Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 4, June 2021:894- 901

Research Article

Estimation of Software Security Risks through CVSS: A Design Phase Perspective

Syed Anas Ansar[a] , Savarni Prakas Srivastava[b] , Jaya Yadav[c] , Mohd. Waris Khan[b] Amitabha Yadav[c] , Raees Ahmad Khan[a]

[a]Department of Information Technology, Babasaheb Bhimrao Ambedkar University, Lucknow 226025, India
[b]Department of Computer Application, Integral University, Lucknow 226026, India
[c]Department of DDUKK, National Post Graduate College, Lucknow 226001, India

Abstract

In today's world, the software makes work more straightforward and more manageable for users, employees, and organizations. It makes our working environment more comfortable, but on the other hand, it has security issues. Over the last two decades, software security has become a prime focus for security organizations as well as security practitioners. As the growth of software increases, the level of security risks compromised with software also increases. In most cases, 'compromising in design' is one of the critical security risks. Software security risks are the weakness in the software that accidentally allow hazardous operations. To give prime concern to the security mechanism, the valuable assets must be protected. Hence, prompt detection and remediation of software security risks is a crucial issue in software security. In this paper, the researchers have laid stressed on Software security risk at the design phase and listed some common software security risks from CWE, i.e., Common Weakness Enumeration. In addition, researchers have also mentioned the CVSS 3.1 vulnerability scoring mechanism and calculated the scores of listed security risks to prioritize the exploitability impact and its base score.

Keywords: Security; SDLC (Software Development Life Cycle); CWE (Common Weakness Enumeration); Security Risk; CVSS (Common Vulnerability Scoring System); CIA (Confidentiality Integrity Availability);

1. Introduction

We are now in the age of digitalization, where software, hardware, and sensors are working together, and nearly all services are provided through computers [1]. Over the last couple of years, the software's role in today's life has grown rapidly. Software provides a vast variety of social, business and financial services, and this dramatic growth in the field of software has led to a revolution in software industries. Software is the decisive factor in the progress of initiatives and business aspects as it is a secret to competitiveness. Generally, it is the control core of complicated structures, offers device functionality and stimulates hardware design. The software drives our world; it allows our banking corporation, power stations, communication channel, and combat or fighting equipment to work. Technology is used in every sector effectively and efficiently, such as in medical equipment, hospitals and critical infrastructures like the electric grid and other 911 systems [2]. Software makes our functioning circumstances more convenient, but on the other hand, it has security issues, and it becomes a primary concern for security experts to deal with it systematically. As we know that secure software development is an arduous as well as an error-prone job, and when we are talking about software, then "data" is a valuable asset.

Nowadays, the software has emerged as the organization's "new-perimeter." financial and customer assets, i.e., "information," have become an important entity, and applications must be secured sufficiently to guard them [3]. So, it is very important to build a proper mechanism to protect these valuable assets. As the software covered a large scope, the range of security vulnerabilities and simplicity of applications have increased due to increasing network access. To use a device's applications, cybercriminals don't need to have physical access to that device [4]. Growing adoption and deployment of software undermines user's security as well as privacy [5]. The urge to maintain security in this current scenario is a prime consideration. Software security risk management comprises of methods which are used to access as well as control risk. Risk identification and prioritization and its analysis are the sub-steps of risk assessment, while risk monitoring, risk management, planning, and risk resolutions are the sub-step of risk control [6]. Risk management is an approach in which project members regularly laid down what could adversely impact the projects. The development of a risk management system is a significant task of maintaining security [7]. It gives disciplined surroundings for strategic decision-making to continuously access what can go incorrect and recognize the risks that are important to address and enforce actions to address them.

