# HACKER PREVENTION

# **#1 - SQL INJECTION**

|  |  |
| --- | --- |
| **Security Concern #** | **SQL Injection** |
| Contributing Team Members: | Lyea Paul, Jinil Parekh, Navaneeth Krishnan, Pavan Mistry |

## DEFINITION:

SQL injection (SQLi) is a web security vulnerability that allows an attacker to interfere with the queries that an application makes to its database allowing an attacker to view data that they are not normally able to retrieve. This might include data that belongs to other users, or any other data that the application can access. In many cases, an attacker can modify or delete this data, causing persistent changes to the application's content or behavior.

In some situations, an attacker can escalate a SQL injection attack to compromise the underlying server or other back-end infrastructure. It can also enable them to perform denial-of-service attacks

## OBJECTIVE OF ATTACK:

A successful SQL injection attack can result in unauthorized access to sensitive data, such as:

* Passwords.
* Credit card details.
* Personal user information.

SQL injection attacks have been used in many high-profile data breaches over the years. These have caused reputational damage and regulatory fines. In some cases, an attacker can obtain a persistent backdoor into an organization's systems, leading to a long-term compromise that can go unnoticed for an extended period

There are lots of SQL injection vulnerabilities, attacks, and techniques, that occur in different situations which include

* Retrieving hidden data, where you can modify a SQL query to return additional results.
* Subverting application logic, where you can change a query to interfere with the application's logic.
* UNION attacks, where you can retrieve data from different database tables.
* Blind SQL injection, where the results of a query you control are not returned in the application's responses.

(<https://portswigger.net/web-security/sql-injection#sql-injection-examples>)

## NOTABLE CASES/VICTIMS:

The GhostShell gang on Saturday posted online what it claims are accounts and records from various financial services, consulting firms, academia, law enforcement, and the CIA. "Team GhostShell final form of protest this summer against the banks, politicians and for all the fallen hackers this year," the post said in part. "One million accounts/records leaked. We are also letting everyone know that more releases, collaborations with Anonymous and others, plus two more projects are still scheduled for this fall and winter. It's only the beginning."

Researchers at Imperva say the attackers appear to have employed mostly SQL injection, but also exploited weak passwords and vulnerable content management systems. The attackers used the popular SQLmap tool, and some of the hacked databases included more than 30,000 records.  
(<https://www.darkreading.com/cyberattacks-data-breaches/ghostshell-haunts-websites-with-sql-injection>)

## EXAMPLES:

There are many ways to hack a database using SQL Injection.

1. **Line comments**

Line comment sql injection works by putting a line comment at the end to comment out the rest of the query. Line comments are typically used to ignore the rest of the original query so you don’t need to worry about valid syntax.

Example: logging in as admin:Injection into the username parameter: admin'--

SELECT \* FROM members WHERE username = '**admin'--**' AND password = 'password'   
If successful, this will log you as the admin user because the rest of the SQL query after -- will be ignored.

1. **Inline comments**

You can use inline comments to comment out the rest of a query as with line comments (by simply not closing the comment). They are also useful for manipulating characters to bypass filtering/blacklisting, remove spaces, and obfuscate queries. In MySQL, you can use its special comment syntax to detect the database and version.

Example: Comment SQL Injection Attack Samples

ID value: 10; DROP TABLE members /\*

Simply get rid of other stuff at the end of the query. Same as 10; DROP TABLE member--

SELECT /\*!32302 1/0, \*/ 1 FROM tablename

Will throw an division by 0 error if MySQL version is higher than 3.23.02

1. **Stacking Queries**

Stacking means executing more than one query in one transaction. This technique can be very useful but only works for some combinations of database server and access method:

Example: SELECT \* FROM members; DROP members--

When successful, this will end one query and start another one.

Stacked SQL Injection Attack Samples

ID value: 10;DROP members --

SELECT \* FROM products WHERE id = 10; DROP members--

This will run DROP members SQL sentences after normal SQL Query.

1. **If Statements**

Get a response based on an IF statement. This is one of the key techniques for Blind SQL Injection. Also very useful to test simpler things blindly yet accurately.

Example: if ((select user) = 'sa' OR (select user) = 'dbo') select 1 else select 1/0 (S)

This will throw a divide by zero error if the user currently logged in is not sa or dbo.

1. **Bypassing login screens**

This is SQL injection 101—here are some typical login tricks that you can use with form fields and parameters:

Example: SELECT \* FROM users WHERE username = '**'' or ''='**' AND password = '**'' or ''='**'

admin' --

admin' #

admin'/\*

' or 1=1--

' or 1=1#

' or 1=1/\*

') or '1'='1--

') or ('1'='1--

Another trick is to log in as a different user:

' UNION SELECT 1, 'anotheruser', 'doesn't matter', 1–

([source:https://www.invicti.com/blog/web-security/sql-injection-cheat-sheet/#LineCommentAttacks](https://www.invicti.com/blog/web-security/sql-injection-cheat-sheet/#LineCommentAttacks))

## PREVENTION:

The only sure way to prevent SQL Injection attacks is input validation and parametrized queries including prepared statements. The application code should never use the input directly. The developer must sanitize all input, not only web form inputs such as login forms. They must remove potential malicious code elements such as single quotes.

Preventing SQL Injection vulnerabilities is not easy. Specific prevention techniques depend on the subtype of SQLi vulnerability, on the SQL database engine, and on the programming language. However, there are certain general strategic principles that you should follow to keep your web application safe.

**Step 1: Train and maintain awareness**

To keep your web application safe, everyone involved in building the web application must be aware of the risks associated with SQL Injections. You should provide suitable security training to all your developers, QA staff, DevOps, and SysAdmins. You can start by referring them to this page.

**Step 2: Don’t trust any user input**

Treat all user input as untrusted. Any user input that is used in an SQL query introduces a risk of an SQL Injection. Treat input from authenticated and/or internal users the same way that you treat public input.

**Step 3: Use whitelists, not blacklists**

Don’t filter user input based on blacklists. A clever attacker will almost always find a way to circumvent your blacklist. If possible, verify and filter user input using strict whitelists only.

**Step 4: Adopt the latest technologies**

Older web development technologies don’t have SQLi protection. Use the latest version of the development environment and language and the latest technologies associated with that environment/language. For example, PHP uses PDO instead of MySQLi.

**Step 5: Employ verified mechanisms**

Don’t try to build SQLi protection from scratch. Most modern development technologies can offer you mechanisms to protect against SQLi. Use such mechanisms instead of trying to reinvent the wheel. For example, use parameterized queries or stored procedures.

([source: https://www.acunetix.com/websitesecurity/sql-injection/](https://www.acunetix.com/websitesecurity/sql-injection/))

# **#2 - CROSS-SITE SCRIPTING (XSS)**

|  |  |
| --- | --- |
| **Contributing Team Members:** | Anish Patel, Vaibhav Baria, Chi-Wei Yin, Ozan Lago-Sereflioglu |

## Definition of Cross-site Scripting (XSS)

Cross-site Scripting (XSS) is a type of security vulnerability typically found in web applications. XSS allows attackers to inject malicious scripts into web pages viewed by other users. The injected code is executed in the context of the user's browser, leading to potential data theft, session hijacking, defacement of websites, and other malicious activities.

## Potential Targets or Objectives of the Hacker

XSS can be a tool for stealing valuable user data from visitors. This could include login credentials, credit card information, or other sensitive details.

Hackers might use it to inject malicious scripts that automatically download malware onto any visitor's computer, essentially creating a botnet for further attacks.They could redirect users to phishing scams designed to steal login credentials.

Beyond direct theft, attackers might use it to disrupt a website's functionality. This could involve defacing the website with malicious content or launching Denial-of-Service (DoS) attacks that overwhelm the site with traffic, making it inaccessible to users.

## Notable Cases or Victims

One of the notable case of an XSS vulnerability were recent attacks targeting outdated versions of the WordPress plugin Popup Builder (versions prior to 4.2.3).

In this incident, attackers leveraged a known vulnerability (CVE-2023-6000) to compromise websites. This vulnerability allowed them to inject malicious scripts that did not originate from the Popup Builder plugin itself. These injected scripts could then manipulate website visitors in various ways.

These scripts could redirect users to fake login pages designed to steal login credentials. Scripts could download and install malware onto visitors' computers.

Also, it could steal sensitive user information, such as credit card details or login credentials.

The impact of this attack was significant. Security researchers identified over 1,170 infected websites, with estimates suggesting a total of over 3,300 compromised sites.

## Examples / Demonstrations

A screenshot of a computer

Description automatically generated

* The attacker injects a payload into the website’s database by submitting a vulnerable form with malicious JavaScript content.
* The victim requests the web page from the web server.
* The web server serves the victim’s browser the page with attacker’s payload as part of the HTML body.
* The victim’s browser executes the malicious script contained in the HTML body. In this case, it sends the victim’s cookie to the attacker’s server.
* The attacker now simply needs to extract the victim’s cookie when the HTTP request arrives at the server.
* The attacker can now use the victim’s stolen cookie for impersonation.

## Methods to Protect Your Code and Users

* **Input Validation**: Ensure that user inputs are validated and sanitized before processing. Only allow expected formats.
* **Output Encoding**: Encode data before displaying it to the user. Use functions like htmlspecialchars() in PHP or htmlEncode() in other languages to prevent the browser from interpreting input as code.
* **Content Security Policy (CSP)**: Implement a CSP to restrict the sources from which scripts can be executed.
* **HTTP-only Cookies**: Use HTTP-only flags for cookies to prevent access via JavaScript.
* **Regular Security Audits**: Perform regular security reviews and code audits to identify and fix vulnerabilities.
* **Frameworks and Libraries**: Utilize secure frameworks and libraries that are less prone to XSS attacks.

## Reference

<https://www.acunetix.com/websitesecurity/cross-site-scripting/>

<https://medium.com/@galiniostech/sql-injection-vs-cross-site-scripting-xss-unraveling-two-critical-web-vulnerabilities-b9be62c25235>

<https://en-ca.wordpress.org/plugins/popup-builder/>

<https://codex.wordpress.org/images/2/2a/WP3.9.4-ERD.png>

<https://www.bleepingcomputer.com/news/security/hackers-exploit-wordpress-plugin-flaw-to-infect-3-300-sites-with-malware/>

# **#3 – RANSOMWARE**

Contributing Team Members: Mick, Irina, Rosalia

## DEFINITION

According to the Canadian Centre for Cyber Security, ransomware is a type of malware

that denies a user’s access to files or systems until a sum of money is paid.

