Computers, Materials & Continua
DOI:10.32604/cmc.2020.012831
Article

Tech Science Press

A Framework for Systematic Classification of Assets for Security Testing

Sadeeq Jan[1,*], Omer Bin Tauqeer[1], Fazal Qudus Khan[2], George Tsaramirsis[2], Awais Ahmad[3], Iftikhar Ahmad[4], Imran Maqsood[5] and Niamat Ullah[6]

[1]National Center for Cyber Security, Department of CS & IT, University of Engineering & Technology, Peshawar, Pakistan
[2]Department of Information Technology, FCIT, King Abdulaziz University, Jeddah, Saudi Arabia
[3]Dipartimento di Informatica (DI), Università Degli Studi di Milano Statale, Via Celoria 18, Milano, Italy
[4]Department of Computer Science & IT, University of Engineering & Technology, Peshawar, Pakistan
[5]Department of Software Engineering, University of Engineering & Technology, Mardan, Pakistan
[6]University of Buner, Buner, Pakistan
[*]Corresponding Author: Sadeeq Jan. Email: sadeeqjan@uetpeshawar.edu.pk
Received: 14 July 2020; Accepted: 07 August 2020

Abstract: Over the last decade, a significant increase has been observed in the use of web-based Information systems that process sensitive information, e.g., personal, financial, medical. With this increased use, the security of such systems became a crucial aspect to ensure safety, integrity and authenticity of the data. To achieve the objectives of data safety, security testing is performed. However, with growth and diversity of information systems, it is challenging to apply security testing for each and every system. Therefore, it is important to classify the assets based on their required level of security using an appropriate technique. In this paper, we propose an asset security classification technique to classify the System Under Test (SUT) based on various factors such as system exposure, data criticality and security requirements. We perform an extensive evaluation of our technique on a sample of 451 information systems. Further, we use security testing on a sample extracted from the resulting prioritized systems to investigate the presence of vulnerabilities. Our technique achieved promising results of successfully assigning security levels to various assets in the tested environments and also found several vulnerabilities in them.

Keywords: Security; security testing; privacy; asset classification

1 Introduction

Complex web-based systems either contain or utilize private and critical information which must remain secure from unauthorized access and tampering. Similarly, basic web applications may also process sensitive information and, are constantly at risk of being attacked. New and complex systems used in cloud computing for data crunching and information gathering may also be vulnerable to various attacks and threats. To ensure the security of these systems and applications, security testing is required. There are various types of security testing techniques that are used to find vulnerabilities. The most common form of testing is Penetration Testing also known as "Pen Testing". Penetration testing is carried out by simulating real attacks on systems to identify exploitable vulnerabilities and the damage they would incur [1,2]. Open Web Application Security (OWASP) is a well-known online community that provides various techniques and tools for securing web-based systems [3]. The most common of these is the document titled as OWASP Top 10 Web Application Vulnerabilities published every 3 to 5 years [4]. The document discusses the most common vulnerabilities that potentially exists in many web applications. It also describes how the vulnerabilities can be exploited by the attackers, along with identifying the key techniques that can be employed as safeguard against such attacks. The fundamental three features of security that are checked during any security testing process are [5,6]:

- Confidentiality: is the assurance that information is not disclosed to unauthorized individuals, processes, or devices.
- Integrity: is provided when data is unchanged from its source and has not been accidentally or maliciously modified, altered, or destroyed.
- Availability: guarantees timely, reliable access to data and information services for authorized users.
- These security principles make the CIA triad which is the most commonly used and oldest security standard around the globe. Over the years with the increase in the complexity and wide variety of systems and applications, more security features have been added such as:
- Authentication: is a security measure designed to establish the validity of a transmission, message, or originator, or a means of verifying an individual's authorization to receive specific categories of information.
- Authorization: provides access privileges granted to a user, program, or process.
- Non-repudiation: is the assurance that none of the partners taking part in a transaction can later deny of its participation.

The focus of providing security should be applied on the web application layer to protect it from unauthorized users by building security across the software development lifecycle security mechanism [7]. The effectiveness of the testing process significantly depends on the tools used to support the process. Testing tools usually automate some of the tasks required by the process, such as test case generation, test case execution and evaluation of the test case result. Several testing tools support the production of useful testing documentation and provide a configuration and management of these tools [8]. The existing approaches for mitigating threats to Web applications can be divided into client-side and server-side solutions. If we look at the server-side security, we can consider an application-level firewall offering protection in case of suspected cross-site scripting (XSS) attacks that attempt to steal a user's credentials [9]. Server-side solutions have the advantage of being able to discover a larger range of vulnerabilities, and the benefit of a security flaw fixed by the service provider is instantly propagated to all its clients. These server-side techniques can be further classified into dynamic and static approaches [10]. Dynamic tools and Perl's taint mode try to detect attacks while executing the audited program, whereas static analyzers scan the Web application's source code for vulnerabilities [11]. Assessment or test of security risks both from outside and within the organization can include someone's access to classified information and transferring it to a USB [12].

Security testing is often performed for a single System Under Test (SUT), however, there are usually more systems or components that needs to be tested in a complex web-based infrastructure. In such cases, it becomes a difficult decision for the tester/organization that which system/component should be tested first among the vast majority of systems [13–18]. For such scenarios, a technique is needed to classify the assets of an organization. In this paper we propose an Asset classification system to assign priority levels to each system based on their security needs, for the web-based Information systems.

Our proposed technique verifies the quality of data that the system stores, analyses, processes and transfers, as well as the criticality of the system determined via a checklist that focuses on such aspect of the system. The technique utilizes information as collected and described during the planning and design stage of the Security Testing and the Software Development Life Cycle (SDLC). This information is further used to consider the exposure to various types of users of the system. All the collected information about various aspects of a SUT is then analyzed to calculate the criticality value of the asset and an appropriate category (High, Medium, Low) is assigned to it. For evaluating the effectiveness of our proposed approach, we performed testing on 400 web based information systems of the province of Khyber Pakhtunkhwa, Pakistan. Finally, we analyzed a sample of the categorized systems for the investigation of OWASP Top 10 vulnerabilities.

