A Metamodel for Web Application Security Evaluation

Shao-Fang Wen and Basel Katt

Norwegian University of Science and Technology
Gjøvik, Norway

shao-fang.wen, basel.katt@ntnu.no

***Abstract*—In the digital era, web applications have become a prevalent tool for businesses. As the number of web applications continues to grow, they become enticing targets for malicious actors seeking to exploit potential security vulnerabilities. Organizations face constant risks associated with vulnerabilities in their web-based software systems, which can result in data breaches, service disruptions, and a loss of trust. Consequently, organizations require an effective and efficient approach to assess and analyze the security of acquired web-based software, ensuring sufficient confidence in its utilization. This research aims to enhance the quantitative evaluation and analysis of web application security through a model-based approach. We focus on integrating the Open Web Application Security Project's (OWASP) Application Security Verification Standard (ASVS) into a structured and analyzable metamodel. This model aims to effectively assess the security levels of web applications while offering valuable insights into their strengths and weaknesses. By combining the ASVS with a comprehensive framework, we aim to provide a robust methodology for evaluating and analyzing web application security.**

I. Introduction

Web applications have emerged as the dominant technology for delivering services and disseminating information online. Numerous businesses across various sectors have embraced this digital platform, transitioning their operations to the web. Examples include social networks, webmail services, banks, and other entities that perform critical operational functions and store sensitive data. The extensive utilization of web applications in contemporary society has attracted the attention of hackers, who seek to exploit vulnerabilities in these applications to carry out malicious activities. Such actions can lead to disruptions and impair the efficiency and effectiveness of business operations [5]. Given the prevalence and importance of web applications in today's landscape, organizations strive for assurance that their software is developed with a strong emphasis on security and reliability. They aim to implement the necessary security mechanisms while minimizing risks to their assets, seeking confidence in the overall robustness of their applications.

To instill the required confidence in web-based software, organizations require a comprehensive methodology for evaluating and analyzing its security. The objective of security evaluation is to deliver precise and dependable results that decision-makers can rely on with confidence. [6]. This process involves the identification and analysis of security threats, vulnerabilities, and risks, while also assessing the effectiveness of security controls and procedures in mitigating them. [9]. To enable stakeholders to effectively utilize this data, it is crucial to present it in a format that aligns with their requirements. Quantitative security evaluation is a specialized discipline that employs computational and mathematical techniques to assess the security level of a system. By leveraging these techniques, stakeholders can gain valuable insights into the quantitative aspects of security evaluation [6, 11]. Quantitative security evaluation endeavors to provide a more precise assessment of the level of effort needed to protect a system and the potential risk of compromise [23]. This type of security assessment model facilitates the generation of quantifiable security scores, offering a clear indication of the effectiveness of a system's protective measures [6].

This paper aims to contribute to the field of research by focusing on the modeling of web application security evaluation, with a particular emphasis on its suitability for quantitative analysis. Specifically, we strive to create a comprehensive and analyzable metamodel that is built upon the Open Web Application Security Project's (OWASP) Application Security Verification Standard (ASVS) [15] to guarantee the optimum security of web applications. OWASP ASVS is widely used for web application security assessment the security requirements elicitation, as it provides a comprehensive overview of all security-related topics [8, 22]. While the ASVS offers advantages for security assessments, there exists a gap in research concerning the generation of meaningful data for analysis. To address this limitation, our model-based approach enables the transformation of ASVS data into informative and comprehensible information. By merging meaningful data sets with robust analytics, security stakeholders can make informed choices that drive organizational decision-making [26]. This paper also showcases the practical application of these models for analyzing security strengths, weaknesses, and quantitative aspects by aggregating ASVS verification results. Through practical demonstrations and illustrations, we highlight the utilization of these models to assess and evaluate the security posture of web applications.

The rest of this paper is organized as follows. Section 2 outlines the OWASP ASVS framework. In Section 3, we

provide an overview of related work. In Section 4, the proposed web application security evaluation metamodel is discussed in detail. Subsequently, Section 5 provides an example of data analytics based on this model to better illustrate it. Lastly, the conclusion and future works are presented in Section 6.

II. OWASP Application Security Verification Framework

The OWASP is a non-profit, community-driven organization that promotes software security through educational materials, open-source software, and other initiatives. The OWASP ASVS is an open standard for performing web application security verification, which is designed to methodically test application and environment-level technical security controls. With this, it is possible to identify various potential vulnerabilities, for example, Cross-Site Scripting (XSS) and SQL injection. The ASVS Project has designed its standard for practical, "commercially workable". With extensive coverage and flexibility, the ASVS can be applied in various situations, from intimate internal security measuring to instructing developers how to suitably implement safety functions or evaluating third-party software and contractual development agreements. The latest stable version of ASVS is 4.0.3 released in October 2021. Fig. 1 depicts the whole data structure of ASVS. The ASVS contains 286 verification requirements that are grouped into 14 higher-level categories (named "Chapter") and sub-categories (named "Section") that are of similar functionality.

Additionally, from version 4.0, ASVS provides a comprehensive mapping to the Common Weakness Enumeration (CWE) [15]. CWE is a list of weaknesses in software that can lead to security issues. While the CWE list is long, it is also prioritized by severity of risk, providing organizations and developers with a good idea about how to best secure applications. Where applicable, ASVS requirements are also mapped to (or aligned with) different security standards, including OWASP Proactive Control [15] and the U.S. National Institute of Standards and Technology (NIST) Digital Identity Guidelines (NIST 800-63) [15]. The former describes the most important control and control categories that every architect and developer should absolutely, while the latter introduces modern, evidence-based, and advanced authentication controls.