There is a lack of understanding of software security in the early phase of the Software Development Life Cycle (SDLC), which should be illuminated as well as handled. Researchers, therefore, failed to create a stable program when implementing best practice software engineering. Not just integrating security requirements right from the beginning of SDLC but also ensures secure software. According to Baker et al., there is a sufficiency of security procedures that help secure software development, but appropriate security quantification tools are scarce [8]. Some researchers have suggested that the security risks must be considered at each and every phase of SDLC and mentioned that the idea of rectification of requirement and design process to bring focus at early phases of SDLC [9]. A number of different methods, as well as procedures so far to measure software security risks, among them a few, are based on CVSS scores and selection benchmarks. C. Wang and W. A. Wulf, as well as R. Ortalo et al., have recommended a technique for the estimation of security risk on the basis of the weakest-link principle [10, 11]. This paper discusses the vulnerabilities of the design phase and evaluated their Base, Temporal as well as Environmental scores by using the CVSS mechanism.

This paper is closely knitted as Section 2 discusses some common software security risks at the design phase along with the evaluation mechanism of CVSS as well the evaluated result of the given security risks with the help of the CVSS 3.1 mechanism. Section 3 shows some significant findings, and at last, section 4 describes the conclusion as well as future work.

2. Major Security Risk

The CWE (Common Weakness Enumeration) provides a list of vulnerabilities in the Design Phase based on vulnerabilities detection, prevention, and mitigation. Few of the vulnerabilities were discussed below.

Table 1. Some Common Software Security Risks at Design Phase

CWE ID	Risk in Design Phase	Likelihood of Exploit	Scope Factors
CWE-640	Mechanism of Weak Password Recovery for Forgotten Password	High	Access Control, Integrity and Availability.
CWE-311	Missing Encryption of Sensitive Data	High	Integrity, Confidentiality.
CWE-494	Download of Code Without Integrity Check	Medium	Confidentiality, Integrity, Availability.
CWE-416	Use After Free	High	Confidentiality Integrity, Availability.
CWE-327	Use of Broken or Risky Cryptographic Algorithm	High	Confidentiality, Integrity, Accountability, and Non-Repudiation.
CWE-426	Untrusted Search Path	High	Confidentiality, Integrity, Availability, Access Control.
CWE-89	Improper Neutralization of Special Elements used in an SQL Command "SQL Injection"	High	Confidentiality, Integrity, Access Control.
CWE-404	Improper Resource Shutdown or Release	Medium	Confidentiality and Availability.

As we have seen above, the number of vulnerabilities introduced in the design phase of software as well as it has a different impact factor based on the CIA. For the assessment or evaluation of the severity level of these vulnerabilities in the software security, FIRST (Forum Incident Response and Security Teams) introduced an open framework named CVSS (Common Vulnerability Scoring System). FIRST.Org is a US-based, non-profit corporation owning and managing CVSS [12]. The several security metrics consider the scores computed with the help of CVSS. CVSS determines the three main rating scores of severities by using an algorithm. The rating score of Metric Group (MG) are Base MG, Temporal MG, and Environmental MG; its score range is varying from 0.0 to 10.0 with the most extreme scores of 10.0 [13]. Many reputed firms, such as NVD, Oracle, and IBM organizations, use the CVSS standard [14]. CVSS has been widely accepted since its initial publication in 2004; research conducted in 2003/2004 through NIAC (National Infrastructure Advisory Council) resulted in the launch of Version 1, i.e., CVSSv1, in the month of February 2005. This initial draft had not been subject to peer review or review by other organizations. NIAC selected the FIRST as the guardian of CVSS for future creation in April 2005 [15]. After all these changes, research started in April 2005 on CVSSv2 (CVSS version 2) and was released in June 2007 on the final specification [16]. In 2012, additional feedback led to work starting with version 3 of CVSS (CVSSv3.0). The latest version of CVSS is CVSSv3.1 that was released in June 2019, and it is now available [17]. For a comprehensive CVSS overview, researchers have provided a detailed diagram to evaluate the score of CVSS.

