Towards a Quantitative Approach for Security Assurance Metrics

Goitom K. Weldehawaryat and Basel Katt

Department of Information Security and Communication Technology
Norwegian University of Science and Technology
Gjøvik, Norway
Email: {goitom.weldehawaryat,basel.katt}@ntnu.no

***Abstract*—The need for effective and efficient evaluation schemes of *security assurance* is growing in many organizations, especially Small and Medium Enterprises (SMEs). Although there are several approaches and standards for evaluating application security assurance, they are *qualitative* in nature and depend to a great extent on manually processing. This paper presents a *quantitative evaluation approach* for defining security assurance metrics using two perspectives, *vulnerabilities* and *security requirements*. While vulnerability represents the negative aspect that leads to a reduction of the assurance level, security requirement improves the assurance posture. The approach employs both *Goal Question Metric* (GQM) and *Common Vulnerability Scoring System* (CVSS) methods. GQM is used to construct measurement items for different types of assurance metrics and assess the fulfillment of security requirements or the absence of vulnerabilities, and CVSS is utilized to quantify the severity of vulnerabilities according to various attributes. Furthermore, a case study is provided in this work, which measures and evaluates the security assurance of a *discussion forum* application using our approach. This can assist SMEs to evaluate the overall security assurance of their systems, and result in a measure of confidence that indicates how well a system meets its security requirements.**

Keywords–Quantitative security assurance metrics; Security testing; Goal question metric (GQM); Common vulnerability scoring system (CVSS); Security metrics.

I. INTRODUCTION

Our society has become increasingly dependent on the reliability and proper functioning of a number of interconnected infrastructure systems [1]. The security of most systems and networks depends on the security of the software running on them. This holds also for web applications that are usually accessible in public networks. However, most of the attacks on these systems occur due to the exploitation of vulnerabilities found in their software applications. The number of vulnerabilities also increase as the systems become more complex and connected [2].

Many organizations are implementing different security processes and procedures to secure their systems; however, some organizations need some evidence that show the security mechanisms are effectively put in place and carry out their intended functions to prevent, detect or divert a risk or to reduce its impact on assets [3][4]. Thus, it is important for organizations to know, on one hand, if their systems are vulnerable to threats, and on the other hand if the protection security mechanisms are effective to fulfill the security requirement and mitigate the threats [5].

Security assurance is the confidence that a system meets its security requirements [6]. The confidence is based on specific metrics and evidences gathered and evaluated with given assurance techniques, e.g., formal methods, penetration testing, or third-party reviews. The main activities in security assurance include threat and vulnerability analysis, definition of security requirements based on risk, testing, and architectural information of the environment where Target of Evaluation (ToE) resides. Although the research focus has been mainly on developing *qualitative* metrics that usually lead to security assurance levels that are either not accurate or not repeatable, recent efforts within the field have been directed towards utilizing *quantitative* indicators to capture the security state of a particular system [7]. However, the research efforts that applied quantitative methods have been mainly focused on vulnerabilities and to a lesser extent on the understanding of *vulnerability-security requirement interactions* [2][8].

This paper presents a quantitative evaluation approach for defining security assurance metrics that provides a high level security assurance evaluation and distinguishes two perspectives: *security requirement metrics* and *vulnerability metrics*. While vulnerability represents the negative aspect that leads to a reduction of the assurance level, security requirement improves the assurance posture. Specifically, the approach utilizes the GQM to construct measurement items for different types of assurance metrics and assess the fulfillment of security requirements or the absence of vulnerabilities, and the CVSS to quantify the severity of vulnerabilities according to various attributes. Furthermore, this work illustrates a case study on conducting security testing and assurance functions on an example application, *discussion forum*. The main contribution is the development of a quantitative evaluation approach for defining security assurance metrics that enables quantifying and estimating the level of security requirement and the degree of vulnerability severity. The metrics reflect the strengths of the protection mechanisms and the severity of vulnerabilities that impact a target of evaluation.

The rest of the paper is organized as follows. Section II presents a related work. Section III describes the quantitative assurance metrics, while Section IV discusses the security assurance process. Section V presents the case study, and Section VI provides a discussion of the quantitative assurance metrics approach. Finally, Section VII concludes the paper and presents future work.

 ISBN: 978-1-61208-661-3

II. Related Work

Research efforts have been made to address systems security assurance from the software development life cycle down to the operational systems level [6]. The reason for this is that without a rigorous and effective way of dealing with security throughout the system development process, the end product cannot be secure. However, the emphasis on design and process evidence versus actual implementation largely overshadows practical security concerns involving the implementation and deployment of operational systems [9].

