اولین کنفرانس بین المللی و چهارمین همایش ملی مدیریت، روان شناسی و علوم رفتاری
تهران - ۳۱ فروردین ۱۴۰۲
1st International & 4th National Conference on Management, Psychology and Behavioral Sciences
20 April 2023 | Tehran

A study on security, and vulnerability on the web

Ali Khodayaran
K. N. Toosi University of Technology
Student of Computer Networks at
Khwaja Nasiruddin Toosi University
Tehran, Iran khodayaran@gmail.com

***Abstract*—Web security This includes cloud security and web application security, which defend cloud services and web applications, respectively.Website Security Check includes site scanning of URLs for potential vulnerabilities and malware through website security software. The world is exceedingly reliant on the Internet. Nowadays, web security is the biggest challenge in the corporate world. It is considered as the principle framework for the worldwide data society. Web applications are prone to security attacks. Web security is securing a web application layer from attacks by unauthorized users. A lot of the issues that occur over a web application is mainly due to the improper input provided by the client. This paper discusses the different aspects of web security and its weaknesses. The main elements of web security techniques such as the passwords, encryption, authentication and integrity are also discussed in this paper.The anatomy of a web application attack and the attack techniques are also covered in details.This paper explores a number of methods for combatting this class of threats and assesses why they have not proven more successful.This paper proposes a better way for minimizing these types of web vulnerabilities. It also provides the best security mechanisms for the said attacks.**

Keywords— Network Security, Network attacks, Firewall, Security Vulnerabilities, Web, CAPTCHAs

I. Introduction

In general, web security refers to the protective measures and protocols that organizations adopt to protect the organization from cyber criminals and threats that use the web channel. Web security is critical to business continuity and to protecting data, users and companies from risk.

The Internet is a dangerous place! With great regularity, we hear about websites becoming unavailable due to denial of service attacks, or displaying modified (and often damaging) information on their homepages. In other high-profile cases, millions of passwords, email addresses, and credit card details have been leaked into the public domain, exposing website users to both personal embarrassment and financial risk. The purpose of website security is to prevent these (or any) sorts of attacks. The more formal definition of website security *is the act/practice of protecting websites from unauthorized access, use, modification, destruction, or disruption*. Effective website security requires design effort across the whole of the website: in your web application, the configuration of the web server, your policies for creating and renewing passwords, and the client-side code. While all that sounds very ominous, the good news is that if you're using a server-side web framework, it will almost certainly enable "by default" robust and well-thought-out defense mechanisms against a number of the more common attacks. Other attacks can be mitigated through your web server configuration, for example by enabling HTTPS. Finally, there are publicly available vulnerability scanner tools that can help you find out if you've made any obvious mistakes. The rest of this article gives you more details about a few common threats and some of the simple steps you can take to protect your site.

Web security is an important aspect for web applications. Today web security is a real concern related to the Internet. It is considered as the principle framework for the worldwide data society. Web applications provide a better interface for a client through a web page .The web page script gets executed on the client web browser. Web applications are a main base of attacks such as cross-site scripting, cookie-session theft, browser attack,self-propagating worms in web email and web sites. These types of attacks are called 'injection attacks' which are attacks by the use of malicious code. Injection attacks have commanded the highest point of web application vulnerability lists for a significant part of the previous decade. There are two most common security vulnerabilities today: SQL injection and cross-site scripting .A security evaluation of application defense center, which had more than 250 e-commerce applications, online banking and the corporate sites came up with a statement that more than 85% of web applications are vulnerable to attacks. CAPTCHA was invented in 2000 at CMU by Luis Von Ahn, Manuel Blum, Nicholas J. Hooper and John Langford. CAPTCHA stands for Completely Automated Public Turing Test to tell Computers and Humans Apart. CAPTCHA is a defensive system that acts as a tool to prevent web bots from abusing online services on the internet including free email providers, wikis, blogs etc. It is a HIP system that is widely used to secure internet based applications. There are 3 basic properties that CAPTCHAs must satisfy:

- It should be easy for human users to pass.
- It should be easy for a tester machine to generate and grade.
- It should be hard for a software robot to pass.

