**A research paper on various web application attacks, its various prevention and protection against vulnerabilities**

T Sai Teja1, Munagala Bhanu Prakash2, Rajesh Sharma3

Student, Faculty of Engineering Design and Automation, GNA University, Phagwara, India

teja3435j@gmail.com1

Student, Faculty of Engineering Design and Automation, GNA University, Phagwara, India

bhanu.prakaah.2002@gmail.com2

Associate Professor, Faculty of Engineering Design and Automation, GNA University, Phagwara, India

rajesh.sharma@gnauniversity.edu.in3

# **Abstract**

In recent years, a large number of firms have embraced the web as a cost-effective platform for disseminating and exchanging data with prospects and customers. Web applications have acquired control over the sections of arrangement atomization and upgrading. It provides numerous forms of help to the application's clients. Web programs are more vulnerable to attack as a result of the communication of secret information during administrations. Notwithstanding their benefits, web apps create a few security problems due to erroneous code. These flaws or loopholes enable thugs to have rapid and unfettered access to information bases in order to beat sensitive information. In this paper we had discussed about various web application attacks and its assault remedies. Another critical component we had explained is determining how to preserve our resources by tying down the applications that produce hackable code and which tools we had employ to be preventative and rely on a secure stage. We had further discussed on how to increase security in our organization and ensure that our obtained data is in accordance with the requirement

1. **INTRODUCTION**

**1.1 Web application**

A web application is a software that is kept on a distant server and delivered via the internet via a program interface. For various reasons, web applications are intended for different purposes and can be utilized by anybody, from an association to a person. A normally utilized application can incorporate webmail, a web-based number cruncher, or internet business shops. The web application can be open to clients through an internet browser, for example, Mozilla Firefox and google chrome. Working a web application requires a web server, server, and information base. As indicated by Tech Target Supporter of store, the required data information base is utilized. HTML5, JavaScript, and Flowing Templates (CSS) are the dialects utilized for client-side programming to construct front-end applications. Python, Java, and Ruby are the dialects utilized for server-side programming to work Back-end applications.[1]

**1.2 web application attacks**

Web applications are progressively turning out to be not difficult to trade data over the web. They give a connection point through which clients can convey. Specialist organizations utilize a data set that stores client-delicate information to deal with client data. Assailants exploit this data set in numerous ways to acquire clients' very own data. Web applications are available for aggressors to get to hidden data sets as they are frequently helpless against assaults. As OWASP, a code infusion assault is the most well-known and hazardous among the best ten web application weaknesses. It has been at the main situation for the beyond seven years, trailed by cross-site prearranging, broken verification, and meeting the board. Web application takes input from the client through textboxes as the name, address, remark criticism, and numerous alternate ways. These data sources are associated through the data set to store the client input values in the data set table. Aggressors endeavour different application-level assaults to think twice about security of web applications to commit misrepresentation or take delicate data.[2]

1. **OWASP Top 10**

OWASP is an international organization that specifies web applications' top 10 vulnerabilities and flaws.[3][4]

* Broken Access Control
* Cryptographic Failures
* Injection
* Insecure Design
* Security Misconfiguration
* Vulnerable and outdated components
* Identification and Authorization Failures
* Software and Data Integrity Failures
* Security logging and Monitoring Failures
* Server-Side Request Forgery (SSRF)

1. **METHODOLOGY ADOPTED**

Attackers perform attacks using a detailed process called the hacking methodology to gain knowledge of a particular web application to compromise it successfully.[5][6]

To collect information about various resources needed to run or access the web application, some of the methodologies are

* Foot printing web application
* Analyse web application
* Bypass client-side controls
* Attack authorization mechanism
* Attack authorization schemes

**3.1 Foot printing web application**

It accumulates total data about a system, its connected parts, and how they work. The infrastructure of a web application is the plan by which it associates different frameworks, servers, etc in the network. Web foundation foot printing is the most vital phase in web application hacking; it helps assaults distinguish and choose vulnerable web applications.

**3.2 Analyse web application**

Once attackers have attempted various possible attacks on vulnerable web servers, they may focus on the web application itself. To hack the web application, first, they need to analyze it to determine its weak areas. If they find any vulnerabilities, attackers try to compromise their security by launching an appropriate attack.

**3.3 Bypass client-side controls**

A web application requires client-side controls to restrict user inputs when transmitting data via client components and implement measures to control the user's interaction with their client. Web developers assume that the data transmitted from the client to the server is within the user's control, and their assumption can make the application vulnerable to various attacks.

**3.4 Attack Authentication Mechanism**

Web applications verify clients through a confirmation instrument, for example, login functionality. During web application investigation, attacker attempt to track down verification weaknesses like powerless passwords. Attackers exploit these weaknesses to get close enough to the web application through brute force attacks, dictionary assaults, cookie replay attacks, and so on.

**3.5 Access Authorization schemes**

the web application contains an authorization mechanism that restricts authenticated users' access to a specific functionality resource. Web applications always perform user authorization followed by authentication. The attacker implements the flawed authorization mechanism in the web application and takes advantage of it to access restricted pages by escalating privileges. The attacker attempts to get close enough to data without legitimate credentials. The attacker will use various attacks to attack the authorization schemes of the web application.