A number of frameworks and standards exist for evaluating security assurance [10][11]. Examples include the Systems Security Engineering Capability Maturity Model (SSE-CMM) [10], OWASP's Software Assurance Maturity Model (OpenSAMM) [12], and the Common Criteria (CC) also known as ISO/IEC 15408 [13]. The CC describes a framework in which developers can specify security and assurance requirements that need to be valuated to determine whether a system meets the claimed security. Although evaluation methods are based on guidelines and best practices, they are done manually to a large extend and result in the creation of large amount of documentation. One major criticism against the CC, for example, is that it evaluates the process more than evaluating the implementation. They are also limited to a few application domains, like smart card security [14]. Furthermore, the assurance levels they define, especially those of Open Web Application Security Project (OWASP), CC and OpenSAMM are abstractly defined and have no quantifiable basis to be measured, which makes it harder for the vendors and third-party assessors to measure the actual security impact and confidence.

Recently, some initiatives have been taken towards developing operational methodologies for the evaluation of IT infrastructure security assurance. Pham *et al*. [15] introduce an attack graph based security assurance assessment system based on multi-agents. In their approach, the authors use attack graph to compute an "attackability" metric value (the likelihood that an entity will be successfully attacked) for static evaluation and define other metrics for anomaly detection at run time. Attack surface estimation [8] is another approach aiming at detecting vulnerabilities within a system. It does not evaluate the security directly, but rather estimates the number of access points to the subject system by counting available interfaces, supported protocols, open ports, open sockets, installed software, etc. The Building Security Assurance in Open Infrastructures (BUGYO) project [16][17] can be cited as the first project that proposed a methodology and tool for continuous security assurance evaluation; security assurance evaluation in the context of BUGYO was aimed at probing the security of runtime systems rather than products. This work investigates a quantitative approach for defining security assurance metrics that provides an overall security assurance evaluation of a target of evaluation.

III. Quantitative Assurance Metrics

Security assurance defines the confidence that a system meets its security requirements based on specific evidence provided by the application of assurance techniques (formal methods, penetration testing, etc) [6]. The need to provide organizations with confidence that deployed security measures meet their requirements at runtime has been acknowledged as a crucial issue [16][18][19]. This is because security mechanisms, even properly identified during the risk assessment stage, may still suffer from an inappropriate deployment that may render them less effective. Although the current evaluation methods rely to a great extent on security experts knowledge and are not adapted to real dynamic operational systems [17], recent research efforts have been directed towards utilizing *quantitative* indicators to capture the security state of a particular system [7]. The gathering of measurable evidence is facilitated by the specification of metrics that are necessary for the normalization of the security assurance levels. We consider three key concepts in our metrics specifications: *vulnerability metrics*, *security requirement metrics* and *assurance metrics*. These metrics will be described in the following subsections.

A. Security Requirement Metrics

Security requirements are associated to the protection of valuable assets in a system. Many authors implicitly assume that security requirements are identical to high-level security goals. Tettero [20] defined security requirements as the confidentiality, integrity, and availability of the entity for which protection is needed. Devanbu and Stubblebine [21] defined a security requirement as "a manifestation of a high-level organizational policy into the detailed requirements of a specific system". Thus, the aim of defining security requirements for a system is to map the results of risk and threat analyses to practical security requirement statements that manage (mitigate or maintain) the security risks of the target of evaluation.

Security requirement metrics reflect the vendor's confidence in a particular security control employed in the target of evaluation to fulfill one or more *security requirements*. Evaluating the confidence level of a security requirement is twofold. First, we need to check whether the current deployed security protection controls *fulfill* the security requirement. This can be manifested as a set of test cases associated with each security requirement. Second, it can be argued that not all security requirements of one application are equally important [22]. Thus, there is a need to consider the importance, or the *weight*, of each of the security requirements. The *weight* for a requirement represent the level of importance of the this requirement to the application in question.

In order to quantify the *fulfillment* factor of the security requirement metrics, we need to connect each requirement to a set of measurable metrics. This can be done using the Goal Question Metric (GQM) approach [23]. GQM provides a clear derivation from security goals to metrics by developing questions that relate to the goals and are answered by metrics [24]. A GQM approach is a hierarchical structure that defines a top-down measurement model based on three levels [23]:

- *Conceptual level* (Goal)
  A goal is defined for an object for various reasons, with respect to various models of quality, from various points of view and relative to a particular environment. Object of Measurement can be: Products, Processes or Resources. In our context, the main goal is to assess the assurance of the target of evaluation. This goal can be split into sub-goals that represent the identified security requirements.
- *Operational level* (Question)
  A set of questions is used to characterize the way the assessment or the achievement of a specific goal is

going to be performed based on some characterizing model. Test cases defined for each security requirement represent questions in our context.