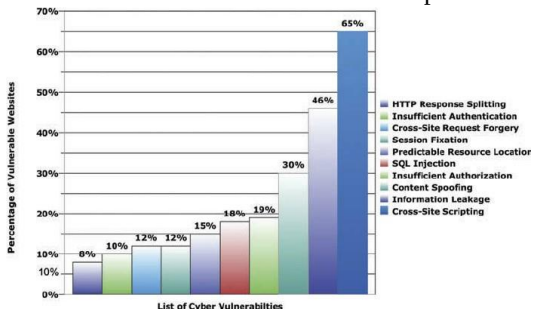

Fig. 1. Percentage of websites containing the vulnerabilities

II. LITERATURE REVIEW

A. Network Security

The main issue in web security research is in enabling a user a safe and trusted platform for communication with the web application. But some people continue to do business with insecure sites. Some organizations or companies don't want to reveal the information about their own security holes. So, it's a very hard task to get reliable information about the state of web security today. There are two common important security vulnerabilities today: SQL injection and cross-site scripting. These types of vulnerabilities directly affect web servers, application servers, and web application environments. OWASP in this paper explores a number of Table 1: Reasons for Attacks[2] methods for detecting threats and assess why they have not proven more successful.A better mechanism for minimizing such type of web vulnerabilities is proposed in this paper. Currently, there are many privacy risks in web applications. Today too many websites are hacked by anonymous people. They target websites because of different types of reasons. They are mentioned in table 1.

RANK	CATEGORY	CWES MAPPED	MAX INCIDENCE RATE	AVG INCIDENCE RATE	MAX COVERAGE	AVG COVERAGE	AVG WEIGHTED EXPLOIT	AVG WEIGHTED IMPACT
A01:2021	BROKEN ACCESS CONTROL	34	55.97%	3.81%	94.55%	47.72%	6.93	5.93
A02:2021	CRYPTOGRAPHIC FAILURES	29	46.44%	4.48%	79.33%	34.85%	7.29	6.81
A03:2021	INJECTION	33	19.09%	3.37%	94.04%	47.90%	7.25	7.15
A04:2021	INSECURE DESIGN	40	24.19%	3.00%	77.25%	42.51%	6.46	6.78
A05:2021	SECURITY MISCONFIGURATION	20	19.84%	4.51%	89.58%	44.84%	8.12	6.56
A06:2021	VULNERABLE AND OUTDATED COMPONENTS	3	27.96%	8.77%	51.78%	22.47%	5.00	5.00
A07:2021	IDENTIFICATION AND AUTHENTICATION FAILURES	22	14.84%	2.55%	79.51%	45.72%	7.40	6.50
A08:2021	SOFTWARE AND DATA INTEGRITY FAILURES	10	16.67%	2.05%	75.04%	45.35%	6.94	7.94
A09:2021	SECURITY LOGGING AND MONITORING FAILURES	4	19.23%	6.51%	53.67%	39.97%	6.87	4.99
A10:2021	SERVER-SIDE REQUEST FORGERY	1	2.72%	2.72%	67.72%	67.72%	8.28	6.72

Attack Goal	%
Stealing Sensitive Information	42%
Defacement	23%
Planning Malware	15%
Unknown	08%
Deceit	03%
Blackmail	02%
Link Spam	03%
Worm	01%
Phishing	01%
Information Warfare	01%

Table. 1. OWASP focussed on identify some vulnerability for the broad array of the organization

III. PROPOSED ARCHITECTURE AND FRAME FOR DETECTING SECURITY VULNERABILITIES

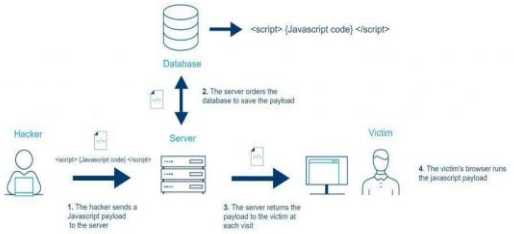

Fig. 2. Website In Vulnerability

The literature survey study of different vulnerabilities defines that two common security vulnerabilities are SQL injection and XSS. Many technology and mechanism are proposed by different researchers to prevent the SQL and XSS attack.During our research we have found out that SQL injection attack and XSS attacks are still possible even after implementing preventing mechanism presently available in the market,and so provide a preventive mechanism we have proposed the architecture shown in figure 2. In this figure we have proposed a scheme through which we will attack any website with SQL injection whether it is prevented by any of the current available mechanisms. In our architecture we describe all the sessions. Client: The client of a web browser is effectively making client requests for pages from servers all over the web. In this

article client login to system normally,client sends request to server and gets response.This happens only in normal scenario.

