# A Study on Incorporating Privacy and Security Practices in the Development of Software

Sampath Yelchuri
*Department of Computer Science*
*Bowling Green State University*
yvenkat@bgsu.edu

***Abstract*—Over the past few decades, various organizations have been facing many challenges and issues in designing secure software(s). Software manufacturers adopt different kinds of Software Development Lifecycles (SDLC) to design and develop their software application (s) or system(s). These traditional SDLC practices do not provide any security aspect in terms of building a software product and may often drive to various software vulnerabilities at later stages. In this research, our primary focus is to study different methodologies to incorporate various security practices into the traditional Software Development Lifecycle (SDLC). We majorly analyze various methods of embedding security activities into the Agile methodology and examine to what extent this would help various organizations to build their software product(s) with the help of various security metrics.**

***Index Terms*—Software Development Lifecycle (SDLC), Software Vulnerabilities, Agile and Security Metrics.**

## I. Introduction

Nowadays, software application(s) or system(s) have great demand in the current market all over the world. With the tremendous increase in the usage of the network and internet services like Cloud Computing, the Internet of Things (IoT), Mobile Edge Computing (MEC), etc. have attracted a huge number of people to make their life simple and easy. It is due to this requirement from the outside world that drives various software industries and organizations to build different application(s) or system(s) that meet the requirements of customers irrespective of the platform [1]. People across the world have started using these software applications or systems to either store, process or share their information. It is good that we have various organizations that have come up with different kinds of software application(s) or system(s) that help people to lead a comfortable life. But we have many incidents that took place during the past few decades where these software applications or systems have been exploited due to which many people hesitate to depend upon the current software application(s) or system(s) [2]. The primary reason behind the customers hesitation is that they are worried about the security and privacy of their information that is associated with these software applications [3].

Customers are progressively facing many ambiguities in understanding how secured and private their personal information is being maintained, processed, shared and handled once their data is being transferred through a common medium called the Internet [4]. However, it is not the deficiency of the user or customer. Many software organizations typically come across many challenges while designing or building different software application(s) or system(s). Among which security and privacy play a crucial role that needs to be taken care of at each step or phase of designing or building software application(s) or system(s). This is due to the reason that the customer(s) may always request or demand the organization(s) for things that are needed to be included in their software [5].

One of these is the functionality of the software application or system which must be correct and delivered rapidly with less cost. This makes various organizations pressurize their product team(s) which increases the chances of making errors or bugs and sometimes this may also lead to neglect some of the problems which may be raised in the future [6]. And one of these problems is security and privacy aspects in a software application(s) or system(s). Security and privacy are considered as a non-functional requirement as it does not constitute any functionality to the software application or system, but they affect the quality product(s) which are being designed and released by the organization. The problem becomes more complex when there is a large-scale development of software application(s) within the organization [7]. Moreover, it is not only that the software is confined to work only in one environment instead there may be various other environments where these software applications or systems may be used once they are out in the market. However, all the organization(s) are very careful in delivering an adaptable, fast and better software application(s) or system(s) with more functionalities that can be delivered rapidly to all their customer(s) or user(s) [8].

On the other hand, it is also important for the organization(s) to handle the business logic and the flow of implementing software application(s) or system(s) properly. For this, the organization must investigate the business functionality of the software and how it is being built. So, it is very crucial for the organization(s) to have these three dimensions of business, process and technical sides for designing and building secured software application(s) and system(s) using different software

development processes. Organization(s) make use of different practices that includes a certain number of steps or phases. These are known as the Software Development Lifecycles (SDLC) [9]. There are different types of SDLCs which can be used designing and developing software application(s) or systems(s) at a large scale. The selection of a proper SDLC plays a vital role in the development of a software application or system. It always depends on the user/customer or the complexity of the software being developed [10]. Five key terms are always included in any of the software development life cycles that the organization follows. The below cyclic representation in figure.1 shows all the five phases that are involved in the development of every software application or system.


Fig. 1. Different steps or phases that involve in a Software Development Lifecycle (SDLC) [11].

In this research, we mainly focus on how to incorporate or embed privacy and security practices in different SDLCs and how does this benefit various organization(s) in terms of providing security and maintaining privacy for different software application(s) or system(s) that are being designed and developed in the future by using some of the security metrics. We have structured this paper as follows: In section II, we will discuss the various existing privacy and security practices that are helpful in the secure development of software application(s) or system(s). Section III describes different ways of embedding or integrating the privacy and security practices into various Software Development Lifecycles (SDLC). In section IV, we discuss the benefits of incorporating privacy and security practices in SDLC. At last, section V and section VI discuss the future scope and conclusion of this research.