The rest of the paper is organized as following. Section 2 provides a succinct summary of the related work. Section 3 provides a background on the OWASP Top 10 vulnerabilities. Sections 4 presents our proposed approach for asset security classification in detail. Section 5 describes the details of our study design including the subjects' selection and methods of analysis. The results are discussed in Section 6. Finally, the conclusion and future work is presented in Section 7.

2 Related Work

Attacks on web-based systems have increased significantly over the last few years. The number of attacks grew from 17 million to 50 million between years 2015 and 2016 [19]. Similarly, the number of new vulnerabilities found in web applications have seen an increase in 2017 by 212% as compared to 2016 [1]. In 2018, Google sent over 45 million notifications to various web administrators alerting them about possible problems with their websites that could affect their appearance in a search. Therefore, there is a dire need to take appropriate security measures to counter these attacks.

A web based platform is a complex system consisting several components, tools, devices, technologies e.g., HTTP/S protocols, application development technologies like PHP, ASP and web clients (browser etc.). Further, almost all types of these systems are continuously being targeted by attackers and therefore organizations use intrusion detection/prevention systems (IDS/IPS) and firewalls to protect and monitors such networks [8]. Although, a number of preventive measures are used to secure the deployed web applications, security testing has become a critical activity at the development phase. The purpose of security testing is to ensure confidentiality and authenticity of the data, as well as ensuring the availability of the services to the end user. Such security testing is used to verify if the web applications fulfills its security requirements in case of malicious user inputs [20]. There are various challenges when carrying out security testing of systems and applications and the newly discovered vulnerabilities are making the task more complicated. Developers/testers need to understand the importance of all such issues/challenges when conducting security testing.

A framework for assessing the risk of vulnerabilities in e-government sites, has been discussed by Anastacio et al. [21]. The authors discussed the benefits and risks of the e-government systems. As per the authors views, the value or importance of an e-government system depends on its difference from other systems and the interactions of the users with the system. Rjaibi et al. [22] provided an analysis technique for the security assessment of e-learning systems. Security requirements such as privacy, non-repudiation, authentication etc. have been identified along with the types of possible attacks such as, buffer overflow, cross site scripting, insecure direct object referencing and information leakage etc. The authors considered the availability as the most important security requirement. Patel et al. [23] proposed a risk assessment modeling technique for modeling the possible attacks and their impact on industrial systems. The technique allowed them to determine financial loses that can occur due to the cyber-attacks on these types of systems. The authors implemented their technique to find the financial loss caused due to an attack on a SCADA based system and found an estimated $454,094 yearly loss possible based on their methodology.

Almadhoob et al. [24] performed a study to analyze cybercrimes and their effects in Bahrain. For this purpose, a survey was carried out by the authors among the different businesses and organizations working in Bahrain. Based on the survey an audit plan was created that if utilized would protect the businesses in Bahrain from cyber-attacks on their systems. The authors found that from 34 total participants, 31 had been affected by phishing attacks. It was also found that most of the participants had not added important security controls to protect their systems. Of the total participants only 13 were found that had controls in place to track changes made to the data hosted on their systems. Saripalli et al. [25] propose a quantitative framework for the calculation of risk and impact on security of different cloud computing environments. The framework measures the security events and categorizes them from among six pre-defined categories. The framework utilizes the wide-band Dolphi method for calculating the measures in quantitative form. According to the authors, the framework would provide the different user types interacting with the cloud environment and regulating agencies with statically usable data. The authors point out that utilization of this framework would require input of risk knowledge and objectives in huge amount.

3 Background of OWASP Vulnerabilities

In addition to the asset classification, we also aim at security testing for vulnerabilities, especially for the web vulnerabilities that may exist in web-based information systems. Therefore, in this section, we provide an overview of the widely known vulnerabilities as listed by the Open Web Application Security Project [3] commonly known as OWASP.

OWASP is a platform developed by and for the IT community. This platform is used to share knowledge and tools for professionals and beginners alike in the pursuance of defending against attacks on web-based systems. OWASP provides open source tools as well as documents focusing on finding security related attacks and vulnerabilities, guard against attacks and further strengthening the security activities protecting the systems. OWASP ZAP [26] is one of the most widely used tool to discover vulnerabilities and attacks that can be used to affect a system. Similarly, OWASP Juice Shop [27], is an application which has been developed with the most common security flaws that affect web applications in mind. It is used as a tool for teaching beginners and new comers to the field of security how various security vulnerabilities can be used by attackers due to the flaws that remain unfixed in the system during the development phase.

Similar to using tools, OWASP also provide documentation for developers to learn about the various vulnerabilities and how to harden the systems against such vulnerabilities [28]. The OWASP testing guide [29] is a useful resource for this purpose and provides detailed best practices for system hardening and security testing. Another documentation project by OWASP is OWASP Top 10 Vulnerabilities [4] that are found in most web applications and demonstrates how a slight coding habit can emerge into a security threat. Most recent OWASP Top 10 list was released in 2017 and lists SQL injection to be the most common and dangerous security.

Following are the OWASP vulnerabilities in the order of their severity.

3.1 Injection

An injection attack allows an attacker to insert malicious data into a program via input sources, e.g., input fields. These attacks are commonly found in SQL, LDAP, XPath etc. In case of SQL attack, the attacker can read, modify, delete the database or execute other queries. In these types of attacks, the coding query handling methods affect the security of the program [30].

3.2 Broken Authentication

Often many web applications require users to login with their credentials. Typical cases require a username and password, that are used to generate a random session id that authenticates all actions as a legitimate user. Disclosure of these credentials occur due to reasons like transmission through insecure channels and security misconfiguration. Upon obtaining such credentials, attackers can impersonate a legitimate user. Therefore, authentication and session management must be managed properly to protect the users' data from unauthorized disclosure or modification [31].