- *Quantitative level* (Metric)
  A set of metrics is associated with every question in order to answer it in a measurable way. In our context, every question can be assigned to a value (for example, Full=1, Average=0.5, Weak=0), and metrics can be given based on *fulfillment* value to the test case.

As an example of applying the GQM for the *authentication* security requirement for web applications, Table I shows questions and metrics for this case. Based on the previous discussion, we define a security requirement metric (Rm_i) for a given security requirement at a specific time instance as:

$$Rm_i = (w_i \times \sum_{j=1}^{m} f_{ij}) \quad (1)$$

where m represent the number of test cases defined for this security requirement, w is the weight of the requirement and f is the fulfillment factor of the requirement. i is the index of the security requirement, and j is the index of the test cases for the security requirement. GQM is used to measure the fulfillment of the security requirements.

As a result, we define the accumulate security requirement metrics of an application at a specific time instance as:

$$RM = \sum_{i=1}^{n} (Rm_i) \quad (2)$$

where n represents the total number of security requirement defined for the ToE.

B. Vulnerability Metrics

A *vulnerability* is defined as a bug, flaw, behaviour, output, outcome or event within an application, system, device, or service that could lead to an implicit or explicit failure of confidentiality, integrity or availability [25]. Thus, vulnerability is a weakness which allows attacker to reduce a system's security assurance. Since organizations usually operate within limited budgets, they have to prioritize their vulnerability responses based on risk value of each vulnerability.

Vulnerability metrics allows to measure the existence and the severity level of system vulnerabilities. Thus assessing vulnerability metrics is twofold. First, there is a need to check the existence of the different types of vulnerabilities that pose a threat to the application. This can be assessed using the GQM method, in which (1) the goal will be to assess the existence of different vulnerabilities in the ToE, (2) the sub goals represent the vulnerability types that pose threat to the ToE, (3) questions represent the test cases that will check the existence of a given vulnerability type, and finally, (4) the quantified answer to the questions represent the metrics. Second, it is essential to quantify the severity level of vulnerabilities in the context of the ToE, which can be represented by the risk value of the vulnerability. For example, the *Common Vulnerability Scoring System* (CVSS) [25][26] cab be used for this purpose. A CVSS score is a decimal number in the range $[0.0, 10.0]$, where the value 0.0 has no rating (there is no possibility to exploit vulnerability) and the value 10.0 has full score (vulnerability easy to exploit). This score is computed using three categories of metrics, which assess the intrinsic characteristics of the vulnerabilities (*base metrics*), its evolution over time (*temporal metrics*), and the user environment in which the vulnerability is detected (*environmental metrics*). These three metric groups can be used to compute the vulnerability severity level of a target of evaluation.

We define a vulnerability metric (Vm_k) for a given security vulnerability at a specific time instance as:

$$Vm_k = (r_k \times \sum_{l=1}^{p} e_{kl}) \quad (3)$$

where, p represent the number of test cases defined for this vulnerability type, r_k is the risk of the k^{th} vulnerability and e_{kl} is the existence factor for l^{th} test case defined for the k^{th} vulnerability. The existence factor can have three values, 0 means that the test case indicates no vulnerability, 1 indicates the existence of the vulnerability for the test case, and 0.5 indicates the partial existence of the vulnerability for the test case.

Thus, the vulnerability metrics of a system at a specific time instance can be calculated using the risk of vulnerabilities and their existence factor as follows:

$$VM = \sum_{k=1}^{d} (Vm_k) \quad (4)$$

where d represents the total number of vulnerabilities defined for the ToE.

C. Assurance Metrics

Assurance Metrics (AM) determine the actual confidence that deployed countermeasures protect assets from threats (vulnerabilities) and fulfill security requirements. We define assurance metrics as the difference between security *Requirement Metrics* (RM) and *Vulnerability Metrics* (VM). Thus, the assurance metrics can be calculated as follows:

$$AM = RM - VM = \sum_{i=1}^{n} (Rm_i) - \sum_{k=1}^{d} (Vm_k) \quad (5)$$

where, AM is the security assurance metrics at a given time instance, RM is the security requirement metrics at a given time instance, and VM is the vulnerability metrics at a time given instance.

From (5), it can be noticed that AM is *minimum* when the following two conditions are met:

- All security requirements are not fulfilled (RM becomes zero), which causes the value of the first term to be minimum (zero), and
- All possible vulnerabilities exist and all have a maximum risk value. This makes the second term to be maximum (VM).

AM, on the other hand, can be *maximum* if (1) VM is minimum for all vulnerabilities, and (2) the protection mechanisms have been found to be effective to fulfill the defined security requirements (RM is maximum) for all requirements.

