using runtime deterministic security which depend on the program itself, rather than depending on previous breaches to determine a zero-day attack as well as OWASP Top Ten Security Vulnerability by verifying the program control flows.

Open Web Application Security Project (OWASP), a non-profit organization aim to improve software security and educate community regarding the most common security concern, so they may implement security policies, reducing the occurrence of known risks. Recently, OWASP has release Top 10 Security Risk and Vulnerabilities 2021 [23], which listed as below:

1) Injection

It happen when an attacker insert invalid data to the web application and interact with database query [24] with the aim to make it to do something it does not programmed to do. It cause by the data used by the web application has not been validate and filtered.

2) Broken Authentication

Attackers can gain control of a system by brute-force tactics. Bad session management can allow an attacker to take control of the entire system or even worse, take over the whole thing. It can cause by poor session management and/or bad password protection.

3) Sensitive Data Exposure

It consist of data that should be protected being compromised. Example of sensitive data: credentials, credit card numbers, medical information, social security number, personal identifiable information.

4) XML External Entities

Form of XML input parsing attack against an application. When a poorly constructed XML parser examines XML input containing a reference to an external object, this attack occurs.
This caused by vulnerable XML processors, vulnerable code, vulnerable integration and vulnerable dependencies.

5) **Broken Access Control**

Access control implies limiting which sections or pages they may access depending of user level of access. Broken access control happen when outsider can access certain section that he should not to.

6) **Security Misconfigurations**

The most common security misconfiguration including unpatched flaws, keeping CMS default configurations, availability of unused pages, has unprotected files and directories and installing unnecessary services.

7) **Cross Site Scripting (XSS)**

XSS attacks occur when attacker putting dangerous client-side scripts into a website and exploiting it to spread malware. The drawbacks of XSS is attacker can change the way a website is presented by running malicious code on a victim's browser. It allows an attacker to upload material into a website and change how it is presented to the victim.

8) **Insecure Deserialization**

Deserialization is the action of attacker replacing byte strings to objects (structure data) in order to give themselves admin privileges. For instances, a super cookie with serialized information about current logged-in user which include information about user’s role. If an attacker successfully deserializes the object, then alter it to assign himself admin role and serialize it again. This activities has potential to threaten entire online application.

9) **Using Components With Known Vulnerabilities**

Personal blogs, for example, now have several dependencies. Failure to update software program on website’s backend and frontend, relatively soon can cause significant security problems. Ignoring an update warning could permitting now-known vulnerability to remain in the system as attacker are eager to look into software and changelogs.

10) **Insufficient Logging And Monitoring**

Insufficient logging and monitor on website can increase website infiltration. There are number of ways to keep website examined on regular basis so that quick action can be taken if something goes wrong.

### 2.3 Challenges and Limitation in Secure Software Development

Studies from [27] shows that best practices described in the literature deviate significantly from real-world security approaches. Best practices are frequently ignored since compliance to those practices would add team’s workload. However, they are making a justifiable cost-benefit trade-off. This situation shows that best security practices suggested from literature not aligned with the industry requirements, therefore this may contribute to software vulnerability and business risk of organization.
Due to extreme rising number of data theft and other sorts of cyber threats, software engineers are more focused on developing more secure software rather than fulfill its functionality. The introduction of specialized techniques into software development, however, has resulted in cost rising [28]. This situation caused by the wide range of cyber threats that consistently thrive which leads to more specialized procedures required to overcome the issues.

Software development is a process of that required continuous improvements. Developers frequently work in tight dateline established by management teams in an attempt to set realistic goals, even though achieving such goals might be challenging. However, developers give their best to develop a secure software, but they may not be able to identify all the flaws before the release date. Delays may be expensive, which make organization launch the initial software version and later address any flaws reported by providing security updates, often known as patches [29].

There are studies that shows application of secure software development practices do not reach satisfactory level in majority of software industry. This scenario happen due to multiple factors including lack of direction, a clear guidelines from company’s top management and there are not specific guidelines on security policy regarding on integration in system developments [30]. This all factors has leads to weak security measures integration and expose the software on threats and cyberattacks.

Open source components such as frameworks and libraries are prebuild product that help developers in reducing development time, but they frequently include unknown dependencies. A lot of open source components are utilized in a ‘black box’ way which might result in establishment of vulnerabilities that lead to unsafe code as well as patching and versioning compatibility issues. Moreover, the action of download from unauthorized sources might cause complexity that difficult to manage [31]. Availability of open source components might be ease of developer’s program writing, but increasing become source of cybersecurity vulnerability and threats if it not treated properly.