3.3 Sensitive Data Exposure

Data exposure occurs when a web application or program does not adequately protect its data and information. This data if accessed by the attackers can result in financial or business loss. An example could be, exposed data by an error message, weak crypto and lack of headers preventing browser caching.

3.4 XML External Entity (XXE)

XML is used to describe data. Two systems that are running on different technologies can communicate with each other using XML. XML External Entity attack takes place when a reference to an external entity is processed by weakly configured parser that may result in information disclosure, Denial of Service (DoS) attacks, port scanning [32].

3.5 Broken Access Control

In access control mechanism, also known as authorization, users are allotted access to resources according to their roles, e.g., admin, employee or guests etc. Broken access control is one of the most common and highly exploitable vulnerability. Access controls are exploited by changing parameter values, giving direct access to unauthorized system object. Most common impact is privilege escalation—A practice of providing users more rights or access than required, hence weakening the system security [29].

3.6 Security Misconfiguration

Security misconfiguration vulnerabilities appear into systems due to the use of weak passwords, encryption, using default configured setting, incomplete or improper configuration of settings, outdated software's or unpatched flaws etc.

3.7 Cross-Site Scripting (XSS)

Cross-site Scripting attacks are a type of injection attacks. The attacker generally injects the malicious code through a browser site script. Nowadays, JavaScript is enabled in most web applications to provide rich functionalities to users. This also provides the attacker an opportunity to exploit and execute their attack. One of the main difficulties in stopping XSS vulnerabilities is proper character encoding where the web applications are unable to filter the character encodings for example there is a possibility that the web application might filter out <script> but won't filter %3script%3e which is another encoding of tags [29].

3.8 Insecure Deserialization

Deserialization is the transformation of incoming serialized data from file, stream or network socket into an object. It is very common and regular process for any web application to serialize and deserialize data. Reserialization of untrusted data by an application can lead to an attacker launching DOS, authentication bypass, etc.

3.9 Using Components with Known Vulnerabilities

This vulnerability occurs when the system is not updated and patched on regular basis. In most cases, once a web application is developed and handed over to the customer, no proper maintenance is carried out afterwards.

3.10 Insufficient Logging and Monitoring

This vulnerability may exist when there is inadequate/insufficient logging and monitoring facilities in the SUT. This may also occur due to a human fault that lead to vulnerability in the system, improper management of logs.

4 Proposed Framework for Asset Security Classification

Our proposed approach consists of the following two phases:

1. Asset Security Classification Framework
2. Testing for OWASP Vulnerabilities in critical assets

We developed a framework for asset security classification that assigns an appropriate security level to assets based on various types of information, e.g., interfaces, dependencies, user access, exposure level. The framework consists of the following components.

4.1 Information Gathering

For calculating the prioritization of assets, we collect various types of information. Our framework gathers information and categorizes it as shown in Fig. 1.

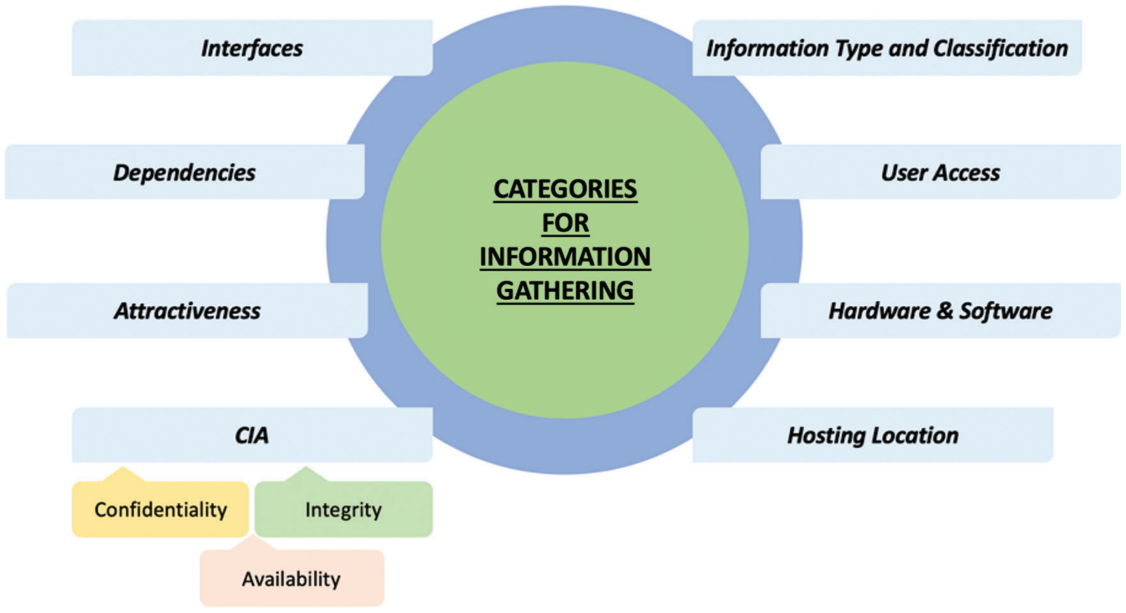

Figure 1: Various types of information collected in our technique

4.1.1 Information Type and Classification

This category consists of information being stored, processed or transmitted by the system. The type of information can be specified as personal information, financial information and system configuration information. Classification is made on the basis of the security level of the information. The information can be classified as either secret, confidential, internal or public information. This category also describes that in case of the information being secret or confidential, whether there exists a possibility for the information to be transferred to a removable device or communicated outside the system through electronic communication media such as email.

4.1.2 User Access

This category describes finding the access to the system that users may be depending upon for their authorization level in the system. This can include users such as the general public, subscribed customers, employees, third party and licensed system operators.

