Understanding the emerging need for SSDLC and Security risk in the current SSDLC model

HARDIK CHUGH
University of Central Florida
Orlando, USA
hardik_chugh@knights.ucf.edu

YASH CHAUDHARI
University of Central Florida
Orlando, USA
yesh0607@knights.ucf.edu

ADITYA PAWAR
University of Central Florida
Orlando, USA
aditya281198@knights.ucf.edu

FATIMA BEGUM FAYHA
University of Central Florida
Orlando, USA
fatimafayha@knights.ucf.edu

ABSTRACT

The Secure Software Development Life Cycle (SSDLC) is an organized procedure that tries to produce high-quality, inexpensive software as quickly as feasible. Over time, other SDLC models have been created, from the waterfall and iterative to, more recently, the agile model. With each new model, deployment has tended to happen more quickly and frequently. Secure SSDLC has added security testing at each software development stage, from design to development, to deployment and beyond. You must make sure that you are developing with potential vulnerabilities in mind at every stage since anyone might probably have access to your source code. So due to this Secure Software Development Life Cycle is so essential and showcasing the emerging need.Despite the benefits of utilizing Secure SDLC, organizations continue to have security challenges. A pressing need for a secure development lifecycle has resulted from this due to non-adaption of SSDLC and improper security practices implementation in the current SSDLC. With this in mind, we researched the factors contributing to the growing demand followed by discussing the discrepancies between the actual implementation of the secure development lifecycle and expectations. As a result, we eventually came up with a technique to fix the present SSDLC model's security problem.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → *Machine learning Libraries*;

KEYWORDS

Understanding the emerging need for SSDLC and Security risk in the current SSDLC model

ACM Reference Format:
HARDIK CHUGH, YASH CHAUDHARI, ADITYA PAWAR, and FATIMA BEGUM FAYHA. 2021. Understanding the emerging need for SSDLC and Security risk in the current SSDLC model. In *Foundations of Computer Security and Privacy*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3463676.3485607

1 INTRODUCTION

2,200 cyberattacks are carried out every day, and 800,000 people have compromised annually solely because of poor security procedures. Due to inadequate design and development processes, enterprises are currently having issues with security vulnerabilities in their IT systems and networks. Organizations must employ new security measures to address The Secure Software Development Life Cycle (SSDLC) is a methodical procedure with the goal of creating an application's high-quality security mechanism. Design, development, deployment, and all subsequent stages of software development are all included in the Secure SDLC. There are 5 phases in Secure Software Development Life Cycle, each phase deals with the different perspective of security -

PHASE 1: REQUIREMENTS - From a variety of stakeholders, needs for new features are gathered. It's crucial to note any security concerns while gathering functional requirements for the new release.

PHASE 2: DESIGN - Criteria those are in-scope are converted into a design for how they should appear in the real application during this phase. Functional requirements define what should occur in this situation, whereas security requirements frequently concentrate on what shouldn't.

PHASE 3: DEVELOPMENT - Secure coding best practices are often outlined, and code reviews ensure that these best practices have been adhered to.

PHASE 4: VERIFICATION - Applications undergo a rigorous testing cycle throughout the verification phase to make sure they adhere to the original design and requirements.

PHASE 5: MAINTENANCE AND EVOLUTION - At this stage, vulnerabilities may also originate from other sources, such as external penetration tests carried out by ethical hackers or contributions from the general public through so-called "bug bounty" programs.