Ransomware can infect your network, and spread to all connected devices.

Usually, you can unknowingly download ransomware onto a computer by opening an

email attachment, clicking an ad, following a link, or even visiting a website that's

embedded with malware.

## OBJECTIVE OF ATTACK

From the definition, we can define two main purposes of ransomware:

- Affect basic computer functions (Locker ransomware)

- Encrypt individual files (Crypto ransomware)

In addition to the fact that the owner loses access to the data, the attackers may

threaten to delete or leak this data unless ransom is paid.

The main objective of the huckers’ attacks is profit. After the WannaCry outbreak in

2017, which is one of the largest ransomware attacks with over 200,000 attacked

computers in 150 countries and estimated costs of USD 4 billion, “71% of companies

have encountered ransomware attacks, resulting in an average financial loss of $4.35

million per incident.”

“In the year 2023 alone, attempted ransomware attacks have targeted 10% of

organizations globally. This marks a notable rise from the 7% of organizations facing

similar threats in the previous year, representing the highest rate recorded in recent

years.”

However, ransom payments are only part of the total cost of a ransomware infection.

According to the IBM Cost of a Data Breach report, the average cost of a ransomware

breach is $5.13 million, which does not include ransom payments.

The average cost exceeds $1.8 million per incident, and costs are expected to balloon

to $265 billion by 2031 because, nowadays, ransomware has become a business.

Affiliates can profit from extortion without having to develop their own malware, and

developers can increase their profits without launching more cyberattacks.

## NOTABLE CASES/VICTIMS

Many cases of ransomware have been notable throughout history. Ever since the first

known case in 1989 on the World Health Organization's AIDS, many other cases have

been made to different public and private companies, governments and even hospitals.

On May 7th, 2021, Colonial Pipeline, an American oil pipeline system, was hit by a

ransomware attack. This forced the company to shut down operations and 2 days later,

a state of emergency was declared in 17 states in the USA. The attackers or attacker,

identified as DarkSide, accessed the systems through a compromised credential for a

legacy VPN and demanded 75 Bitcoins as payment for the ransom. Around that time

this was worth $4.4 million.

Unfortunately, there was no way to break from this attack, and the company was forced

to pay the amount requested, under the supervision of the FBI. The impact of this attack

had lasting consequences on the country.

Another case that’s very notable is the attack that was enacted towards the Costa Rican

government on April 17, 2022. This attack was directly targeted towards the government

institutions, affecting the Ministry of Finance, the Ministry of Science, Innovation,

Technology, and Telecommunication, and the state internet provider RACSA. This

forced the government to shut down operations on many of their systems, significantly

slowing down services and limiting what could be done. The former president, Carlos

Alvarado, refused to pay the ransom of $10 million. This led to the perpetrators,

identified as the hacker group Conti, to leak almost all the 672 GB of data it stole. At the

end of this situation, the government ended up losing around $30 million, and to this day

the consequences of said attack are still felt in Costa Rica.

## EXAMPLES

Ransomware attacks have become increasingly prevalent, posing significant threats to

individuals, businesses, and governments worldwide. Among the numerous

ransomware strains, three of the most notorious ones are the AIDS Trojan, WannaCry,

and CryptoLocker. These ransomware attacks have caused substantial damage and

have been widely studied due to their impact and unique characteristics.

1. AIDS Trojan

One of the earliest examples of ransomware, the AIDS Trojan, was created by Dr.

Joseph Popp. This malicious software was disseminated via infected floppy disks

labeled as "AIDS Information Introductory Diskette." Upon installation, the Trojan

modified the AUTOEXEC.BAT file on the victims' computers. After 90 reboots, it

concealed directories and encrypted file names, rendering the system unusable and

demanding a $189 payment for a "license renewal." The decryption key was embedded

within the code, yet the victims were unaware of this. Dr. Popp later claimed he intended

to donate the ransom proceeds to AIDS research, but he was subsequently declared

mentally unfit for trial

2. WannaCry Ransomware

WannaCry emerged on May 12, 2017, as an encrypting ransomware worm that rapidly

became a global crisis. This ransomware exploited the EternalBlue vulnerability in

outdated Microsoft Windows systems, enabling it to spread autonomously. The ransom

demanded ranged from $300 to $600 in Bitcoin for the decryption of affected files.

Despite prompt patching efforts and the discovery of a kill switch, WannaCry managed

to infect approximately 200,000 computers across 150 countries. The resultant

damages were estimated to be in the range of hundreds of millions to billions of dollars.

A significant factor contributing to its widespread impact was the inadequate patching

practices prevalent at the time. Multiple countries attributed the attack to North Korea.

3. CryptoLocker Ransomware

CryptoLocker, active from September 2013 to May 2014, targeted Microsoft Windows

systems, spreading primarily through email attachments and the Gameover ZeuS

botnet. Once infected, files were encrypted using RSA public-key cryptography, with the

attackers demanding ransom payments in Bitcoin or prepaid cash vouchers for

decryption. Removing the malware did not restore access to the encrypted files, as the

decryption key was necessary. The international effort known as Operation Tovar

eventually dismantled the botnet, recovering the private keys and enabling victims to

decrypt their files without payment. Despite this, CryptoLocker successfully extorted

approximately $3 million from its victims during its operational period.

Example Scenario: What A Real Ransomware Attack Looks Like - YouTube (video)

A retiree victim with limited tech knowledge, received an email claiming he needed to

download an attachment to verify a suspicious bank transaction. Unaware of the risks,

he clicked the link in his email's spam folder and downloaded the file. Shortly after

opening it, a message popped up on his screen stating that all his files were encrypted.

It demanded a payment in cryptocurrency to unlock them. Panicked and confused, The

unaware user realized he had fallen victim to a ransomware attack.

## PREVENTION

These are the best practices to be aware of in order to best prevent ransomware

attacks at any level, whether they occur personally or inside a business:

● Backup your data. From time to time, backing up any important information to an

external drive or a cloud service can help a lot to avoid the loss of data in the

case of a ransomware attack.

● Keep all systems and softwares up to date. Avoid using outdated softwares, as

this can be an easy entry to systems and networks for ransomware malware.

● Use/Install antivirus softwares and firewalls.

● Whitelist applications that you will use in your system and/or network.

● Limit user access privileges. By limiting these, you can control the access of

data.

● Protect your email; pay attention when opening emails or attachments. This

method is the most common way to access data, by sending files or links that

hide the malware in it.

● Segment your network. This will prevent the attack from spreading too far into the

network and potentially isolate the attack to a small section of the network.

● Regularly run security testing. This will help assess any vulnerability in the

system and create new security protocols.

● Prioritize endpoint security: antivirus, data encryption and data loss prevention,

real-time security alerts, web browser security.

● Implement security awareness training, for both users and employees. Teaching

others how to safely browse websites, recognize suspicious emails and create

secure passwords, it’s a good start in preventing these types of attacks.

## Sources & Additional Links:

● Canadian Centre for Cyber Security

● CheckPoint: Ransomware Attack – What is it and How Does it Work?

● CheckPoint: The CISO’s Guide to Ransomware Prevention

● IBM: What is ransomware? + IBM Definitive Guide to Ransomware

● IBM Cost of a Data Breach report

● TechTarget: The 10 biggest ransomware attacks in history

● GetAstra: 10 of the Biggest Ransomware Attacks in History

● Wikipedia: Colonial Pipeline ransomware attack

● What A Real Ransomware Attack Looks Like - YouTube

● UpGuard: How to Prevent Ransomware Attacks: Top 10 Best Practices

● GetAstra: What can You Do to Avoid Ransomware Attacks?

# **#4 - URL MANIPULATIONS ATTACKS**

|  |  |
| --- | --- |
| Contributing Team Members: | Bhavya Patel, Kwasi Apenteng, Maria Alejandra Espinoza Huamaní, Pranjal Lokhande |

## DEFINITION

URL manipulation is also known as HTTP manipulation. It is a set of attacks focusing on attempts to get access to unauthorized information based on the manipulation of the URL. In simple terms, A URL manipulation attack is when someone edits the URL text in the browser's location bar, to probe a website. URL manipulation attacks are part of a larger group of application-oriented attacks, including cross-site scripting attacks and SQL injection attacks. The Internet has turned into the demand of current associations. The Internet architecture is focused on performance and not security. Some users leave their systems vulnerable to compromise by using easy and general passwords, leaving design features in default mode, switching off firewalls etc. All these weaknesses make it easy to access the root information by the attacker.

URL Manipulation attacks are very frequent in the cyberspace world nowadays. The increasing use of URL Manipulation attacks has put computer and network services at greater risk than ever before. It is one of the easiest attacks to perform. It can be performed by users who are just curious or by hackers who want access to the data with wrong intentions. URL Manipulation attacks do not depend on specific rules. Therefore, the stopping of URL Manipulation attacks is a challenging issue. With an estimated 2.4 billion internet users worldwide, the risk of misuse and abuse of website content involving URLs has increased almost 700 per cent since 2000 (Internet World Stats, 2012). So, making a differentiation between the fake URL and the genuine URL becomes the main responsibility of everyone.

## OBJECTIVE OF ATTACK:

They are mostly financially motivated and usually their objective is to steal information.

Attacks usually target the private sector such as:

1. People in general, but especially the ones who can steal money
2. Websites with low security

Motives:

They steal confidential data such as personal information from personal accounts, financial details, and sensitive data. The latter can also lead to identity theft. They use URL manipulation to examine directories and file extensions, read sensitive information, and no-authorized sections of a website. Disrupting the code from the website can also lead to malfunction and exposing hidden content.

## NOTABLE CASES/VICTIMS:

In May 2019, the First American Financial Corp was found to be vulnerable to URL manipulation attacks. The vulnerability was due to the website having a Insecure Direct Object Reference. The website failed to verify whether users had authorization to view documents. Due to the lack of verification, if users had the URL for a specific document, they could freely view it. In addition, the document keys were sequential, meaning that, by changing the reference number in the URL you were able to view other customer’s documents. Fortunately, Ben Shoval, a real estate developer who discovered the leak, notified the security investigator KrebsOnSecurity. KrebsOnSecurity verified that approximately 855 million files were easily accessible, including documents containing banking information, driver's licenses, and social security numbers. As far as we know the vulnerable data was not exploited. First American Financial Corp was fined around $500 000 by the Securities and Exchange Commission.