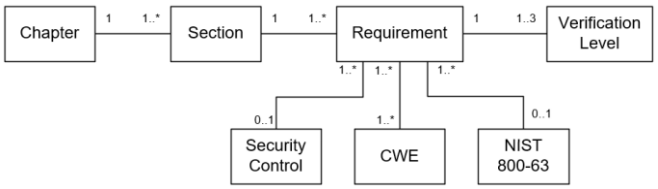

Fig. 1. ASVS data structure

III. Related Work

There is a wealth of research on security assurance and evaluation methods. Over the years, numerous frameworks and standards have been developed to analyze security. Common Criteria (CC) [9] is one of the most well-known efforts in this area. CC is an international ISO/IEC 15408 standard for the security evaluation of IT products. The standard outlines a clear set of guidelines and specifications that provide organizations with the necessary information to accurately specify their security functional requirements and security assurance requirements. [4, 28]. In addition, there are several security maturity models available for the software security domain, such as the Building Security In Maturity Model (BSIMM) [12] and OWASP Software Assurance Maturity Model (OpenSAMM) [17]. BSIMM is a research initiative that investigated the various approaches to software security employed by businesses, leading to the development of a framework featuring 116 activities and 12 practices. Like BSIMM, openSAMM is an open software security framework developed by OWASP, which provides guidelines on which software security practices should be used and how to assess them. Such maturity models provide frameworks, especially in a qualitative fashion, to evaluate the security posture of the process and culture practiced in an organization.

Although various research studies have been conducted on web application security evaluation, few attempts have been made to establish a generic approach that quantifies the results systematically. Below are several papers that discuss this research area. The authors in [7] presented a security evaluation framework for web-portal security assessment, which integrates ISO/IEC 15408 [10] and OWASP evaluation model Common Criteria Web Application Security Scoring (CCWAPSS) [3]. This framework facilitates numerical rankings via the use of a scoring system to assess the significance of each factor within the criteria. By doing so, it provides practical security evaluations that web portal developers can quickly understand and implement. Okamura et al [13] discussed a quantitative security evaluation approach for software systems from the vendor's viewpoint, centering on the analysis of collectible vulnerability data. They apply a stochastic model using a non-homogeneous Poisson process to explain this data, and then use numerical examples to evaluate the security measures relative to the content management system of an open-source project. Yautsiukhin et al. [27] introduced a method of computing the security qualities of software architectures with the adoption of security patterns. The core metric used in this evaluation was threat coverage, and an algorithm was proposed to aggregate low-level measures associated with these patterns into a single high-level indicator. Lastly, Banaei and Khorsandi [2] presented a hierarchical structure for web service security, complete with a model that evaluates various aspects of security from an analytical perspective. We use the Analytical Hierarchy Process (AHP) Theory to prioritize weighted averaging of critical security properties, such as authorization, confidentiality, and availability — all to provide greater levels of customization in terms of provider/consumer needs.

Furthermore, alternative methods for quantitative security assurance of IT systems have been proposed by some researchers. These concepts could be applied in software systems/web applications. For instance, Katt and Prasher [25] outlined a quantification method to evaluate the security assurance of systems. This framework measures two parts: (1) the confidence that existing mechanisms are sufficient to meet security requirements; and (2) which potential security threats

might leave a system vulnerable. The framework has been validated through case studies on public REST APIs. Ouedraogo et al. [14] utilized quantitative risk measurement techniques to create indicators that can be used to assess IT infrastructure security, alongside aggregation procedures. The primary algorithms used to perform operational aggregation are the recursive minimum, maximum, and weighted sum algorithms. Each of these tools has been designed to take into consideration a wide range of datasets when consolidating information. Pham and Riguidel [18] introduced an aggregational method that can be applied in the calculation of the security assurance value of the whole system when combining several entities, which have been evaluated independently. The effects of the emergent relations are taken into account in the calculation of the security assurance value of an attribute in the context of a system.

IV. The Proposed Security Evaluation and Analysis Model

To achieve a comprehensive security evaluation and analysis, it is crucial to examine the strengths and weaknesses of the system's security. Our approach focuses on quantifying OWASP ASVS by dividing it into two fundamental components: security strength evaluation and security weakness evaluation. The objective is to obtain measurable insights that enhance our understanding of the ASVS verification results. By assessing both aspects, we can obtain a more holistic perspective on the system's security posture. The proposed security evaluation and analysis model for a System of Interest (SoI) is depicted in Fig. 2. In essence, the security-strengths model offers a quantifiable measure of the SoI's resilience against attacks, assuring its security. On the other hand, the weaknesses model focuses on identifying the potential consequences that may arise when the security mechanisms are inadequately implemented. By incorporating both models, a more holistic understanding of the system's security can be achieved, enabling organizations to address vulnerabilities and enhance their overall security posture.

In the subsequent sections, we outline our methodology for modeling the evaluation component of the system. We explore

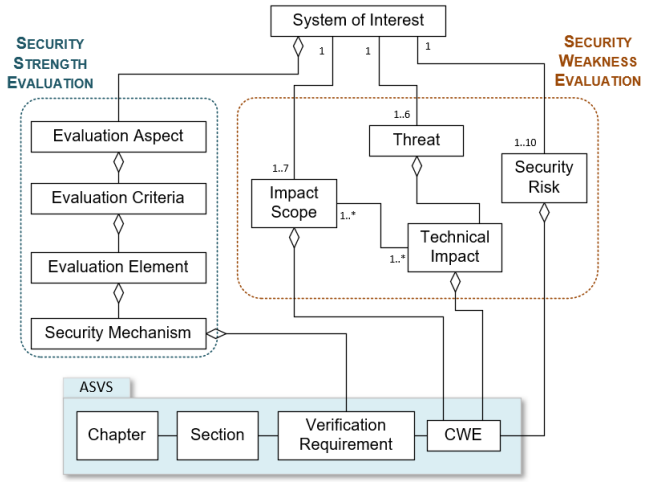

Fig. 2. The complete security evaluation metamodel

the structural representation of the security mechanisms and elucidate how the weaknesses in system security can be derived from the ASVS.

A. Security Strength Evaluation Model

The security strength of a system refers to its level of preparedness and resilience in implementing security measures to counter potential threats [20]. To evaluate the security strength, we utilize a hierarchical structure consisting of five levels, as depicted in Fig. 3. This allows us to comprehensively assess the system's security capabilities. The evaluation is divided into three aspects: structure, environment, and process. Each aspect incorporates a two-level categorization system, enabling the classification of security mechanisms based on their connection to the ASVS requirements. The evaluation process starts by assigning scores to the ASVS requirements and then aggregating these scores using an Average scheme. This allows for the rating of evaluation components at each level of the hierarchy. Score aggregation is a valuable technique as it helps minimize subjective bias in evaluating claims and provides a more objective approach to assessing the accuracy of these claims [1]. The overall score of the SoI is determined by calculating a weighted average using the scores of evaluation components and their corresponding weighting factors. This calculation results in a single value that serves as an objective measure of the system's security level. The specific notation and detailed evaluation process will be discussed in the subsequent sections.