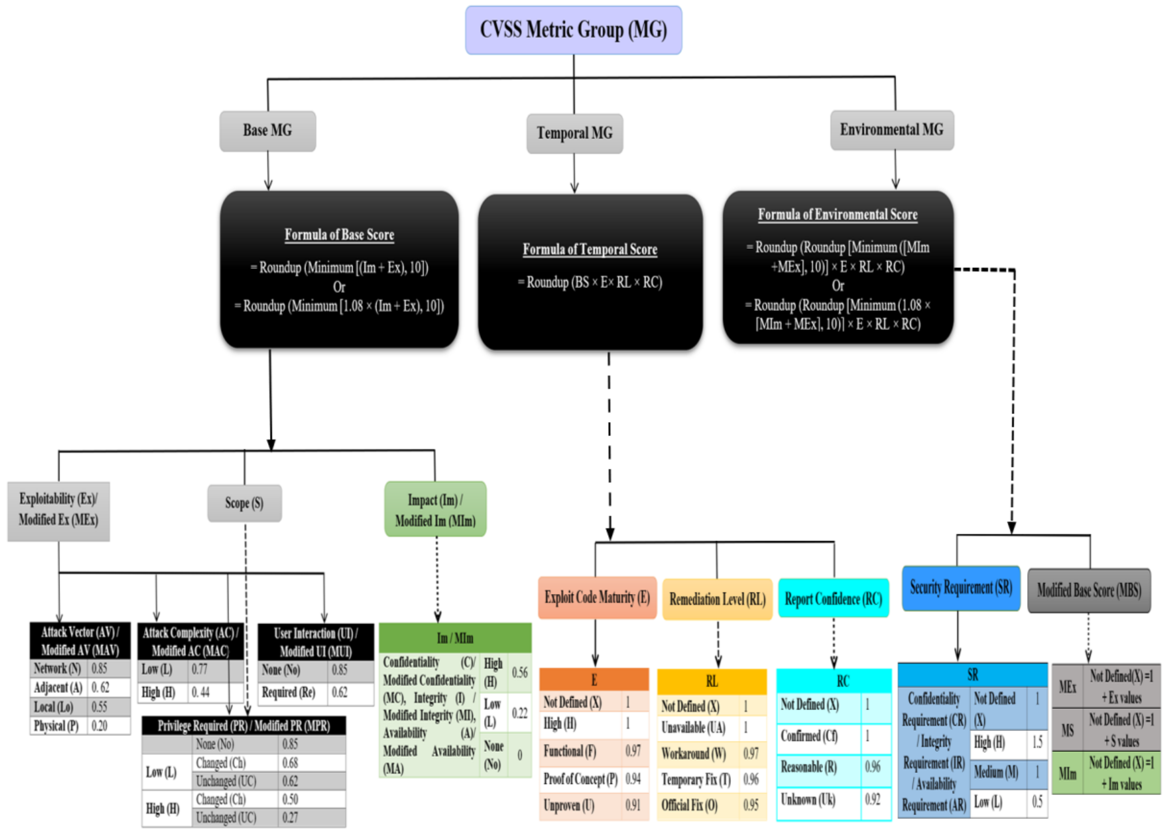
CVSS Metric Group (MG)
Base MG
Temporal MG
Environmental MG
Formula of Base Score
= Roundup (Minimum [(Im + Ex), 10])
Or
= Roundup (Minimum [1.08 × (Im + Ex), 10])
Formula of Temporal Score
= Roundup (BS × E × RL × RC)
Formula of Environmental Score
= Roundup (Roundup [Minimum ([MIm +MEx], 10)] × E × RL × RC)
Or
= Roundup (Roundup [Minimum (1.08 × [MIm + MEx], 10)] × E × RL × RC)
Exploitability (Ex)/ Modified Ex (MEx)
Scope (S)
Impact (Im) / Modified Im (MIm)
Attack Vector (AV) / Modified AV (MAV)
Network (N) 0.85
Adjacent (A) 0. 62
Local (Lo) 0.55
Physical (P) 0.20
Attack Complexity (AC) / Modified AC (MAC)
Low (L) 0.77
High (H) 0. 44
User Interaction (UI) / Modified UI (MUI)
None (No) 0.85
Required (Re) 0.62
Privilege Required (PR) / Modified PR (MPR)
None (No) 0.85
Low (L) Changed (Ch) 0.68
Unchanged (UC) 0.62
High (H) Changed (Ch) 0.50
Unchanged (UC) 0.27
Im / MIm
Confidentiality (C)/ Modified Confidentiality (MC), Integrity (I) / Modified Integrity (MI), Availability (A)/ Modified Availability (MA)
High (H) 0.56
Low (L) 0.22
None (No) 0
Exploit Code Maturity (E)
Remediation Level (RL)
Report Confidence (RC)
Security Requirement (SR)
Modified Base Score (MBS)
E
Not Defined (X) 1
High (H) 1
Functional (F) 0.97
Proof of Concept (P) 0.94
Unproven (U) 0.91
RL
Not Defined (X) 1
Unavailable (UA) 1
Workaround (W) 0.97
Temporary Fix (T) 0.96
Official Fix (O) 0.95
RC
Not Defined (X) 1
Confirmed (Cf) 1
Reasonable (R) 0.96
Unknown (Uk) 0.92
SR
Confidentiality Requirement (CR) / Integrity Requirement (IR) / Availability Requirement (AR)
Not Defined (X) 1
High (H) 1.5
Medium (M) 1
Low (L) 0.5
MEx Not Defined(X) =1 + Ex values
MS Not Defined (X) =1 + S values
MIm Not Defined (X) =1 + Im values