4.1.3 Hardware and Software

In this category, we consider the hardware and software specifications of the systems. This can include the portable devices, IT and network equipment, operating systems, other software and development tools that are used in the system.

4.1.4 Hosting Location

Information such as the hosting location of the system, environments such as any testing and training environments are present etc., is gathered in this section. Also, any information about specific hardware necessary to access or run the system is collected here as well.

4.1.5 Interfaces

In this category, the information about any interfaces in the system for different Software Development Kit (SDK) and Application Programming Interface (API) libraries integration is gathered. Proper identification of interfaces is necessary as they expose the system to vulnerabilities when proper protection steps are not taken.

4.1.6 Dependencies

Information about the systems dependency on other systems, networks and services is gathered in this category. Often dependencies can have a major impact on the security/protection of the system.

4.1.7 Attractiveness

In this category the information about how much an attacker would be attracted to the system is discussed. Gathering information about attractiveness of the system to attackers helps in the identification of weak points and possible defenses that can be put in place. Here questions, such as use of system capabilities to exploit other systems, are also discussed.

4.1.8 Confidentiality, Integrity and Availability (CIA)

The CIA triad consists of confidentiality, integrity and availability of information.

- Confidentiality: As defined in Section 1, measuring confidentiality of the system information and data allows for creating safeguards against leakage of the data. Impact or loss of confidentiality and the likelihood of the loss is measured using this category. Loss of confidentiality can have an adverse effect on the reputation of the system and massive impact on system's reliability.
- Integrity: Like confidentiality, here the effects of loss of integrity of the data is measured. Loss of data integrity can affect systems' reputation and expose it to financial and legal repercussions.
- Availability: As described in the introduction section and like the above-mentioned confidentiality and integrity related problems, a breach of system availability will have a massive impact on it's overall reputation and create a disastrous situation for the organization.

4.2 Parameters

As previously described, the proposed technique utilizes basic information collected regarding each of the Systems Under Test to measure, in quantitative terms, the security requirements and sort the systems based on these values. For this purpose, we calculate the following three main variables.

4.2.1 Exposure Factor (EF)

This variable measure, in quantitative terms, the access of the system to the world. This refers to the interaction of users in its current security state. If the systems exposure level is high, its risk of becoming an attractive target for a threat agent increases as well as the potential of having more vectors for gaining possible entries into the system. If the threat agents gain entry into the system, the least they can do is to deface the system resulting in a long-lasting stain on the reputation of the organization to which the system or site belongs.

Mathematically, we can express the Exposure Factor as $\mathcal{E}f = \frac{\sum \alpha}{\tau}$ where, α represents the total user access and τ is the predefined threshold value. These values of the user access and threshold are assigned based on a predefined set of security related questions, and their corresponding weightage about the SUT. The resulting $\mathcal{E}f$ value is in percentage 1–100%.

4.2.2 Cumulative CIA (CCIA)

Security triad is a globally accepted standard for measuring the security requirements of any digital system. The standard helps in identifying and measuring the criticality of the system to the organization. This also helps in assessing the amount of possible damage to inflict on the organization if any of the three security standards are breached.

We represent CCIA mathematically as: $CC = Max\ (C, I, A)$ where cC represents cumulative CIA. The terms C, I, A refers to the calculations of Confidentiality, Integrity and Availability score calculated via three equations, i.e., $C = \frac{\Sigma C \times A\gamma}{\tau}$, $I = \frac{\Sigma I \times A\gamma}{\tau}$ and $A = \frac{\Sigma I \times A\gamma}{\tau}$ where $\sum C$, $\sum I$ and $\sum A$ represent the total scores of Confidentiality, Integrity and Availability respectively. Similarly, is the Asset Value Level, and τ is the threshold value.

4.2.3 Asset Value Level ($A\gamma$)

The Asset Value Level ($A\gamma$) is our own developed scale for measuring the value of the system or site to the organization. It is a normalized weighted score calculated from the combination of two parameters namely $\mathcal{E}f$ and CC. The $A\gamma$ score helps in assessing the overall requirements of the system and the features of the system in its current state. In addition, allows us to not only consider the value of a single system but also to group together systems having the same value level. These groups can further be assigned to different teams based on the value of the systems and the skillset of the team members allowing for a faster and efficient assessment of the system.

By identifying and measuring these terms in quantitative values, we believe will increase the probability of correctly identifying the critical factors of the system for prioritizing them from the pool of systems that need to be assessed further.

4.3 Clustering Based on Calculated Parameters

To make the results easier to understand, a generalized approach is used to cluster the assets into 3 types based on their security requirements. For this purpose, we have defined custom ranges for each of the three parameters, i.e., Exposure Factor ($\mathcal{E}f$), Cumulative CIA (CC) and Asset Value Level ($A\gamma$) based on our requirements/specifications. Fig. 2 shows these ranges/measurements of scale for each of the three factors. The assets are finally assigned a security level of High, Medium or Low based on the average values as shown in Fig. 3.