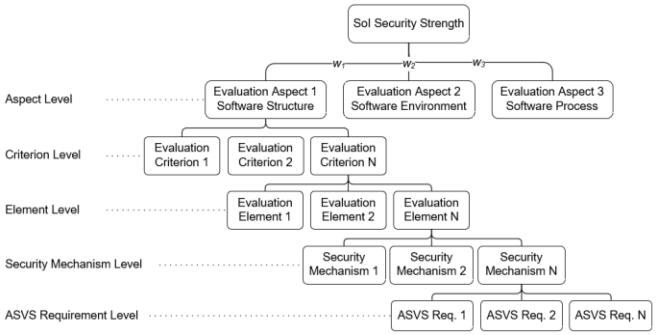

Fig. 3. Illustration of the layers that make up the hierarchical approach for the security-strength evaluation.

1)Evaluation of ASVS Requirements: Initially, each ASVS verification requirement is mapped to one verification case to determine its fulfillment. Results for verification cases are quantified as 1, 0, and 0.5, depending on the level of fulfillment. The score 1 is given to the cases that pass the verification, indicating the corresponding requirements are fully fulfilled, while 0 means the requirements are not fulfilled (i.e., the verification case failed). A score of 0.5 implies the requirement is considered a partial fulfillment. Partial fulfillment means that the actual result matches its expected result, however, there might be unnecessary (or superfluous) exceptions/ messages that are caught during the test-case execution. Such a test execution state is usually applied in the context of manual testing, heavily reliant on the tester's judgment [19].
At this step, we use $S(ASVS_i)$ to denote the score of the i^{th} ASVS requirement, which can be expressed as:

$$S(ASVS_i) \in \{0, 1, 0.5\}$$

2)Evaluation of Security Mechanisms: A deficiency we identify in ASVS is the lack of capability of system diagnosis for subject-of-matter at a granular level. Rather than analyzing scattered descriptive statements, we suggest that security requirements should be organized into a synthesizable and analyzable format. In the security evaluation and analysis approach, we attempt to use a more fine-grained "Security Mechanism" than descriptive ASVS requirements. Security mechanisms can be treated ted the fundamental means and methods that are designed to achieve security-relevant purposes. While ASVS requirements are designed for verification, security mechanisms, on the other hand, are for analysis purposes. To provide analyzability, the mechanism must be small and simple enough to be evaluated. The exemplary security mechanisms for "Password Security" with the associated ASVS requirements can be found in Table I.

Now to calculate the scores of security mechanisms, let C(*SecurityMechanism*$_i$) denotes a set of ASVS scores associated with the *i*th security mechanism, defined by the following equation:

$$\mathrm{C}(SecuriyMechanism_i) = \{\mathrm{S}(ASVS_j) \rightarrow SecuriyMechanism_i\}$$

We use S(*SecurityMechanism*$_i$) to represent a measurement to reflect the actual (calculated) score of the security mechanism. Following equation represents the calculation of the *i*th security mechanism, which uses the average function to derive the score.

$$\mathrm{S}(SecuriyMechanism_i) = \frac{\sum \mathrm{C}(SecuriyMechanism_i)}{|\mathrm{C}(SecuriyMechanism_i)|}$$

3)Evaluation of Criterion and Element Levels: The term "criteria" as used in this model refers to a higher, more abstract level of meaning that can be thought of as a standard in the SoI's application domain. These criteria are part of the "target" that the work is planned to achieve. These criteria are selected, tested, and measured to confirm the sufficiency of system security to be offered to users. Table II lists the corresponding evaluation criteria for each evaluation aspect. Evaluation criteria are then narrated in detail by a set of evaluation elements. Some examples of evaluation elements are presented in Table III.

TABLE II. CORRESPONDING EVALUATION CRITERIA FOR EACH EVALUATION ASPECT

Evaluation Aspect	Evaluation Criteria
Software Structure	Authentication
	Access Control
	Input Validation and Output Encoding
	Session Management
	Cryptography
	Error Handling and Logging
	Web Service and API Security
Software Environment	Environment Management
	Communication Hardening
	Configuration Hardening
Software Process	Security Requirement
	Secure Design
	Secure Coding
	Secure Code Review
	Secure Build and Deployment

TABLE III. EVALUATION ELEMENTS IN EVALUATION CRITERIA

Evaluation Criteria	Evaluation Element
Authentication	Authentication Architecture
	Password Security
	Authenticator Security
	Credential Storage
	Authentication Logging
	Service Authentication
Access Control	Access Control Architecture
	Operation Level Access Control
	HTTP Request Access Control
	Access Control Logging

Similar to the algorithm is the previous level, the score of the *i*th element-level component, denoted by S(*Element*$_i$) is calculated using the following formulas:

$$\mathrm{S}(Element_i) = \frac{\sum \mathrm{C}(Element_i)}{|\mathrm{C}(Element_i)|}$$

where:

$$\mathrm{C}(Element_i) = \{\mathrm{S}(ASVS_j) \rightarrow Element_i\}$$

Consequently, the formula for calculating the score of the *i*th criterion-level components is as follows:

$$\mathrm{S}(Criterion_i) = \frac{\sum \mathrm{C}(Criterion_i)}{|\mathrm{C}(Criterion_i)|}$$

where:

$$\mathrm{C}(Criterion_i) = \{S(Element_j) \rightarrow Criterion_i\}$$

TABLE I. EXEMPLARY SECURITY MECHANISMS WITH ASSOCIATED ASVS REQUIREMENTS

Security Mechanism	ASVS Requirement
Password strength policy	V2.1.1-Verify that the user-set passwords are at least 8 characters in length (after multiple spaces are combined).
	V2.1.2-Verify that passwords of at least 64 characters are permitted, and that passwords of more than 128 characters are denied.
	V2.1.4-Verify that any printable Unicode character, including language-neutral characters such as spaces and Emojis, are permitted in passwords.
	V2.1.7-Verify that passwords submitted during account registration or password change are checked against an available set of, at least, the top 3000 passwords.
	V2.1.10-Verify that the application does not require periodic credential rotation.
Password input functionality	V2.1.11-Verify that "paste" functionality, browser password helpers, and external password managers are permitted.
Password changing functionality	V2.1.5-Verify users can change their password.
	V2.1.6-Verify that password change functionality requires the user's current and new password.
Password processing logic	V2.1.3-Verify that passwords are not truncated.