Attacker:Attacker is an unauthorized user. Typically This kind of attacker would be a proficient programmer or engineer with sufficient technical knowledge to understand the weak points in a security system. In this article Attacker attacksthe website through SQL injection and XSS.Uses of SQL injection and XSS by the attacker is mentioned below.

A. SQL Injection Attack:

SQL injection is one of the most common techniques for web hacking, where the hacker injects malicious code into the database connected to it through an entry port on the site. This portal can be a registration page and fields that request information from the user.

That is, the hacker enters your site like a normal user and enters the registration section instead of the name, username, password, etc., and enters malicious codes. Based on the programming and coding of the site, all these entries are placed in your database assuming that they are correct values and ultimately cause malfunctions and disasters.

To better understand this, imagine that your data center or site is a bank that is visited by thousands of people every day. Now, a vandal enters your bank wearing normal clothes like a normal person and bringing a device causes the security systems and cameras to fail. The entrance door of the bank is the same as the registration portals of the site, and the device that disrupts the security systems is the same malicious code.

The common use of SQL injection attack is to abuse web pages that allow users to input data into form fields for database queries. Injection is an unintended command sent to an interpreter. Attackers can enter the modified SQL query for user information. The queries directly communicate with database for operations on data like data delete, create and change .The queries create link of the static part and value intended for attack .For example: suppose there are two form field, one for entering the username and one for password, the authentication is done as follows:

1. String strr ="select count(*)"Form(user) Where uname=' " ?? "'& Password='" ? " ' ";
2. SELECT * FROM users WHEREemail = 'xx?@x?x.x?x' AND password = md5('???') OR 1 = -1]');

If user has to be allowed to enter apostrophe(') ,use replace functions of String class: “string strrr = UserInput.Replace(" ? ", " ? ");”

B. Url Injection Attack

URL Injection occurs when a hacker creates or injects new pages into an existing website. These pages often contain code that redirects users to other sites or implicates the business in attacks against other sites. These injections can be done through software vulnerabilities, insecure directories or plugins.

Query url is also a way of attack which is a well crafted attack url . If we have a web page with the url. For example:
if you get an URL like http://xyz.&.in/word /abc/abc.html

Then it means, we do not have any vulnerable points in the page. But if the URL is like http://xyz.s&.com/pro.php?u_id=’xxx’

then ‘u_id=xxx’ is a string type query for the url for can be altered by an attacker. The attacker can then enter his query in url which can give him access to the database, causing an attack. SELECT * FROM obj WHERE u_id=’xxx’.

‘u_id’ is parameter of this query and xxx is its value .It is fixed type of parameter but attacker can modify its value, which makes it vulnerable. For Example:http://localhost1/?E_Id=’xxx'; String E_id= “DROP TABLE EmpTable”

Another type of attack is when the attacker uses a UNION query and merges the special crafted query with the original query used by the user. http://localhost/?EmpId=‘ UNION .

This url will change the following SQL statement SELECT e_info FROM E_Table WHERE E_ID = " UNION.

c. Cross Site Scripting Attack(XSS)

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.

An attacker can use XSS to send a malicious script to an unsuspecting user. The end user’s browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.

Cross site scripting(XSS) is also a serious problem of web applications that can be used by an attacker .The attacker can insert the malicious script in a web application through any external resource.

The web browser executes the malicious code as a legitimate code.For example:

The hacker can modify the URL and execute the malicious code in URL box.
http://xyz1.com/index23.asp?search=

The attacker can add a modified statement to the URL and hijack the client to his domain.

1. {get Element sByTagName("formpage"[02].act io =}
2. ">><script>document.location='http://www.xyz.com/bin1/cookies.cgi_?' +document.cookies</script>"
3. varmsg = '<p style="color: red</p>'; msg.addInfoMessage(msg);

The attacker uses this type of script code for cookie theft with the stolen cookie and it helps in accessing the users account.