 ISBN: 978-1-61208-661-3

TABLE I. EXAMPLE OF GQM IN ASSESSING THE AUTHENTICATION VERIFICATION REQUIREMENTS

Goal		Question	Metrics
Purpose	assessing authentication	Are credentials and all pages/functions that require a user to enter credentials transported using a suitable encrypted link ?	Full, average or weak
Issue or Component	authentication	Do all pages and resources by default require authentication except those specifically intended to be public?	Full, average or weak
Object or Process	web application authentication	Do password entry fields allow, or encourage, the use of long passphrases or highly complex passwords being entered?	Full, average or weak
viewpoint	stakeholder, user, organization	Do all authentication controls fail securely to ensure attackers cannot log in?	Full, average or weak
		Does the changing password functionality include the old password, the new password, and a password confirmation?	Full, average or weak
		Is information enumeration possible via login, password reset, or forgot account functionality?	Full, average or weak

Rating Score: Full=1, Average=0.5, Weak=0

IV. SECURITY ASSURANCE PROCESS

An assurance process defines the different activities that need to be performed in order to assess the level of confidence a system meets its security requirements. Our assurance process deals with three types of metrics: *vulnerability metrics*, *security requirement metrics* and *assurance metrics*. Similar to the methodology defined in [17], the assurance process consists of five main activities: *application modelling, metric selection and test case definition, test case execution and measurement collection, assurance metrics and level calculation, evaluation and monitoring*. The input is an operational system running a target of application to be evaluated.

1) **Application modelling:** The application modelling allows decomposing the application in order to identify critical assets. An efficient way of identifying those critical components is an a priori use of a threat modelling and risk assessment methodology. Security functions and threats related to the basic security concepts of the application and its environment can be analysed. This results the security requirements expected to be present and running correctly in the system to fulfill security goals and protect assets from threats.
2) **Metric selection and test case definition:** A metric is based on the measurement of various parts or parameters of security functions implemented on the system with its service and operational environment. Depending on the measurements being performed, metrics can be classified as follows:
   - *Security requirement metrics* relate to a measurement that evaluates whether security protection mechanisms exist and fulfill defined security requirements using the GQM method.
   - *Vulnerability metrics* relate to a measurement that evaluates the weaknesses/severity and vulnerabilities existence in the systems using the CVSS and GQM methods.

   Test cases for both metrics can be defined to test the vulnerabilities and verify the security requirements on the target of evaluation.
3) **Test case execution and measurement collection:** Test case execution and measurement collection consist in deploying specific probes to implement the test cases on a target of evaluation and its operational environment. These probes can help to collect raw data from the system. This step will result in a measurement that will be normalized to produce an assurance level in step 4.
4) **Assurance metrics and level calculation:** Once the security requirement and vulnerability metrics are determined in step 3, the overall security assurance of the target of evaluation can be calculated using (5) presented in Section III.
5) **Evaluation and monitoring:** This step involves comparing the current value of the assurance level to the previous measure, or to a certain threshold and issuing an appropriate message. It can also provide a real time display of security assurance of the service to help the evaluator identify causes of security assurance deviation and assist him/her in making decisions.

V. CASE STUDY

This section presents the proposed assurance approach and processes applied for the web discussion forum developed for this purpose.

A. Application and Threat Modelling

Figure 1 shows a screenshot of the discussion forum. The forum has users who create topics in various categories, and other users who can post replies. Messages can be posted as either replies to existing messages or posted as new messages. The forum organizes visitors and logged in members into user groups. Privileges and rights are given based on these groups.

The tools used to develop the forum includes *PHP, MySQL*, and *Apache*. *WAMP* was used to do an all-in-one installation of Apache, MySQL, and PHP on a Windows 7 virtual instance. A Kali Linux [27] virtual instance was used as a security testing machine. The discussion application forum was tested using a number of tools such as *OWASP Zed Attack Proxy (ZAP), WebScarab, OpenVAS* and *Sqlmap*, and manual testing since some vulnerability types can only be found through testers observations. The infrastructure required to create a realistic environment for conducting the testing and assurance functions of the ToE was built using OpenStack cloud computing platform [28].