4)Evaluation of Aspect Level. Although ASVS provides a categorical view of the security evaluation, it does not come up with a broader perspective and more strategic point of view. Instead of facing the process-technology intertwining information at the first sight, a common analysis approach is to start by analyzing macro aspects, from which a *governing thought* is arrived at. This is the most important idea that needs to be captured first. "Aspects" are the viewpoints about how stakeholders can describe the security strength at the highest level.

While defining the aspects, we include the three predominant attributes of SoI, that is, software structure, software environment, and software process. The software structure is the core subset of the software system, meaning any source code or object code made to perform a specific task(s). The evaluation of the software structure aims to access the sufficiency of the technical security mechanisms of the software system itself, including security architectures and security functionalities. The evaluation criteria under the software structure are, for example, authentication, access control, and cryptography. The evaluation of the software environment entails an examination of the environmental factors that contribute to the production and maintenance of the software system. organizational and physical facilities (for example, development, production, delivery, and operation) are among these factors.

In addition to the security aspect described above, developing and maintaining secure systems rely on the processes linking people and technologies. Therefore, a secure system should also provide evidence that it is developed and operated using adequate software processes, and conformance to implementation standards. The evaluation of software processes is not necessarily tied to the specific functionality of the software structure and environments, but rather to deal with the organizational processes used in the development and operation of core functionalities and infrastructures, conformance to coding standards, adequate testing, verification and validation, and suitable specification and documentation for all system aspects. While evaluating the software systems, stakeholders could decide whether to take the aspect of the software process into the security evaluation. For example, in the context, of the open-source software (OSS) security evaluation, investigating the software-process aspect is generally not possible.

The formula for calculating the score of the i^{th} aspect-level components is as follows:

$$\mathrm{S}(Aspect_i) = \frac{\sum \mathrm{C}(Aspect_i)}{|\mathrm{C}(Aspect_i)|}$$

where:

$$C(Aspect_i) = \{S(Criterion_j) \rightarrow Aspect_i\}$$

5)Evaluation of SoI. Based on the definitions established earlier, we can compute the overall score for the SoI by aggregating the scores of the individual aspects and deriving a single, comprehensive measure. The higher the value, the more reliable the security strength is considered. Assessing this score provides useful insight into assessing trustworthiness within an organization. The SoI score is obtained through a weighted average aggregation process, which is highly intuitive and comprehensible. This method allows us to prioritize certain aspects more than others in the evaluation procedure, based on their relative importance. The exact weighting values are calculated or assigned based on the opinions of stakeholders by applying decision-making techniques that must be carried out based on the verification context. Finally, we standardize the score by scaling the value in the range of [0, 10].

The overall score of the SoI, represented by S(*SoI*) is calculated by the following formula:

$$\mathrm{S}(SOI) = \sum_{i=1}^{3} \mathrm{S}(Aspect_i) \times w_i \times 10$$

where:

w_i: the weight corresponding to the i^{th} evaluation aspect ($0<w_i<1$ and $\sum w_i = 1$)

In order to improve clarity, we have incorporated a discrete rating system into the final SoI score. This approach makes it easier for users to understand the ratings. Table IV is adapted from the NVD Vulnerability Severity Ratings [24]. In NVD, the higher score represents greater severity. However, our table shows the opposite definition, i.e., levels of security. With this table, we can be used to convert the score to a textual form.

TABLE IV. SECURITY LEVEL

Score	Security level
[0.0 – 1.0)	No Security
[1.0 – 4.0)	Low Security
[4.0 – 7.0)	Moderate Security
[7.0 – 9.0)	Good Security
[9.0 – 10.0]	Excellent Security

B. Security Weakness Evaluation Model

The security weakness evaluation aims to describe the consequence of the found weakness in the SoI. This involves defining the taxonomy of effects, including security risks, potential threats, and the impact scopes (i.e., the violated security properties), covering the relationships among them. The comprehensive mapping to CWE in ASVS allows us to derive a set of (negative) components, resulting from the weakness (depicted in Fig. 4), including impact scopes (i.e., the violated properties), technical impacts, threats, and security risks. Our modeling approach leverages existing CWE databases and well-known threat and vulnerability analysis methodologies to help derive a threat and security risk catalog for each ASVS element. The following sections explain the components, along with the associated derivation rules and evaluation formulas.

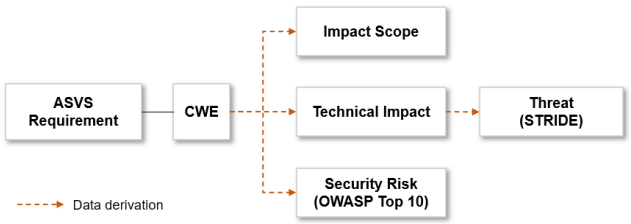

Fig. 4. Security Weakness Evaluation Model

1)Evaluation of CWE. Scores for security weakness components are calculated starting from the CWE identities, by adding up individual values throughout all calculation tasks. In security weakness evaluation, we focus on analyzing the severity of the weakness to the SoI. The summation function counts every occurrence in the ASVS verification that represents the significance of the weakness component.

Let *CWE i* denote a CWE ID that existed in the CWE repository. The following Equation defines the set of ASVS scores mapped to a given CWE.

$$\mathrm{C}(CWE\ i) = \{\mathrm{S}(ASVS_j) \rightarrow CWE\ i\}$$

For each ASVS with a score of 0 (i.e., not fulfilled requirements), the corresponding CWE is assigned the value 1. The total score for CWE i is calculated by accumulating the ASVS score with the following formula:

$$\mathrm{S}(CWE\ i) = \sum_{j \in C(CWE\ i)} e_j$$

where:

$$e_j = \begin{cases} 1, & if\ \mathrm{S}(ASVS_j) = 0, \\ 0, & otherwise. \end{cases}$$

2)Evaluation of Impact Scope. Each SoI comprises the main security objective that needs to be achieved, like confidentiality and availability, which are commonly **captured** by a *Security Property*. For every security property, the impact must be evaluated. The *Impact Scope* element identifies the security property that is violated due to the existence of the weakness. As for the evaluation component, the "Impact Scope' is used to evaluate the severity of weakness with generic/abstract security requirements for the SoI. In the CWE model, the impact scope can be found in the attributes of "Common Consequences". For example, CWE-116: "Improper Encoding or Escaping Output" impacts the security properties of Integrity, Confidentiality, Availability, and Access Control. Other impact scopes defined in CWE are Authentication, Authorization, and Non-repudiation.