## EXAMPLES:

URLs are easily edited and often follow a pattern. This makes them inviting targets for manipulation.   
Manipulations can include:

* Adding or changing query parameters: Attackers modify URL query parameters to inject malicious input or access unauthorized data.

"<http://example.com/search?query=apple>"  
 “query=apple", "query=admin”

* Searching for new web pages: By manipulating URLs, attackers can discover hidden web pages or directories not intended for public access.

http://example.com/admin  
 <http://example.com/login>

http://example.com/backup

* Enumerating values in database: Attackers change URL parameters to sequentially access or enumerate values in a database, potentially revealing sensitive information.

<http://example.com/user?id=1>  
 <http://example.com/user?id=2>  
 http://example.com/user?id=345

* Escalating access privileges: Manipulating URLs can sometimes allow attackers to gain higher access levels or privileges on a website.

<http://example.com/profile?user=guest>  
http://example.com/profile?user=admin

* Detecting installed software: Attackers can manipulate URLs to probe for specific server responses that indicate the presence of particular software or versions.  
   <http://example.com/phpmyadmin>

http://example.com/cgi-bin

* Accessing the server’s file system: URL manipulation can be used to attempt directory traversal attacks, where attackers try to access restricted files on the server.

<http://example.com/view?file=index.html>  
 http://example.com/view?file=../../etc/passwd

* Trying to use values for remote code execution: By crafting specific URLs, attackers can exploit vulnerabilities to execute arbitrary code on the server remotely.

<http://example.com/search?query=apple>

http://example.com/search?query=system('ls');

URL attack techniques may include:

* **Trial and error**
  + Search for directories making it possible to administer the site:

http://target/admin/

<http://target/admin.cgi>

* + Search for a script to reveal information about the remote system:

<http://target/phpinfo.php3>

* + Search for backup copies. The .bak extension is generally used and is not interpreted by servers by default, which can cause a script to be displayed:

<http://target/.bak>

* + Search for hidden files in the remote system. On UNIX systems, when the site's root directory corresponds to a user's directory, the files created by the system may be accessible via the web:

http://target/.bash\_history

http://target/.htaccess

* **Directory transversal**
  + The user may be forced to gradually move back through the tree structure, particularly in the event that the resource is not accessible, for example:

<http://target/base/test/ascii.php3>

<http://target/base/test/>

http://target/base/

* + On vulnerable servers, attackers can simply move back through the path with several "../" type strings:

http://target/../../../../directory/file

## PREVENTION:

* Never consider a URL to be private
* Not linking to a URL does not mean it will not be found
* Use access controls to restrict access, not URL visibility
* Every URL should enforce proper limits
* Consider edge cases and expect unexpected
* Every URL should have robust error handling
* Configure web server to gracefully handle errors and unfound URLs
* Prevent the browsing of pages located below the website's root (chrootmechanism);
* Disable the display of files present in a directory that does not contain an index file
* ("Directory Browsing");
* Delete useless directories and files (including hidden files);
* Make sure the server protects access to directories containing sensitive data;
* Delete unnecessary configuration options;
* Make sure the server accurately interprets dynamic pages, including backup files
* (.bak);
* Delete unnecessary script interpreters;
* Prevent HTTP viewing of HTTPS accessible pages.

## References

Chin, K. (2022, August 5). *Biggest Data Breaches in US History*. UpGuard. <https://www.upguard.com/blog/biggest-data-breaches-us>

Desk, I. (2023, November 1). *URL Manipulation: Construction, Attack Methods Countermeasures*. ITsecurity Demand. <https://www.itsecuritydemand.com/insights/security/url-manipulation-construction-attack-methods-countermeasures/#:~:text=Data%20Theft%3A%20Hackers%20use%20URL>

EasyDmarc. (2022, June 26). *What is URL Manipulation or URL Rewriting?* EasyDMARC. <https://easydmarc.com/blog/what-is-url-manipulation-or-url-rewriting/>

*First American Financial Corp. Leaked Hundreds of Millions of Title Insurance Records — Krebs on Security*. (2018). Krebsonsecurity.com. <https://krebsonsecurity.com/2019/05/first-american-financial-corp-leaked-hundreds-of-millions-of-title-insurance-records/>

Sharma, P., & Nagpal, B. (2015). A STUDY ON URL MANIPULATION ATTACK METHODS AND THEIR COUNTERMEASURES. *International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)*, *15*, 976–1353. <https://www.ijetcse.com/admin/uploads/A%20STUDY%20ON%20URL%20MANIPULATION%20ATTACK%20METHODS%20AND%20THEIR%20COUNTERMEASURES_1605600760.pdf>

Shepherd, A., & Afifi-Sabet, K. (n.d.). *How do hackers choose their targets?* IT PRO. Retrieved June 6, 2024, from <https://www.itpro.com/security/hacking/357971/how-do-hackers-choose-their-targets>

Sundar, V. (2023, March 28). *The 27 Most Notorious Hacks History | Indusface Blog*. Indusface. <https://www.indusface.com/blog/notorious-hacks-history/>

*URL Manipulation | CodePath Cliffnotes*. (n.d.). Guides.codepath.com. Retrieved June 6, 2024, from https://guides.codepath.com/websecurity/URL-Manipulation

*URL manipulation attacks Introduction to URLs*. (n.d.). <https://www.idc-online.com/technical_references/pdfs/data_communications/URL_manipulation_attacks.pdf>

# **#5 - PHISHING/SPOOFING**

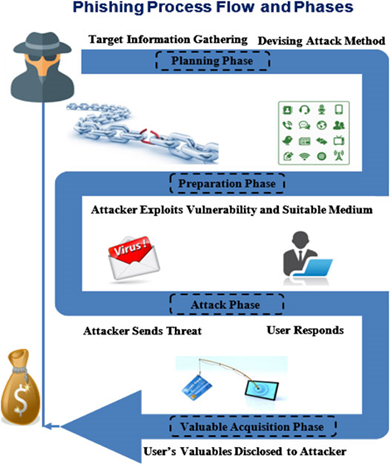
|  |  |
| --- | --- |
| Contributing Team Members: | Disha Kolapate, Joel Tuffour, Rutisha Patel, Saran Kanaga Sabapathy |

## **DEFINITION**

Phishing is a cyber attack where attackers impersonate legitimate entities (such as authorized users, banks, companies, or government agencies) to deceive users into revealing sensitive information (like passwords, credit card details, or personal data).

Spoofing involves disguising communication to appear as if it comes from a trusted source. It can apply to email addresses, websites, or caller IDs.

Notably, a phishing attempt may begin with a spoofing attack.



*Citations: https://www.frontiersin.org/files/Articles/563060/fcomp-03-563060-HTML-r1/image\_m/fcomp-03-563060-g001.jpg*

## **OBJECTIVE OF ATTACK:**

Phishers target everyone, not just tech companies. We all have gotten Spam/Fraudulent Emails. Phishers do everything to gain your trust and exploit your sensitive data.

Some of the objectives of the attack are.

· Identity Theft: The main objective is to trick a victim into disclosing sensitive information like their Social Security Number or Bank Information.

· Deployment of Malware: The motives behind malware vary. Malware can be about making money off you, sabotaging your ability to get work done, making a political statement, or just bragging rights.

· Data Harvesting and Selling: Phishers, after successfully executing one or more phishing attacks, begin to collect and sell user data such as login credentials, financial information, and personal identification. This data is then sold on the dark web or used to fuel future cybercrimes.

· Disruption of Services: Phishers do this as a protest or to sow chaos. Phishers access critical systems, then alter, delete, or ransom data, crippling normal operations. This will usually lead to halted services, creating a domino effect of financial losses and customer dissatisfaction.

Citations:

Target of phishing - <https://trustifi.com/blog/who-is-the-target-of-phishing-scams/>

Objectives Of Phishing Attack - <https://www.guidepointsecurity.com/education-center/phishing-attack/>

## **NOTABLE CASES/VICTIMS:**

**CASE 1:**

In January 2016, an employee of FACC, an Austrian manufacturer of aircraft parts, received an email asking the organization to transfer €42 million to another account as part of an “acquisition project".The message appeared to be from the CEO of the organization, Walter Stephan, but was a hoax. Unaware of the true nature of the email, the employee agreed to the request. Few details have been released about what went wrong, but there were reasons to believe Stephan was at least partially to blame. This is because the FACC reported him after an internal investigation, claiming he had "seriously breached his duties". It also fired its chief financial officer.FACC demanded 10 million euros in legal compensation from the directors, but Austrian courts rejected the claim.

**CASE 2:**

In November 2014, the hacking group ‘Guardians of Peace’ leaked 100 terabytes of data from Sony Pictures. Stuart McClure, CEO of Cylance, revealed that the hackers had sent fake Apple emails to Sony executive’s months earlier, capturing their login details. They accessed sensitive data, including employee information, private emails, and unreleased films, and used a malware “Shamoon” to erase Sony's systems. The hackers, linked to North Korea, demanded Sony cancel "The Interview," a comedy about assassinating Kim Jong-un, and threatened attacks on theaters showing the film. The attack cost Sony over $100 million.

## **EXAMPLES:**

**Email phishing**

Most common form of phishing.Scammers can quickly obtain thousands of email addresses by harvesting or from email addresses stolen in data breach incidents. Phishing emails contain links to a malicious website or an infected attachment.

Example of an AppleID phishing email

* Email phishing can lead to malware infection via an infected attachment in the email. If the recipient clicks on the attachment, the malware will look for vulnerabilities in software on the user's device and exploit these flaws to install the malicious software.
* Email phishing can lead to stolen login credentials via an associated spoof website, a user taken to this site if they click on a link in the email or an attachment.
* Phishing emails are increasingly challenging to detect as they are designed to evade end user detection. For example, infected attachments such as Word and Excel documents are now less common, and instead, fake image files (.jpeg and .png) are increasingly used to bring malware into people's inboxes.

**Spear phishing**

Phishers take advantage of a lack of IT skills in an organization to exploit a stressed and tired workforce. Suppose a scammer can get the castle's keys (login credentials to corporate networks/apps); they can make a lot of money and cause damage. Spear-phishing targets those in an organization who have access to sensitive corporate resources, such as system administrators, C-level executives, and those working in accounts payable.