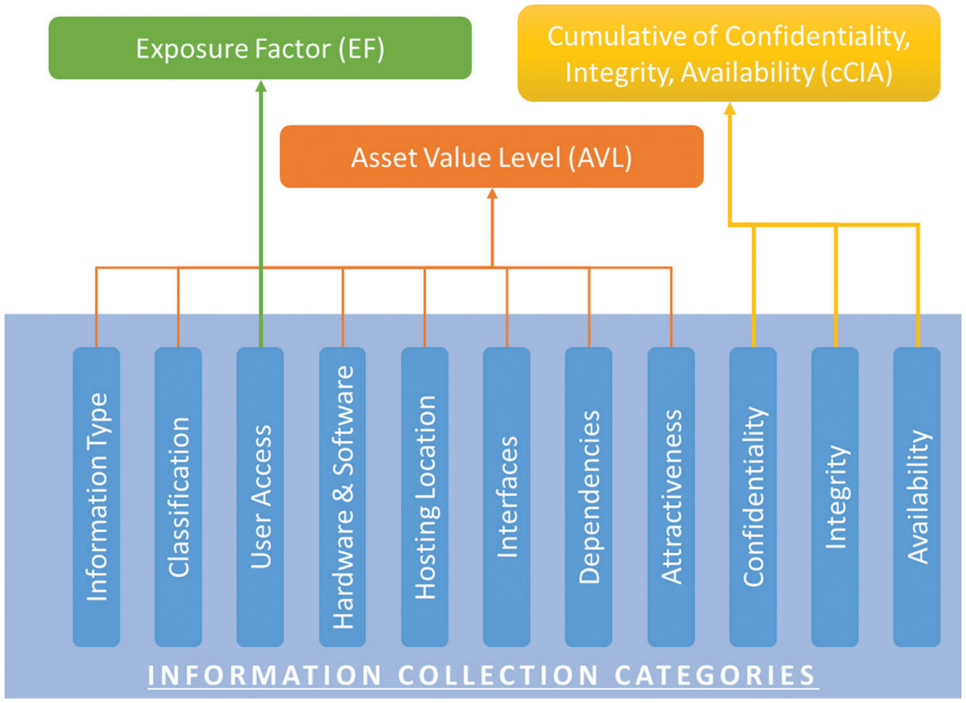

Figure 2: Parameters for asset security classification

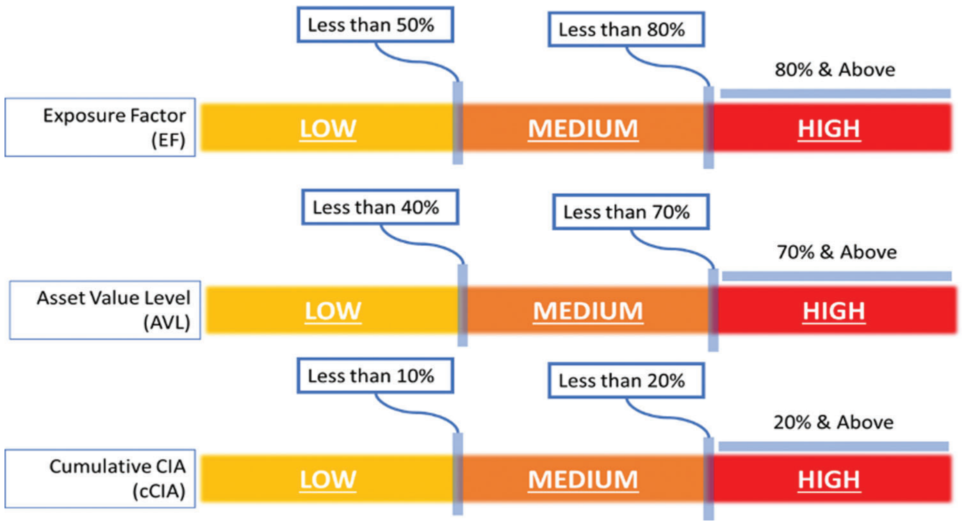

Figure 3: Ranges for the parameters of asset classification framework

5 Study Design

For evaluation of our proposed asset security framework, we used the questionnaire developed by KP CERC [33]—An initiative of the KPITB, to collect data as defined in Section 4. After the data collection, analysis was performed to measure the required parameters and evaluate our methodology for asset classification based on security requirements of the systems. We used Google Forms in our study to gather information about each web-based system and its interaction with other systems. The calculation of the parameters Exposure Factor ($\mathcal{E}f$), Cumulative CIA (CC) and Asset Value Level ($A\gamma$) and various types of analysis was performed on the collected data in Google Sheets. The methodology steps are shown in Fig. 4.

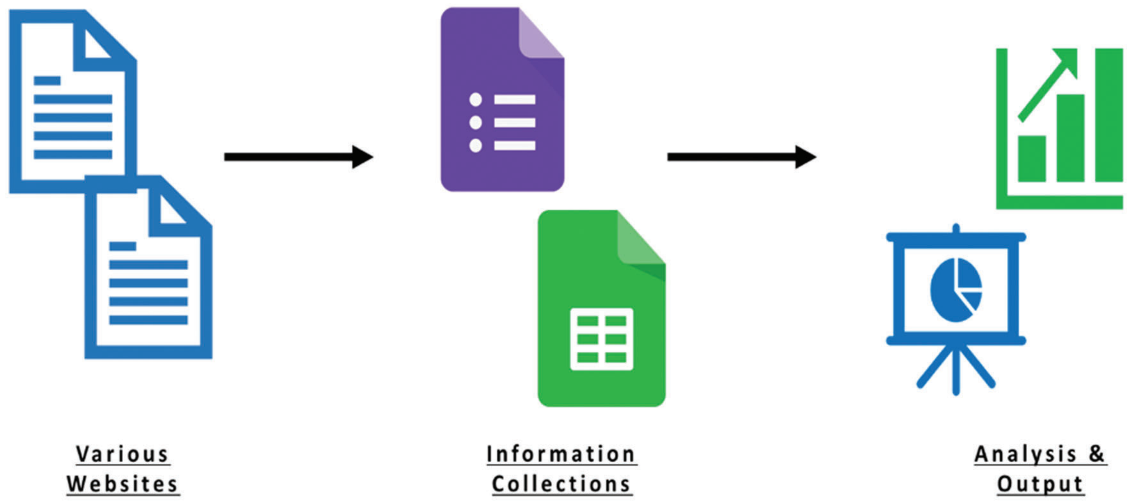

Figure 4: Ranges for the parameters of asset classification framework

We evaluated the proposed framework for asset security classification on a case study consisting of 451 web-based systems related to various organizations of the province of Khyber Pakhtunkhwa, Pakistan. However, for confidentiality reasons, we anonymize the names of these systems in this paper.