**3.6 Attack Access Control**

It is part of the application's security mechanism based logically on authentication and session management. The attacker walks through a website to identify.

* individual access to a particular subset of data
* Levels of grant access
* Administrator functionality to configure and monitor.

1. **RESULTS**

**4.1 SQL Injection**

SQL injection is a cyber-attack that focus on online applications that communicate with databases using SQL (Structured Query Language). The attack involves the insertion of SQL statements into input fields of a web application, which are then executed by the database server. SQL injection attacks are threat to web applications that use SQL, as they can result in serious consequences such as data breaches, financial loss, reputational damage, and legal liability. To prevent SQL injection attacks, developers can implement several best practices such as using parameterized queries, input validation, and sanitization techniques. It is also crucial for organizations to conduct regular security audits and vulnerability assessments to identify and remediate any vulnerabilities in their web applications. SQL injection attacks can involve exploring the different techniques used by attackers to inject malicious SQL statements into a web application, the impact of successful SQL injection attacks, preventive measures that can be taken by developers and organizations, and real-world case studies of SQL injection attacks. By understanding the nature of SQL injection attacks and implementing preventive measures, organizations can protect themselves from the potentially devastating consequences of a successful SQL injection attack.[7]

**4.2 Dos Attack**

A DoS attack is a cyber-attack that seeks to make a web application inaccessible to its intended users. The attacker does this by sending a greater number of requests to the target system which causing it to slow down, crash, or go down. Several tactics, such as flooding the target system with massive volumes of traffic or delivering faulty packets to the system, can be used to carry out the assault. DoS attacks can be launched from a single computer or from a botnet, which is a network of hacked machines. DoS attack is a sort of cyber-attack that seeks to make a website or service inaccessible to its intended users. The attacker does this by flooding the target system with traffic or requests, causing it to slow down, crash, or go down. Several tactics, such as flooding the target system with massive volumes of traffic or delivering faulty packets to the system, can be used to carry out the assault. DoS attacks can be launched from a single computer or from a botnet, which is a network of hacked machines. DoS attacks can have serious consequences for the target system and its users. As such, it is important for organizations to take steps to protect their systems from DoS attacks, such as implementing firewalls and intrusion detection systems, and regularly testing their systems for vulnerabilities. Some of the DoS assaults are as follows:

Ping Flood: This attack includes flooding a target system with ping queries, overloading it with traffic and rendering it unusable.

SYN Flood: The attacker sends a huge number of SYN requests to the target system, prompting it to dedicate resources for the incoming connection requests, resulting in a denial of service.

Smurf Attack: Sending ICMP (Internet Control Message Protocol) queries to a network's broadcast address causes all devices on the network to respond and overload the target machine.

Application Layer Attacks: These attacks target specific applications, such as web servers or email servers, by exploiting vulnerabilities in the application to overload the system or crash it.

Preventing DoS attacks can be challenging, but there are some strategies that organizations can adopt to reduce their risk of becoming a victim of a DoS attack. Implementing firewalls and intrusion detection systems, monitoring network traffic for odd activity, and frequently upgrading software and hardware to patch known vulnerabilities are examples of these. Additionally, organizations can work with their internet service providers to mitigate the impact of DoS attacks by utilizing traffic filtering and redirection techniques.[8][9]

**4.3 DDoS Attack**

A DDoS (Distributed Denial of Services) assault is a cyber-attack in which several sources of traffic or requests overload a target system. DDoS assaults may be carried out using a variety of strategies, including amplification and reflection attacks. In an amplification assault, the attacker sends a tiny quantity of traffic to a susceptible system, such as a DNS server, which responds with a substantially higher amount of traffic to the target system, overloading it. The attacker spoofs the originating IP address of a request to a susceptible system in a reflection attack, forcing the system to react with much more traffic to the target system. DDoS attacks can have serious consequences for the target system and its users. They can cause the system to become unavailable, leading to financial losses, reputation damage, and the loss of important data. Additionally, DDoS attacks can be difficult to defend against, as the traffic is coming from multiple sources and may be difficult to distinguish from legitimate traffic. Organizations can take steps to protect themselves from DDoS attacks, such as implementing network security measures, regularly testing their systems for vulnerabilities, and working with their internet service providers to mitigate the impact of attacks. This may include using traffic filtering and redirection techniques, or working with third-party vendors to provide DDoS protection services. DDoS attacks have become increasingly sophisticated in recent years, with attackers using techniques such as botnets composed of Internet of Things (IoT) devices, which can include cameras, routers, and other connected devices. These devices often have weak security protections and can be easily compromised by attackers to launch DDoS attacks. It can be challenging to detect and mitigate, but there are some strategies that organizations can adopt to reduce their risk of becoming a victim. These include implementing network security measures, monitoring network traffic for unusual activity, regularly updating software and hardware to patch known vulnerabilities, and utilizing DDoS protection services. By implementing robust security measures and staying vigilant for signs of an attack, organizations can reduce their risk of becoming a victim of a DDoS attack.[10]