Server: A server is a program that uses HTTP to serve the files that form web pages to users, in response to their requests, which are forwarded by their system HTTP client. In this article client sends request to server and gets a response.Attacker tracks the session id of user by sending http request to server (eg: GET/user/profie_session_id="xyz") using several malicious code.After this request, server will respond to the user's session_id (eg: _session_id="xyz"). Finally, attacker will attaches malicious script into a database(commands) and gets response for the query accordingly

How to prevent SQL injection?

Many SQL injection attacks can be prevented by using parameterized queries instead of primitive strings. For example, the following code snippet is vulnerable to SQL injection because the user input is directly connected to the query.

; "" + String query = "SELECT * FROM products WHERE category = '" + input

;()Statement statement = connection.createStatement

;ResultSet resultSet = statement.executeQuery(query)

But you can easily remove the direct connection between the user input and the query structure by using the following code string.

;PreparedStatement statement = connection.prepareStatement("SELECT * FROM products WHERE category = ?")

; statement.setString(1, input)

;()ResultSet resultSet = statement.executeQuery

Parameterized queries can be used in any situation where unreliable input may appear as data and be used within queries. But it should be noted that they cannot be used to handle invalid entries in other parts of the query, such as tables and column names.

An application that operates by leaving invalid data in blind parts or queries requires a different approach (such as using new logic when encountering data or using a list for allowed input values). Also, pay attention to the fact that for parameterized queries to be effective, fixed hard-coded strings should be used and should never include variable data from different sources.

IV. PROTECTION AGAINST SQL INJECTION ATTACKS

An SQL injection attack is a web security vulnerability that allows a hacker to interfere with queries that an application makes to its database.It generally allows an attacker to view data that would normally be unrecoverable. This may include user data or any other data that the application has access to.In many cases, an attacker can modify or delete this data, causing permanent changes to the application's content or behavior.In some situations, an attacker can compromise the main server or other back-end infrastructure with a SQL injection attack or perform a denial-of-service attack.

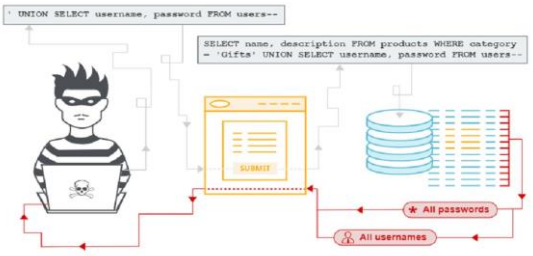

Fig. 3. sql-injection

A successful SQL Injection attack can lead to unauthorized access to sensitive data such as passwords, credit card information, or personal user information.

SQL injection types

- Hidden data retrieval, where you can modify an SQL query to return additional results.
- Disrupting the program logic, where you can modify a question to change the program logic.
- UNION attack, where you can retrieve data from different database tables.

- Check the database, where you can extract information about the version and structure of the database.
- Blind SQL injection, where your query results are not controlled in the application response.

Malicious attacks make web applications less secure because the intruder can harm the integrity of the database by applying malicious queries.

PreparedStatement obj; Pobj = con.prepareStatement("select * from std where userid = ?"); pobj.setString(01, ".?.");

The above-mentioned query is a simple way to curb application attacks. This can be formed by applying simple changes into the server site code. Binding variables is one more way for control SQL injection attacks and through binding variables we can improve web applications performance .The developer should use this type of variable in all SQL statements and also to Java language which provides better method called prepared statement .Prepared statement also uses bind variables.To defend against the SQL injection attacks, we should avoid passing the input directly into SQL queries. Instead users should use parameterized statements or sanitized input filtered carefully. In order to sanitize the provided user input, it should be bound to a parameter and input must be done through a filtering or sanitizing method .The main purpose of this method is that it adds a backslash('\') against all malicious code.

How to detect SQL injection vulnerabilities

Most SQL injection vulnerabilities can be found quickly and reliably using the Burp Suite web vulnerability scanner.

SQL Injection can be detected manually using a systematic set of tests against each application entry point. It typically includes the following:

- Post single quote characters and look for errors or other anomalies.
- Send some special SQL routines that evaluate to a base value from the entry point and to another value and look for systematic differences in application responses.
- Send boolean conditions like 1 = 1 and 1 = 2 and look for the difference in the program's response.
- Sending loads with the purpose of creating a delay when executing a query in SQL and looking for a difference in the time it takes to respond.
- Sending OAST payloads is designed to create out-of-band network interactions when executing an SQL query and monitor any resulting interactions.