Threat modelling is a systematic process of identifying, analysing, documenting, and mitigating security threats to a software system [29]. Analysing and modelling the potential threats that an application faces is an important step in the process of designing a secure application. Some of these threats are very specific to the application, but other threats

 ISBN: 978-1-61208-661-3

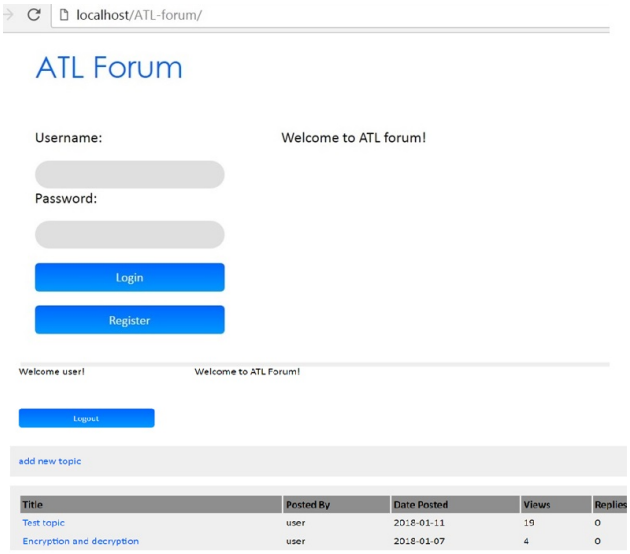

Figure 1. Discussion forum.

are directly or indirectly related to the underlying platforms, technologies or programming languages. The main steps of threat modelling process consists of the following three high-level steps: *characterizing the system, identifying assets* and *access points, and identifying threats* [30].

a) Characterizing the system: Characterizing the system involves understanding the system components and their interconnections, and creating a system model emphasizing its main characteristics. *Data Flow Diagram (DFD)* is used to model the application which dissects the application into its functional components and indicates the flow of data into and out of the various parts of system components. Figure 2 shows a flow diagram of the discussion forum, which was modelled with Microsoft threat modelling tool 2016.

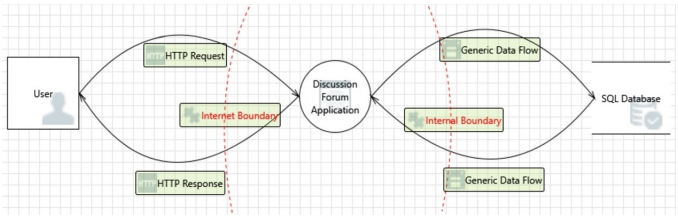

Figure 2. Data flow diagram for the discussion forum.

b) Identifying assets and access points: An asset is an abstract or concrete resource that a system must protect from misuse by an adversary. Identifying assets is the most critical step in threat modelling because assets are threat targets. Examples of identified list of assets of the application that may be targeted by attackers include:

- Forum users and assets relating to forum users
- User login details and the login credentials that a user will use to log into the discussion forum
- The discussion forum website and assets relating to the website

Access (entry) points are interfaces through which potential attackers can interact with the system to gain access to assets. Examples of access points include user login interfaces, HTTP Port, configuration files, file systems and hardware ports. It is also important to determine the *trust boundaries* in the system. A trust boundary is a boundary across which there are varied levels of trust. For example, administrators are trusted to do more than normal users.

c) Identifying threats: A threat is what an adversary might try to do to a system [31]. Threats can be identified by going through each of the identified critical assets and creating threat hypotheses that violate confidentiality, integrity, or availability of the assets. The output of threat identification process is a threat profile for a system, describing all the potential attacks, each of which needs to be mitigated or accepted.

A threat categorization is useful in the identification of threats by classifying attacker goals such as: *Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege* (STRIDE). An incomplete identified list of threats for the application are given in Table III .

Security requirements: Security requirements are driven by security threats. Although security requirements can also be extracted from standards, it is still important to conduct a thorough risk management to discover which threats are realistic, and analyse the suitability of security requirements to the system. However, these security requirements need to be validated and measured to achieve the security goals of the application. In this work, *Application Security Verification Standard (ASVS)* [11] verification requirements are considered as a source of security requirements, and GQM approach is used to measure/verify the fulfillment of these security requirements for the application. Security requirements used for the application are given in Table II.

B. Metric Selection and Test Case Definition

The metrics are categorized as (i) security requirement metrics, (ii) vulnerability metrics, and (iii) security assurance metrics. The first two are discussed in the first two subsections, and the third one is presented in subsequent subsection concentrating on security assurance metrics. OWASP ASVS is used to define the test cases for the security requirements, and OWASP Testing Guide and OWASP Cheat Sheets [32] were used as a reference while choosing the testing techniques and developing the vulnerability test cases.

a) Security requirement test cases: As the discussion forum is not a critical application, Level 1 is used to verify the security requirements. Level 1 consists of 68 security requirements to be verified. All of the requirements were analysed to find out how many of these requirements are applicable to the discussion forum, and the security requirements that are applicable to the application were assessed to measure the their fulfillment. Example test cases for the authentication requirement verification are given as follows:

- Verify that the weak lock out mechanism to mitigate brute force password guessing attacks
  - attempt an invalid log in by using the incorrect password a number of times, before using the correct password to verify that the account was locked out. An example test may be as follows:
    1) Attempt to log in with an incorrect password 3 times.