**4.4 XSS**

Cross-Site Scripting (XSS) is a security flaw which allows an attacker to insert suspicious code into web page that other users are seeing. This sort of attack is particularly risky since it allows the attacker to obtain sensitive data such as login passwords, personal information, or financial information. An XSS attack often involves the attacker exploiting a vulnerability in a web application to inject malicious code, such as JavaScript, into a web page. When a victim views a hacked page, the malicious code is performed in their browser, letting the attacker to steal data or do other harmful operations on their behalf. XSS attacks are classified into two types:

Persistent XSS: It is an XSS in which the malicious code is stored on the web server and executed whenever a user visits the compromised page

Reflected XSS: It is an XSS in which the malicious code is injected into the URL and executed when the victim user clicks on a link containing the injected code. XSS attacks are classified into three categories: stored XSS, reflected XSS, and DOM-based XSS. Reflected XSS attacks do not require the malicious script to be stored in the server's database. Instead, the attacker sends a link containing the malicious script to the victim user, who then clicks on the link, executing the script in their browser. Both DOM-based XSS and reflected XSS are similar, but they target the Document Object Model (DOM) rather than the server's database. This type of attack involves manipulating the DOM to execute malicious code. XSS attacks can have serious consequences, such as data theft, identity theft, or the compromise of confidential information. They can also be used to spread malware or launch phishing attacks.[8][11]

1. **RECOMMENDATIONS**

**5.1 Broken Authentication and session management**

* Limit the login attempts and lock the account for a specific period after several failed attempts.
* Make sure to check weak passwords with cryptographic password hash algorithms.
* Log authentication failures and send alerts whenever probable attacks are detected.
* Confirm whether every one of the clients' characters and certifications put away in a hashed structure
* Try to really look at powerless passwords against a rundown of the top terrible passwords

**5.2 Sensitive Data Exposure**

* Try not to make or utilize frail cryptographic calculations
* Create encryption keys disconnected and store them safely
* Use proper key management and ensure all the keys are in place.
* Disable caching techniques for requests that contain sensitive information.
* Guarantee that encoded information put away on the disk isn't not difficult to decode

**5.3 XML External Entity**

* XML unmarshaller should configure securely
* Parse the document with a securely configured parser
* Update and patch the latest XML processors and libraries
* Make sure the XML file upload function validates the XML using XSD validation.

**5.4 Broken Access Control**

* Avoid using insecure IDs to prevent the attacker from guessing them
* Provide a session timeout mechanism
* Limit file permissions to authorized users to avoid misuse
* Avoid client-side caching mechanism
* Remove session tokens on the server-side on user logout

**5.5 Insecure Deserialization**

* Deserialization of trusted data must cross a trust boundary
* Developers must re-architect their application
* Avoid serialization for security-sensitive classes
* Guard sensitive data during deserialization
* Filter untrusted serial data.

**5.6 Using components with known vulnerabilities**

* Apply security patches regularly
* Scan the parts with a security scanner frequently
* Enforce security policies and best practices for details use
* Make sure to obtain details from official sources and accept only signed packages.

1. **CONCLUSION**

In this, we performed an SQL injection attack, DoS attack, DDoS attack, and XSS (Cross-Site Scripting) attack. Web applications are more at now than before, so executives must take the proper steps to secure their web applications. Leveraging technology and innovative methodologies can go a long way toward minimizing the effects of even the most dangerous cyber-attacks. We also discussed the countermeasures regarding web application attacks like Broken conformation, validation, sensitive information exposure, XML External Entity, and Broken Access Control.

1. **REFERENCES**

[1] T. G. Author, “Common Web application attacks,” 2020. https://www.tripwire.com/state-of-security/most-common-website-security-attacks-and-how-to-protect-yourself

[2] Williams, “Malicious web application attacks,” 2022. https://itbrief.com.au/story/malicious-web-application-attacks-climb-88-report#:~:text=The report findings

[3] O. Project, “OWASP.” https://owasp.org/www-project-top-ten/#

[4] O. Moradov, “Web Application Vulnerabilities,” 2013. https://brightsec.com/blog/web-application-vulnerabilities/#:~:text=Common Types of Web Application Vulnerabilities,-1.&text=SQL vulnerabilities allow attackers to,by web applications or services.

[5] E. of Encyclopedia, “Britannica,” 2023. https://www.britannica.com/technology/operating-system

[6] Acunetix, “Acunetix,” *Invicti*. https://www.acunetix.com/websitesecurity/web-application-attack/

[7] P. Swigger, “PortSwigger.” https://portswigger.net/web-security/sql-injection/union-attacks

[8] S. Authors, “Synopsys,” 2022. https://www.synopsys.com/glossary/what-is-cross-site-scripting.html

[9] K. Thompson, “TripWire,” 2023. https://www.tripwire.com/state-of-security/most-common-website-security-attacks-and-how-to-protect-yourself

[10] Comptia, “Comptia,” 2023. https://www.comptia.org/content/guides/what-is-a-ddos-attack-how-it-works

[11] T. T. Contributor, “TripWire,” 2019.