Initially we collected the web related information about each system in a total of 451 links including some dual links that were still open after the system development period expired. Each of these websites were then individually examined and information was gathered by use of the questionnaire for all the links collected except duplicates. However, some of the information could only be provided by the administrator of the application. As we were not able to get access to the administrators, we assumed thresholds for such information under advisory of cyber security experts.

6 Results

In this section, we describe our results in the following two parts:

6.1 Results for the Proposed Technique for Asset Classification

We attempted to manually access all of the 451 links initially, however 38 systems (8%) were not accessible via the internet, and therefore our further investigation was performed on the remaining 413 (92%) of the systems. There are various reasons for the non-accessibility of these systems, e.g., power outage, no or limited public interface of the systems.

We tried to collect detailed information of the remaining 413 websites. However, there were some highly sensitive systems for which we did not have permissions and therefore they could not be classified into any category using the proposed technique. These "Not Classified" systems accounted for 30% (171) of the total 413 systems. The remaining 242 systems were analyzed in detail by collecting all information related to the categories defined in the proposed framework.

All the required data was collected, and those parameters were calculated for the 242 systems for their security classification. As shown in Fig. 5, 82% of the systems were assigned Medium Security level based on our calculations. A high portion of these systems include static sites with integrity and availability as the main requirements. On the other hand, 15 (6%) assets (applications) were categorized as High in our analysis. This includes the systems that interact with some private information of the citizens and information as well as those from among the departments. Hence, applications grouped as High category have higher requirements of security and should be given more focus for testing and implementing security features. 12% of the systems achieved Low category. Applications in this category are easily available to public and do not have security requirements.

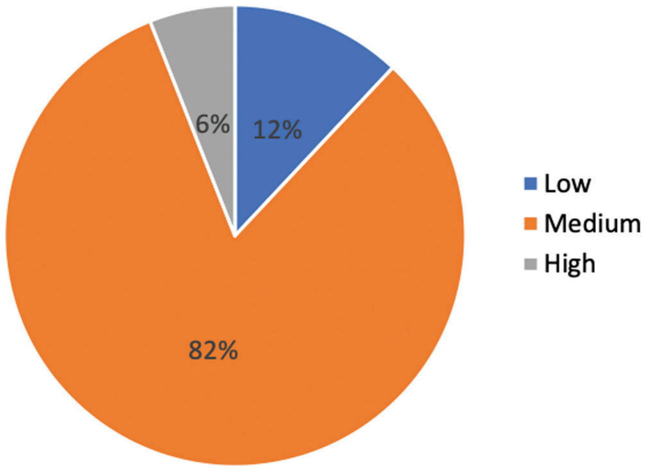

Figure 5: Security level classification of the studied systems

6.2 Testing of Study Cases

As stated previously, our approach also includes security testing of web-based information systems. We picked a sample of 20 systems for vulnerability testing belonging to different organizations and categories as listed in Tab. 1. We also ensured to select systems from both High and Medium Categories as assigned using our proposed technique of asset classification.

It is important to mention here that the vulnerability testing was performed manually and we did not use any automated penetration testing tools to ensure that no damages/disruptions are caused to these systems. Tab. 1 lists some basic information about our sample applications tested for vulnerabilities. As can be observed from the table, the actual URLs of the sites are anonymized (Site 1, Site 2, etc.) because of confidentiality reasons.

Fig. 6 depicts the summary of the results of our vulnerability testing on the sampled applications. As can be seen in the figure, the vulnerability of broken access control is found in large number of applications, followed by sensitive data exposure, using components with known vulnerabilities and security misconfiguration. On the other hand, the cross-site scripting, injection and broken authentication is found only in few applications.

Tab. 2 lists the detailed results of our vulnerability testing, i.e., for each site with the site name anonymized (Site 1, Site 2, etc.). From the table, it can be observed that the minimum number of vulnerabilities found were regarding broken authentication, injection and cross-site scripting in 1, 2 and 3 applications respectively. Vulnerabilities such as the sensitive data exposure, broken access control and using components with known vulnerabilities were found to be affecting more than half of the tested web applications. The three vulnerabilities Insufficient logging and monitoring, XXML external entities and insecure deserialization could not be tested as they either require an automated tool or in-depth analysis that might disrupt the SUTs.

Table 1: Sample of selected sites for vulnerabilities testing

S. No	Category	URL	HTTP/HTTPS	#pages	Certificates	Certificate authority
1	Health	Site 1	HTTP	8	No	N/A
2	Management	Site 2	HTTPS	25	Yes, Valid till…	cPanel, Inc.
3	Education	Site 3	HTTP	35	No	N/A
4	Government	Site 4	HTTPS	46	Yes, Valid till…	cPanel, Inc.
5	Welfare & Security	Site 5	HTTPS	40	Yes, Valid till…	cPanel, Inc.
6	Finance	Site 6	HTTP	42	No	N/A
7	Management	Site 7	HTTPS	43	Yes, Valid till…	COMODO CA limited
8	Development	Site 8	HTTP	34	No	N/A
9	Government	Site 9	HTTP	43	No	N/A
10	Health	Site 10	N/A	–		
11	Education	Site 11	HTTP	21	No	N/A
12	Education	Site 12	HTTP	27	No	N/A
13	Finance	Site 13	HTTP	58	No	N/A
14	Management	Site 14	HTTP	76	No	N/A
15	Welfare & security	Site 15	HTTP	–	No	N/A
16	Management	Site 16	HTTP	17	No	N/A
17	Development	Site 17	HTTPS	43	No	N/A
18	Development	Site 18	HTTP	14	No	N/A
19	Development	Site 19	HTTP	–	No	N/A
20	Education	Site 20	HTTP	45	No	N/A

