

Get unlimited access to the best of Medium for less than \$1/week. [Become a member](#)



TryHackMe | Zeek — Write-up



igor_sec · [Follow](#)

12 min read · Jul 12, 2023

Listen

Share

More

Introduction to hands-on network monitoring and threat detection with Zeek (formerly Bro).

Link: <https://tryhackme.com/room/zeekbro>

Zeek (formerly Bro) is an open-source and commercial network monitoring tool (traffic analyser).

The official description; “Zeek (formerly Bro) is the world’s leading platform for network security monitoring. Flexible, open-source, and powered by defenders.” “Zeek is a passive, open-source network traffic analyser. Many operators use Zeek as a network security monitor (NSM) to support suspicious or malicious activity investigations. Zeek also supports a wide range of traffic analysis tasks beyond the security domain, including performance measurement and troubleshooting.”

The room aims to provide a general network monitoring overview and work with Zeek to investigate captured traffic. This room will expect you to have basic Linux familiarity and Network fundamentals (ports, protocols and traffic data). We suggest completing the “Network Fundamentals” path before starting working in this room.

Task 2: Network Security Monitoring and Zeek

Each exercise has a folder. Ensure you are in the right directory to find the pcap file and accompanying files. Desktop/Exercise-Files/TASK-2

What is the installed Zeek instance version number?

Ans: 4.2.1

```
zeek -v
```

```
ubuntu@ip-10-0-10-10:~$ sudo -s
root@ip-10-0-10-10:/home/ubuntu# zeek -v
zeek version 4.2.1
root@ip-10-0-10-10:/home/ubuntu# zeekctl
Warning: new zeek version detected (run the zeekctl "deploy" command)

Welcome to ZeekControl 2.4.0
```

What is the version of the ZeekControl module?

Ans: 2.4.0

We got the answer from executing the command from the previous question.

Investigate the “sample.pcap” file. What is the number of generated alert files?

Ans: 8

```
zeek -C -r Desktop/Exercise-Files/TASK-2/sample.pcap
```

```
root@ip-10-10-10-10:/home/ubuntu# ls -l
total 72
drwxr-xr-x 3 ubuntu ubuntu 4096 Apr  7  2022 Desktop
drwxr-xr-x 2 ubuntu ubuntu 4096 Feb 27 2022 Documents
drwxr-xr-x 2 ubuntu ubuntu 4096 Apr  7  2022 Downloads
drwxr-xr-x 2 ubuntu ubuntu 4096 Feb 27 2022 Music
drwxr-xr-x 2 ubuntu ubuntu 4096 Feb 27 2022 Pictures
drwxr-xr-x 2 ubuntu ubuntu 4096 Feb 27 2022 Public
drwxr-xr-x 2 ubuntu ubuntu 4096 Feb 27 2022 Templates
drwxr-xr-x 2 ubuntu ubuntu 4096 Feb 27 2022 Videos
-rw-r--r-- 1 root   root   11375 Jun 27 10:29 conn.log
-rw-r--r-- 1 root   root    763 Jun 27 10:29 dhcp.log
-rw-r--r-- 1 root   root   2923 Jun 27 10:29 dns.log
-rw-r--r-- 1 root   root   2532 Jun 27 10:29 ntp.log
-rw-r--r-- 1 root   root    254 Jun 27 10:29 packet_filter.log
-rw-r--r-- 1 root   root    530 Jun 27 10:29 snmp.log
-rw-r--r-- 1 root   root    703 Jun 27 10:29 ssh.log
-rw-r--r-- 1 root   root   1559 Jun 27 10:29 syslog.log
```

Task 3: Zeek Logs

Each exercise has a folder. Ensure you are in the right directory to find the pcap file and accompanying files. Desktop/Exercise-Files/TASK-3

```
zeek -C -r sample.pcap
```

```
ls -l
```

```
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-3# zeek -C -r sample.pcap
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-3# ls -l
total 444
-rwxr-xr-x 1 ubuntu ubuntu    46 Apr  3  2022 clear-logs.sh
-rw-r--r-- 1 root   root  11381 Jun 27 10:45 conn.log
-rw-r--r-- 1 root   root    759 Jun 27 10:45 dhcp.log
-rw-r--r-- 1 root   root   2923 Jun 27 10:45 dns.log
-rw-r--r-- 1 root   root   2530 Jun 27 10:45 ntp.log
-rw-r--r-- 1 root   root    254 Jun 27 10:45 packet_filter.log
-rw-r--r-- 1 ubuntu ubuntu 407510 Mar  3  2017 sample.pcap
-rw-r--r-- 1 root   root    530 Jun 27 10:45 snmp.log
-rw-r--r-- 1 root   root    703 Jun 27 10:45 ssh.log
-rw-r--r-- 1 root   root   1561 Jun 27 10:45 syslog.log
```

Investigate the sample.pcap file. Investigate the dhcp.log file. What is the available hostname?