## PREVENTION:

**Process for Email Phishing**

Spammers can spoof your domain or organization to send fake messages that impersonate your organization. DMARC tells receiving mail servers what to do when they get a message that appears to be from your organization, but doesn't pass authentication checks, or doesn’t meet the authentication requirements in your DMARC policy record. Messages that aren't authenticated might be impersonating your organization, or might be sent from unauthorized servers.

DMARC is always used with these two email authentication methods or checks:

* **Sender Policy Framework (SPF)** lets the domain owner authorize IP addresses that are allowed to send email for the domain. Receiving servers can verify that messages appearing to come from a specific domain are sent from servers allowed by the domain owner.
* **Domain Keys Identified Mail (DKIM)** adds a digital signature to every sent message. Receiving servers use the signature to verify messages are authentic, and weren't forged or changed during transit.

**Process for Spear phishing**

Two factor Authentication(2FA) is the most effective method for countering phishing attacks, as it adds an extra verification layer when logging in to sensitive applications. 2FA relies on users having two things: something they know, such as a password and username, and something they have, such as their smartphones. Even when employees are compromised, 2FA prevents the use of their compromised credentials, since these alone are insufficient to gain entry

**Code Examples**

**2FA using node.js**

**Step 1:** npm install express speakeasy

**Step 2 Configure Express:**

const express = require('express');  
 const app = express();  
  
 app.use(express.json());  
 app.use(express.static('public'));  
 app.listen(3000, () => {  
 console.log('Server started on port 3000');  
 });

**Step 3 User Model:**

const users = [];  
  
 class User {  
 constructor(id, name, email, password, secret) {  
 this.id = id;  
 this.name = name;  
 this.email = email;  
 this.password = password;  
 this.secret = secret;  
 }  
 }  
 module.exports = { users, User };

**Step 4 Create Registration Route. In this it will generate a secret key in database:**

const { users, User } = require('./user');  
 const speakeasy = require('speakeasy');  
 const QRCode = require('qrcode');  
  
 app.post('/register', (req, res) => {  
 const { name, email, password } = req.body;  
 // Generate a new secret key for the user  
 const secret = speakeasy.generateSecret({ length: 20 });  
 // Save the user data in the database  
 const user = new User(users.length + 1, name, email, password, secret.base32);  
 users.push(user);  
 // Generate a QR code for the user to scan  
 QRCode.toDataURL(secret.otpauth\_url, (err, image\_data) => {  
 if (err) {  
 console.error(err);  
 return res.status(500).send('Internal Server Error');  
 }  
 // Send the QR code to the user  
 res.send({ qrCode: image\_data });  
 });  
 });

**Step 5: Login Route. In this we will validate user credentials and ask for a verification code.**

const { users } = require('./user');  
 const speakeasy = require('speakeasy');  
  
 app.post('/login', (req, res) => {  
 const { email, password, token } = req.body;  
 // Find the user with the given email address  
 const user = users.find(u => u.email === email);  
 // Validate the user's credentials  
 if (!user || user.password !== password) {  
 return res.status(401).send('Invalid credentials');  
 }  
 // Verify the user's token  
 const verified = speakeasy.totp.verify({  
 secret: user.secret,  
 encoding: 'base32',  
 token,  
 window: 1  
 });  
 if (!verified) {  
 return res.status(401).send('Invalid token');  
 }  
 // User is authenticated  
 res.send('Login successful');  
 });

**Step 6 Create Middleware to verify if the user has successfully logged in**

const speakeasy = require('speakeasy');  
  
 exports.requireToken = (req, res, next) => {  
 const { token } = req.body;  
 // Find the user with the given email address  
 const user = users.find(u => u.email === req.user.email);  
 // Verify the user's token  
 const verified = speakeasy.totp.verify({  
 secret: user.secret,  
 encoding: 'base32',  
 token,  
 window: 1  
 });  
 if (!verified) {  
 return res.status(401).send('Invalid token');  
 }  
 // Token is valid, proceed to the next middleware or route handler  
 next();  
 }

app.post('/protected', requireToken, (req, res) => {  
 // This route handler will only be called if the user's token is valid  
 res.send('Protected resource accessed successfully');  
 });

Two-factor authentication (2FA) is a powerful security mechanism that adds an extra layer of protection to user accounts. By requiring users to provide two different authentication factors, such as a password and a code sent to their mobile device, 2FA can help prevent unauthorized access to sensitive information.

## REFERENCES

Email phishing -<https://support.google.com/a/answer/2466580?hl=en>

Spear phishing and spoofing -<https://www.imperva.com/learn/application-security/phishing-attack-scam/>

<https://blog.devgenius.io/two-factor-authentication-with-node-js-and-express-secure-your-app-ca6de34a6fcb>

Target of phishing - <https://trustifi.com/blog/who-is-the-target-of-phishing-scams/>

Objectives Of Phishing Attack - https://www.guidepointsecurity.com/education-center/phishing-attack/

# **#6 - BROKEN ACCESS CONTROL**

**Contributing Team Members:** Alejandro Castro, Reddy Nagendra Darsha, Abdul Rehman Kasim (Ark), Breno

## DEFINITION –> Alejandro

Broken Access Control occurs when unauthorized users can access, modify, or delete data they shouldn’t be able to access. If an application manages users and roles poorly, hackers can bypass authorization and perform tasks as actual logged-in users.

## OBJECTIVE OF ATTACK: [Reddy]

The main objective of attackers is to exploit broken access control vulnerabilities with specific objectives aimed at undermining the security, integrity, and functionality of applications. Understanding these objectives can help devise effective defenses and mitigate risks.

1. **Unauthorized Account Access:**

Attackers aim to access confidential information they shouldn't have access to. This could involve personal data, financial records, or proprietary business information. By exploiting weak access controls, attackers can steal sensitive data, leading to data breaches and significant financial and reputational damage to organizations.

1. **Privilege Escalation:**

There are two main types of privilege escalation. Vertical privilege escalation occurs when a user gains higher-level access than permitted, such as a regular user obtaining admin rights. Horizontal privilege escalation happens when a user accesses another user's data at the same privilege level. Both scenarios can allow attackers to perform unauthorized actions and access restricted resources, potentially compromising the entire system.

1. **Service Disruption:**

Exploiting access control flaws attackers can disrupt services, potentially leading to Denial of Service attacks. By accessing and altering critical system settings or functions, attackers can render services unavailable, causing operational downtime and financial loss.

1. **Url Manipulation:**

URLs are not just addresses for browsers and servers to use as users go from page to page using links. They are requests from the browser to the server which act as a low-level form of programming. When the browser requests X from the server, the server responds with Y.

1. **Manipulation of Data:**

Attackers can modify data in unauthorized ways. This could involve altering account balances, manipulating transaction records, or changing user permissions. Such actions can disrupt business operations, lead to financial losses, and undermine the integrity of data systems.

## NOTABLE CASES/VICTIMS [Ark]:

Phreaking, Steve Job, and Steve Wozniak during their college days came up with one of the earliest forms of hacking with a device called the blue box which started its life as a whistle, A blue box generates audio tones (2600hz) that mimics signals used by telephone exchanges to connect calls and using this exploit one could place call anywhere in the world for a whopping $0.00.  
  
Mydoom was a worm that targeted Windows computers. It was first sighted on January 26, 2004, and the worm is believed to have originated from Russia to this day the person behind this worm has never been found but they had one intent to open a backdoor in as many computers as possible and carry out a denial-of-service attack   
  
Stuxnet was first discovered in 2010 but has been believed to have started development back in 2005, Stuxnet was primarily made for the disruption and destruction of Iran’s nuclear program, The objective of this worm was to grant access to the systems within the test facility and give CIA and FBI the access to the system.

## EXAMPLES:

We have four different categories, which are:

- URL Manipulation

- Exploiting Endpoints

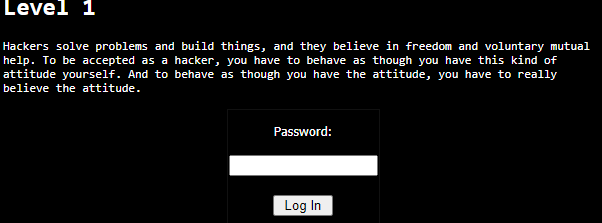
- Elevating User Privilege

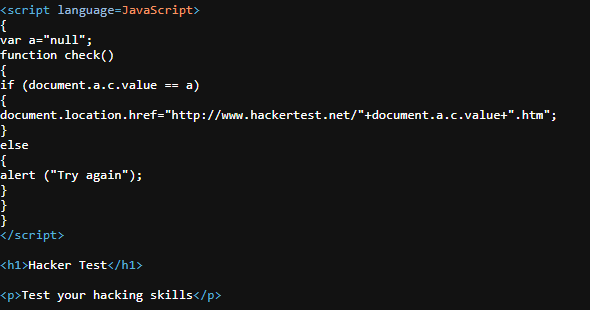
- Insecure Direct Object References

To be shown as a brief explanation we are breaking down the code into two categories, **URL Manipulation** and **Elevating User Privileges**.

Changing the URL parameters on a vulnerable website is an easy approach to getting unauthorized access to its pages and data. An attacker might change the value of the user's ID in a query parameter on an e-commerce website, for instance, to view the cart for a different user. The URL for accessing the current user's shopping cart includes the user's ID.

Example 1: For this example, we’re using the website called Hacker Test, at level 1. We are supposed to use a password to bypass the credentials which initially we do not have access

…

To manipulate this website, our first step is to investigate the console and page source, with this we can see the code logic of the page showing that is unprotected and vulnerable

Following the logic of this If statement inside the function **check** we can conclude that the variable a is the key to bypass the first level by using URL manipulation, altering the URL parameters to access different resources, or executing unintended actions.

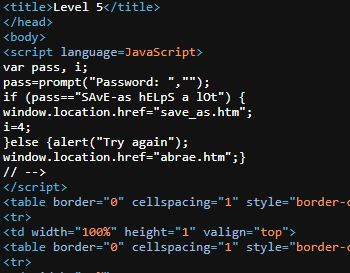
Example 2:

Once initial access has been obtained, privilege escalation techniques aim to obtain higher-level access within a system or application. This needs taking advantage of software bugs, configuration errors, or logical errors to elevate a user from low-privilege to administrative status.