Even with the proper implementation of everything organizations are facing lot of vulnerabilities on their production environment just due to non-adaption of Secure SDLC, improper security implementations and lack of Secure Development Life Cycle awareness. **In our study**, we looked at the importance and the emerging need for Secure Software development life cycles across a range of organizations all over the world and not having a negative impact related to security on the organization. To enlighten the emerging need we have also used a National Vulnerability Archive (NVD) dataset to **showcase the rising number of vulnerabilities on the production environment based on the year and severity using Machine learning methodology that includes using libraries i.e. Seaborn and matplotlib, along with data pre-processing methods making it suitable to showcase the graphs in a re-presentable manner, along with that the latest log4j vulnerability scenario is used to illustrate the effects of not adapting the Secure SDLC**. Likewise, we discussed the early and current adoption of the SSDLC by organizations and the challenges they encounter in doing so based on the passive survey. Understanding the advantages and disadvantages of a secure SSDLC by Exploring the gaps between expectations and reality and Analysing the limitations of SSDLC. We discovered a new concept of security that is lacking in organizations after studying the current Secure SDLC models and the ways these models provide security. To improve the organization's security, We also discussed the security issue that is not given the proper priority in most organizations and we will make theoretical effort to put this strategy into practice that has not yet been implemented by anybody in the SSDLC model till now, along with the theoritcal implementation **we have also showcase some findings to prove our security mechanism that this new security method is able to find these based on the findings**. We are able to grab multiple findings so we have showcase two findings and those are MySQL credentials disclosure and Employee organization Gmail Account Credentials leakage of the big reputed Multi national companies that proves that the method we proposed should be implemented in the current SSDLC.

2 RELATED WORK

Every firm must follow a secure development lifecycle in order to create a secure environment. As the software application is implemented in several industries including healthcare, transportation, and many others, it is being utilized more and more frequently in the contemporary digital age [12] and it's important also for [3] application resistance against malicious assaults brought on by the exploitation of vulnerabilities is the main goal of software security. Security is a key notion in the development life cycle, thus software engineers must incorporate security from the start of the software development process [1], routing in [5] software development process, Routing in [4] increasing the likelihood of producing additional expenses, lengthening development times, and harming corporate reputation if secure model solution is not adopted. Organizations used to experience a lot of difficulties and financial loss as a result of vulnerability assaults because they were ignorant of security procedures or lacked understanding in this area. According to the survey paper [7], the majority of organizations around 2010 did not adopt SDLC and Organizations began applying the SDLC in response to the increase in knowledge and security awareness in the current environment, however there are a lot of difficulties that they are running into while implementing [16]. Any model or technique has a weakness known as a limitation, and when it comes to security procedures, limits can result in significant financial loss as well as reputational damage. Anuradha Misra [14] highlights the limitations of the SDLC in her work by highlighting the many restrictions in the various phases of the SDLC. In 2008 Goertzel [9] releases an enhanced SDLC methodology that reduces the number of high targets and denies attackers access to the software, but his methods will eventually become obsolete due to advancements in technology. For enhanced application security Thomas Loruenser [11] has created a cryptographic engineering technique to embed during the development phase. Hossein Shirazi [15] is also developed a new SDLC model in which he attempts to adopt a new Secure Software Development based on already-known vulnerabilities. However, none of the upgraded SDLC models employ our passive security testing method, which we will describe how to adapt in the SDLC and where to do so.

3 APPROACH

To successfully complete our project task, we are employing a systematic strategy. Our first strategy is to shed light on and discuss the **emerging need for Secure Software Development Lifecycle**, including why SSDLC is so important for organizations presently and what **challenges developers in organizations face when adopting this security practice model**. Then based on the **National Vulnerability Archive (NVD) dataset** we have enlightened up the emerging need of Secure Software Development Lifecycle with the machine learning libraries and methodologies to stabilize the dataset to showcase the graph, along with also use log4j 0-day vulnerability example to strengthening the emerging need. With the surveys already being conducted, we have done a passive survey and showed up for the current situation of the consequences of integrating SSDLC model. We also made an effort to contrast how corporations have changed SSDLC models in the past and present in order to supply and enhance corporate security. Additionally, we tried conducting our own survey by contacting software developers and cyber security consultants who work in the IT industry via more than 250+ mails, but we did not get the responses to showcase the active survey. In order to raise awareness, we looked into the gaps between expectations and realities of guarded cyber security models. We also made the SSDLC's constraints more understandable and offered solutions to overcome them. We had identified the security risk in the current Secure SSDLC model after analyzing and demonstrated a way to overcome the security constraints and showcase the theoretical implementation how we can achieve the new security method and also **showcase some findings to prove our security method.**