Figure 1. CVSS Computation Mechanism

The value of all three scores, i.e., (BS) Base Score, (TS) Temporal Score, as well as (ES) Environmental Score, is important to evaluate the vulnerability CVSS Score. The final CVSS score is determined by vulnerability information of **BS** and **TS**, provided by security service as well as by Host/Asset group of **ES**, provided by user [18].

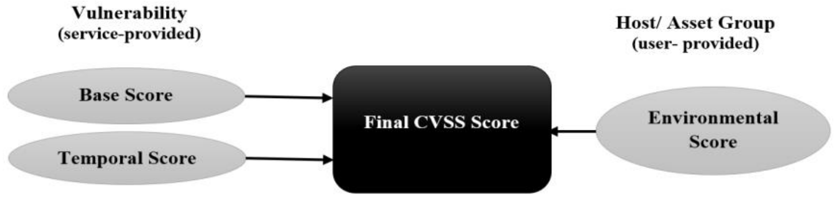

Figure 2. Major Components of CVSS

For measuring the final CVSS score, it questioned about the BS and TS score from service provided to determine the scores of vulnerabilities, including the value from the Environmental metrics provided by users from ES. Base MG demonstrates the vulnerability characteristics that are intrinsic and consistent over time between various user environments [19]. It consists of two metrics group- Exploitability, Scope and Impact MG. Exploitability demonstrates how easy it is to use the vulnerability of any software to exploit its component. The value of Exploitability is determined by calculating the value of Attack Vector (AV), Attack Complexity (AC), Privileges Required (PR), and User Interaction (UI). AV can be exploited at Network layer (N) of OSI, or via the mutual physical network at Adjacent Vector (A), or locally based on Local vector (L) by user interactions, or it can be physically manipulated and managed by the attacker at Physical Vector (P). AC defines the circumstances that are beyond the control of an attacker to take advantage of weakness. If an attacker will be able to succeed to exploit the weakness more than once, then the AC score is Low (L); otherwise, it is High (H) [20]. PR indicates the number of privileges that an attacker would successfully exploit. It defined three values- None (No), Low (L), and High (H). Suppose an attacker is unauthorized before any cyber-attack and does not need any access. In that case, it is None (No), in the case to access only non-sensitive resources, then it is called Low (L), and if vulnerable component allowing access to wide component and files, then it is High (H). In Low and High value, scope metric is also defined within PR named as - Changed (Ch), and Unchanged (UC) [6]. UI metric is user-oriented, which can be exploited and determines whether a vulnerability can exclusively be taken into account by the attacker or involvement of a separate user in some manner [10]. It can be evaluated on the basis of two values None (No) and Required (Re). The Impact MG measures the impact and implications for the components affected in one way or another way by the successful vulnerability exploitation. Its final result is calculated with three Impact MGs together, that is well-known CIA (Confidentiality, Integrity, and Availability) triad. "*For evaluating the Base MG score, first we have to calculate Impact Sub Score (ISS) or final impact that can be measured by CIA triad, ISS would help to evaluate its Impact score and Exploitability score. On the basis of Impact (Im) and Scope (S), we get the value of Base score*" [21].