To determine the score of "Impact Scope", we add up the corresponding CWE scores with the following equation:

$$\mathrm{S}(ImpacScope_i) = \sum \mathrm{C}(ImpactScore_i)$$

where:

$$\mathrm{C}(ImpacScope_i) = \{S(CWE\ j) \rightarrow ImpacScore_i\}$$

3)Evaluation of Technical Impact. *Technical Impact* is the potential result that can be produced by the weakness, assuming that the weakness can be successfully reached and exploited. This is expressed in terms that are more fine-grained than confidentiality, integrity, and availability. The technical impact is an important criterion that can be useful to any organization that needs reasonable security assurance for their software-based solutions. The CWE-Common Consequence also describes the Technical Impact that arises if an adversary succeeds in exploiting this weakness. Security weaknesses can cause a lot of damage if they are successfully exploited. This information then evaluates the different types of damage that can be caused, and how severe the damage can be. Examples of technical impact are: modify data, read data, unreliable execution, resource consumption and execute unauthorized commands.

Similar to "Impact Scope", the "Technical Impact" score is yielded by summing the results of the relevant CWEs:

$$\mathrm{S}(TechnicalImpct_i) = \sum \mathrm{C}(TechnicalImpact_i)$$

where:

$$\mathrm{C}(TechnicalImpact_i) = \{\mathrm{S}(CWE\ j) \rightarrow TechnicalImpact_i\}$$

4)Evaluation of Threat. To have a clear picture of the dangers, it is important to formulate an assessment of the threats to the SoI. Threat assessment is often performed on a higher level, by especially addressing legal or business-related issues. In our test-based approach, threats are identified and evaluated based on the catalogs of known CWEs, deriving from the relevant verification results of ASVS. CWE with its Common Consequences provides a point where we could start. In terms of threat categories, we use the STRIDE framework [21], which is a mature and optimal approach, to classify threats in areas where mistakes are often made. The acronym "STRIDE" stands for the threat categories of Spoofing, Tampering, Repudiation, Information Disclosure, Denial of 'Service, and Elevation of privilege.

The CWE Schema offers an alternative method for mapping between the CWE and the STRIDE, mediated by the attribute of "Technical impact". We map CWE in the dataset against STRIDE using the "Technical Impact" attribute elicited from the previously mapped CWE. Each STRIDE category had a relationship with one or more enumerations of the Technical Impact. The mapping of STRIDE to CWE Technical Impact is presented in Table V.

Based on the mapping table, threat scores are calculated as:

$$\mathrm{S}(Threat_i) = \sum \mathrm{C}(Threat_i)$$

where:

$$\mathrm{C}(Threat_i) = \{\mathrm{S}(TechnicalImpact_j) \rightarrow Threat_i\}$$

*5)Evaluation of Security Risk***.** While the mapped CWE list in ASVS is extensive, it can be grouped and ranked by risk severity. The OWASP Top 10 categories provide an easy, clear at-a-glance summary of the ten most critical application vulnerabilities, which are arranged according to their impact and the security risk involved. The condensing of the numerous kinds of CWS into a small number of categories gives an easier way to analyze the security weakness in the software system. Instead of making an effort to eradicate all vulnerabilities, one can decide which of the ten risks is either more or less hazardous to the organization. This provides analyzers with a good idea about how to draw stakeholders' attention to certain issues that are the most common problems at the time.

In our model, "Security risk" is derived from the "Memberships" attribute in CWE. The evaluation of security risks involves the quantification of risks and the associated Criticality factor. When security risks are identified, it is difficult to remove all of them simultaneously due to the limited resources available for vulnerability mitigation. Criticality is a numerical value that we give to a security risk that

TABLE V. MAPPING OF STRIDE CATEGORIES BASED ON CWE -TECHNICAL IMPACT

STRIDE Category	CWE/Technical Impact
Spoofing	Gain Privileges or Assume the identity
Tampering	Modify Application Data Modify Memory Modify Files or Directories Unexpected State Alter Execution Logic
Repudiation	Hide Activities
Information Disclosure	Read Application Data Read Memory Read Files or Directories
Denial of Service	DoS: Instability DoS: Resource Consumption (CPU) DoS: Resource Consumption (Memory) DoS: Crash or Exit or Restart DoS: Resource Consumption (Other)
Elevation of Privilege	Execute Unauthorized Code or Commands Bypass Protection Mechanism

communicates how serious it is and determines the mitigation to be applied first. The higher the criticality, the more urgent the need to act. A common criticality assessment method is based on the probability of failure and consequences. Criticality can be calculated using the following equation:

$$Criticality = Probability \times Serverity$$

The equation to derive the score of a security risk is defined as:

$$S(SecurityRisk_i) = \sum C(CWE\ j) \times c_i = \sum C(CWE\ j) \times p_i \times s_i$$

where:
c_i: the criticality that corresponds to the i^{th} *SecurityRisk*
p_i: the probability that corresponds to the i^{th} *SecurityRisk*
s_i: the severity that corresponds to the i^{th} *SecurityRisk*

To evaluate the criticality, we refer to the data factors listed for each of the OWASP Top 10 categories [16], which are systematically derived using CVSS v3. Two data factors are considered: “Average Incidence Rate” and “Average Weighed Impact”. The former represents the Probability while the latter is the Severity.

V. SAMPLE CASE STUDY

To showcase the effectiveness of the suggested methodology, a manual security evaluation and analysis were performed on a particular web application. It should be noted that certain mechanisms were not implemented in the software, leading to non-compliance with the requirements specified by ASVS. For example, requirements of V2.6.1 to V2.6.3 in the Authentication criteria define the security mechanism of “Look-up secrete security”. However, the application does not feature the specified functionality. Therefore, the relevant requirements may be excluded from the verification scope. As a result of this, these requirements were marked as "Not Applicable." Examples of non-applicable ASVS requirements in this case study are listed in Table VI. In summary, there are 261 out of 286 ASVS requirements have been determined to be "applicable" to the security verification.