## II. Related Work

The process of designing software application(s) or system(s) is not independent. It constitutes different units or modules that need to be integrated and coordinated with each other to build a large software application or system. To design these units or modules more securely, various software engineering methodologies have been introduced at different phases of software development. This is due to the reason that the organizations had to meet the requirements of their customers by delivering efficient and secure software. Below are some of the methodologies that have been proposed by various researchers for incorporating security aspects in the software development lifecycle (SDLC):

Anuradha Sharma et al. [12] have explored different aspects of incorporating security at various levels in the software development lifecycle (SDLC). In this research, the authors provide a reference for twenty-one directed rules for expanding the knowledge of software security to various teams that are involved in designing and developing the software application(s) or system(s). These rules provide in-depth training and knowledge of various security engineering practices. The authors provide some of the best security practices like Threat Modeling, Architectural Risk Analysis, and Vulnerability Attack Trees. According to researchers, Threat modeling helps to identify and analyze how an external entity or attacker finds scope for exploiting vulnerabilities in software by looking at various points to enter and attack the software. The practice of Architectural risk analysis helps the developer in finding threats during the design phase by understanding the pre-requisites that must be included to prevent various security attacks. The Vulnerability Attack Trees help in determining the pathway of an attacker and provides information on how various vulnerabilities of software are linked with each other. Moreover, this research also provides a comparative analysis of Secure SDLC with the traditional SDLC model.

Dr. Dheerendra Singh et al. [13] in their research provide an analysis of integrating security in the software development lifecycle (SDLC). Researchers provide a study of various roles of architects and developers in secure SDLC practices that help in Access and Identity Control, Classification of Risks, Assessment of Risks and maintaining security. According to their investigation, the Requirement Phase must accommodate a group of members from the security team whose role is to supervise the security issues and provide guidance to the other members to minimize the vulnerabilities at the early stages of building a software product. The contributors provide a set of principles that must be involved in the Design Phase. Some of the principles are adding and removing access to various service(s) based on the task or functionality, restoring the service or system to default state if the action performed by the attacker fails and declining access and information regarding the practice or process followed should not be shared among external entities. The role of developers in the Development Phase has a major impact on the security of software. Developers must be very keen about their code that may late lead to major threats. The Testing Phase also has a crucial role in identifying the different kinds of security concerns that would raise once the software is delivered. The team which is responsible for Testing Phase also needs to verify whether the product meets the requirements of the customer or not.

According to Vaishali et al. [14], the customers always interested in the software product(s) or service(s) which are free from bugs and vulnerabilities. The main motive of their research is to introduce security and risk assessment practices in the regular SDLC model. Their study indicated that most of the traditional SDLC practices perform security testing at later stages of software development which results in different software bugs and introduces new risks to the software. Researchers have proposed a new Secure Software Development Lifecycle (SecSDLC) which involves two major phases. The initial phase is nothing but involving testing of security aspects at every phase in traditional SDLC at early stages. The authors classify the process of Risk Management into three categories: Risk Assessment, Risk Mitigation, and Risk Evaluation. These three practices are responsible for various risk management tasks like assessing Requirements, Development, Implementation, and Maintenance of the software product(s) or service(s) that being developed. Their research also provides a comparison between different development methods like Open Software Assurance Maturity Model (OpenSAMM), Microsoft-SDL and SecSDLC based on the Common Criteria (CC).

Arwa Albuolayan et cl. [15] in their research have proposed an extended model of the traditional software development lifecycle (SDLC) with the help of a case study. This extended model ensures maintaining the security aspects in the SDLC at earlier stages. A case study of analyzing practical observations was used to investigate that the newly proposed Secure SDLC can be used by various organizations for the development of any software product or service. The main goal behind their research is to involve various software engineers to integrate security in the SDLC and bring project managers into the picture to review these security policies used at different stages of SDLC. They define the term Unit Analysis which states that at each developmental phase of the software product or service the software engineers and the project manager play a crucial role in reviewing the security policies. They have also designed a framework for the Secure SDLC approach that includes Security Standards and Policies, Tools, Processes, and Skillsets which are maintained by various engineers and managers.