How does Mimecast Web Security work?

Mimecast Web Security serves as a web security gateway, inspecting web requests and filtering these URLs using Mimecast's threat intelligence and security analytics and through an organization's acceptable use controls, security policies, and bypass exceptions to determine whether a website is safe and appropriate. Mimecast blocks access to unsafe or inappropriate sites and provides users with an explanation via a block page. Users are allowed to visit safe sites immediately, without any delay in response time.

What are Mimecast Web Security's biggest benefits?

- Block access to sites that contain malware, phishing and other threats.
- Prevent users from visiting certain categories of websites that are inappropriate for business use.
- Help to ensure that files downloaded from the web are free of threats and malware.
- Block compromised devices from communicating with attackers using the web.
- Protect data from exfiltration attacks.
- Improve understanding of employee use of the web.
- Simplify administration by managing web security from a single, cloud-based console and by consistently applying security policies throughout the organization.
- Combine email and web security in a single, easy-to-use solution.
- Deploy a web security solution quickly – in less than 60 minutes – and at a cost that is a fraction of alternative solutions.

V. PROTECTION AGAINST CROSS-SITE SCRIPTING ATTACKS

Hackers inject malicious code into a web form or web application link in a web script injection (XSS) cyber attack. This code, written in a scripting language such as JavaScript or PHP, is highly malicious to cause havoc on the website it is intended to use and can perform password or other application credentials.

Today's websites are usually created by running a bunch of code in the browser itself at load time. XSS programs also exploit this feature of websites. As a result, such attacks are not simple.

In an XSS attack, hackers take advantage of the interaction between the user and the website to execute malicious code on the user's system. For example, consider the following URL:

https://www.google.com/search?q=CSO+online

After writing this address in the address bar of the browser, you will see the search results for the term CSO Online in Google, as if you entered the term CSO Online in the browser search bar or the main page of the Google website. You'll also see this phrase appearing in various parts of the page, including the search bar at the top of the page.

Suppose that instead of the simple and harmless phrase CSO Online, the address you wrote contains a malicious JavaScript code such as:

https://www.google.com/search?<script>doEvil()</script>

Here doEvil() is hypothetically substituted for a malicious function. In this case, instead of displaying the term CSO Online on the page resulting from entering this address, Google should use the mentioned malicious codes in processing the page. In this way, this malicious script will be created in your browser and will bring unpleasant consequences. Of course, Google has implemented special security measures by hiring professional web security experts to prevent such attacks, but not all websites have such forces at their disposal. So sometimes these attacks are executed successfully.

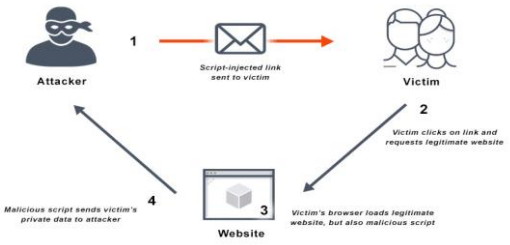

Fig. 4. Cross Site Scripting (XSS)

Nowadays cross-site scripting attacks occur because the developers add some vulnerability to the code. Every developer is responsible for attacks because developers should understand what kind of attacks are possible on web applications .Never trust user input because the user can insert any type of characters and always filter metacharacters as it reduces the XSS attacks. Developers should convert what's written between any two tages, which are enclosed in '<' and '>'. XSS holes can damage your application because the attackers will disclose these types of holes to the public and often everyone can see your personal information. Filtering does not provide a proper solution for cross site scripting attacks. But if developers use) and (, to " ; , ' to ' and convert # and & to #(#) and & (&).

VI. CAPTCHA

CAPTCHA stands for Completely Automated Public Turing Test To Tell Computers and Humans Apart and means "public automated test to distinguish humans from computers". Captcha was created in 2000 at Carnegie Mellon University by Luis von Ahn and his colleagues and was first used on the Yahoo site.

To prevent password hacking, to prevent brute force attacks, to distinguish between robots and humans, other than CAPTCHA is used on sites, CAPTCHA is used in various forms such as: mathematical problems, images, letters and numbers.