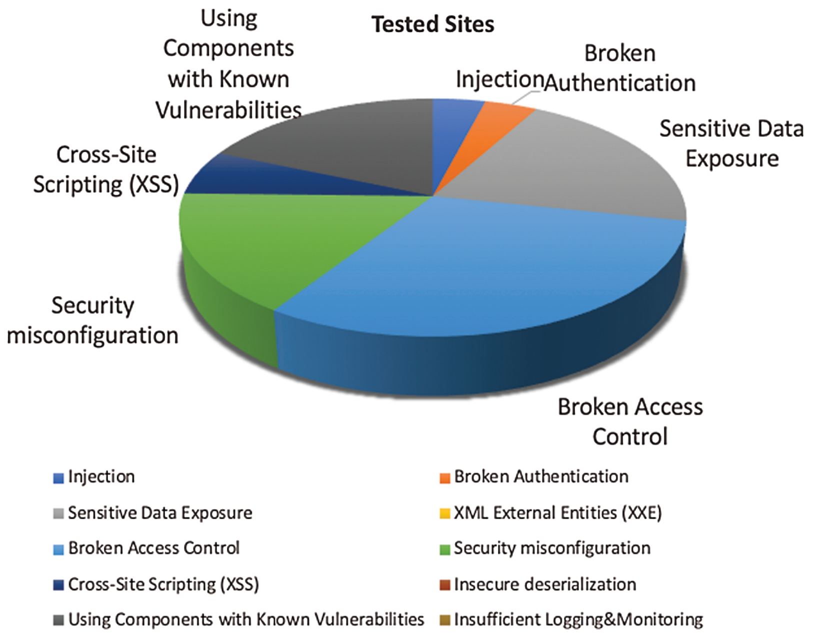

Figure 6: Penetration testing results

Table 2: Results of security testing for sample sites

Site	Injection	Broken authentication	Sensitive data exposure	XML external entity	Broken access control	Security misconfiguration	Cross-site scripting	Insecure deserialization	Using components with known vulnerabilities	Insufficient logging & monitoring
Site 1	No	No	Yes	–	Yes	Yes	No	–	Yes	–
Site 2	No	No	Yes	–	Yes	Yes	No	–	Yes	–
Site 3	No	No	Yes	–	Yes	Yes	No	–	Yes	–
Site 4	No	No	Yes	–	Yes	No	No	–	No	–
Site 5	Yes	No	Yes	–	Yes	Yes	Yes	–	Yes	–
Site 11	No	No	Yes	–	Yes	No	No	–	No	–
Site 12	Yes	No	Yes	–	Yes	Yes	Yes	–	Yes	–
Site 13	No	Yes	No	–	Yes	Yes	Yes	–	Yes	–
Site 14	No	No	Yes	–	Yes	No	No	–	Yes	–
Site 15	No	No	Yes	–	Yes	No	No	–	Yes	–
Site 16	No	No	No	–	Yes	No	No	–	No	–
Site 17	No	No	Yes	–	Yes	Yes	No	–	No	–
Site 18	No	No	Yes	–	Yes	No	No	–	Yes	–
Site 19	No	No	Yes	–	Yes	Yes	No	–	Yes	–
Site 20	No	No	No	–	Yes	No	No	–	No	–

7 Conclusion and Future Work

In this paper, a framework is proposed for Asset Security Classification of information systems. The framework is based on the collection and analysis of a huge data about the System Under Test (SUT) and assigned a security level (High, Medium, Low) to the SUT to prioritize it for security testing. The framework calculates the confidentiality, integrity, availability, and exposure level of the SUT using the formulated mathematical equations before automatically assigning a security level to the SUT. The framework is evaluated on a collection of 451 web-based information systems. A total of 242 systems were assigned appropriate Security Level (High, Medium, Low) as the remaining systems were either not accessible or not allowed to be tested for security reasons. We further performed security testing on a sample extracted from the prioritized systems to investigate the presence of OWASP (Open Web Application Security) Top 10 vulnerabilities. Our results revealed the presence of several vulnerabilities in these systems.

The proposed approach is beneficial as it provides organization a detailed overview about the security requirements for their assets and to prioritize their assets for security testing.

Although, our focus in this work is to prioritize and test the web-based systems, the proposed technique is generic and can be applicable to other types of systems with little modification. In addition, automation of the proposed framework can also be performed in future to collect all the data automatically from the web based systems, analyze it and assign an appropriate security level.

Acknowledgement: We are very thankful to the team of KP CERC (Khyber Pakhtunkhwa Cyber Emergency & Response Center), an initiative of the KPITB, for providing the questionnaire and other resources to collect data about the web-based systems for asset classification. We are also thankful to our M.Sc. students Bilal Shahzad for data collection for assets classification and Syeda Warda Asher for vulnerabilities testing.

Funding Statement: This research is funded by the Higher Education Commission (HEC), Pakistan through its initiative of National Center for Cyber Security for the affiliated Security Testing—Innovative Secured Systems Lab (ISSL) established at University of Engineering & Technology (UET) Peshawar, Grant No. 2(1078)/HEC/M&E/2018/707.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] B. Arkin, S. Stender and G. McGraw, "Software penetration testing," *IEEE Security & Privacy*, vol. 3, no. 1, pp. 84–87, 2005.

[2] D. D. Bertoglio and A. F. Zorzo, "Overview and open issues on penetration test," *Journal of the Brazilian Computer Society*, vol. 23, no. 2, pp. 1–16, 2017.

[3] OWASP Foundation, "Open web application security project," 2004. Available: https://owasp.org/.

[4] D. Wichers and J. Williams, "OWASP top-10 2017," OWASP Foundation, 2017. [Onine]. Available: https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/.

[5] M. Felderer, M. Büchler, M. Johns, A. D. Brucker, R. Breu *et al.*, "Security testing: A survey, " in *Advances in Computers*. vol. 101, Elsevier, Cambridge, MA, USA, pp.1–51, 2016.

[6] P. Herzog, "OSSTMM 3: The open source security testing methodology manual-contemporary secutiy testing and analysis," ISECOM, 2010. [Online]. Available: https://www.isecom.org/OSSTMM.3.pdf.