Chandramohan Muniraman et al. [16] have performed research on exploring and suggesting few methods for embedding security into a software application(s) or services(s) right from the initial till the end phase of developing the software product. The main goal of the authors is to make use of the Threat Modeling by using the Abuse and Misuse cases that typically help in managing the risks of the software product or service at the Requirement Phase itself. They highly prefer to refer to the requirement analysis and specification documents that are generated by various group client meetings. Each group includes at least one head from each department like database administrative, security administrator, developer, and architect. This kind of approach showed an impressive approach to develop the product where teams collaborate on whether to proceed with the other phases of software development. The researchers emphasized more on the Design Phase, where most of the security aspects are needed to be answered before the development begins. Some of the aspects are maintaining access rights based on the user(s) role, design of architecture and functionality, and Application or Product Security. Moreover, the researchers have investigated that the Testing Phase must include multiple code reviews, removing unwanted functional properties and involvement of trusted third parties for maintaining security policies.

## III. Methodology

In traditionally Software Development Lifecycles (SDLC) like Waterfall, Model, Iterative Model, Spiral Model, security is treated as a collection of operational activities which in-turn means operating Firewalls, the Intrusion Detection System (IDS), Intrusion Prevention System (IPS). These are some of the most common security practices that most of the organizations usually adopt to defend their infrastructure. Moreover, many software industries periodically perform audit operations where they have various policies and procedures that the developers are supposed to follow and many times there might be chances that they do not follow all the policies.

In this paper, we typically follow an extended Systematic Literature Study (SLS) [17] as our research methodology. This methodology is very helpful in finding, analyzing, comparing and integrating various previous related works that are interlinked with each other to answer a research question. Our SLS involves the following stages as mentioned below:

- Formulate Research Question(s).
- Search and Analyze Proposed Mechanism(s).
- Focus on a Specific SDLC model(s).
- Integrate appropriate Mechanism(s).
- Document/ Report Result(s).

Given below is a pictorial representation in figure 2 states the entire control flow of our research methodology. In this section, we mainly focus on analyzing some of the security aspects that should be embedded in one of the most common software development practices that are currently being followed by various software industries known as Agile. We try to provide some security metrics to show the benefits of incorporating security practices into the traditional SDLC models.

Given below are the two research questions that this study tries to analyze:

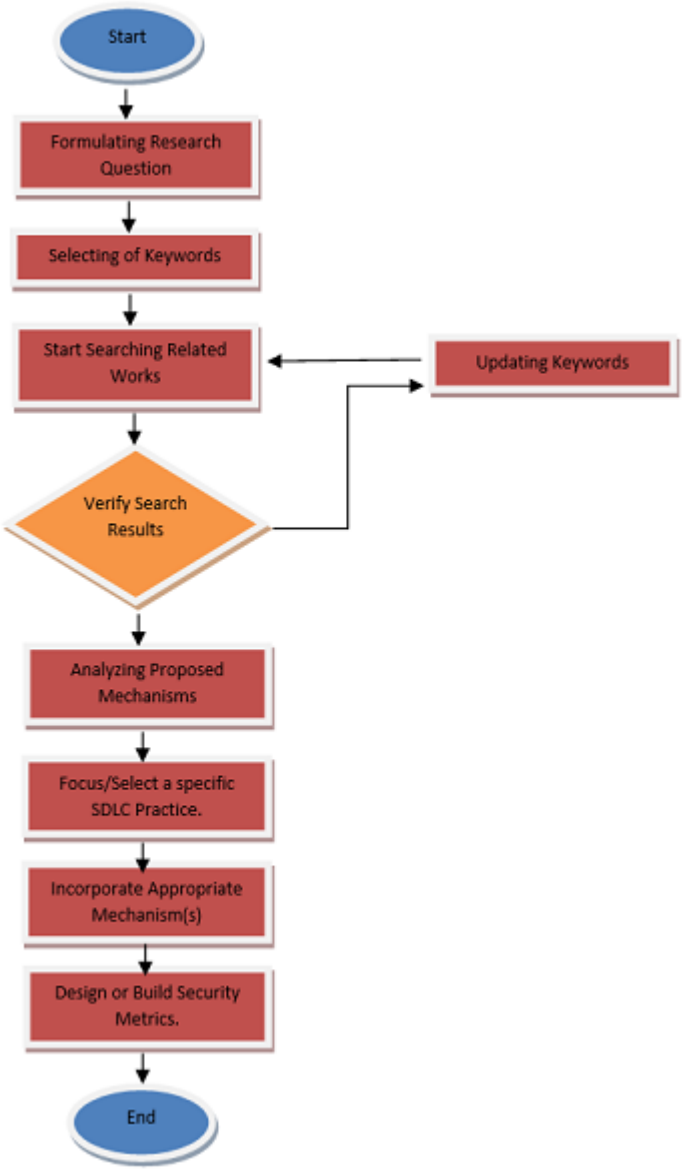


Fig. 2. Different steps involved in our extended SLS process [18].

**Research Question 1:** What kind of strategies are used to incorporate security aspects into the Agile Software Development Life Cycle?

**Research Question 2:** How do we define or build the security metrics for the modern SDLCs like Agile?

So, we are quite familiar with the traditional Software Development Life Cycles (SDLC) where the process typically starts with an idea of building software or application, then designing it may be on a piece of paper, followed by coding, testing, and eventually deploy it into the unsuspecting real world. As shown in the below figure 3 , the traditional practices follow the approach of inserting security gates across the phases of conventional SDLC. The term gated security deals with the integration of various security practices with the standard development process.

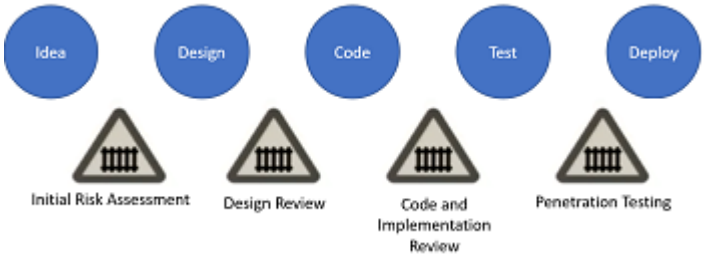


Fig. 3. Gated Security approach in traditional SDLC practice [19].

We initially start with inserting the risk assessment between the ideas and design phases and finally end up injecting penetration testing between the test and deployment phase. The integration of gated security with the traditional SDLC like waterfall makes the process longer for the development of software products or systems. So the biggest challenge is whether the integration of these gated security activities supports the modern Agile development framework. The gated security practices like code reviews and penetration testing might take around 10-days to 3-weeks of time to complete the process of finding bugs and vulnerabilities in a software product [20]. But this is one of the significant reasons behind not integrating the gated security practices where you are working on a two-week iteration process or continuous deployment environment like Agile. Hence, there is a need to rethink the way of integrating the gated security approach for the Agile development life-cycle.

Before we look into the integration, the following are some of the common misconceptions which need to be cleared at the initial stage before incorporating the security aspects into the Agile framework [21]:

- Ignoring security and doing it later is not the same as managing security.
- We as a Software Developer or Software Engineer are tiny objects which often fail at building security in the world of the Internet.
- Knowing that providing 100 percent security is impossible, so why do we try knowing that hackers are coming.

So let us consider a case study where an attacker develops an exploit tool that will break into your system by performing some actions using scripts on the network. The most important part is managing hundreds and thousands of these malicious scripts or tools perhaps the attackers out there doing strange things on the internet. One of the biggest myths is that agility increases the amount of risk. The reason

behind this that we can't the traditional gated approach where you can have the penetration testing every time we have a release every two weeks. Now, there is another side to the agility that if we are deploying three or four times a day and we have got mechanisms in place to spot bugs or vulnerabilities in our code quickly, the for identifying these things, resolving, and deploying a newer version is very small. The following are some of the steps that an organization can follow for a better, stronger, and more secure:

- **Knowing Developer Technology Stack :**
  As we know that all the technologies in the current world have security problems but all of these have their abilities and strengths as well. A Software Developer or Engineer needs to know what's there in their stack at every layer right from the operating system till the way up to the libraries, framework, and Content Management Systems (CMS) used to build the software product [22]. Every part of your stack is indeed prone to introduce vulnerabilities and it is essential to keep track of the technologies and keep on updating your source based upon the updated of the technology stack to prevent threats.