2) Successfully log in with the correct password, thereby showing that the lockout mechanism doesn't trigger after 3 incorrect authentication attempts.

- Verify that the forgotten password function and other recovery paths do not reveal the current password and that the new password is not sent in clear text to the user
- Verify that information enumeration is not possible via login, password reset, or forgot account functionality

b) Vulnerabilities test cases: This subsection specifies vulnerabilities test cases that can prevent the achievement of the security requirements. The severity impact of each vulnerability is measured based on the CVSS base score. Example vulnerability test cases for the application that transmits cleartext and uses default credentials are given as follows:

- Test for credentials transported over an unencrypted channel
  - Sending data with POST method through HTTP and trying to intercept the username and password by simply sniffing the network with a tool like Wireshark
- Test for default credentials of the application
  - Test for default credentials of common applications (as an example try the following usernames - "admin", "root", "system", "guest", "operator", or "super"), and an empty password or one of the following "password", "pass123", "admin", or "guest"

C. Test Case Execution and Measurement Collection

The discussion forum application was tested based on the the test cases developed to verify the security requirement and vulnerabilities in subsection V-B. After the test cases execution, security requirement measurement is collected from the application for all the security requirements. Due to space limitation, the measurement details for all test cases in each security requirement and each vulnerability type are not included; however the total measurement collected for each security requirements and vulnerabilities are summarized in Table II and Table III, respectively.

D. Assurance Metrics and Level Calculation

The *Assurance metrics* is calculated as the difference of *security requirement metrics* and *vulnerability metrics* using (5).

$$AM = RM - VM \quad \text{that is,}$$

$$AM = \left(\sum_{i=1}^{n}(w_i \times \sum_{j=1}^{m} f_{ij})\right) - \left(\sum_{k=1}^{d}(r_k \times \sum_{l=1}^{p} e_{kl})\right)$$

Thus, the assurance metrics for the discussion forum application can be calculated using the security requirement measurement in Table II and the vulnerability measurement in Table III as follows:

$$AM = \left(\sum_{i=1}^{11}(w_i \times \sum_{j=1}^{m} f_{ij})\right) - \left(\sum_{k=1}^{10}(r_k \times \sum_{l=1}^{p} e_{kl})\right)$$
$$= 73 - 108.1 = -35.1$$

TABLE II. MEASUREMENT COLLECTED FOR ALL SEC. REQUIREMENTS

No.	Security Requirements	Weight	Fulfillment	RM = 73
1	Architecture, design and threat modelling	10	1	10
2	Authentication	8	3	24
3	Session management	5	4	20
4	Access control	7	1	7
5	Malicious input handling	5	1	5
6	Cryptography at rest	4	0	0
7	Error handling and logging	7	1	7
8	Data protection	4	0	0
9	HTTP security configuration	4	0	0
10	File and resources	4	0	0
11	Configuration	4	0	0

AM can be *minimum* if RM is minimum (zero) and VM is maximum. AM, on the other hand, can be *maximum* if VM is minimum and RM is maximum. Thus, the *minimum value* of the assurance metrics (AM_{min}) for the case study can be calculated as follows:

$$AM_{min} = RM_{min} - VM_{max} = 0 - 142.5 = -142.5 \quad (6)$$

The *maximum value* of the assurance metrics (AM_{max}) for this case study can also be calculated as follows:

$$AM_{max} = RM_{max} - VM_{min} = 255 - 0 = 255 \quad (7)$$

The normalized assurance metrics (AM_{norm}) can be calculated in the range of 0 to 10 using the *min-max normalization* formula as follows :

$$\text{AM}_{norm} = (\frac{AM - AM_{min}}{AM_{max} - AM_{min}}(AM_{newmax} - AM_{newmin}) + AM_{newmin}) = (\frac{-35.1-(-142.5)}{255-(-142.5)}(10-0)+0) = \frac{1074}{397.5} = 2.7$$

Security assurance levels: For some purposes, it is useful to have a textual representation of the security assurance metric value of an application. Five subjective categories of assurance metrics and their corresponding values can be represented as follows:

1) Very low (0-0.9)
2) Low (1 - 3.9)
3) Medium (4 - 6.9)
4) High (7 - 8.9)
5) Very high (9 - 10)

As an example, the security assurance metric of the discussion forum application (2.7) has an associated security assurance level of *Low*.