Performing data input validation is one of secure practices that carried out by developer but programming language, Java for example prohibit code execution inside of JVM instance by utilizing static strong typing architecture. However, other programming languages, execute all the native code on metal which expose to risk of malicious application gaining control on machine's system software. This also particularly troublesome in mobile industry, where user may be completely unaware that their devices have been hijacked until it is too late [32].

### 2.4 Vulnerabilities Prevention and Detection Practices

Cybersecurity attack strategies are constantly being evolve and current ones are expanding, making cybersecurity issues incredibly unexpected and dynamic. SQL injection suggested to be most significant threats for web application security as the attack involved for exposure of sensitive data to unauthorized users, which can cause serious harm to legitimate user [24]. Due to this issues, there are improvement in awareness among developers regarding on the importance of implementing secure coding practices in software development at the same time practice based on secure coding guidelines [25]. However, software program are written by human, and they are inherently flawed. Nobody creates software that fully error-free, which leaving gaps for prospective attackers. Therefore, software security practices should be encourage prevent vulnerabilities from being introduced into software and detect
vulnerabilities injected during development. Studies from [26] shows the prevention (table 1) and detection (table 2) of vulnerabilities practices that being used.

Table 1: Practices for prevent vulnerability [26]

<table>
<thead>
<tr>
<th>Problem</th>
<th>Practices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bugs</td>
<td>Utilize a top-N bug list (real data preferred)</td>
</tr>
<tr>
<td></td>
<td>Follow secure coding guidelines</td>
</tr>
<tr>
<td></td>
<td>Average (bugs)</td>
</tr>
<tr>
<td>Flaws</td>
<td>Create and publish security features</td>
</tr>
<tr>
<td></td>
<td>Convert compliance restrictions into requirements</td>
</tr>
<tr>
<td></td>
<td>Work with software security group (SSG) to develop architecture</td>
</tr>
<tr>
<td></td>
<td>Develop data classification scheme and inventory</td>
</tr>
<tr>
<td></td>
<td>Unify regulatory pressures</td>
</tr>
<tr>
<td></td>
<td>Establish security standards</td>
</tr>
<tr>
<td></td>
<td>Make policy for security</td>
</tr>
<tr>
<td></td>
<td>Collect and employ attack intelligence</td>
</tr>
<tr>
<td></td>
<td>Develop SSG capability to solve difficult design problems</td>
</tr>
<tr>
<td></td>
<td>List out potential attackers</td>
</tr>
<tr>
<td></td>
<td>Implement and monitor control for compliance</td>
</tr>
<tr>
<td></td>
<td>Containerize applications</td>
</tr>
<tr>
<td></td>
<td>Identify a personal-identifiable-information data inventory</td>
</tr>
<tr>
<td></td>
<td>Create technology stacks standard</td>
</tr>
<tr>
<td></td>
<td>Identify open source in apps</td>
</tr>
<tr>
<td></td>
<td>Identify and utilize an architectural-analysis process</td>
</tr>
<tr>
<td></td>
<td>Build and maintain a top-N potential attacks</td>
</tr>
<tr>
<td></td>
<td>Uniform architectural descriptions (including data flow)</td>
</tr>
</tbody>
</table>

Table 2: Practices to identify vulnerability [26]

<table>
<thead>
<tr>
<th>Problem</th>
<th>Practices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bugs</td>
<td>External penetration testers to identify issues</td>
</tr>
<tr>
<td></td>
<td>Ensure that edge or boundary value condition testing is supported by quality assurance (QA).</td>
</tr>
<tr>
<td></td>
<td>Request SSG to conduct an ad hoc evaluation</td>
</tr>
<tr>
<td></td>
<td>Utilize penetration testing tools internally</td>
</tr>
<tr>
<td></td>
<td>Use both automated and manual tools to review</td>
</tr>
<tr>
<td></td>
<td>All projects are required for code review</td>
</tr>
<tr>
<td></td>
<td>Integrate black-box security tools into the QA process</td>
</tr>
<tr>
<td></td>
<td>Perform fuzz testing customized to application APIs</td>
</tr>
<tr>
<td></td>
<td>QA automation should included in security check</td>
</tr>
<tr>
<td></td>
<td>Employ automation to do what attackers will do</td>
</tr>
<tr>
<td></td>
<td>Average are bugs</td>
</tr>
<tr>
<td>Flaws</td>
<td>External penetration testers to identify issues</td>
</tr>
<tr>
<td></td>
<td>Examine security features</td>
</tr>
<tr>
<td></td>
<td>Use penetration testing tools internally</td>
</tr>
</tbody>
</table>
Secure software development is essential to follow all SDLC phases carefully for the successful implementation of software and hardware [42-44]. This is equally important for the various types of software such as web-based systems, Server based systems, Operating systems, wireless devices, and Internet of Things devices [45-48]. Since the software is the first baseline for any tool.