The evaluation process commences with the utilization of the assessment model to calculate the security strength. This involves aggregating the verification findings from the ASVS. The summary of the SoI and evaluation aspect scores is presented in Table VII. The SoI score of 7.721 signifies a "Good Security" rating for the system. Weight factors for the three evaluation aspects are determined through a subjective weighting approach. In this particular case, stakeholders assigned a higher weight to the "Software Structure" aspect among the three aspects.

TABLE VI. EXAMPLES OF NON-APPLICABLE VERIFICATION CASES

Criteria	Element	Security Mechanism	ASVS Req.
Authentication	Authenticator Security	Look-up secrete security	V2.6.1-V2.6.3
		Out-of-band verifier security	V2.7.1-V2.7.6
		OTP verifier security	V2.8.1-V2.8.7
Input Validation and Sanitization	Input Validation	Input validation for LDAP Query	V5.3.7
		XPath query parameterization	V5.3.10
Privacy and Data Protection	Server-side Data Protection	Health Data Encryption	V6.1.2
		Financial Data Encryption	V6.1.3
Web Service and API Security	SOAP Web Service Security	Add integrity check to SOAP payload	V13.3.2
	GraphQL	GraphQL logic	V13.4.1-V13.4.2

TABLE VII. SUMMARY OF EVALUATION-ASPECT SCORES

Score of SoI	Security Level	Evaluation Aspect	Weight	Score
7.721	Good Security	Software Structure	0.6	0.45
		Software Environment	0.3	0.26
		Software Process	0.1	0.06

In Fig. 5, we provide an illustrative example of the "next level" analysis of security strength, specifically focusing on the software structure aspect. The figure presents the evaluation criterion scores alongside the distribution of verification-case fulfillment. Among the 11 evaluation criteria, "Files and Resource Security" attains the highest score of 1.00, indicating strong compliance. Conversely, "Intrusion Detection and Prevention" obtain the lowest score of 0.438, suggesting areas for improvement. The evaluation criterion "Authentication" has the highest number of verification cases and achieves a moderate score of 0.708. This analysis provides valuable insights into the specific strengths and weaknesses within the software structure aspect of the security evaluation.

The security strength model incorporates a hierarchical structure that allows for a comprehensive breakdown and facilitates the identification of implemented and functioning security mechanisms. Fig. 6 illustrates the drill-down scenarios in the security strength analysis, revealing specific areas of concern. For example, upon closer examination of the low-scoring "Credential Update" (score = 0.25), it is revealed that

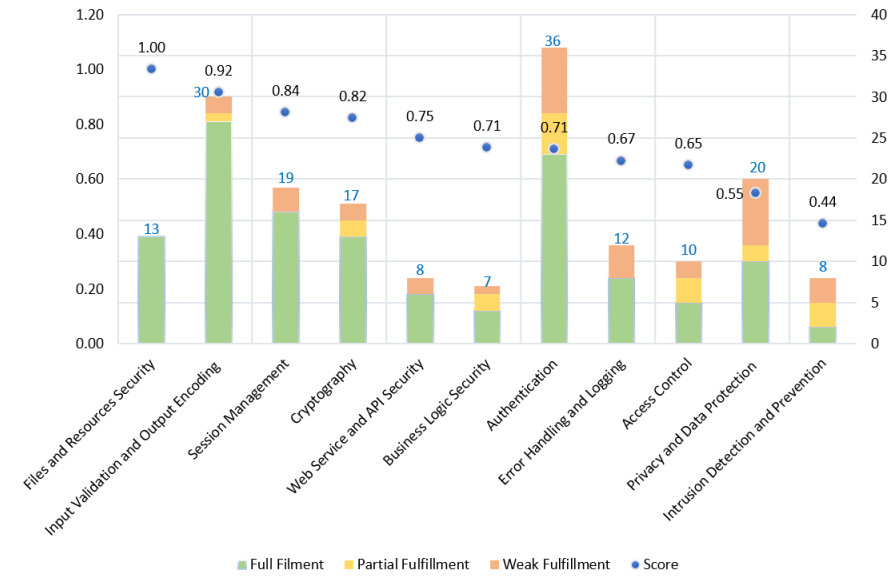

Fig. 5. Analysis of evaluation criteria scores

Criteria	Score
Authentication	0.71
Access Control	0.65
Input Validation and Output Encoding	0.92
Session Management	0.84
Cryptography	0.82
Intrusion Detection and Prevention	0.44
Error Handling and Logging	0.67
Privacy and Data Protection	0.55
Business Logic Security	0.71
Files and Resources Security	1.00
Web Service and API Security	0.75

Element	Score
Authentication Architecture	0.75
Authenticator Security	0.60
Password Security	0.68
Credential Recovery	1.00
Credential Storage	0.70
Credential Update	0.25
Service Authentication	1.00
Authentication Logging	0.00

Security Mechanism	Score
Password Strength	0.64
Password Input Functionality	0.00
Password Changing Functionality	1.00
Password Processing Logic	1.00

Security Mechanism	Score
Notification Functionality of Credential Update	0.25

Element	Score
Data Protection Architecture	0.75
Server-Side Data Protection	0.71
Client-Side Data Protection	1.00
Sensitive Data Transmission	1.00
Data Backup and Retention	0.00
User-Managed Data	0.25
Sensitive Data Logging	0.00

Security Mechanism	Score
Cached Data Protection	0.00
Memory Data Protection	1.00
Sensitive Data Encryption	1.00
Data Collection Consent	1.00
Limited Access to Privacy-Related Features	1.00

Fig. 6. Drill-down analysis based on the security strength evaluation model

the "Notification Functionality of Credential Update" is deficient.

Similarly, the evaluation highlights that the "Password Input Functionality" (score = 0) is not adequately addressing "Password Security." Furthermore, within the "Privacy and Data Protection" criteria, it is observed that "Cache Data Protection" is the only security mechanism in the "Server-Side Data Protection" category that does not meet the required standards. These detailed findings from the evaluation process provide valuable insights into specific vulnerabilities and areas that require attention within the security strength analysis.

