Table of Content

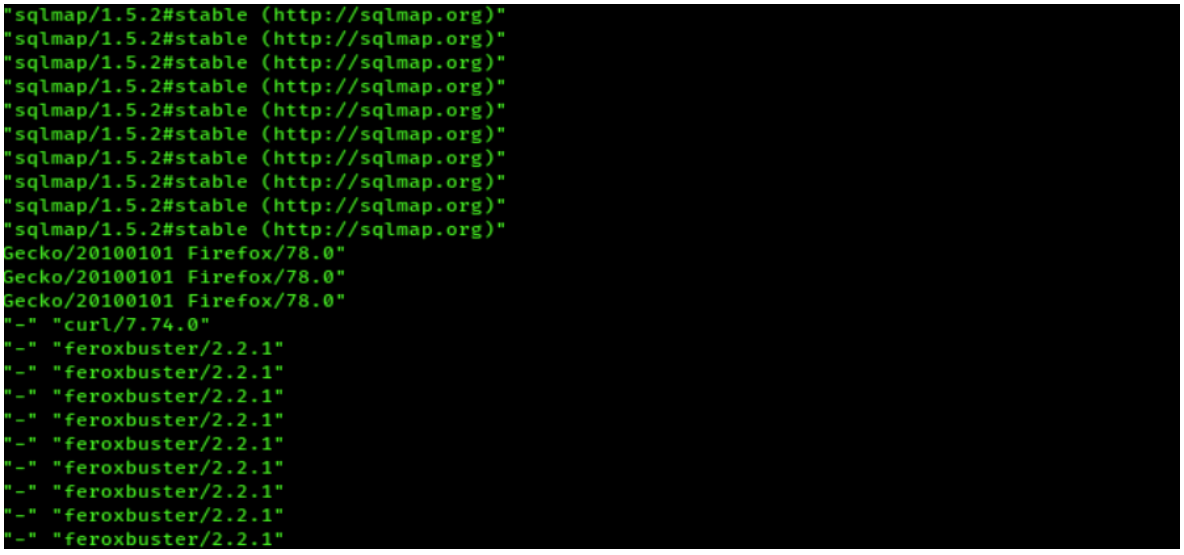
Reconnaissance

Q1: What tools did the attacker use? (Order by the occurrence in the log)

Navigating to the access.log file, and following the User-Agent Headers, we can start our reconnaissance to answer this question.
In order to concentrate on the User-Agent Headers, we could use the following command:

```
awk '{print $(NF -1), $NF}' access.log | less
```

Once the output is simpler to read, we can look at one page at a time to find the tools that were used.



ANSWER: *nmap, hydra, sqlmap, curl, feroxbuster*

Q2: What endpoint was vulnerable to a brute-force attack?

By looking for occurrences of "Hydra", we can easily see the endpoint which the GET requests are being sent to.

```
┌──(kali㉿kali)-[~/Desktop/thm/Juicy_Details]
└─$ grep 'Hydra' access.log
::ffff:192.168.10.5 - - [11/Apr/2021:09:16:27 +0000] "GET /rest/user/login HTTP/1.0" 500 - "-" "Mozilla/5.0 (Hydra)"
::ffff:192.168.10.5 - - [11/Apr/2021:09:16:27 +0000] "GET /rest/user/login HTTP/1.0" 500 - "-" "Mozilla/5.0 (Hydra)"
::ffff:192.168.10.5 - - [11/Apr/2021:09:16:27 +0000] "GET /rest/user/login HTTP/1.0" 500 - "-" "Mozilla/5.0 (Hydra)"
::ffff:192.168.10.5 - - [11/Apr/2021:09:16:27 +0000] "GET /rest/user/login HTTP/1.0" 500 - "-" "Mozilla/5.0 (Hydra)"
::ffff:192.168.10.5 - - [11/Apr/2021:09:16:27 +0000] "GET /rest/user/login HTTP/1.0" 500 - "-" "Mozilla/5.0 (Hydra)"
```

ANSWER: */rest/user/login*

Q3: What endpoint was vulnerable to SQL injection?

Similar to the previous question, by looking for "sqlmap" occurrences we can get the answer to this question.

```
┌──(kali㉿kali)-[~/Desktop/thm/Juicy_Details]
└─$ grep 'sqlmap' access.log
::ffff:192.168.10.5 - - [11/Apr/2021:09:29:14 +0000] "GET /rest/products/search?q=1 HTTP/1.1" 200 - "-" "sqlmap/1.5.2#stable (http://sqlmap.org)"
::ffff:192.168.10.5 - - [11/Apr/2021:09:29:15 +0000] "GET /rest/products/search?q=1&QKqc=7074%20AND%201%3D1%20UNION%20ALL%20SELECT%201%2CNULL%2C%27%3Cscript%3Ealert%28%22XSS%22%29%
3C%2Fscript%3E%27%2Ctable_name%20FROM%20information_schema.tables%20WHERE%202%3E1--%2F%2A%2A%2F%3B%20EXEC%20xp_cmdshell%28%27cat%20..%2F..%2F..%2Fetc%2Fpasswd%27%29%23 HTTP/1.1" 20
```

ANSWER: */rest/products/search*

Q4: What parameter was used for the SQL injection?
Looking closely at the output of our last command, we can see the parameter that was used.

```
"GET /rest/products/search?q=1 HTTP/1.1"
```

ANSWER: *q*

Q4: What endpoint did the attacker try to use to retrieve files? (Include the /)
By researching both "access.log" and "vsftpd.log" we can see and confirm which endpoint was enumerated and later used by the attacker to retrieve the folders.