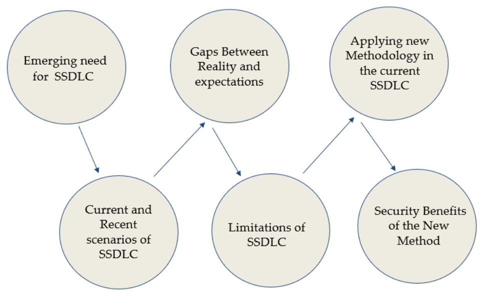

Emerging need for SSDLC
Current and Recent scenarios of SSDLC
Gaps Between Reality and expectations
Limitations of SSDLC
Applying new Methodology in the current SSDLC
Security Benefits of the New Method

Figure 1: Flow chart of our approach

4 EVALUATION

Based on the strategy mentioned in the technique module above, we will present our work so far in this module.

4.1 Emerging need of SSDLC

In 2020, Estée Lauder, a massive producer [8] of skincare and cosmetics, left their customer base with more than 440 million records that were open to the public. Microsoft acknowledged the leak of a company database storing more than 250 million anonymous customer records over a 14-year period in January 2020 and This type of data breach occurred several times as a result of secure SDLC poor implementing procedures.

Each organization aspires to maintain its position at the top in the cutthroat corporate environment. Many software vendors strive to offer secure software to their customers. Due to the rising risks of utilizing insecure products, businesses are being forced to turn their attention to secure SDLC. It effectively incorporates security into the methods used to produce software.

Based on the data set from the google we try to represent the cost to fix bugs in different phases with the pie chart visualization in figure 2. The sooner phase we work to fix the bug, the less money the company will lose in the security attacks.

The average cost to fix a vulnerability [6] that arises after the development process is 100 times more than it would be during the testing phase so it is abundantly obvious that a lower cost per defect is associated with detecting bugs earlier in the development process. «Data about cost to fix bugs at Google»

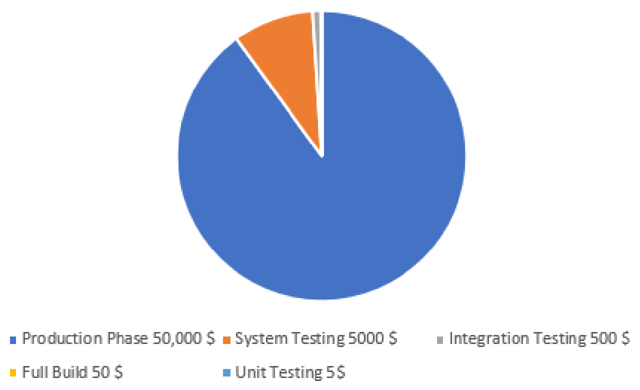


Figure 2: The cost to fix bugs at Google

4.1.1 *Showcasing the SSDLC need with the use of Machine Learning Libraries* -.

In the present security environment, the need for a secure software development life cycle is crucial. Using Seaborn and Matplotlib, two machine learning methodology libraries, we have created a bar graph based on data from the National Vulnerability Archive (NVD), which contains all vulnerabilities disclosed in the NVD database between 1999 and 2021. The dataset «Dataset Link» contains the CVE ID, vulnerability summary, CVSS scores, vulnerability severity, and matching CWE IDs. **The dataset was not in suituable manner so we have to implement data pre-processing that is a early phase machine learning methodology to make the dataset suitable to plot the graph.**