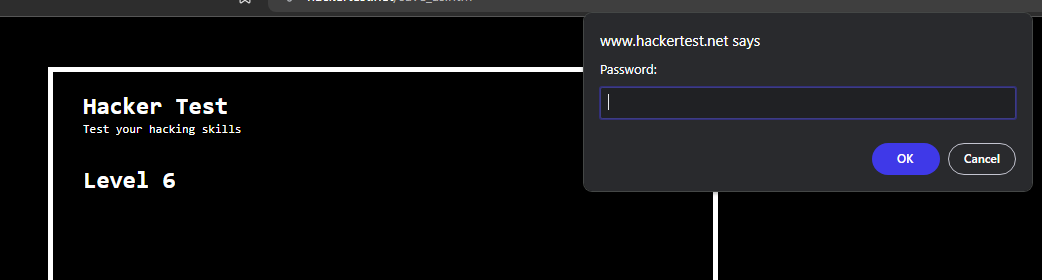
For this example let’s use Hacker Test, Level 4. **Taking advantage of the configuration error** of the website, we can reach level 6 without even using the password as shown in the images below:



Pressing Ctrl + U you can have access to the page source of level 5 even if the password popup is open on your screen, with this we can reach this point:



You can copy the window.location.href which is the condition if the password is put, so with this, you can reach level 6.



## PREVENTION -> Alejandro:

To prevent this security concern, there are 4 key methods

### **1. Implementing the Principle of Least Privilege**

A cybersecurity concept aims to give a user the bare minimum levels of access necessary for their user experience.

### **2. Secure Session Management and Authentication Controls**

Securing a unique session for a user and discarding it on log-out or inactivity drastically lowers the probability of a hacker hijacking a session. Additionally using tools like multi-factor authentication ensures that users are not being impersonated, even if their username and password become leaked

### **3. Regular Access Control Audits and Reviews**

Regular audits ensure roles are up to date. They are opportunities to review and apply the principle of least privilege. It will also be helpful to verify activity logs to scan for unauthorized behavior

### **4. Proper Error Handling and Logging**

By using proper error handling. The error messages will provide insights into what was trying to be accessed. With error handling in place, unauthorized information will not leak since the error is being handled.

Logging actions like successful and unsuccessful login attempts, changes to access rights, and attempts to access restricted resources are important. These logs can be analyzed to identify suspicious behavior.

## References:

<https://www.youtube.com/@DisruptReality>

[https://brightsec.com/blog/broken-access-control-attack-examples-and-4-defensive-measures/#:~:text=Broken%20access%20control%20vulnerability%20is,critical%20web%20application%20security%20risks.](https://brightsec.com/blog/broken-access-control-attack-examples-and-4-defensive-measures/#:~:text=Broken%20access%20control%20vulnerability%20is,critical%20web%20application%20security%20risks)

<https://guides.codepath.com/websecurity/URL-Manipulation>

<https://heimdalsecurity.com/blog/broken-access-control/>

<https://medium.com/@kamalkannanares/tryhackme-owasp-broken-access-control-7985ecede0d9>

# **#7 - DENIAL OF SERVICE ATTACKS**

|  |  |
| --- | --- |
| **Contributing Team Members:** | Georgios Toufexis, Sumin Shrestha, Ritika Mehta, Ayushi Behl |

## DEFINITION

A denial-of-service (DoS) attack is a type of cyber attack in which a malicious actor aims to render a computer or other device unavailable to its intended users by interrupting the device's normal functioning.

DoS attacks typically function by overwhelming or bombarding a targeted machine with requests until normal traffic is unable to be processed, resulting in denial-of-service to its end users.

A DoS attack is characterized by using a single computer to launch the attack.A distributed denial-of-service (DDoS) attack is a type of DoS attack that comes from many distributed sources.

## OBJECTIVE OF ATTACK:

The primary goal of a DoS attack is not to steal information but to overload a website or network or to take it down, with the aim of degrading its performance or even making it completely inaccessible to its intended users.

Another objective of such an attack can be to **explore and exploit** the vulnerabilities that exist in the system.

The attackers' motivations are diverse, ranging from simple fun, to financial gain and revenge/ blackmail.

These attacks are often targeted at web servers of high-profile organizations such as banking, commerce, and media companies, or government and trade organizations. DoS attacks do not typically result in the theft or loss of significant information or other assets,  they can cost the victim a great deal of time and money to handle.

## NOTABLE CASES/VICTIMS:

***February 5th, 2024, Pennsylvania, USA***

A DDoS attack that targeted Pennsylvania’s Supreme Court was reported after several of the court’s computer systems and web services, including an electronic case filing portal were down. The Supreme Court said that there was no data interruption or theft, however these attacks forced the services to work with the traditional paper and mail filing.

**January 18th 2024, Davos, Switzerland**

During a World Economic Forum that takes place in Switzerland yearly, several DDoS attacks were reported, on websites run by the Federal Government of Switzerland. According to the Swiss National Cybersecurity Center, these attacks mainly disrupted the access to the websites. The hacking group called NoName took responsibility for the attack because of political reasons, and claimed that they also targeted hotels, airports and railways. These attacks were mainly used for the group to gain media attention for their cause, and not to steal any data or other sources from the websites.

***November 9th, 2023***

OpenAI reported that it was experiencing periodic DDoS attacks that as a result had millions of people who use  ChatGPT hanging. With multiple companies and individuals using ChatGPT for coding practices, including debugging, this incident raised the concern of security researchers. The individual(s) behind the attack took responsibility and revealed that this DDoS was motivated my political reasons.

## EXAMPLES:

Attackers may use one or more of the following strategies to DDoS their targets:

1. **Application-layer attacks**, also known as **layer 7 DDoS attacks,** create a denial of service by **overwhelming the target’s server and network resources** with legitimate-seeming HTTP requests.
2. **Protocol attacks**, or **state-exhaustion attacks**, overwhelm network equipment and infrastructure by using **layer 3 or 4 protocols (e.g. ICMP)** to send a flood of unwanted traffic to their target.
3. Volumetric attacks use amplification techniques — for example, deploying a botnet or exploiting common networking protocols — to consume all of the target’s available bandwidth.

## PREVENTION:

**Network Segmentation**

Segmenting the network into smaller segments provides security teams with increased control over the traffic that flows into their systems and can limit the impact of DoS attack. This can be done by creating VLANs, and firewalls can limit the spread of an attack. The optimal solution is zero trust microsegmentation which builds smaller, more secure zones on a network, enabling the organization to create policies that minimizes flow between workloads. This limits a hacker's ability to move between compromised applications and reduces the complexity of managing network segmentation. Adding a device-level and device-cloaking firewalling, external to the operating system remains the most reliable form of DoS protection.

**Real-time, Adaptive Threat Monitoring:**

Log monitoring can help pinpoint potential threats by analyzing network traffic patterns, monitoring traffic spikes or other unusual activity, and adapting to defend against anomalous or malicious requests, protocols, and IP blocks.

**Rate Limiting**

Limiting the rate of network traffic over a specific time period to reach a server or resource can prevent a DoS attack from overwhelming it. This will stop concurrent users or connections from exhausting network resources and can be implemented using some kind of filter.

**Load Balancing**

Distributing traffic across multiple servers which reduces the strain on each server and makes the server more efficient, speeding up performance and reducing latency. So, a DoS attack can be prevented from overwhelming a single server or resource. Load balancing can be achieved using hardware or software solutions, where hardware-based solutions require the installation of dedicated load balancing device and software-based solutions can run on a server, on a virtual machine or in the cloud.

**IP blocking**

By blocking or restricting the access to the network from a known or suspected malicious sources can prevent a DoS traffic from reaching its target. Most websites use this method to restrict access to a specific individual depending on factors like their geography and the reliability of their IP address.

## CITATIONS:

<https://therecord.media/swiss-websites-targeted-ddos-attacks-davos>

<https://www.cloudflare.com/learning/ddos/glossary/denial-of-service/>

<https://www.researchgate.net/publication/331529715_A_Case_Study_of_the_Impact_of_Denial_of_Service_Attacks_in_Cloud_Applications>

<https://statescoop.com/pennsylvania-restores-court-system-ddos-cyber-attack-2024/>

<https://www.scmagazine.com/news/ddos-attack-on-chatgpt-sparks-concerns-over-coding-productivity-disruptions>

<https://www.paloaltonetworks.com/cyberpedia/what-is-a-denial-of-service-attack-dos>  [https://www.cloudflare.com/learning/ddos/how-to-prevent-ddos-attacks/#:~:text=Attack%20surface%20reduction%3A%20Limiting%20attack,ports%2C%20protocols%2C%20and%20applications.](https://www.cloudflare.com/learning/ddos/how-to-prevent-ddos-attacks/#:~:text=Attack%20surface%20reduction%3A%20Limiting%20attack,ports%2C%20protocols%2C%20and%20applications)

# **#8 - CROSS-SITE REQUEST FORGERY (CSRF)**

|  |  |
| --- | --- |
| Definition | Navjot |
| Objective Of Attack | Vincent Chan |
| Notable Cases/Victims | Cherry |
| Examples | Bhargav Suthar |
| Prevention | Deryn |

## DEFINITION:

Cross-Site Request Forgery (CSRF) is an attack that force a user to perform unwanted actions on a web application while they are currently authenticated A successful CSRF attack has the ability to compel a typical user, as the victim, to execute state-changing actions such as transferring payments or altering their email address, among other things. The entire web application may be compromised by CSRF if the victim is an administrative account.

## OBJECTIVE OF ATTACK:

The attacker in a CSRF is to force an innocent user to submit a state-changing request which the user doesn’t intend to do. Those requests include submitting or deleting a record, submitting a transaction, purchasing a product, changing a password, sending a message. After the user logs in a web application such as a bank website and gets a validation token from the server, A CSRF attack can be launched by sending a forge request to the user and tricking the user into clicking a hyperlink that contains a maliciously crafted, unauthorized request for that web application.

Since the user is validated, any request forged by the attacker can be executed acting like a normal operation by the user.Since the user is validated, any request forged by the attacker can be executed acting like a normal operation by the user.The users have no idea of what the hyperlink did to themself because the intention of the attack is hidden.