- **Add, Adapt and Abandon :**
  When we know there is a process we need to do such as code review or architecture review or penetration testing [22] i.e whenever we have a security aspect that needs to be included, observe the work flow of the software and feel free to add and adapt it to the software product or screw it all together to abandon it entirely. To be more precise, it is suggested not to add or adapt any security aspect with the software product if it does not make any insights.

- **Communication :**
  A Software Developer or Software Engineer needs to handle and prioritize [22] all the information in the following categories:

  - Low Priority
  - Medium Priority
  - High Priority
  - Critical Information
  - Resolve Now
  - Resolve Later
  - Ignore

- **Manage Technical Debt :**
  Managing technical debt is the concept of handling the unnecessary features that you don't need to put or incorporate into your software systems. Injecting undesired features into the software system may lead some vulnerabilities once you pile a set of security aspects on it if do not adapt to it properly [22].

## IV. Security Metrics for Modern SDLC's

In this section, we try to understand the security metrics used throughout the modern Software Development Life-Cycles (SDLCs). These security metrics help in maintaining informed security during the design, development, and deployment phases of every modern SDLCs. According to Jaquith, "a metric is a measure that quantifies the resultant data by providing some insights." [23]. Below given figure 4 represents a formal description of showing a metric. A metric can store several dimensions of a specific task or an object.

| | |
|---|---|
| **Metric** | Name of the metric |
| **Metric Description** | Description of what is measured |
| **Measurement Procedure** | How is the metric measured |
| **Measurement Frequency** | How often is the measurement taken |
| **Thresholds Estimation** | How are the thresholds calculated |
| **Current Thresholds** | Current range of values considered |
| **Target Value** | Best possible value of the metric |
| **Units** | Units of measurement |

Fig. 4. Formal Description of a Metric [24].

Security Metrics are measures that help various organizations and executives to understand the level of application security, risks, cost, and various other aspects as well. The reason behind having security metrics is that the business of the organizations wants to look into the risk management of the newly built software product and security aspects of the application. To build a security metric for software application we follow the approach as stated in figure 5 that consists of three phases.

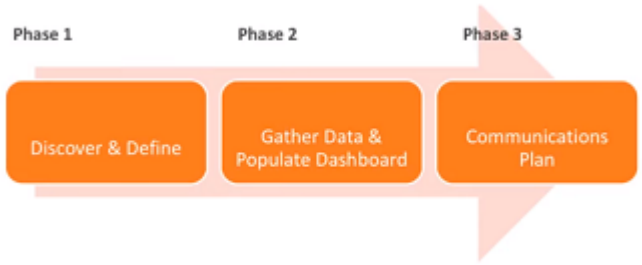


Fig. 5. Phases of Building a Security Metric [25].

In the initial phase, we define and discover your security metric, then followed by gathering data related to the possible threats and vulnerabilities of a software application and finally designing communication plans for solving the problems. Before we build our security metrics, it is necessary that we initially carry out various tests on the newly built software to identify the defects and if there are any defects, then we try to fix and prevent them from happening in the future.This approach helps in evaluating the capability and maturity of the software product.

## V. Conclusion

The primary focus behind adopting these Secured Software Development Lifecycle (SSDLC) is to reduce the number of software vulnerabilities at the early stages of developing the software. This could benefit most of the organization outside the world in terms of cost and reliability of their product in the market. Since Agile is one of the most commonly used methodologies used for developing software products, we have investigated different methodologies of embedding security practices into the Agile Software Development model. Coming down the line we will also make a study on various other software development models that could improve the design and build quality of their software product by enhancing security aspects as well.

## References


[1] A. Senarath and N. A. G. Arachchilage, The Unheard Story of Organizational Motivations Towards User Privacy, Security, Privacy, and Forensics Issues in Big Data Advances in Information Security, Privacy, and Ethics, pp. 280303, 2020.

[2] M. G. Nagler, Negative Externalities, Competition And Consumer Choice*, The Journal of Industrial Economics, vol. 59, no. 3, pp. 396421, 2011.

[3] S. Brooks, M. Garcia, N. Lefkovitz, S. Lightman, and E. Nadeau, An introduction to privacy engineering and risk management in federal systems, 2017.

[4] R. Madge, GDPR's global scope: the long story, Medium, 26-May-2018. [Online]. Available: https://medium.com/mydata/does-the-gdpr-apply-in-the-us-c670702faf7f. [Accessed: 15-Sep-2019]

[5] J. Yang, A. Lodgher, and Y. Lee, Secure Modules for Undergraduate Software Engineering Courses, 2018 IEEE Frontiers in Education Conference (FIE), 2018.