In Figure 5, you can see an example of CAPTCHA.

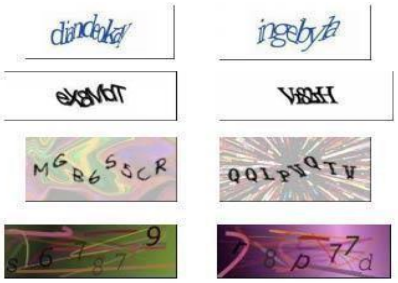

Fig. 5. CAPTCHA

VII. CONCLUSION

This research paper provides a complete survey of current research results under web application security. We have covered all properties of web application development, understood the important security functions and properties that secure web applications should use and divided existing works into three major classes .We also discuss a few issues that still need to be considered.

security on the Internet is improving. The increasing use of the Internet for commerce is improving the deployed technology to protect financial transactions. Extension of the basic technologies to protect multicast communications is

possible and can be expected to be deployed as multicast becomes more widespread.

Control over routing remains the basic tool for controlling access to streams. Implementing particular policies will be possible as multicast routing protocols improve. Cryptography is a tool which may alleviate many of the perceived problems of using the Internet for communications. However, cryptography requires the safe implementation of complex mathematical equations and protocols, and there are always worries about bad implementations. A further worry is that users are integral to securing communications, since they must provide appropriate keys.

To access a few out of the box features in web applications various programming concepts and tools are taking place that cause essential security aspects to our applications .Apart from this security researchers are applying required efforts to extend security features to web applications by several tools and techniques.

Generally, our logic and crucial codes reside at client side, that is our browser that exposes programmer concepts .Thus for attackers it becomes easy to intercept the logic and cause total damage to the server-side state of the application.

In this article, we examined the SQL injection attack from the code injection category, which is very popular among hackers. By looking at the basic concepts and types of this attack, we find out that the intruder is able to bypass the program with the simplest tricks and use the logic of the program against itself to be able to extract information from the database or carry out operations beyond its access level. . You will not be able to recover the damages caused by this attack afterwards, so the best way to avoid damages is to implement security policies and rules before the attack.

In many sites and blogs, the use of captcha has an effect on increasing spam comments and preventing hacking, but on the other hand, incorrect use of captcha and non-compliance with its principles can reduce users' communication with your site. Sometimes we are faced with questions and images on internet pages that will be very difficult and time-consuming to recognize even for humans. In this situation, many users do not show themselves and leave your site. Also, using other methods such as identifying spam comments and stored in memory can be a suitable alternative to captcha on low-visited sites.

39% of all WordPress security flaws are related to cross-site scripting issues. Companies spend millions of dollars fighting cross-site scripting attacks. To prevent XSS attacks targeting your website, it is important to know the nature of cross-site scripting and take preventive measures.

To protect your website from cross-site scripting, you need to secure your input fields. In addition, you can install security and dedicated anti-XSS plugins that will help you keep your WordPress website safe.

References

[1] Nunes P, Medeiros I, Fonseca JC, Neves N, Correia M, Vieira M. Benchmarking static analysis tools for web security. IEEE Transactions on Reliability. 2018 June 28;67(3):1159-75.

[2] Akhawe D, Barth A, Lam PE, Mitchell J, Song D. Towards a formal foundation of web security. In2010 23rd IEEE Computer Security Foundations Symposium 2010 Jul 17 (pp. 290-304). IEEE.

[3] Rubin AD, Geer DE. A survey of Web security. Computer. 1998 Sep;31(9):34-41.

[4] Friedman B, Hurley D, Howe DC, Felten E, Nissenbaum H. Users' conceptions of web security: a comparative study. InCHI'02 extended abstracts on Human factors in computing systems 2002 Apr 20 (pp. 746-747).

[5] Fonseca J, Seixas N, Vieira M, Madeira H. Analysis of field data on web security vulnerabilities. IEEE transactions on dependable and secure computing. 2013 Sep 6;11(2):89-100.

[6] Fonseca J, Vieira M, Madeira H. Evaluation of web security mechanisms using vulnerability & attack injection. IEEE Transactions on dependable and secure computing. 2013 Oct 11;11(5):440-53.