The impact of identified CWEs on security properties is analyzed and represented in a bar chart, showcasing the scope of their effects. Fig. 7 illustrates this impact analysis, where the horizontal axis represents the number of CWEs. Unlike the positive connotation of security strength scores, in the evaluation of security weaknesses, higher scores indicate more severe weaknesses, threats, or security risks.

Therefore, all weakness components result in a negative effect on the overall result. From the figure, it is evident that the system's flaws have the most significant impact on the security properties of "Access Control" and "Confidentiality." This analysis helps prioritize the areas requiring immediate attention and highlights the vulnerabilities that have the most significant potential impact on the system's security.

We conducted an assessment to determine the relationship between CWEs and OWASP's Top 10 security risks, and the results are summarized in Table VIII and depicted in Fig. 8. Our evaluation revealed that six out of the ten critical risks are associated with the SoI. These six risks were further ranked based on their scores. Upon analyzing the table, it is evident that although the number of CWEs related to "Identification and Authentication Failure" is the highest (10), its criticality rating is relatively low (0.17). Consequently, this risk is ranked second according to the calculated score (1.66). Among the six risks, "Broken Access Control" is identified as the most critical one,

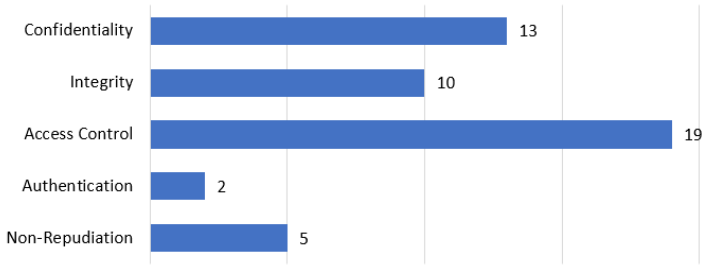

Fig. 7. Analysis of Impact Scopes

TABLE VIII. SUMMARY OF SECURITY RISKS

Security Risk	Number of CWEs	Criticality	Score	Rank
A01-Broken Access Control	8	0.23	1.81	1
A02-Cryptographic Failures	3	0.31	0.92	4
A04-Insecure Design	2	0.20	0.41	6
A07-Identification and Authentication Failures	10	0.17	1.66	2
A08-Software and Data Integrity Failures	3	0.16	0.49	5
A09-Security Logging and Monitoring Failures	5	0.32	1.62	3

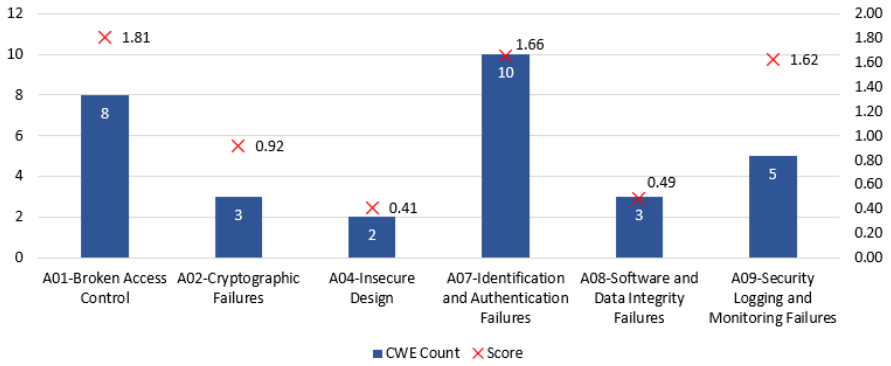

Fig. 5. Analysis of security risk

receiving a score of 1.81, while "Insecure Design" is determined to be the least critical. This evaluation provides insights into the specific CWEs that contribute to the OWASP's Top 10 security risks, allowing stakeholders to prioritize their efforts in addressing the most critical risks affecting the system's security.

The analysis of threats is presented in Table IX, showcasing the severity of threats relevant to the system. Among the six threats evaluated, "Information Disclosure" emerges as the most serious. It is identified as a relatively significant threat, with the most substantial technical impact being "Read Application Data." This analysis enables stakeholders to understand and prioritize the threats that pose the highest risk to the system's security, allowing them to allocate resources and implement appropriate mitigation measures accordingly.

VI. CONCLUSION

This paper introduces a model for quantifying the security evaluation of web applications. Through the utilization of techniques such as aggregation, scoring consolidation, and analytics, organizations can enhance their understanding of the security posture of the system of interest. This improved understanding enables informed decision-making regarding security measures and risk mitigation strategies. Our approach enables the integration of ASVS operational data into a knowledge-based framework, facilitating the extraction of valuable information regarding the security strength of a system and the identification of potential vulnerabilities and threats. This integration enhances the effectiveness of security evaluations and empowers organizations to take proactive measures to mitigate risks and enhance their overall security posture.

By adopting this security evaluation approach, the ASVS requirements are given thorough consideration, offering a holistic perspective of system security encompassing aspects such as "structure," "environment," and "process." This method involves breaking down the security challenges, pinpointing the underlying components, and placing special emphasis on critical or essential security mechanisms at a granular level. These security mechanisms serve as vital connectors between the descriptive ASVS requirements and their subsequent

TABLE IX. SUMMARY OF THREAT SCORES WITH CORRESPONDING TECHNICAL IMPACTS

Threat	Score	Technical Impact	Score
Spoofing	12	Gain Privileges or Assume Identity	12
Tampering	15	Modify Application Data	7
		Modify Memory	0
		Modify Files or Directories	4
		Unexpected State	3
		Alter Execution Logic	1
Repudiation	4	Hide Activities	4
Information Disclosure	17	Read Application Data	12
		Read Memory	0
		Read Files or Directories	5
Denial of Service	7	DoS: Instability	0
		DoS: Resource Consumption (CPU)	3
		DoS: Resource Consumption (Memory)	1
		DoS: Crash, Exit, or Restart	1
		DoS: Resource Consumption (Other)	2
Elevation of Privilege	11	Execute Unauthorized Code or Command	1
		Bypass Protection Mechanism	10

analysis, ensuring a comprehensive and effective evaluation of the system's security. Furthermore, in our test-based approach, we seamlessly integrate the ASVS framework into the evaluation process of identified vulnerabilities, which are mapped to CWE. By leveraging existing CWE databases and employing effective mapping techniques, we generate threat and risk catalogs that align with each ASVS element. This modeling approach leverages the verification results as explicit inputs for evaluation, enabling a more precise and targeted assessment of potential negative impacts and ultimately enhancing the overall analysis outcomes.