## NOTABLE CASES/VICTIMS:

Paypal Vulnerability (2012)

A CSRF vulnerability on Paypal, a major online payment platform is discovered by a security research in 2012

Attack worked by embedding malicious links on external websites. Logged-in PayPal users who visited these sites would trigger the site to send unauthorized requests to PayPal’s servers to alter their account settings using their credentials

Through the alteration of user account settings, these attackers could change the security questions and answers which meant there will be a potential risk of an account takeover - severe financial security threat

To resolve this issue, PayPal implemented additional security checks and required re-authentication for critical actions. They reassured users no account were compromised

## EXAMPLES:

For a CSRF attack to be possible, three key conditions must be in place:

1. **A relevant action:**  There is an action within the application that the attacker has a reason to induce. This might be a privileged action (such as modifying permissions for other users) or any action on user-specific data (such as changing the user's own password).
2. **Cookie-based session handling:** Performing the action involves issuing one or more HTTP requests, and the application relies solely on session cookies to identify the user who has made the requests. There is no other mechanism in place for tracking sessions or validating user requests.
3. **No unpredictable request parameters:** The requests that perform the action do not contain any parameters whose values the attacker cannot determine or guess. For example, when causing a user to change their password, the function is not vulnerable if an attacker needs to know the value of the existing password.

If, We Understand using an example:

Alice wishes to transfer $100 to Bob using the bank.com web application that is vulnerable to CSRF. Maria, an attacker, wants to trick Alice into sending the money to Maria instead. The attack will comprise the following steps:

1. Building an exploit URL or script
2. Tricking Alice into executing the action with Social Engineering

GET Scenario

Original Url

GET http://bank.com/transfer.do?acct=BOB&amount=100 HTTP/1.1

Maria takes the original command URL and replaces the beneficiary name with herself, raising the transfer amount significantly at the same time:

http://bank.com/transfer.do?acct=MARIA&amount=100000

The social engineering aspect of the attack tricks Alice into loading this URL when Alice is logged into the bank application. This is usually done with one of the following techniques:

* Sending an unsolicited email with HTML content
* Planting an exploit URL or script on pages that are likely to be visited by the victim while they are also doing online banking

The exploit URL can be disguised as an ordinary link, encouraging the victim to click it:

<a href="http://bank.com/transfer.do?acct=MARIA&amount=100000">View my Pictures!</a>

Or as a 0x0 fake image:

<img src="http://bank.com/transfer.do?acct=MARIA&amount=100000" width="0" height="0" border="0">

If this image tag were included in the email, Alice wouldn’t see anything. However, the browser will still submit the request to bank.com without any visual indication that the transfer has taken place.

POST scenario

Such a request cannot be delivered using standard A or IMG tags, but can be delivered using a FORM tags:

<form action="http://bank.com/transfer.do" method="POST">

<input type="hidden" name="acct" value="MARIA"/>

<input type="hidden" name="amount" value="100000"/>

<input type="submit" value="View my pictures"/>

</form>

This form will require the user to click on the submit button, but this can be also executed automatically using JavaScript:

<body onload="document.forms[0].submit()">

<form./>.

## PREVENTION:

**Synchronizer Token Pattern (STP):** Each session, the server generates a random CSRF token which is included in the request and a cookie. Once the request is sent, the server compares the CSRF token in the cookie to the one in the request to validate its authenticity. This prevents CSRF attacks but can create a vulnerability for other attacks if not implemented correctly.

**One-Time Tokens:** A variation of STP where the server generates a CSRF token for every request. This is a more effective approach but can greatly increase the processing load of the server and cause usability issues.

**Double-Submit Cookies:** The server sends two cookies to the client, one for authentication and one for CSRF prevention which includes a random token that is also included in the form submission. When the request is received, the server compares the CSRF token to the token in the request to confirm authenticity. The ***Signed* Double-Submit Cookie** pattern is the most secure type of this method and uses a secret key only known to the server ensuring that an attacker cannot inject their own CSRF.

**Referer Header Checking:**  This approach verifies that the referer header in the HTTP request originates from the same site as the requested resources. However hackers can change or disable the referrer header.

**Custom Header Checking:** Validates a request’s authenticity by using client-side code to add custom HTTP headers in every request. This method is much more secure than Referer Header Checking.

**IP Address Validation:** Verifies that the request originates from the same IP address as previous requests made by the user. This will cause usability issues if the user's IP address changes frequently.

## REFERENCES

Example: <https://portswigger.net/web-security/csrf#how-does-csrf-work>

<https://owasp.org/www-community/attacks/csrf>

Prevention: <https://medium.com/@yadav-ajay/cross-site-request-forgery-csrf-64066cddbfb3>

<https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html>

Objective: <https://www.synopsys.com/glossary/what-is-csrf.html#:~:text=An%20attacker's%20aim%20for%20carrying,Submitting%20a%20transaction>.

<https://spanning.com/blog/cross-site-forgery-web-based-application-security-part-2/>

<https://www.scmagazine.com/news/popular-websites-fall-victim-to-csrf-exploits>

**(Notable Cases) News Link:**

<https://netplatforms.co.uk/cyber-post-2/>

**Presentation Link:**

<https://docs.google.com/presentation/d/1Vw7COSjdXLDvmDbawYPS7HJJOffS67JAZx_MniF3Jas/edit?usp=sharing>

# **#9 - SERVER-SIDE REQUEST FORGERY (SSRF)**

|  |  |
| --- | --- |
| Contributing Team Members: | Akash Sharma, Anant Chauhan, Noah Choi, Kathan Patel |

## DEFINITION

Server-side request forgery is when hackers get a server to make HTTP requests to internal resources or other servers to gain access and information. This is done by creating a custom-made URL for a server to access and return the results to the hacker, retrieving the URL's contents from the request. URLs are sent to unexpected hosts or ports, making it seem like the server is sending the request, thereby forging the server-side request.

## OBJECTIVE OF ATTACK:

A server-side request forgery attack relies on the server’s internal resources as its main target. These are often behind firewalls but are bypassed by this attack. The attacker abuses the server functionality to access or modify resources. These often include data that allows URLS to import and read data.

The main objective of the server-side request forgery depends on the attacker's intentions. Hence this attack is ideally used to supply or modify a URL that runs on the server, in which the code running on the server will read or submit data. The attacker can read server configurations such as AWS metadata and even connect to internal services. Like HTTP database bases or perform POST requests to internal services that should not be intended to be exposed.

The most used target would be in the form of XML data format, which is used to transmit structured data from the client to the server often by external entity injection (XXE), which is most vulnerable to SSRF.

## NOTABLE CASES/VICTIMS:

Microsoft Exchange Server- On March 2, 2021, Microsoft detected multiple zero-day(A zero-day is a security flaw for which the vendor of the flawed system has yet to make a patch available to affected users) exploits being used to attack on-premises versions of Microsoft Exchange Server. Over the next few days, over 30,000 organizations in the US were attacked as hackers used several Exchange vulnerabilities to gain access to email accounts and install web shell malware, giving the cybercriminals ongoing administrative access to the victim’s servers. They compromised Microsoft Exchange servers outlook web access (OWA)] giving them access to victims' entire servers and networks as well as to emails and calendar invitations, only at first requiring the address of the server, which can be directly targeted or obtained by mass-scanning for vulnerable servers.

2019 Capital One Cyber Incident - On July 19, 2019, there was unauthorized access by an outside individual who obtained certain types of personal information relating to people who had applied for our credit card, and to Capital One credit card customers.

## EXAMPLES:

For Capital One Cyber Incident

(1) The threat actor accessed the network via anonymizing services such as TOR and VPN.

(2) They then performed a server-side request forgery or SSRF attack that tricked the server into executing commands as a remote user.

(3) Next, a scanning tool identified misconfigured WAF(web application firewall) and relayed commands to gain access to the AWS platform and receive temporary credentials to the environment.

(4) The SSRF and WAF misconfiguration were combined, and the attacker obtained valid credentials, which allowed her to run commands through the AWS CLI.

(5) She identified data stored in S3 buckets.

(6) she copied 30 GB of customer data across 700 different S3 buckets.

## PREVENTION:

SSRF is one of the most dangerous web application attacks that you must consistently protect your web applications from, and this can be achieved with different best practices, tools, techniques, and methodologies. To mitigate the risk of SSRF attacks, consider the following measures: To mitigate the risk of SSRF attacks, consider the following measures:

Proper Implementation of input validation and sanitation: Ensure that any data, text, URLs and other characters supplied by the users are validated and sanitized to avert mischievous input that would lead to undesired results.

Building allow lists for URLs and IP addresses: As a measure of combating the malicious attack and misuse by unauthorized users, one should limit the accessibility to URLs and IP addresses of their internal resources including domain names of all the identified and accredited applications.

Securing third-party services and libraries: In making this check, it is vital to ensure that all the third-party websites, which make use of cookies and preferences that are used in application, are set to the latest and configured securely.

Monitoring and Logging: It is useful for detecting and mitigating SSRF attacks by sending data packets to a specified URL and analyzing the response.

It must be noted that to minimize the threat doubt associated with SSRF, real-time monitoring and logging mechanisms could be installed by an analyst, which would immediately indicate signs of such an attack or odd patterns.

Performing regular security testing: Conduct application security penetration testing, that is by white hat hacker or gray hat hacker to identify vulnerabilities in your web applications or protect the organization from SSRF attacks. With the help of cutting-edge technologies and innovative security solutions, one can prevent malicious attacks on SSRF.

## Sources:

<https://heimdalsecurity.com/blog/server-side-request-forgery-attack/>

<https://medium.com/purplebox/a-complete-guide-to-server-side-request-forgery-ssrf-58b1dbdd064e>

<https://owasp.org/www-community/attacks/Server_Side_Request_Forgery>

<https://cyberlaw.ccdcoe.org/wiki/Microsoft_Exchange_Server_data_breach_(2021)>

<https://www.evolvesecurity.com/blog-posts/how-to-prevent-server-side-request-forgery>

<https://cheatsheetseries.owasp.org/cheatsheets/Server_Side_Request_Forgery_Prevention_Cheat_Sheet.html>

# **#10 - BRUTE FORCE ATTACKS (& Dictionary Attacks)**

|  |  |
| --- | --- |
| Contributing Team Members: | Joshua Moshood, Yuto Arimori, Evneet Kaur, Karishma Patel |

## **DEFINITION:**

A brute force attack is a hacking method that uses trial and error to crack passwords, [login credentials](https://www.fortinet.com/resources/cyberglossary/login-credentials), and encryption keys. It is a simple yet reliable tactic for gaining unauthorized access to individual accounts and organizations’ systems and networks. The hacker tries multiple usernames and passwords, often using a computer to test a wide range of combinations, until they find the correct login information. Despite being an old cyberattack method, brute force attacks are tried and tested and remain a popular tactic with hackers.