Table 2. Conditions for the Calculation of CVSS.

Base MG	**Temporal MG**	**Environmental MG**

Base MG depends on three parameters [24]-
✓ Impact Sub Score (ISS)
✓ Impact (Im)
✓ Exploitability (Ex)
 	
BaseMG()
{
string S;
float ISS, Im, Ex, BS;
ISS= 1-((1-C)*(1-I)*(1-A));
if(S="Ch")
{
Im= (7.52*(ISS-0.029)) –(3.25*((ISS-0.02)^15)));
}
else if (S="UC")
{
Im=(6.42*ISS);
	Temporal MG depends on four parameters [24]-
	✓ Base Score (BS)
	✓ Exploit Code Maturity (E)
	✓ Remediation Level (RL)
	✓ Report Confidence (RC)
	
	TemporalMG()
	{
	float TS;
	TS= Round(BS*E*RL*RC);
	}
		Environmental MG depends on three parameters [24]-
		✓ Modified Impact Sub Score (MISS)
		✓ Modified Impact (MIm)
		✓ Modified Exploitability (MEx)
		EnvironmentalMG()
		{
		string MS;
		float MISS, MIm, MEx, ES;
		MISS= min(1.0-((1.0-(CR*MC))*(1.0-(IR*MI))*(1.0-(AR*MA)),0.915);
		if(MS="Ch")
		{
		MIm= (7.52*(MISS-0.029)) – (3.25*(((MISS*0.9731)-0.02)^13)));
		}
		else if (S="UC")
		{

```
}
Ex= 8.22*AV*AC*PR*UI;
if (Im<=0)
{
BS=0;
}
else if( S=’Ch”)
{
BS= round(min(1.08*(Im+ Ex),10.0)));
}
else if (S=’UC” )
{
BS= round(min((Im+ Ex),10.0)));
}
}
MIm=(6.42*MISS);
}
MEx= 8.22*MAV*MAC*MPR*MUI;
if (MIm<=0)
{
ES=0;
}
else if( S=’Ch”)
{
ES=            round(round(min(1.08*(MIm+
MEx),10.0)) *E*RL*RC);
}
else if (S=’UC” )
{
ES= round(round(min((MIm+ MEx),10.0))
*E*RL*RC);
}
}
```

The Temporal MG demonstrates the vulnerability characteristics which are not consistent and can change over time between various user environments [8]. It consists of three metrics group- Exploit Code Maturity (E), Remediation Level (RL), and Report Confidence (RC). In order to determine the score of Temporal MG, the value of the Base Score (BS) must be determined first. After that, by using the score of BS, E, RL, and RC, the Temporal score (TS) is calculated. On the basis of their metric value such as Not Defined (X), High (H), Functional (F), Unavailable (UA), Workaround (W), Temporary Fix(T), Confirmed (Cf), and Reasonable (R) etc. numerical value is used which is defined as its metric value is constant [22].

Environmental MG demonstrates the vulnerability properties that are important and uniquely relevant to a particular or specific user [8]. It consists of two metrics group- Security requirements (SR) and Modified Base Score (MBS). SR score is determined by a well-known CIA triad that can be measured in three possible values, i.e., Low (L), Medium (M), and High (H). SR is an indicator of the business criticality of an asset, measured in terms of CIA. The overall environmental impact is measured by the corresponding MBS, which can be evaluated in the same manner as Exploitability, and Impact metrics are measured except there is one modification it defined new value, i.e., Not Defined (X). This means that these measurements amend the environmental metric by overriding the Base MG before the Security requirements (SR) are enforced in Environmental metrics. “*For evaluating the Environmental MG score, first we have to calculate Modified Impact Sub Score (MISS) that can be measured by CR, MC, IR, MI, AR, and MA, MISS would help to measure its Modified Impact score (MIm) and Modified Exploitability score (MEx). On the basis Modified Impact (MIm) and Modified Scope (MS), we get the value of Environmental score*” [23].