[7] G. A. Di Lucca, A. R. Fasolino, F. Faralli and U. De Carlini, "Testing web applications," in *Proc. Int. Conf. on Software Maintenance*, Montreal, Quebec, Canada, 2002.

[8] X. Li and Y. Xue, "A survey on web application security," Nashville, TN USA: Vanderbilt University, Technical Report, 2011. [Online]. Available: https://www.isis.vanderbilt.edu/sites/default/files/main_0.pdf.

[9] N. Jovanovic, C. Kruegel and E. Kirda, "Pixy: A static analysis tool for detecting web application vulnerabilities," in *Proc. of IEEE Sym. on Security and Privacy*, Berkeley/Oakland, CA, USA, 2006.

[10] B. Chowdhury, M. Chowdhury and J. Abawajy, "Securing a smart anti-counterfeit web application," *Journal of Networks*, vol. 9, no. 11, pp. 2925–2933, 2014.

[11] D. Yadav, D. Gupta, D. Singh, D. Kumar and U. Sharma, "Vulnerabilities and security of web applications," in *Proc. of ICCCA*, pp. 1–5, 2018.

[12] C. Osborne, "Heathrow airport fined £120,000 over USB data breach debacle | ZDNet," 2019. [Online]. Available: https://www.zdnet.com/article/heathrow-airport-fined-120000-over-usb-data-breach-debacle/.

[13] B. Grobauer, T. Walloschek and E. Stocker, "Understanding cloud computing vulnerabilities," *IEEE Security & Privacy*, vol. 9, no. 2, pp. 50–57, 2010.

[14] N. Hoque, M. H. Bhuyan, R. C. Baishya, D. K. Bhattacharyya and J. K. Kalita, "Network attacks: Taxonomy, tools and systems," *Journal of Network and Computer Applications*, vol. 40, no. 3, pp. 307–324, 2014.

[15] R. Montesino, S. Fenz and W. Baluja, "SIEM-based framework for security controls automation," *Information Management & Computer Security*, vol. 20, no. 4, pp. 248–263, 2012.

[16] P. Radanliev, D. C. D. Roure, J. R. C. Nurse, R. M. Montalvo, P. Burnap *et al.*, *Design Principles for Cyber Risk Impact Assessment from Internet of Things (IoT)*. Oxford: University of Oxford, 2019.

[17] A. Srivastava, T. Morris, T. Ernster, C. Vellaithurai, S. Pan *et al.*, "Modeling cyber-physical vulnerability of the smart grid with incomplete information," *IEEE Transactions on Smart Grid*, vol. 4, no. 1, pp. 235–244, 2013.

[18] A. Fielder, E. Panaousis, P. Malacaria, C. Hankin and F. Smeraldi, "Decision support approaches for cyber security investment," *Decision Support Systems*, vol. 86, no. 1, pp. 13–23, 2016.

[19] A. Talalaev, "5 reasons why website security is important," 2018. [Online]. Available: https://www.webarxsecurity.com/5-reasons-website-security-important-2018/.

[20] A. Jaiswal, G. Raj and D. Singh, "Security testing of web applications: Issues and challenges," *International Journal of Computer Applications*, vol. 88, no. 3, pp. 26–32, 2014.

[21] M. Anastacio, J. A. Blanco, L. Villalba and A. Dahoud, "E-government: Benefits, risks and a proposal to assessment including cloud computing and critical infrastructure," in *Proc. of ICIT*, Amman, Jordan, pp. 1–8, 2013.

[22] N. Rjaibi, L. B. A. Rabai, A. B. Aissa and M. Louadi, "Cyber security measurement in depth for e-learning systems," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 2, no. 11, pp. 107–120, 2012.

[23] S. Patel and J. Zaveri, "A risk-assessment model for cyber attacks on information systems," *Journal of Computers*, vol. 5, no. 3, pp. 352–359, 2010.

[24] A. Almadhoob and R. Valverde, "Cybercrime prevention in the Kingdom of Bahrain via IT security audit plans," *Journal of Theoretical and Applied Information Technology*, vol. 65, no. 1, pp. 274–292, 2014.

[25] P. Saripalli and B. Walters, "Quirc: A quantitative impact and risk assessment framework for cloud security," in *Proc. IEEE 3rd Int. Conf. on Cloud Computing*, Miami, FL, USA, pp. 280–288, 2010.

[26] S. Bennetts, R. Pereira and R. Mitchell, "OWASP zed attack proxy (ZAP)," 2015. [Online]. Available: https://owasp.org/www-project-zap/.

[27] B. Kimminich, "OWASP juice shop," 2017. [Online]. Available: https://owasp.org/www-project-juice-shop/.

[28] S. Kazerooni, D. Cuthbert, A. Van der Stock and K. Raja, "OWASP application security verification standard," 2014. [Online]. Available: https://www.owasp.org/images/5/58/OWASP_ASVS_Version_2.pdf.

[29] M. Meucci and A. Muller, "OWASP testing guide v4, OWASP foundation," 2014. [Online]. Available: https://www.owasp.org/images/5/58/OWASP_ASVS_Version_2.pdf.

[30] J. Clarke-Salt, *SQL Injection Attacks and Defense*. Burlington, MA, USA: Elsevier, 2012.

[31] D. Huluka and O. Popov, "Root cause analysis of session management and broken authentication vulnerabilities," in *Proc. of WorldCIS-2012*, Guelph, ON, pp. 82–86, 2012.

[32] A. Andreu, *Professional Pen Testing for Web Applications*. New York, USA: John Wiley & Sons Inc., 2006. [Online]. Available: https://www.amazon.com/Professional-Pen-Testing-Web-Applications/dp/0471789666.

[33] KPCERC Cyber Emergency & Response Center, "KP ITBoard," 2018. [Online]. Available: https://kpcerc.com/.