To facilitate future research endeavors, it is crucial to acknowledge the limitations of this study. Firstly, the model developed in this work primarily emphasizes technical security mechanisms and may not encompass human factors such as social engineering attacks or insider threats. Furthermore, it is important to note that the model does not guide risk mitigation strategies, as its primary purpose is to serve as an analysis tool rather than a prescriptive guide. In summary, although the model presented in this study offers valuable insights for assessing security posture, it is essential to supplement it with other frameworks and considerations to establish a comprehensive security strategy. A potential future direction would be to enhance the model by developing practical security metrics and integrating them into a dedicated security analytics application, as suggested in the research [26]. The proposed application enhances data analysis capabilities, enabling organizations to conduct comprehensive assessments of crucial security elements. By leveraging this tool, organizations gain a deeper understanding of the necessary actions to ensure security and compliance. Furthermore, automating the security evaluation process could be a valuable step to increase effectiveness and enable real-time monitoring. This represents a significant advancement in continuously improving the system's security measures through the provision of well-structured metrics and analytics.

ACKNOWLEDGMENT

This research work is financially supported by the SFI Norwegian Centre for Cybersecurity in Critical Sectors (NORCICS, NFR project number: 310105).

REFERENCES

[1] Andrews, R., G.A. Boyne, and R.M. Walker. 2006. "Subjective and objective measures of organizational performance: An empirical exploration". Public service performance: Perspectives on measurement and management, pages 14-34.

[2] Banaei, O. and S. Khorsandi. 2012. "A new quantitative model for web service security". in 2012 IEEE 14th International Conference on Communication Technology. IEEE.

[3] Charpentier, F., "Common Criteria Web Application Security Scoring CCWAPSS"; Available from: https://dl.packetstormsecurity.net/papers/web/ccwapss_1.1.pdf. (Accessed on Feb. 3, 2023)

[4] Ekclhart, A., et al. 2007. "Ontological mapping of common criteria's security assurance requirements". in IFIP International Information Security Conference. Springer.

[5] Erşahin, B. and M. Erşahin. 2022. "Web application security". South Florida Journal of Development, volume 3, issue 4, pages 4194-4203.

[6] Gritzalis, D., M. Karyda, and L. Gymnopoulos. 2002. "Elaborating quantitative approaches for IT security evaluation". Security in the Information Society: Visions and Perspectives, pages 67-77.

[7] Hai, H.D. and P.T. Nga. 2018. "Evaluating the security levels of the Web-Portals based on the standard ISO/IEC 15408". in Proceedings of the 9th International Symposium on Information and Communication Technology.

[8] Harrison, S., et al. 2016. "A security evaluation framework for UK e-goverment services agile software development". arXiv preprint arXiv:1604.02368.

[9] Herrmann, D.S. 2002. "Using the Common Criteria for IT security evaluation". CRC Press.

[10] ISO/IEC, "Information security, cybersecurity and privacy protection — Evaluation criteria for IT security — Part 1: Introduction and general model"; Available from: https://www.iso.org/standard/72891.html. (Accessed on Jan. 21, 2023)

[11] LeMay, E., et al. 2011. "Model-based security metrics using adversary view security evaluation (advise)". in 2011 Eighth International Conference on Quantitative Evaluation of SysTems. IEEE.

[12] McGraw, G., B. Chess, and S. Migues. 2009. "Building security in maturity model". Fortify & Cigital.

[13] Okamura, H., M. Tokuzane, and T. Dohi. 2013. "Quantitative security evaluation for software system from vulnerability database".

[14] Ouedraogo, M., et al. 2009. "Security assurance metrics and aggregation techniques for it systems". in 2009 Fourth International Conference on Internet Monitoring and Protection. IEEE.

[15] OWASP, "OWASP Proactive Controls"; Available from: https://owasp.org/www-project-proactive-controls/. (Accessed on Feb. 3, 2023)

[16] OWASP, "OWASP Top10 Introduction"; Available from: https://owasp.org/Top10/A00_2021_Introduction/. (Accessed on Apr. 27, 2022)

[17] OWASP, "Software Assurance Maturity Model v2.0"; Available from: https://www.opensamm.org/. (Accessed on Apr. 30, 2022)
[18] Pham, N. and M. Riguidel. 2007. "Security assurance aggregation for it infrastructures". in 2007 Second International Conference on Systems and Networks Communications (ICSNC 2007). IEEE.
[19] Reddy, N. "An Excellent Compilation of Software Testing Concepts (Manual Testing)".
[20] Schechter, S.E. 2004. "Computer security strength and risk: a quantitative approach". volume: Harvard University.
[21] Shostack, A. 2014. "Threat modeling: Designing for security". volume: John Wiley & Sons.
[22] Sönmez, F.Ö. 2019. "Security qualitative metrics for open web application security project compliance". Procedia Computer Science, volume 151, pages 998-1003.
[23] Vache, G. 2009. "Vulnerability analysis for a quantitative security evaluation". in 2009 3rd International Symposium on Empirical Software Engineering and Measurement. IEEE.
[24] W3C, "RDF 1.1 XML Syntax"; Available from: https://www.w3.org/TR/rdf-syntax-grammar/. (Accessed on Jan. 26, 2022)
[25] Weldehawaryat, G.K. and B. Katt. 2018. "Towards a quantitative approach for security assurance metrics". in The 12th International Conference on Emerging Security Information.
[26] Wen, S.-F., A. Shukla, and B. Katt. 2022. "Developing Security Assurance Metrics to Support Quantitative Security Assurance Evaluation". Journal of Cybersecurity and Privacy, volume 2, issue 3, pages 587-605.
[27] Yautsiukhin, A., et al. 2008. "Towards a quantitative assessment of security in software architectures". in Nordic Workshop on Secure IT Systems (NordSec), Date: 2008/10/01-2008/10/01, Location: Copenhagen, Denmark.
[28] Zhou, C. and S. Ramacciotti. 2011. "Common criteria: Its limitations and advice on improvement". Information Systems Security Association