Furthermore, if software security is not considered at any phase of the software development, which easily leads the software to issues and can provide various vulnerabilities and opportunities to hackers. As mentioned, that software security is important even if belongs to any group of applications [49-55], whether they belong to any health applications or wireless applications.

Mainly the application software which is designed using AI and machine learning for smart homes, smart devices, tiny wireless devices, and the Internet of Things, [56-60] to make them intelligent and to take smart decisions at the right time are also prone to the software security, in case if the secure SDLC is not followed properly. This is equally important [61-64] for the software, which is being used for Industry 4.0, cyber-physical systems, in the cloud for virtual machines etc. OWASP is an open platform which provides a different range of attacks on the software for various applications due to the vulnerabilities and loopholes, and these all-impact different applications [65-67] while they are present due to the lack of secure software development life cycle implementation.

### 3.0 Methodology

Methodology technique will determine the reliability and validity of gathered information. In this studies, document survey be conducted for achieve understanding on secure software development process and activities. The source of information being retrieve and gathered from different reliable resources of scientific research databases such as IEEE, Science Direct and Google Scholar. Moreover, trusted web resources also be referenced to support the argument in literature study.

Data collection be conducted by making keyword search related to research scope for narrowing the searching process. For this information gathering activities, keyword such as ‘secure software development lifecycle’, ‘secure coding practices’, ‘current security issue’ and ‘security threats and vulnerabilities’ being used to find information related to this study. Then, all the search result being analyse and interpret to gain understanding on the articles findings to form reasonable conclusions. The following are some of the articles we looked into:

<table>
<thead>
<tr>
<th>Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>For high-risk applications, do design review</td>
</tr>
<tr>
<td>Integrate black-box security tools into the QA process</td>
</tr>
<tr>
<td>Request SSG in charge for design review process</td>
</tr>
<tr>
<td>Implement automated tools with custom rules</td>
</tr>
<tr>
<td>QA automation should included security checks</td>
</tr>
<tr>
<td>Construct multiple analysis that feed into one reporting or remediation process</td>
</tr>
<tr>
<td>Automate malicious-code detection</td>
</tr>
<tr>
<td>Employ automation to do what attackers will do</td>
</tr>
<tr>
<td>Average in term of faults</td>
</tr>
<tr>
<td>Average use of all 16 practices</td>
</tr>
</tbody>
</table>
1) “Exploring Software Security Approaches in Software Development Lifecycle: A Systematic Mapping Study”. The study analysed 118 academic research papers on software security approach and conclude that most of studies do security practice measures in coding phase on software development. It also stated static analysis and dynamic analysis are the most techniques used for security examination on coding phase [16].

2) “A Preventive Secure Software Development Model for a Software Factory: A Case Study”. The study analyse and compare the existing secure software development models and propose a new secure software development methodology. Experiment be conducted on all the models and resulted reduce of 68.42% of vulnerabilities on proposed solution [14].

3) "Security in Software Development Lifecycle”. The study examine the real life security practice in workplace. The result shows there are significant different on security practice apply in workplace with the best practice suggested in academic research. [27]

This method is utilized due to ease of access of wide range information with reliable and trusted resources for research purposes due to in-depth of expertise and information. Moreover, all the researches were supported by real case studies with evidences of analysis and findings that contains graphs, chart and pictures that assist in clarifying and proven the research purposes. In addition, all the research studies proven to be reliable since it be appropriately monitored and finalized by expert panels in the academic area. Meanwhile, web resources provide author with current information of security in industry written by technical expertise in security.

4.0 Finding and Discussion

Based on software development lifecycle, security approaches are included in every phases of development to assist in minimizing of underlying business risks of organization. A secure SDLC facilitates in effective prevention of most security vulnerabilities, hence securing an organization from variety of cyberattacks. Discussion below will talk through on recommended security practices in each phases of SDLC [27] [33] [34] [20].

1) Concept and planning
   • Establishing project security and compliance goals to ensure development team be able to resolve security vulnerability as soon as feasible
   • Write a list of project security requirements, which include technical and regulatory standard such as including access control, operational boundaries, policy and privacy [35]
   • Training session regarding software security to expose development team on threat and vulnerability awareness
   • Risk assessment conducted to identify the potential risk, establish technical feasibility and quality assurance [36].