**Type of Brute force attack:**

***Dictionary Attack*** : A dictionary attack is a basic form of brute force hacking in which the attacker selects a target, then tests possible passwords against that individual’s username. The attack method itself is not technically considered a brute force attack, but it can play an important role in a bad actor’s password-cracking process.

The name "dictionary attack" comes from hackers running through dictionaries and amending words with special characters and numbers. This type of attack is typically time-consuming and has a low chance of success compared to newer, more effective attack methods.

**Reference link:** <https://www.fortinet.com/resources/cyberglossary/brute-force-attack>

## **OBJECTIVE OF ATTACK:**

**Brute Force Attack**:

The main purpose of a Brute Force Attack is to hack passwords, login information, or encryption keys. They use this way to get credential information of individual accounts, or organization’s systems and networks. API keys or SSH logins are also their targets. The advantage of this attack is that this is simple to attack. They can know the security level by the amount of time it takes to brute force into a system.

**Dictionary Attack**:

A dictionary attack is done to break into a password-protected computer, network, or other IT resource by systematically entering every word in a dictionary. A dictionary attack can also be used in an attempt to find the key necessary to decrypt an encrypted message or document. This strength is that it is simple and fast. Some words and phrases as passwords are often used. They just compare the popular password with the popular words in the dictionary.

**References:**

<https://www.cloudflare.com/learning/bots/brute-force-attack/>

<https://www.fortinet.com/resources/cyberglossary/brute-force-attack#:~:text=A%20brute%20force%20attack%20is,and%20organizations'%20systems%20and%20networks>.

<https://www.techtarget.com/searchsecurity/definition/dictionary-attack#:~:text=A%20dictionary%20attack%20is%20a,an%20encrypted%20message%20or%20document>.

<https://www.geeksforgeeks.org/rainbow-table-attack-vs-dictionary-attack/#:~:text=Advantages%20of%20Dictionary%20Attack&text=These%20are%20frequently%20effective%20due,it%20is%20simple%20and%20fast>.

## **NOTABLE CASES/VICTIMS:**

1. **Company & year: Dunkin’ Donuts (2015)**:

**Impact**: Dunkin’ Donuts pays over half a million in penalties, Cyberattackers got access to 19,715 user accounts.

Dunkin’ Donuts digital customer accounts were targeted by hackers who used a leaked list of previously stolen credential information and ran brute force algorithms. They gained access to 19,715 user accounts for the customer loyalty application and stole tens of thousands of dollars of rewards cash. Source: <https://www.strongdm.com/blog/brute-force-attack#:~:text=Examples%20of%20Brute%20Force%20Attacks,-Dunkin'%20Donuts%20pays&text=The%20result%20of%20the%20brute,security%20protocols%20for%20the%20application>.

1. **20.6 million accounts were compromised at Alibaba(In 2016)**,

Impact: a team of hackers used a previously breached database with over 99 million credentials for multiple web applications. Taking advantage of weak passwords and users implementing the same password across other accounts, they used brute force and credential stuffing to successfully access nearly 20% of all the targeted accounts.

Source: <https://www.strongdm.com/blog/brute-force-attack#:~:text=Examples%20of%20Brute%20Force%20Attacks,-Dunkin'%20Donuts%20pays&text=The%20result%20of%20the%20brute,security%20protocols%20for%20the%20application>

1. **Magento (2018)**

**Impact**: More than 1000 account credentials were found on the dark web. The affected users’ credit card information were scraped

Source: <https://www.makeuseof.com/brute-force/>

1. **Canadian Revenue Agency (2020)**

**Impact**: 11,000 accounts belonging to the CRA and other government-related services. The attack targeted the Canada Revenue Agency (CRA) and Government of Canada Key service (GCKey), agencies that enable Canadians to access various government programs and services in the country.

## **EXAMPLES:**

**Demonstration of Brute Force Attack**:

Attackers assume a password on their victim in brute force attacks. Initially, they try to guess with the use of the 4-character password which consists of lowercase letters (a-z) and digits (0-9). They begin generating short combinations of letters and numbers like “abc1”, “abc23” among others. If not, they increase the length of the expected passwords and try to test as many passwords as possible. The attackers keep on trying a series of passwords to break into the target’s account until they succeed. Once they reach and check out for it from the targets’ site, it becomes very simple for them to access the individual's system.

**Demonstration of Dictionary Attack**:

In dictionary attack scenarios, hackers usually create passwords starting from common names used by people in everyday life, progressing to phrases or even more popular words that are likely to be found in any internet user’s personal data, such as bank accounts, social media profiles, and emails. After creating such a list/password base, attackers start checking each word from it. In case when the first password doesn’t match the previous one then the attacker goes further without testing twice any word again. The process continues until one does not work anymore.

## **PREVENTION:**

<https://www.crowdstrike.com/cybersecurity-101/brute-force-attacks/>

**Multi-Factor Authentication (MFA)**:

Requiring multiple forms of verification for access, MFA makes it significantly harder for attackers to gain unauthorized entry.

**Rate Limiting**:

Limiting the number of login attempts prevents brute force attacks by slowing down or blocking repeated access attempts from the same source.

**Creating strong and complex passwords**:

A brute Force Attack is a random attack. They repeat try and error until they find the passwords, so we need to create strong and complex passwords such as combining uppercase and lowercase letters, numbers, and symbols.

**Avoiding easy passwords**:

We also need to avoid using passwords that they can guess easily such as your name or birthdate. Using a password manager is another way to prevent dictionary attacks. Dictionary attacks are a way to use words in a dictionary. If we use a password manager, it creates a strong master password to protect our password storage file.

**Importance of combination**:

We need to use long unique passwords. Passwords consisting of completely random characters should be of at least 10 characters. Using a passphrase with four to six random words offers similar protection but is easier for the average user to remember. A simple password of eight lowercase characters means 26^8 combinations. Eight characters including uppercase, lowercase, numbers, and symbols are 95^8 combinations.

<https://www.sapphire.net/insights/dictionary-attacks/#:~:text=Use%20Strong%20and%20Complex%20Passwords,letters%2C%20numbers%2C%20and%20symbols>.

<https://www.techtarget.com/searchsecurity/definition/dictionary-attack#:~:text=A%20dictionary%20attack%20is%20a,an%20encrypted%20message%20or%20document>

# **#11 - CRYPTOGRAPHIC FAILURES**

|  |  |
| --- | --- |
| Contributing Team Members: | GANGA SURESH  ISIBOR EROMOSELE  JERRY BORGES JACOB  NEELIMA AKOJU |

## DEFINITION

Cryptographic Failures (previously known as Sensitive Data Exposure) indicate the vulnerabilities, misconfigurations, or weaknesses in cryptographic systems that lead to exposure of sensitive data. Poor cryptography directly affects the security of an application and makes information available to malicious hackers.

Some causes for Cryptographic Failures are:

* Inadequate key Management:

Weak or poorly managed encryption keys make the cryptographic systems vulnerable to brute-force attacks and decryption.

* Weak Algorithms:

Algorithms with known vulnerabilities compromise data security.

* Random Number Generation Issues:

Poor random number generation makes the encryption key guessable.

* Side Channel Attacks:

Hackers use unintended side effects like power consumption or time variations to access insight into cryptographic processes and sensitive information.

Source:

<https://www.softwaresecured.com/post/introduction-to-cryptographic-failures>

<https://www.linkedin.com/pulse/cryptographic-failures-understanding-pitfalls-impact-zahid-ali/>

## OBJECTIVE OF ATTACK:

Hackers may attempt to bypass the security of a cryptographic system by discovering weaknesses and flaws in cryptography techniques, cryptographic protocol, encryption algorithms, or key management strategy.

**Types of attack**

* Passive attack - Passive cryptographic attacks intend to obtain unauthorized access to sensitive data or information by intercepting or eavesdropping on general communication. In this situation, the data and the communication remain intact and are not tampered with. The attacker only gains access to the data.
* Active attack - while active cryptography attacks involve some kind of modification of the data or communication. In this case, the attacker not only gains access to the data but also tampers with it.

Depending on the type of cryptographic system in place and the information available to the attacker

* Brute force attack
* Ciphertext-only attack
* Chosen plaintext attack
* Known plaintext attack
* Key and algorithm attack

Source:<https://www.packetlabs.net/posts/cryptography-attacks/#:~:text=These%20attacks%20aim%20to%20retrieve,algorithms%2C%20or%20key%20management%20strategy.>

## NOTABLE CASES/VICTIMS:

**Exactis Data Breach**

Exactis is a marketing company based in Florida established in the year 2015. They aggregate and sell lists of data. The data appears to be collated from every survey every citizen and businessperson ever took. Evidently, they have and store more than 3.5 billion business and consumer records. The Exactis data breach was discovered by security expert Vinny Troia who was testing the security protocols of ElasticSearch. The Exactis database was entirely unsecured and accessible. More than 340 million records were out in the open for anyone to steal. The disturbing thing was there were 400 entries for each person.

Source: <https://www.idstrong.com/sentinel/exactis-data-breach/>

**Facebook Cryptographic Failure**

* It was revealed that over 540 million records related to Facebook users were accidentally leaked by two third-party Facebook app developers. These apps, without any malicious intent, posted the records in very much plain sight on Amazon’s cloud service. The exposed records included Facebook users’ account names, IDs, friends, photos, location check-ins, and passwords.
* On March 21, 2019, cyber security writer Brian Krebs reported in his KrebsOnSecurity blog that hundreds of millions of Facebook users had their account passwords stored in plain text and searchable by thousands of Facebook employees — in some cases dating back to 2012. Between 200 million and 600 million users may have had their account passwords stored in plain text and searchable by more than 20,000 employees. The article also mentioned that access logs showed some 2,000 engineers or developers made approximately 9 million internal queries for data elements that contained plain text user passwords. The issue with storing passwords in plain text and unencrypted is that it leaves the passwords wide open to cyber attacks or potential employee abuse.

Source:<https://www.cshub.com/data/articles/incident-of-the-week-facebook-fails-to-secure-passwords>

## EXAMPLES/DEMONSTRATE THE PROCESS:

**Storing Sensitive data as plain text**

The most common mistake that leads to Cryptographic Failures is storing data as plain text. This usually occurs when data is used for testing, and then pushed onto the live server without encrypting it.