Researchers have calculated all the three CVSS scores, i.e., Base, Temporal and Environmental Scores of above software security risks mentioned in Table 1 by using ***CVSS version 3.1*** are mentioned in Table 3, Table 4 and Table 5, respectively.

Table 3. Calculated Base Score of Security Risks Factors mentioned in Table 1.

CWE ID	AV	AC	PR	UI	S	C	I	A	SCORE
CWE-640	N	L	No	No	U	H	H	H	9.8
CWE-311	N	L	No	No	U	H	No	No	7.5
CWE-494	Lo	L	L	No	U	H	H	H	7.8
CWE-416	Lo	L	No	Re	U	No	No	H	5.5
CWE-	Lo	L	L	No	U	H	H	H	7.8

327									
CWE-89	N	L	L	No	U	H	H	H	8.8
CWE-426	Lo	L	L	No	U	H	H	H	7.8
CWE-404	N	L	No	No	U	No	No	H	7.5

Table 4. Calculated Temporal Score of Security Risks Factors mentioned in Table 1.

CWE ID	E	RL	RC	SCORE
CWE-640	H	O	UA	8.6
CWE-311	H	UA	UA	6.9
CWE-494	H	O	Cf	7.5
CWE-416	H	X	UA	5.1
CWE-327	H	UA	R	7.5
CWE-89	H	UA	R	8.5
CWE-426	X	T	Cf	7.5
CWE-404	H	O	UA	6.6

Table 5. Calculated Environmental Score of Security Risks Factors mentioned in Table 1.

CWE ID	CR	IR	AR	MAV	MAC	MPR	MUI	MS	MC	MI	MA	SCORE
CWE-640	H	H	H	N	L	No	No	U	H	H	H	8.6
CWE-311	M	H	H	N	L	X	Re	U	L	L	H	8.1
CWE-494	H	H	H	N	L	No	Re	U	L	L	H	8.4
CWE-416	M	L	H	X	L	X	Re	U	L	No	H	6.9
CWE-327	M	M	H	A	L	No	Re	U	L	H	H	7.7
CWE-89	H	H	H	Lo	L	No	Re	U	L	H	H	7.5
CWE-426	X	M	H	A	L	No	No	X	L	H	H	8.5
CWE-404	M	M	L	P	L	No	No	U	H	H	L	5.6

3. Major Findings

Nowadays, the emergence of software creates a new way in an organization and makes work more straightforward, but as well as it has also created a drastic problem, i.e., security risks. Security risks in software applications became a primary concern for organizations, researchers, and security experts all around the world. Some of the significant findings given below:

- Vulnerability, i.e., risks within the software, must be identified because it is a weakness within the system, and attackers can exploit this to disrupt the process.
- In the majority of software, security procedures are not taken from the initial phase of SDLC.
- To mitigate risk, security effectiveness needs to be strengthened, and this needs to prior identification of threats and vulnerability assessment to determine significant risks for secured systems.
- It is clear from the literature survey we cannot avoid security risks altogether after taking all preventive measures.
- Assessment of the IT risk is not something that can be taken lightly. The assessment must be reviewing its function periodically to manage security.
- For the development of an appropriate framework to reduce risk, researchers must integrate the lowest-cost strategy and introduce the most effective controls, with minimal effects on the organization's resources.

4. Conclusion and Future Work

Recently, we have seen tremendous growth in the field of software. Nowadays, technology has made complicated work much more accessible and more straightforward, yet any untidiness in software development leads to a matter of life and death. As we know that secure software development is an arduous as well as an error-prone job, and when we are talking about software, then “data” is a valuable asset. So, it is very important to build a proper mechanism to protect these valuable assets. The security risks need to be assessed in a regular period and must be prior aware in the initial phase of software development. In addition, researchers have suggested that it should not be taken lightly by the security experts. It is clear from the literature survey various software security frameworks have been developed for risk assessment and risk management regularly. A maximum among them is in vague condition, and a very few of them are tested on a large scale. So, there is an urgent need to develop a framework to reduce risk, which uses a low-cost strategy, introduces the most effective controls with minimal effects, and should be tested on a large scale. This paper represents a systemic identification of software security risks at the design phase (i.e., from CWE) and evaluation (i.e., on the basis of CVSS 3.1). In addition, researchers have suggested that if the security issues are taken into measure from the beginning of SDLC, then it will undoubtedly assist in the mitigation of security infringements. So, as a result, an appropriate approach for developing secure software must be taken into account. In the future, researchers are planning to develop a framework to provide optimum security by minimizing the risk to provide an efficient, secure, and reliable mechanism.