TABLE III. MEASUREMENT COLLECTED FOR ALL VULNERABILITIES

No.	Threats/vulnerabilities	Av	Ac	Pr	UI	S	C	I	A	CVSS Base Score	Existence	VM=108.1
1	Web server generic XSS	N	L	N	R	C	L	L	N	6.1	2	12.2
2	Web server transmits cleartext credentials	N	L	N	N	U	L	L	L	7.3	1	7.3
3	Application error disclosure	N	H	N	N	U	L	N	N	3.7	2	7.4
4	Directory browsing	N	L	L	N	U	H	N	N	6.5	2	13
5	Cookie no HttpOnly flag	N	L	N	N	U	H	N	N	7.5	1	7.5
6	SQL injection vulnerability for the SQL-Database	N	L	L	N	C	L	L	N	6.4	1	6.4
7	Lack of input validation	N	L	N	R	C	L	L	N	6.1	5	30.5
8	Data tampering	N	H	N	N	U	H	H	N	7.4	1	7.4
9	Elevation of privilege using remote code execution	L	L	L	N	U	H	H	H	7.8	1	7.8
10	Network eavesdropping/Sniffing	N	L	N	N	U	H	L	L	8.6	1	8.6

VI. DISCUSSION

In this study, we have investigated the quantitative security assurance metrics using two dimensions of metrics to represent an overall security assurance: *vulnerability* and *security requirement*. In particular, vulnerability represents the negative aspect that leads to a reduction of the assurance level, and security requirement improves the assurance posture. Our approach employed both GQM and CVSS methods for metric definition and calculation. While the GQM is used to construct measurement items for security requirement metrics and assess the fulfillment of security requirements, CVSS is used to quantify the severity of vulnerabilities according to various attributes.

A case study is provided in this work, which measures the overall security assurance of the discussion forum application using our approach. The security assurance process was followed to measure and evaluate the degree of trustworthiness of the application. Specifically, we conducted a systematic threat modelling processes of the discussion forum application, test case definition, measurement collection, and security assurance metrics calculation. However, this work did not consider the security of the underlining infrastructure, non-technical attack vectors, new attacks, etc.

VII. CONCLUSION AND FUTURE WORK

This paper has presented a *quantitative approach* for defining security assurance metrics that provides a high level security assurance evaluation and distinguishes two perspectives: *security requirement metrics* and *vulnerability metrics*. The metrics reflect the strengths of the protection mechanisms and the severity of vulnerabilities that impact a target of evaluation. Specifically, we adopted the GQM method to estimate and quantify the level of protection, and the CVSS to quantify the vulnerability severity. The methodology described a process for security assurance evaluation emphasizing the role of security requirement metrics to probe the correctness of deployed security measures, vulnerability metrics to measure a severity level of a system vulnerability, and security assurance metrics. The computation of the assurance metric is focused on the current security state of a target of evaluation in order to consider the system dynamics in a particular time, e.g., the level of protection mechanisms and vulnerabilities.

This work has also conducted a case study on the discussion forum using the approach, and the results show that it is important to utilize a variety of tools, as well as conduct manual testing in order to find and test the most number of vulnerabilities and verify security requirements in a web application. This can assist organizations to evaluate the overall security assurance of a system and result in a measure of confidence that indicates how well a given system at a particular time meets particular security goal.

Our future work aims at automating the security assurance process and developing a platform that creates a network of systems based on testing scenarios, records the test execution and analyses its results, and scores the assurance level.

ACKNOWLEDGMENT

This work was partially supported by the the *Regional Research Fund* (RFF) Innlandet and *Playtecher* of Norway. The authors would like to thank the anonymous referees for their review comments that helped to improve the presentation of the paper.

REFERENCES

[1] G. K. Weldehawaryat and S. D. Wolthusen, "Modelling interdependencies over incomplete join structures of power law networks," in 2015 11th International Conference on the Design of Reliable Communication Networks (DRCN), March 2015, pp. 173–178.