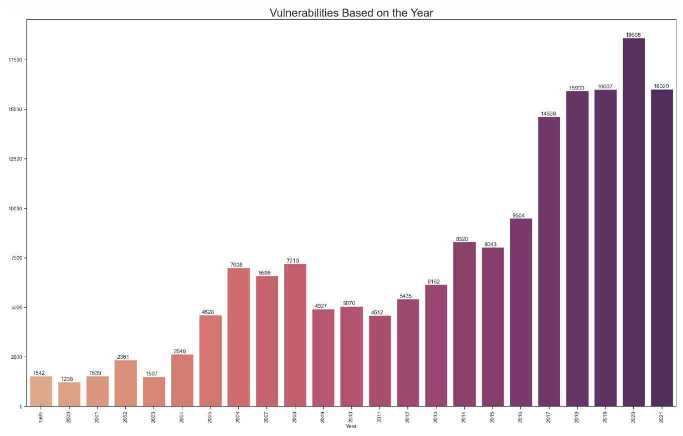


Figure 3: Vulnerabilities based on the year

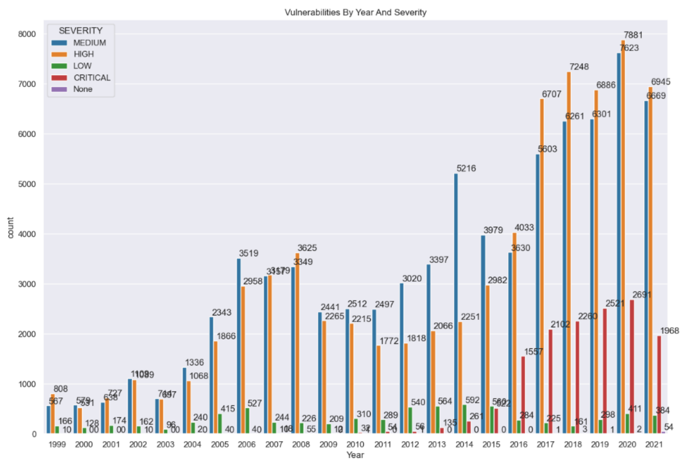


Figure 4: Vulnerabilities based on the year and severity

4.1.2 *Showcasing the SSDLC need with the Log4j vulnerability* - .

One more scenario we will discuss for the emerging need i.e. Log4j vulnerability that was recent 0-day vulnerability affected the organizations security across all over the globe. The Java Naming and Directory Interface (JNDI) injection bug is exploited by Log4j. JNDI offers an interface to naming and directory services such as Remote Method Invocation (RMI) and Lightweight Directory Access Protocol (LDAP) (RMI). This vulnerability allows adversaries to send the malicious payload using the JNDI expression and send an LDAP request to the server that will get store into the logs and execute that leads to malicious activities onto the server.

This security vulnerability arises due to improper security checks during the Software Development Life Cycle. So this also shows another essential and emerging need for the Secure Software Development Lifecycle.

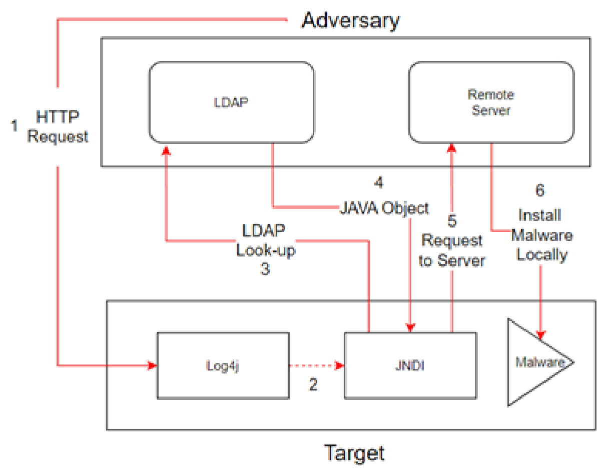


Figure 5: Log4j Vulnerability Working

4.2 Different scenarios of SSDLC

Prior to the development of technology, businesses believed that protecting their networks and hardware would be sufficient to protect them from cyberattacks but there were other application misconfigurations also present like SQLI, XXE, LFI, RCE, DDOS, Session Hijacking, etc. that can also harm the organization in a huge prospect. Due to the lack of security procedures in the Software Development Lifecycle, which developers utilize to create applications, firms have experienced vulnerabilities assaults again after installing their commercial applications. After facing or encountering the security organizations started adopting SSDLC not by all of the organizations but some of them started adopting but faced a lot of issues while adopting.