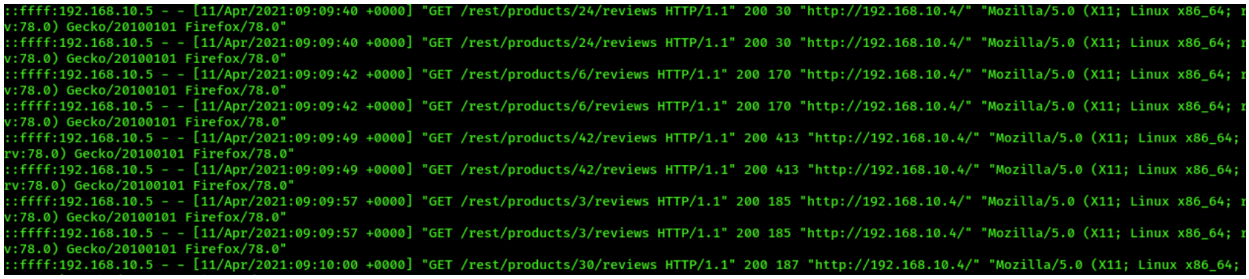
```
┌──(kali㉿kali)-[~/Desktop/thm/Juicy_Details]
└─$ grep 'feroxbuster' access.log
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:33 +0000] "GET /a54372a1404141fe8842ae5c029a00e3 HTTP/1.1" 200 1924 "-" "feroxbuster/2.2.1"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:33 +0000] "GET /3e72ead66df04ca5bff7c9b741883cfbd3044c03e5114f7589804da12c36e5bafa6807b272cf4288ae1316f157b1fab2 HT
TP/1.1" 200 1924 "-" "feroxbuster/2.2.1"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:33 +0000] "GET /api HTTP/1.1" 500 - "-" "feroxbuster/2.2.1"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:33 +0000] "GET /administartion HTTP/1.1" 200 1924 "-" "feroxbuster/2.2.1"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:33 +0000] "GET /login HTTP/1.1" 200 1924 "-" "feroxbuster/2.2.1"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:33 +0000] "GET /admin HTTP/1.1" 200 1924 "-" "feroxbuster/2.2.1"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:33 +0000] "GET /backup HTTP/1.1" 200 1924 "-" "feroxbuster/2.2.1"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:33 +0000] "GET /promotion HTTP/1.1" 200 6586 "-" "feroxbuster/2.2.1"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:33 +0000] "GET /ftp HTTP/1.1" 200 4852 "-" "feroxbuster/2.2.1"

┌──(kali㉿kali)-[~/Desktop/thm/Juicy_Details]
└─$ grep 'ftp' access.log
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:33 +0000] "GET /ftp HTTP/1.1" 200 4852 "-" "feroxbuster/2.2.1"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:40 +0000] "GET /ftp/www-data.bak HTTP/1.1" 403 300 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Fir
efox/78.0"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:43 +0000] "GET /ftp/coupons_2013.md.bak HTTP/1.1" 403 78965 "-" ""Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20
100101 Firefox/78.0"

┌──(kali㉿kali)-[~/Desktop/thm/Juicy_Details]
└─$ grep -i 'download' vsftpd.log
Sun Apr 11 09:35:45 2021 [pid 8154] [ftp] OK DOWNLOAD: Client "::ffff:192.168.10.5", "/www-data.bak", 2602 bytes, 544.81Kbyte/sec
Sun Apr 11 09:36:08 2021 [pid 8154] [ftp] OK DOWNLOAD: Client "::ffff:192.168.10.5", "/coupons_2013.md.bak", 131 bytes, 3.01Kbyte/sec
```

Stolen data

Q1: What section of the website did the attacker use to scrape user email addresses?
Investigating the “access.log” for successful requests, by using the following command -

```
grep '200' access.log | less
```

We can see an interesting section that correlates with where users might have their emails used.