[7] Saini BS, Bala A. A review of bot protection using CAPTCHA for web security. IOSR Journal of Computer Engineering. 2013;8(6):36-42.

[8] Marashdih AW, Zaaba ZF, Omer HK. Web security: detection of cross site scripting in PHP web application using genetic algorithm. International journal of advanced computer science and applications. 2017;8(5).

[9] Bugliesi M, Calzavara S, Focardi R. Formal methods for web security. Journal of Logical and Algebraic Methods in Programming. 2017 Feb 1;87:110-26.

[10] Alsmadi I, Abu-Shanab E. E-government website security concerns and citizens' adoption. Electronic Government, an International Journal. 2016;12(3):243-55.

[11] Kumar S, Mahajan R, Kumar N, Khatri SK. A study on web application security and detecting security vulnerabilities. In2017 6th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO) 2017 Sep 20 (pp. 451-455). IEEE.

[12] A. Azfar, K. K. R. Choo, and L. Liu, “An android communication app forensic taxonomy,” Journal of Forensic Sciences, vol. 61, no. 5, pp. 1337–1350, 2016.

[13] A. Azfar, K. K. R. Choo, and L. Liu, “Forensic taxonomy of popular Android mHealth apps,” in Proceedings of the 21st Americas Conference on Information Systems (AMCIS ’15), San Juan, Puerto Rico, August 2015.

[14] YongJoonPark ,JaeChul Park , “Web Application Intrusion Detection System For Input Validation Attack” , Third 2008 International

Conference On Convergence And Hybrid Information Technology ,IEEE, PP 498-504.

[15] Stein LD. Web security. Addison-Wesley, Massachusetts. 1998;26:1-4.

[16] Zhou W, Jia Y, Peng A, Zhang Y, Liu P. The effect of iot new features on security and privacy: New threats, existing solutions, and challenges yet to be solved. IEEE Internet of things Journal. 2018 Jun 15;6(2):1606-16.

[17] Yang W, Zuo W, Cui B. Detecting malicious URLs via a keyword-based convolutional gated-recurrent-unit neural network. IEEE Access. 2019 Jan 29;7:29891-900.

[18] Kurt Alfred Kluever. Evaluating the Usability and Security of a Video CAPTCHA. Master's thesis, Rochester Institute of Technology, Rochester, NY, August 2008.

[19] Sandhya S, Purkayastha S, Joshua E, Deep A. Assessment of website security by penetration testing using Wireshark. In2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS) 2017 Jan 6 (pp. 1-4). IEEE.

[20] A. Barth, J. Caballero, and D. Song, "Secure content sniffing for web browsers, or how to stop papers from reviewing themselves," in SP '09: Proceedings of the 2009 30th IEEE Symposium on Security and Privacy. Washington, DC, USA: IEEE Computer Society, 2009, pp. 360–371.

[21] M.K. Gupta, M.C. Govil, G. Singh, Static Analysis Approaches to Detect SQL Injection and Cross Site Scripting Vulnerabilities in Web Applications: A Survey, IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014), 2014, pp. 1- 5

[22] J. Arlat, A. Costes, Y. Crouzet, J.-C. Laprie, and D. Powell, "Fault Injection and Dependability Evaluation of Fault-Tolerant Systems," IEEE Trans. Computers, vol. 42, no. 8, pp. 913-923, Aug. 1993. J. Arlat, A. Costes, Y. Crouzet, J.-C. Laprie, and D. Powell, "Fault Injection and Dependability Evaluation of Fault-Tolerant Systems," IEEE Trans. Computers, vol. 42, no. 8, pp. 913-923, Aug. 1993.

[23] Kiesling E, Eckelhart A, Kurniawan K, Ekaputra F. The SENSES knowledge graph: an integrated resource for cybersecurity. InInternational Semantic Web Conference 2019 Oct 26 (pp. 198-214). Springer, Cham.

[24] Crumpler W, Lewis JA. The cybersecurity workforce gap. Washington, DC, USA: Center for Strategic and International Studies (CSIS); 2019 Jan.

[25] Wang B, Dabbaghjamanesh M, Kavousi-Fard A, Mehraeen S. Cybersecurity enhancement of power trading within the networked microgrids based on blockchain and directed acyclic graph approach. IEEE Transactions on Industry Applications. 2019 May 29;55(6):7300-9.

9