4.2.1 ***Early Scenario of organizations about SSDLC***. According to the [14] Fagan reports, despite the significant benefits of Secure Development Lifecycle, fewer than 30% of firms are implementing this methodology to secure their operations [4] to find out why businesses aren't implementing secure development lifecycles to address the various issues they face, as shown in figure 6. This is because, according to the results, 19.6 % of businesses are unaware of security methodologies, 23.9 % say it's time-consuming for some businesses, and a few other responses were also present.

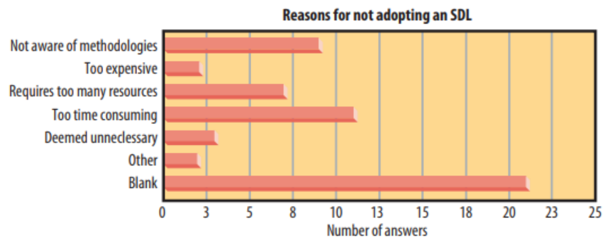


Figure 6: Errata survey about the reasons behind organizations not implementing SDL (Secure Development Lifecycle)

4.2.2 ***Current Scenario about SSDLC in organizations*** . According to the Veracode Secure Development Survey [16], surveys were conducted in a number of nations, with the United States, the United Kingdom, and Germany serving as the key countries and corporate sizes. They made remarks in the poll on the difficulties experienced by developers and devops managers during the application development process. Based on their observations 36.8 % of developers concern about security of the applications, 52 % of the developers say adapting secured development cycle threatens the development deadlines and 24 % describes that their team doesn't have authority to conduct security practices. Various additional issues, such as the high percentages of the United States (55.5% and 66.0%) and (45.5% and 48.5%), are depicted in figure 7. S ecurity practices add complexity and prolong time to market, according to EMEA developers and devops managers, who also believe that Security practices risks deadlines and slows development.

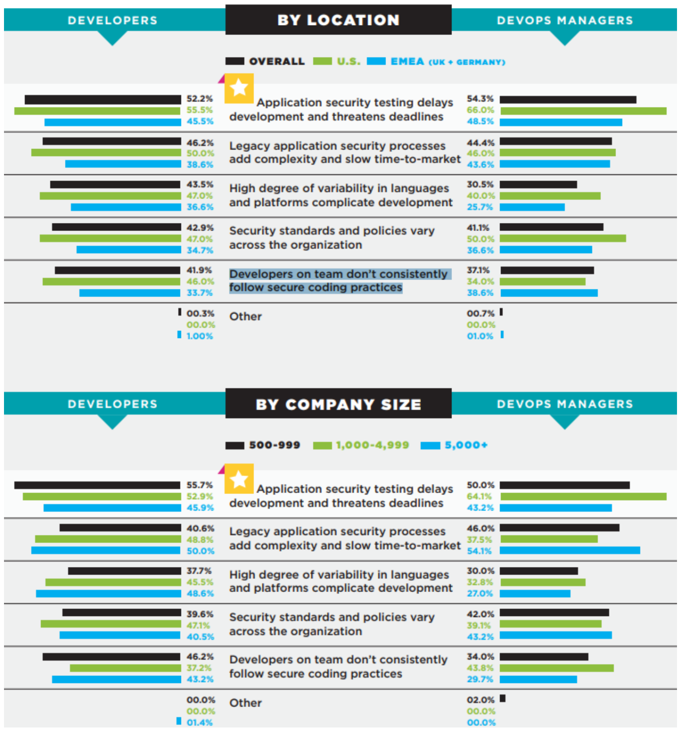


Figure 7: Veracode survey about the reasons behind organizations not implementing Security practices or SDL (Secure Development Lifecycle)