Answer: *product reviews*

Q2: Was their brute-force attack successful? If so, what is the timestamp of the successful login? (Yay/Nay, 11/Apr/2021:09:xx:xx +0000)
Looking for Hydra successful attempts by status code in “access.log”, using the following command -

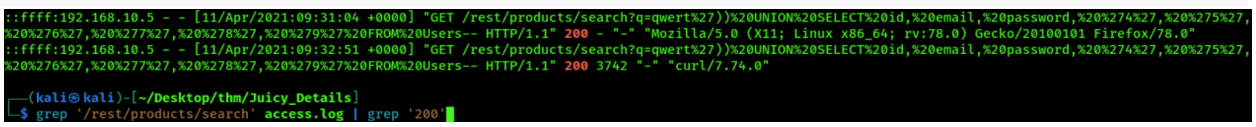
```
grep 'Hydra' access.log | grep '200'
```

We find this -

ANSWER: *Yay, 11/Apr/2021:09:16:31 +0000*

Q3: What user information was the attacker able to retrieve from the endpoint vulnerable to SQL injection?
As we already know, the endpoint vulnerable to SQL injection is /rest/products/search therefore, we can continue investigating the log file to look for successful requests that might contain valuable information.

```
grep '/rest/products/search' access.log | grep '200'
```



ANSWER: *email, password*

Q4: What files did they try to download from the vulnerable endpoint? (endpoint from the previous task, question #5)
That's an easy one! We've already seen these files previously. Do you remember?
Let's take a look at the GETs from the /ftp endpoint one more time.

```
┌──(kali㉿kali)-[~/Desktop/thm/Juicy_Details]
└─$ grep 'ftp' access.log
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:33 +0000] "GET /ftp HTTP/1.1" 200 4852 "-" "feroxbuster/2.2.1"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:40 +0000] "GET /ftp/www-data.bak HTTP/1.1" 403 300 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Fir
efox/78.0"
::ffff:192.168.10.5 - - [11/Apr/2021:09:34:43 +0000] "GET /ftp/coupons_2013.md.bak HTTP/1.1" 403 78965 "-" ""Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20
100101 Firefox/78.0"
```

ANSWER: *coupons_2013.md.bak, www-data.bak*

Q5:What service and account name were used to retrieve files from the previous question? (service, username)

As we already know what service was used, let's go back to our "vsftpd.log" to find out what was the username the attacker used.

```
┌──(kali㉿kali)-[~/Desktop/thm/Juicy_Details]
└─$ cat vsftpd.log
Sun Apr 11 08:13:32 2021 [pid 6335] CONNECT: Client "::ffff:127.0.0.1"
Sun Apr 11 08:13:40 2021 [pid 6334] [anonymous] FAIL LOGIN: Client "::ffff:127.0.0.1"
Sun Apr 11 08:15:09 2021 [pid 6478] CONNECT: Client "::ffff:127.0.0.1"
Sun Apr 11 08:15:14 2021 [pid 6477] [anonymous] FAIL LOGIN: Client "::ffff:127.0.0.1"
Sun Apr 11 08:15:29 2021 [pid 6483] CONNECT: Client "::ffff:127.0.0.1"
Sun Apr 11 08:15:33 2021 [pid 6482] [anonymous] FAIL LOGIN: Client "::ffff:127.0.0.1"
Sun Apr 11 08:15:55 2021 [pid 6529] CONNECT: Client "::ffff:127.0.0.1"
Sun Apr 11 08:15:58 2021 [pid 6526] [ftp] OK LOGIN: Client "::ffff:127.0.0.1", anon password "?"
Sun Apr 11 08:18:04 2021 [pid 6628] CONNECT: Client "::ffff:127.0.0.1"
Sun Apr 11 08:18:07 2021 [pid 6627] [ftp] OK LOGIN: Client "::ffff:127.0.0.1", anon password "ls"
```

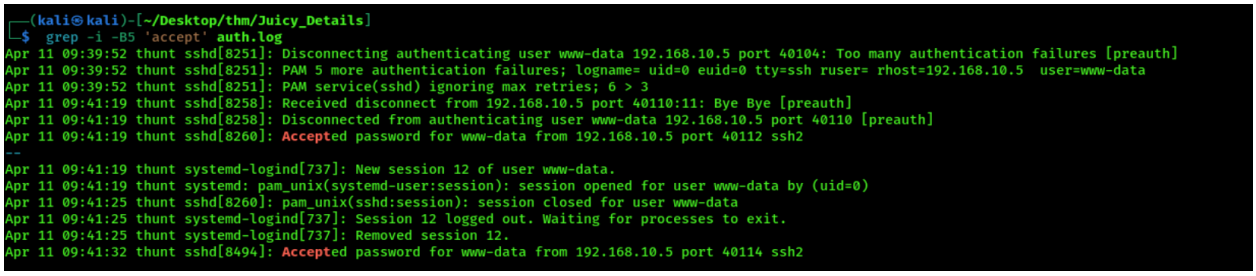
ANSWER: *ftp, anonymous*

Q6: What service and username were used to gain shell access to the server? (service, username)

Let's dive into the "auth.log" and look for a successful connection. Here we can check for successful logins and see the service and the user that were used to gain shell access.

```
grep -i -B5 'accept' auth.log
```

*That username looks familiar, do you remember where we might've seen it earlier?



ANSWER: ssh, *www-data*