References

1. Rehman, Shafiq Ur & Gruhn, Volker. (2018). An Effective Security Requirements Engineering Framework for Cyber-Physical Systems. Technologies. 6. 65. 10.3390/technologies6030065.
2. Available at: https://www.veracode.com/blog/managing-appsec/role-applications-todays-digital-world#:~:text=Software%20is%20truly%20powering%20our,to%20operate%20effectively%20and%20efficiently].
3. Al, Abdullah & Chowdhury, Murad & Arefeen, Shamsul. (2020). Software Risk Management: Importance and Practices.
4. McGraw, Gary. (2002). Managing Software Security Risks. Computer. 35. 99 - 101. 10.1109/MC.2002.993782
5. Brass, Irina & Sowell, Jesse. (2020). Adaptive governance for the Internet of Things: Coping with emerging security risks. Regulation & Governance. 10.1111/rego.12343
6. Madachy, Raymond. (1994). A software project dynamics model for process cost, schedule and risk assessment /.
7. Semin, Valeriy & Shmakova, Elena & Los, Alexei. (2017). The information security risk management. 106-109. 10.1109/ITMQIS.2017.8085774.
8. W. Baker, L. Rees and P. Tippett, Necessary measures: Metrics-driven information security risk assessment and decision making, Communications of the ACM, vol.50, no.10, pp.101-106, 2007.
9. P. T. Devanbu and S. Stubblebine, Software engineering for security: A roadmap, Proc. of the Conference on the Future of Software Engineering, Limerick, Ireland, pp.227-239, 2000.
10. C. Wang and W.A. Wulf, "Towards a framework for security measurement," in 20th National Information Systems Security Conference, Baltimore, MD, pp. 522-533, 1997
11. R. Ortalo, Y. Deswarte and M. Kaâniche, "Experimenting with quantitative evaluation tools for monitoring operational security," Software Engineering, IEEE Transactions On, vol. 25, pp. 633-650, 1999.
12. Available at: https://beyondsecurity.com/vulnerability-assessment-requirements-cvss-explained.html?cn-reloaded=1, Last visited 20 Nov. 2020.
13. Available at: https://searchsecurity.techtarget.com/definition/CVSS-Common-Vulnerability-Scoring-System, Last visited 20 Nov. 2020.
14. Available at: https://www.imperva.com/learn/application-security/cve-cvss-vulnerability/, Last visited 20 Nov. 2020.
15. Available at: https://www.first.org/cvss/v1/, Last visited 20 Nov. 2020.
16. Available at: https://www.first.org/cvss/v2/history, Last visited 20 Nov. 2020.

17. Available at: https://www.first.org/cvss/v3-0/, Last visited 20 Nov. 2020.
18. Available at: https://qualysguard.qg2.apps.qualys.com/qwebhelp/fo_portal/setup/cvss_scoring.htm, Last visited 20 Nov. 2020.
19. Available at: https://www.oracle.com/security-alerts/cvssscoringsystem.html, Last visited 20 Nov. 2020.
20. Available at: https://www.balbix.com/insights/base-cvss-scores/, Last visited 20 Nov. 2020.
21. Available at: https://medium.com/habilelabs/cvss-calculator-software-vulnerability-scoring-process-25c6a3356751, Last visited 20 Nov. 2020.
22. Available at: https://www.balbix.com/insights/temporal-cvss-scores/, Last visited 20 Nov. 2020.
23. Available at: https://www.balbix.com/insights/environmental-cvss-scores/, Last visited 20 Nov. 2020.
24. Available at: https://www.first.org/cvss/specification-document, Last visited 20 Nov. 2020.