4.3 Exploring the gaps between expectations and reality with the limitations of SSDLC

Commonly in most of the organisations software testing happens at the end of the software development lifecycle. Since the release date is approaching, there is a ton of pressure at this phase coupled with several criteria like the budget, team size, amount of time required, project difficulty, technology used, documentation, and training, due to which the company might not be able to implement the most optimal security practices or adoption of SSDLC, which is going to result in a gap between the expected and actual security practices. Hence it becomes extremely important to focus on integrating security at every stage of the software development lifecycle.

To overcome this issue there should be proper timeline check of security practices in all the different development phases.

4.4 Improvement in the existing SSDLC model

Although the earlier SSDLC models are so much well equipped to provide high security i.e. modern driven architecture [2] to the application but even though organizations are encountering security attacks that leads to organizations big reputations as well as financial loss even with the remeditation in SSDLC [13] like **refactoring the code, applying a proxy in inbound and outbound connections, and to generate protection code linked to the application being analyzed**.

It is happening due to a fact that every model just focuses over the active security testing none of the model being made so far doesn't focus on the passive side of the security aspects. passive security consists of a lot of vulnerable paths that would lead to give organization a financial loss, for example, source code leakage through public code sharing platforms, dark web monitoring, social media nad phishing attacks.

Every aspect of security is essential to provide a full proof security mechanism but from the SSDLC improvement point of view source code leakage fits into the SSDLC model that leads to provide more security towards the more security from SSDLC.

Some developer use code-sharing platforms and uploads the internal source of the application while building the application just for the replica of the code but forgot to make those repositories private. Those source codes are obvious confidential and In addition those source codes might contain secret keys, SQL queries, credentials, API keys, access keys, internal paths, Personal Information, dependencies information, multiple confidential data.

This approach can be added into the testing phase of the secure software development life cycle where everything is done just they have to check whether any of their developer team has made the source code public or not, after analyzing if some repositories found vulnerable they should have remediate those to provide a passive security also to the SSDLC model.

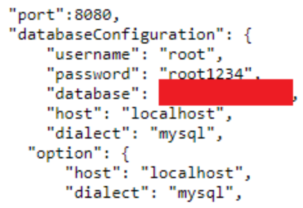
Why You Should Be Concerned About Code Leaks - The most precious asset for any corporation is its source code [10], which is also known as the design or blueprint for its proprietary technology. Leaking or stealing source code could give your opponent companies an advantage in creating new solutions and open the door for hackers to take advantage of its flaws. **Factors a organization should keep in mind if there is a source code leakage are** Reputation of organization, Data Misuse by un-identified users, Theft of Intellectual Property (IP), un-authorized access of core systems and Customer server infection.

4.4.1 *Theoretical implementation of Passive Security.*

There are many ways to implement passive security, but for the Secure Software Development Lifecycle passive security can be done by integrating a script or tool that analyzes all the developers' code sharing platforms for the information related to the software that is being developed or staged, or it can also analyze by the organizations' keyword to detect for the security confidential data that can lead to harm the organization business by chaining that data to perform active vulnerability attacks. We can also perform this security practice in a Manual Way that includes **Github Dorking** that helps to gather all the confidential data which you are looking for.