2) Architecture and design
   • Develop threat modelling to incorporates evaluating potential attack scenarios and appropriate preventive measure into application architecture
   • Validate design document and subsequent modification in security requirements to discover features that are vulnerable to security threats
• Checking third-party software on regular basis to identify areas that vulnerable to compromised components to immediately apply patches.

3) Implementation
• Enforcing secure coding standards to remove minor flaws and allow focus on more crucial activity
• Perform Static Application Scanning Testing (SAST) on newly written code to identify flaws before application builds
• Review code manually in timely manner to identify and resolve any issues before proceeds to other tasks

4) Testing and fixing
• Incorporate Dynamic Application Scanning Testing (DAST) with Interactive Application Security Testing (IAST). This technique combine runtime scanning with monitoring of the program’s executed code and data flow. It's detect common vulnerabilities and configuration issues that compromise security.
• Perform fuzzing testing to strengthen defensive capabilities against attacks that take advantage of faulty inputs
• Incorporate with third-party security expert to stimulate possible attacks that development team might overlook

5) Release and maintenance
• Security monitoring that cover environment of system, not just the application
• Produce incident response plan that specifies action that must be taken in case of security breaches as fast execution of action plan is important
• Perform continuous security checks since new form of vulnerabilities are constantly being identified

6) End of life
• Check company retention policies for compliance with legal requirements as some data types are subjected to government-defined retention policies
• Properly destroyed all sensitive data such as encryption keys and personal information stored in application

5.0 Proposed Solution
Vulnerabilities and threat in software program are caused by human error. There is no software program that error-free since technology landscape is constantly changing which make cybersecurity trends dynamic and unpredictable. These are some suggestion of solution to deals with current issue in secure software development:

1) Educate and train development team on secure coding practices
Software program coded by developer, therefore if developer familiar of applying secure coding practices, software vulnerability can be minimized. However, some of them do not realise of the consequence of their actions. They think they already know all they need to know. They lack of understanding on how dangerous the vulnerability can cause to software developed as well as unaware the important of security practice in software development. This
scenario happen as they has no or little comprehension on foundational lesson of security application. In order to overcome this shortcomings, training program can be conducted to educate them on secure coding practice. This exposure will enable them to learn, understand and implement the techniques in encoding program to minimize security risk.

2) **Utilizing automated tools for software testing**

Availability of commercial automated tools for vulnerability testing has shifted the burden of developers to conduct manual testing because of capability of automate tool to produce testing result immediately with generated comprehensive reports. Moreover, utilizing automated tools ensure high-quality delivered product as well as improve efficiency in overall software development [37]. This be proven by research study that automate software testing has demonstrate that it can save approximately 68% of overall software testing automation process and shortening product launch cycle [38]. In addition, developer can conduct in-depth test that analyse complicated use cases and lengthy tests with automated testing that generally be avoided during manual testing.

3) **Automated on threat and vulnerability scanning continuously**

Pandemic has significantly increase the traffic of network usage as everyone stay connected virtually via Internet. The use of technology has is speeding up, as seen by the emergence of multi-perception technologies, artificial intelligence’s appeal and cloud solutions [39]. Therefore, report shows that there is increasing in the number of cybersecurity attacks as attackers see these situation as an opportunity to make profit [22]. Moreover, advancement of technology has made technology such as Internet of Things (IoT) and network based wireless sensor has growing rapidly which has expose to more vulnerability in term of data transmission, reliability of data and excess to physical cyberattacks [40][41]. Therefore, author would like to suggest that automate threat and vulnerability scanning being running continuously on software environment so that immediate automated actions can be carried out once cyberattack occur.

6.0 **Conclusion**

Pandemic COVID-19 has significantly impact the cyber security threat and digital ecosystem. Increasing number of software utilizing in daily activities make software security become a priority. Therefore, author would like to suggest that secure software practices should not only be part of software development phase only (testing phase), vice versa incorporate in every phases and activities in software developments. With the current situation of unpredictable cyberattack, security precaution and measures should be emphasized. Software program written by human, therefore educate and train development team is essential as cyber security is constantly changing along with cyber threat. It is also encourage for developers to follow the current trend in security world. There is no perfect secure software application, therefore developer need to continuously learn and update their knowledge and skills regarding software security. Moreover, availability of automated security tools should be utilize wisely in order to improve security quality. Even though there are challenges in implementing secure software practices, organization should put an effort to make this practice as a culture to produce secure quality software.
References