 ISBN: 978-1-61208-661-3

[2] R. M. Savola, H. P. äinen, and M. Ouedraogo, "Towards security effectiveness measurement utilizing risk-based security assurance," in 2010 Information Security for South Africa, August 2010, pp. 1–8.
[3] S. M. Furnell, "The irreversible march of technology," Inf. Secur. Tech. Rep., vol. 14, no. 4, November 2009, pp. 176–180.
[4] G. Stoneburner, "Underlying technical models for information technology security," Special Publication (NIST SP)-800-33, 2001, pp. 1–28.
[5] M. Ouedraogo, R. M. Savola, H. Mouratidis, D. Preston, D. Khadraoui, and E. Duboi, "Taxonomy of quality metrics for assessing assurance of security correctness," Software Quality Journal, vol. 21, no. 1, March 2013, pp. 67–97.
[6] "Common criteria for information technology security evaluation part 3: Security assurance components, version 3.1 rev1," pp. 1–86, 2006.
[7] M. Ouedraogo, H. Mouratidis, D. Khadraoui, E. Dubois, and D. Palmer-Brown, "Current trends and advances in it service infrastructures security assurance evaluation," 2009, pp. 132–141.
[8] M. Pendleton, R. Garcia-Lebron, J.-H. Cho, and S. Xu, "A survey on systems security metrics," ACM Comput. Surv., vol. 49, no. 4, December 2016, pp. 62:1–62:35.
[9] W. Jansen, "Directions in security metrics research - NISTIR 7564," 2009, pp. 1–21.
[10] G. B. Regulwar, V. S. Gulhane, and P. M. Jawandhiya, "A security engineering capability maturity model," in 2010 International Conference on Educational and Information Technology, vol. 1, Sept 2010, pp. 306–311.
[11] OWASP, "Application security verification standard (ASVS)," 2015, pp. 1–70.
[12] C. Kubicki, "The system administration maturity model - SAMM," in Proceedings of the 7th USENIX Conference on System Administration, ser. LISA '93. Berkeley, CA, USA: USENIX Association, 1993, pp. 213–225.
[13] "Common criteria for information technology security evaluation," pp. 1–93, 2012.
[14] M. Vetterling, G. Wimmel, and A. Wisspeintner, "Secure systems development based on the common criteria: The palme project," in Proceedings of the 10th ACM SIGSOFT Symposium on Foundations of Software Engineering, ser. SIGSOFT '02/FSE-10. New York, NY, USA: ACM, 2002, pp. 129–138.
[15] N. Pham, L. Baud, P. Bellot, and M. Riguidel, "A near real-time system for security assurance assessment," in 2008 The Third International Conference on Internet Monitoring and Protection, June 2008, pp. 152–160.
[16] E. Bulut, D. Khadraoui, and B. Marquet, "Multi-agent based security assurance monitoring system for telecommunication infrastructures," in Proceedings of the Fourth IASTED International Conference on Communication, Network and Information Security. ACTA Press, 2007, pp. 90–95.
[17] S. Haddad, S. Dubus, A. Hecker, T. Kanstren, B. Marquet, and R. Savola, "Operational security assurance evaluation in open infrastructures," in Proceedings of the 2011 6th International Conference on Risks and Security of Internet and Systems (CRiSIS), ser. CRISIS '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 1–6.
[18] C. Criteria, "Common criteria for information technology security evaluation, v3.1, part 1-3," 2017, pp. 1–106.
[19] A. M. B. N. I. L. Nabil Seddigh, Peter Pieda and A. Hatfield, "Current trends and advances in information assurance metrics," 2004, pp. 197–205.
[20] O. Tettero, D. Out, H. Franken, and J. Schot, "Information security embedded in the design of telematics systems," Computers & Security, vol. 16, no. 2, 1997, pp. 145 – 164.
[21] P. T. Devanbu and S. Stubblebine, "Software engineering for security: A roadmap," in Proceedings of the Conference on The Future of Software Engineering, ser. ICSE '00. New York, NY, USA: ACM, 2000, pp. 227–239.
[22] K.-Y. Park, S.-G. Yoo, and J. Kim, "Security requirements prioritization based on threat modeling and valuation graph," in Convergence and Hybrid Information Technology, G. Lee, D. Howard, and D. Ślezak, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 142–152.
[23] V. R. Basili, G. Caldiera, and H. D. Rombach, "The goal question metric approach," Encyclopedia of software engineering, vol. 2, no. 1994, 1994, pp. 528–532.
[24] S. Islam and P. Falcarin, "Measuring security requirements for software security," in 2011 IEEE 10th International Conference on Cybernetic Intelligent Systems (CIS), September 2011, pp. 70–75.
[25] P. Mell, K. Scarfone, and S. Romanosky, "A complete guide to the common vulnerability scoring system version 2.0," 2007, pp. 1–24, [Accessed: 20/07/2018].
[26] FIRST, "Common vulnerability scoring system v3.0: Specification document," pp. 1–21, 2015, [Accessed: 20/07/2018].
[27] "Offensive security ltd. kali linux," https://www.kali.org/, 2018, [Accessed: 19/07/2018].
[28] "Openstack open source cloud computing software," https://www.openstack.org/, 2018, [Accessed: 18/07/2018].
[29] A. Marback, H. Do, K. He, S. Kondamarri, and D. Xu, "A threat model-based approach to security testing," Softw. Pract. Exper., vol. 43, no. 2, Feb. 2013, pp. 241–258.
[30] S. Myagmar, A. J. Lee, and W. Yurcik, "Threat modeling as a basis for security requirements," in in Proceedings of the IEEE Symposium on Requirements Engineering for Information Security, 2005, pp. 1–8.
[31] F. Swiderski and W. Snyder, Threat Modeling (Microsoft Professional), 2004.
[32] OWASP, "OWASP testing guide v4," pp. 1–224, 2014, [Accessed: 20/07/2018].