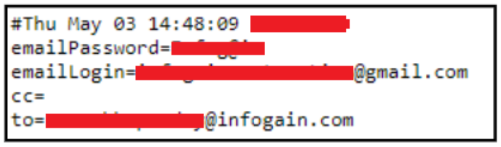
4.4.2 *Security findings to prove our improved security mechanism.*

We have attempted to put our passive security approach into practice in a manual way in order to collect some results to demonstrate our enhanced security mechanism. We have been successful in gathering many findings for various businesses, but we are unable to reveal the identities of the firms. As a result, we are disclosing the confidential information while maintaining anonymity.



```
"port":8080,
"databaseConfiguration": {
    "username": "root",
    "password": "root1234",
    "database": [redacted],
    "host": "localhost",
    "dialect": "mysql",
  "option": {
      "host": "localhost",
      "dialect": "mysql",
```

Figure 8: My_SQL credential leakage of Big MNC

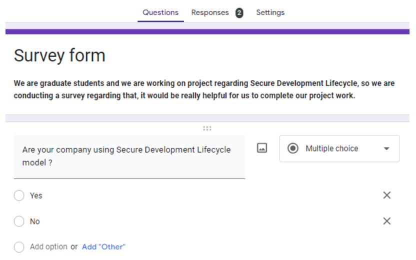


```
#Thu May 03 14:48:09 [redacted]
emailPassword=[redacted]
emailLogin=[redacted]@gmail.com
cc=
to=[redacted]@infogain.com
```

Figure 9: Employee org Gmail Account credential leakage

5 LIMITATIONS

Regardless of the encouraging results, which highlight the growing demand for Secure development life cycle, we won't be able to highlight the active survey on whether organizations are adopting SDLC or not due to the low response rate. To make the active survey happen we deployed a survey form. We had sent mails to 250+ that includes security advisors, security consultants and software developers, but we only got 2 responses.



Survey form

We are graduate students and we are working on project regarding Secure Development Lifecycle, so we are conducting a survey regarding that, it would be really helpful for us to complete our project work.

Are your company using Secure Development Lifecycle model ?

Yes

No

Figure 10: Low Responses on our survey form

5.1 Questions asked into the survey

- Are your company using Secure Software Development Lifecycle model ?
- If not using Secure Software Development Lifecycle model what is the reason your organization in not implementing ?.
- If using Secure Software Development Lifecycle, are you facing any difficulties while using Secure Development Lifecycle ?
- If using Secure Software Development Lifecycle, are you facing any difficulties while using Secure Development Lifecycle ?

6 DISCUSSION

The project's team leader is Hardik Chugh, and after discussing the project idea with each team member, he divides the work according to each person's knowledge, interests, and enthusiasm for a specific topic.

First, we outline the goals that each step of our project must reach. Then we talk about how we can demonstrate the emerging demand for SDLC, and if there is a growing need, we also talk about how Secure Software Development Lifecycle has been adopted by businesses in the past and in the present, so for that we have done passive analyzing and active scanning is on the process. We also talked about lighten up the gap between reality and expectations, as well as analyzing the limitations of Secure Development Lifecycle. We also discussed about the things we will try to implement in the final milestone and we managed to do so- In the emerging need for SSDLC we will showcase the essential need by lightening up the Log4j 0-day vulnerability example and impact, along with the Log4j vulnerability, we also hope to demonstrate an emerging requirement with the aid of a machine learning algorithm, if we are successful in obtaining the data set, will demonstrate the effects of not employing a Secure SDLC.

7 WORK DISTRIBUTION

We divided up the work among the team members according to their skills and areas of interest.

Hardik Chugh : Passive Analysis of whether the organizations are using SDLC to provide security to their organization. Showcase the emerging need of SDLC with the help of machine learning libraries. Improvement in SDLC model and theoretical implementation of passive scanning methodologies.

Aditya Pawar : Analysis of recent security attack(Log4j Vulnerability). Researching the passive scanning methodology.

Yash Chaudhari : Analysis of recent security attack (Log4j vulnerability). Analysis of Secure Development Lifecycle and its working.

Fatima Begum Fayha : Exploring the gaps between reality and expectations of organizations. Passive survey whether organizations are adopting SDLC or not. Analysis of limitations of Secure development lifecycle model.

Each team member put in a lot of time and effort to finish the tasks assigned to them, and they all tried to accomplish their roles as best as they could.