**Weak key generation**

Cryptographic systems rely on random numbers for generating keys. If the logic for the algorithm is flawed, this can lead to weak key generation. Attackers can exploit this weakness to try and decrypt the data stored in the databases.

Key generation algorithms rely on various factors, such as time and place of generation, to create unique keys. But, if this algorithm is flawed, it can lead to duplicate keys being generated.

Attackers can simulate these conditions and recreate the encryption keys. This helps them easily decrypt your encrypted data and access them.

## PREVENTION:

1. Use strong encryption algorithms to store sensitive data
2. Encryption keys should be stored in a secure way and not a plain text
3. Use security protocols to send data over the internet (like HTTPS)
4. Regularly update security protocols to prevent attacks
5. Conduct Penetration Testing and Security audits to ensure there are no bugs in the software that can lead to cryptographic failures

Source: [OWASP Top Ten: Cryptographic Failures (pentestpeople.com)](https://www.pentestpeople.com/blog-posts/owasp-top-ten-cryptographic-failures)

# **#12 - VULNERABLE AND OUTDATED COMPONENTS**

|  |  |
| --- | --- |
| Contributing Team Members: | Amanpreet Kaur, Jaskaran Kaur, Ali Syed, Ting Liu |

## DEFINITION

Open-source or proprietary code that has software flaws or is out-of-date is referred to as having vulnerable and obsolete components. For online applications, this code can take the shape of libraries or frameworks, such as Django (Python), Angular (JavaScript), and Laravel (PHP). Unfortunately, this code is frequently developed with little to no regard for security, endangering the reputation of businesses and possibly having disastrous effects on application users. These components can be exploited by attackers to gain unauthorized access to sensitive data or take control of the system. On the other hand, outdated components are no longer supported by the vendor. As a result, such components lack security patches addressing uncovered vulnerabilities, making them low-hanging fruit for threat campaigns.

Vulnerabilities over such components may exist for several years before they are discovered, fixed, or disclosed. Sometimes, a vulnerability may not even be identified until it has been exploited. Vulnerabilities over such components may exist for several years before they are discovered, fixed, or disclosed.

Types of vulnerable and outdated components are categorized and mapped to the following three Common Weaknesses and Enumerations:

• **Failure to maintain components -** Flaws that occur due to the presence of unused features along with insufficient patch management and other preventative maintenance controls on modern application development pipelines

• **Out-of-date support systems** - Relying on outdated versions of software packages in the deployment

• **Component misconfiguration** - Third-party package components causing conflict with the overall system configuration.

## OBJECTIVE OF ATTACK:

Vulnerable and outdated components often used by hackers for such as:

* Data breaches or Mass disclosure of records intended for private access
* Distributed Denial-of-Service
* Remote Code Execution attacks
* Data integrity violations
* Malware infections
* Compromised systems.

It can range from a small cross site scripting (XSS) attack to full remote access to the server by exploiting a remote code execution (RCE).

These are some real-life examples of each of the [Top 10 Vulnerabilities and Cyber Threats for 2021](https://owasp.org/Top10/) according to The Open Web Application Security Project (OWASP).

## NOTABLE CASES/VICTIMS:

**Log4j2 Vulnerability**

Log4j2 is an open-source logging library provided by Apache Foundation. Lots of major online service providers use it, including Amazon, Microsoft, IBM and Google.

There is a remote code execution vulnerability found in the versions 2.14.1 on 9, December, 2021.

It’s easy for threat actors to exploit. They just need to parse a string into a URL, then they are able to take over the servers to steal data and passwords, install malware and crypto-mining software, they can even sell the access to third parties for profit.

The Apache Software Foundation released an update, Log4j 2.17.1 on 27 December, 2021 to address all known Log4j2 vulnerabilities to date. But the widespread use of the Log4j2 library means eradicating the threat will be a long-term exercise.

**Australian Census Hack**

The Australian government’s census website was hacked in 2019, the main reason is the vulnerability in a library “SimplePie”, which was used to parse RSS and Atom feeds.

This vulnerability had been known for a couple months but the Australian Bureau of Statistics failed to patch it before being hacked.

The hack was discovered on August 9, 2019, and the ABS immediately took the census website offline. The ABS later announced that no personal information had been stolen in the hack, but the incident raised concerns about the security of the Australian government's online services.

**Swagger UI**

Swagger UI is a library that is widely used to display interactive API documentation. There is a feature which allows the user to specify a url pointing to a yaml or json file and return to the user the specifics of the API.

On version 3.14.1 to 3.38.0, this feature is vulnerable to reflected XSS injections because it uses a library called DOMPurify to clean up user inputs. This library is used to prevent certain attacks like XSS, but bypasses have been found.

## EXAMPLES:

**Injection:** An attacker can execute unintended commands or gain access to sensitive data by injecting malicious data as part of a command or query. This usually happens when a website fails to filter, validate or sanitize users’ inputs or implement parameterization**.**

Example: A team of attackers used SQL injection to penetrate corporate systems at several companies, primarily the 7-Eleven retail chain, stealing 130 million credit card numbers.A diagram of a device and a magnifying glass

Description automatically generated

1. **Dos attack: Its** an attack meant to shut down a machine or network, making it inaccessible to its intended users. DoS attacks accomplish this by flooding the target with traffic, or sending it information that triggers a crash

**Example:** Major Ukrainian online bank Monobank faced a [distributed denial-of-service attack](https://www.scmagazine.com/brief/switzerland-subjected-to-noname-ddos-attacks) over the weekend that involved 580 million service requests, marking the bank's largest DDoS attack.

A diagram of a computer network

Description automatically generated

1. **Buffer overflow:** Buffer overflow vulnerabilities occur when the amount of data in a buffer memory exceeds the assigned storage capacity. Attackers abuse buffer overflow vulnerabilities to corrupt and manipulate data in adjacent memory buffers since overflowing data flows into these memory units. In instances where attackers gain access to a vulnerable component’s memory layout, they can overwrite executable codes stored in buffers with malicious scripts that allow them to cause system crashes, alter security controls or obtain escalated privileges for deeper attacks.

**Example:** The first key example of a widespread buffer overflow attack is the Morris worm. In 1988, this worm traveled across the nascent internet to bring down 10% of the then “internet” in just two days. Across two years, this computer worm affected 60,000 computers.

A screenshot of a computer

Description automatically generated

## PREVENTION:

* **Software Composition Analysis:** Finding and making visible each open-source software component within the application code is the goal of Software Composition Analysis (SCA). It is also advised to test each of these components for known security vulnerabilities and faults while doing SCA. Finding vulnerable design flaws that could result from merging several open-source software packages and determining whether they have an overall effect on the environment's security compliance, posture, and code quality is the optimum way to go about doing this.
* **Patch management:** Most threat campaigns target unpatched versions of software components used in installations. These components frequently have known vulnerabilities, which makes it simple for threat actors to plan effective exploits. Patch management makes ensuring that out-of-date component versions are either upgraded to secure versions or replaced. Security teams may proactively stop data integrity breaches, unsecured third-party component introductions, and other sporadic threats with a scheduled patch management lifecycle.
* **Multi-factor Authentication:** The majority of third-party components function with root access since they execute privileged code. In these situations, if an attacker manages to access one weak point in the application, they may be able to access other parts as well.

By implementing several authentication mechanisms to give resource permissions at the component level, multi-factor authentication helps lower the attack blast surface while bolstering crucial access control measures to avoid such interwoven chain abuses.

* **Secure Design Patterns:** The use of safe design patterns forces developers to follow best practices while creating reusable, secure apps that are meant to thwart common weaknesses and attack vectors. By avoiding known security holes in application code, these design patterns assist security teams in defining a course of action to take in the event that an exploit is successful. Adopting secure design principles guarantees that no susceptible components are employed at any point during the Software Development Lifecycle (SDLC) and helps define the parameters of an efficient patch management lifecycle.
* **Deploy with a minimal Setup:** Installing pointless plugins and functionality makes it more difficult to maintain a strong security posture since an overly complex setup increases the labor-intensive nature of vulnerability monitoring and patching. The attack surface of an application stack is further increased by new software packages, which can create system-level conflicts that may be exploited for potential assaults. Auditing the configuration on a regular basis is crucial as advised to ensure that only needed features and plugins are installed and that the attack surface is minimized by removing any unnecessary functionality.

## Resources

* Forsite CyberSecurity –(<https://foresite.com/blog/owasp-top-10-vulnerable-and-outdated-components/>)
* Real Life Examples of Web Vulnerabilities (OWASP Top 10)(<https://www.horangi.com/blog/real-life-examples-of-web-vulnerabilities>)
* OWASP Top 10 Vulnerable and Outdated Components Explained(<https://www.securityjourney.com/post/owasp-top-10-vulnerable-and-outdated-components-explained>)
* Vulnerable and Outdated Components: OWASP Top 10 #6
* (<https://www.vaadata.com/blog/vulnerable-and-outdated-components-owasp-top-10-6/>)
* OWASP - Top 10 Vulnerabilities and Cyber Threats for 2021(<https://owasp.org/Top10/>)

# **ADDITIONAL HACKER PREVENTION RESOURCES**

**OWASP.org** (Open Web Application Security Project)

A great resource known for their Top 10 list of vulnerabilities and numerous cheat sheets to help guide hacker prevention.

“The Open Web Application Security Project (OWASP) is a [501(c)(3)](http://www.irs.gov/Charities-&-Non-Profits/Charitable-Organizations/Exemption-Requirements-Section-501(c)(3)-Organizations) worldwide not-for-profit charitable organization focused on improving the security of software. Our mission is to make software security [visible,](https://www.owasp.org/index.php/Category:OWASP_Video) so that [individuals and organizations](https://www.owasp.org/index.php/Industry:Citations) are able to make informed decisions.”

**10 Most Common Web Security Vulnerabilities**

<https://www.toptal.com/security/10-most-common-web-security-vulnerabilities>

An article by Gergely Kalman with succinct descriptions to help you avoid common mistakes.

**Test Hacking Sites**

* <http://www.hackertest.net/>
* <https://picoctf.com/>
* <http://hackergames.net/>
* <https://www.hackthissite.org/>

**dvwa.co.uk (Damn Vulnerable Web Application)**

A downloadable PHP/MySQL application that is “damn vulnerable”.

*“Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a class room environment.”*