[6] M. C. Oetzel and S. Spiekermann, A systematic methodology for privacy impact assessments: a design science approach, European Journal of Information Systems, vol. 23, no. 2, pp. 126150, 2014.

[7] R. Selby, Enabling reuse-based software development of large-scale systems, IEEE Transactions on Software Engineering, vol. 31, no. 6, pp. 495510, 2005.

[8] A. Steffens, H. Lichter, and J. S. Dring, Designing a next-generation continuous software delivery system, Proceedings of the 4th International Workshop on Rapid Continuous Software Engineering - RCoSE 18, 2018.

[9] M. Kumar Sharma, A study of SDLC to develop well engineered software, International Journal of Advanced Research in Computer Science, vol. 8, no. 3, pp. 14, 2107.

[10] Mohamed Sami, Choosing the right Software development life cycle model, Mohamed Sami, 18-Aug-2019. [Online]. Available: https://melsatar.blog/2012/03/21/choosing-the-right-software-development-life-cycle-model/. [Accessed: 16-Sep-2019].

[11] What are the SDLC phases?, GoodFirms, 26-Jul-2017. [Online]. Available: https://www.goodfirms.co/glossary/sdlc/. [Accessed: 15-Sep-2019].

[12] P. K. Misra, Aspects of Enhancing Security in Software Development Life Cycle, Advances in Computational Sciences and Technology, vol. 10, pp. 203210, 2017.

[13] G. Raj, D. Singh, and A. Bansal, Analysis for security implementation in SDLC, 2014 5th International Conference - Confluence The Next Generation Information Technology Summit (Confluence), 2014.

[14] M. Khari, Vaishali and P. Kumar, "Embedding security in Software Development Life Cycle (SDLC)," 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, 2016, pp. 2182-2186.

[15] N. S. A. Karim, A. Albuolayan, T. Saba, and A. Rehman, The practice of secure software development in SDLC: an investigation through existing model and a case study, Security and Communication Networks, vol. 9, no. 18, pp. 53335345, 2016.

[16] C. Muniraman, M. Damodaran, "A Practical Approach to Include Security in Software Development", Issues in Information Systems, vol. 8, no. 2, pp. 193-199, 2007.

[17] R. Khaim, S. Naz, F. Abbas, N. Iqbal, and M. Hamayun, A Review of Security Integration Technique in Agile Software Development, International Journal of Software Engineering Applications, vol. 7, no. 3, pp. 4968, 2016.

[18] B. Kitchenham, R. Pretorius, and D. Budgen, Systematic literature reviews in software engineering A tertiary study, Information and Software Technology, vol. 52, pp. 792805, 2010.

[19] Joshi, S.D.. (2015). Incorporating Security into SDLC Phases Using Security Analysis. International Journal of Innovative Research in Computer and Communication Engineering. 03. 6423-6431. 10.15680/ijircce.2015.0307010.

[20] Efe, Ahmet Mhrdarolu, Nisanur. (2018). Secure Software Development in Agile Development Processes of E-Government Applications. The Journal of International Scientific Researches. 3. 73-84. 10.23834/isrjournal.396735.

[21] Madampe, Kashumi. (2017). Successful Adoption of Agile Project Management in Software Development Industry. International Journal of Computer Science and Information Technology Research. 5. 27-33.

[22] R. Sass, How to Balance Between Security and Agile Development the Right Way, WhiteSource, 15-May-2019. [Online]. Available: https://resources.whitesourcesoftware.com/blog-whitesource/how-to-balance-between-security-and-agile-development-the-right-way. [Accessed: 03-Dec-2019].

[23] G. Hayslip, G. Hayslip, and IDG Contributor Network, Security metrics: telling your value story, CSO Online, 06-Feb-2018. [Online]. Available: https://www.csoonline.com/article/3253332/security-metrics-telling-your-value-story.html. [Accessed: 03-Dec-2019].

[24] McQueen, Miles Boyer, Wayne McBride, Sean Farrar, Marie Tudor, Zachary. (2008). Measurable Control System Security through Ideal Driven Technical Metrics.

[25] Jain, Smriti and Maya Ingle. A Review of Security Metrics in Software Development Process. (2011).