8 CONCLUSION

We used passive methodologies to examine the significance and growing demand for Secure Software development lifecycles in a variety of enterprises throughout the globe. With the help of machine learning libraries like Seaborn and Matplotlib as well as the Log4j - 0-day vulnerability example, it is vital to reduce the impact and necessity of SDLC. We made sure to underline the SDLC's constraints and the discrepancy between reality and expectations checks in addition to the graphics. It also covered how organizations adopted the SDLC early on and how they still use it now, as well as the difficulties they face in doing so. We studied the current SDLC models and found a new idea of security that is lacking in businesses. We will make a theoretical attempt to put this strategy into practice to increase the security of the organization, but none has yet done so using the SDLC paradigm. We also talked about the lack of emphasis given to security issues in most organizations and the challenges businesses and developers face when putting the SDLC into practice.

REFERENCES

[1] Mamdouh Alenezi and Sadiq Almuairfi. 2019. Security risks in the software development lifecycle. *International Journal of Recent Technology and Engineering* 8, 3 (2019), 7048–7055.

[2] Muhammad Asad and Shafique Ahmed. 2016. Model driven architecture for secure software development life cycle. *International Journal of Computer Science and Information Security (IJCSIS)* 14, 6 (2016).

[3] Hala Assal and Sonia Chiasson. 2018. Security in the software development lifecycle. In *Fourteenth symposium on usable privacy and security (SOUPS 2018)*. 281–296.

[4] Juan de Vicente Mohino, Javier Bermejo Higuera, Juan Ramón Bermejo Higuera, and Juan Antonio Sicilia Montalvo. 2019. The application of a new secure software development life cycle (S-SDLC) with agile methodologies. *Electronics* 8, 11 (2019), 1218.

[5] AR Shehab Farhan and GM Mostafa Mostafa. 2018. A methodology for enhancing software security during development processes. In *2018 21st Saudi Computer Society National Computer Conference (NCC)*. IEEE, 1–6.

[6] Lynda Gaines. 2021. Cost of fixing vs. preventing bugs. (2021).

[7] David Geer. 2010. Are companies actually using secure development life cycles? *Computer* 43, 6 (2010), 12–16.

[8] TATEEDA GLOBAL. 2021. The Growing Importance of Software Development Security. *TATEEDA GLOBAL* (2021).

[9] Karen Mercedes Goertzel, Theodore Winograd, et al. 2008. Enhancing the development life cycle to produce secure software. *US. Department of Defense, pp. i-iv* (2008).

[10] Mackenzie Jackson. 2021. Source Code Leaks: The Real Problem Nobody Is Paying Attention To. (2021).

[11] Thomas Loruenser, Henrich C Pöhls, Leon Sell, and Thomas Laenger. 2018. CryptSDLC: Embedding cryptographic engineering into secure software development lifecycle. In *Proceedings of the 13th international conference on availability, reliability and security*. 1–9.

[12] Muhammad Danish Roshaidie, William Pang Han Liang, Calvin Goh Kai Jun, Kok Hong Yew, et al. 2020. Importance of Secure Software Development Processes and Tools for Developers. *arXiv preprint arXiv:2012.15153* (2020).

[13] Gabriel Serme, Anderson Santana De Oliveira, Marco Guarnieri, and Paul El Khoury. 2012. Towards assisted remediation of security vulnerabilities. In *The Sixth International Conference on Emerging Security Information, Systems and Technologies*. 49–56.

[14] Anuradha Sharma and Praveen Kumar Misra. 2017. Aspects of enhancing security in software development life cycle. *Advances in Computational Sciences and Technology* 10, 2 (2017), 203–210.

[15] HM Shirazi. 2009. A new model for secure software development. *International Journal of Intelligent Information Technology Application* 2, 3 (2009).

[16] Veracode. 2016. Veracode Secure Development Survey. *Veracode Blog* (2016), 19.