Ans: Microknoppix

```
cat dhcp.log
cat dhcp.log | zeek-cut host_name
```

```
root@lp-1:~/Desktop/Exercise-Files/TASK-3# cat dhcp.log
#separator \x09
#set_separator ,
#empty_field  (empty)
#unset_field -
#path_dhcp
#open 2023-06-27-10-45-33
#fields ts    uids   client_addr   server_addr   mac     host_name      client_fqdn    domain  requested_addr  assigned_addr  lease_time    client_message  server_messa
ge   msg_types   duration
#types time  set[string]   addr   string string string  addr   interval   string string vector[string]  interval
1488571051.699148  CcvtJunONeaxQuKsf,C1vChpDUOcWHSrlB  192.168.30.11  192.168.30.1  00:21:70:e9:bb:47  Microknoppix -  webernetz.net  192.168.30.11  192.168.30.11  86400.000000  -  DISCOVER,OFFER,REQUEST,ACK  0.022753
```

```
root@ip-1:~/Desktop/Exercise-Files/TASK-3# cat dhcp.log | zeek-cut host_name
Microknoppix
Microknoppix
```

Investigate the dns.log file. What is the number of unique DNS queries?

Ans: 2

```
cat dns.log
cat dns.log | zeek-cut query
```

```
root@lp-1:~/Desktop/Exercise-Files/TASK-3# cat dns.log
#separator \x09
#set_separator ,
#empty_field  (empty)
#unset_field -
#path_dns
#open 2023-06-27-10-45-33
#fields ts    uid    id.orig_h    id.orig_p    id.resp_h    id.resp_p    proto  trans_id    rtt    query  qclass  qclass_name  qtype  qtype_name  rcod
e   rcode_name  AA    TC    RD    RA    Z    answers  TTLs  rejected
#types time  string  addr  port  addr  port  enum  count  interval  string  count  string  count  string  count  string  bool  bool  bool  bool
t  vector[string]  vector[interval]  bool
1488571051.943258  Cs0eF82rv027cYvVbg  192.168.121.2  51153  192.168.120.22  53  udp  46282  0.001263  blog.webernetz.net  1  C_INTERNET  1  A
0  NOERROR F  F    T    T    0  5.35.226.136  18180.000000  F
```

```
root@ip-10-0-10-11:~/Desktop/Exercise-Files/TASK-3# cat dns.log | zeek-cut query
blog.webernetz.net
ip.webernetz.net
```

Investigate the conn.log file. What is the longest connection duration?

Ans: 332.319364

```
cat conn.log
```

```
root@ip-10-0-10-11:~/Desktop/Exercise-Files/TASK-3# cat conn.log
separator '\x09'
• et_separator ,
empty_field  (empty)
#unset_field
#path conn
#open 2023-06-27-10-45-33
#fields ts      uid      id.orig_h      id.orig_p      id.resp_h      id.resp_p      proto      service duration      orig_bytes      resp_bytes      conn_state      local_orig
#types time      string    addr        port        enum       string    interval      count      count      string   bool      count      string   count      count      count      count      set[st]
```

We will use sort from the highest (-r) according to string numerical values (-n) then pipe the result with head command to show the first value.

```
cat conn.log | zeek-cut duration | sort -n -r | head -n 1
```

```
root@ip-10-0-10-11:~/Desktop/Exercise-Files/TASK-3# cat conn.log | zeek-cut duration | sort -n -r | head -n 1
332.319364
```

Task 5: Zeek Signatures

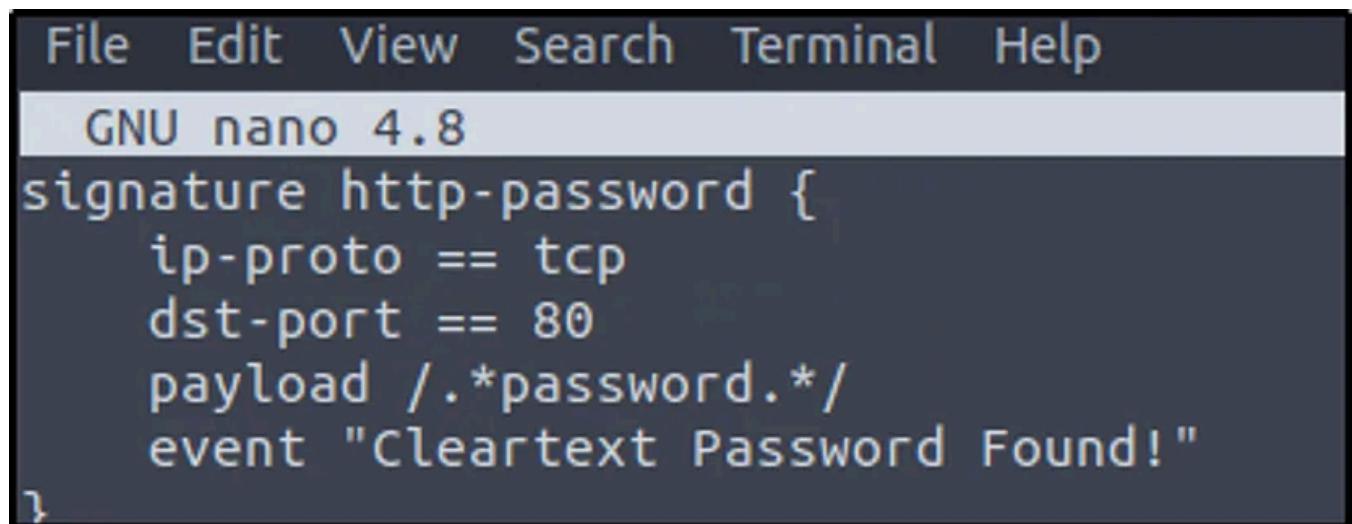
Each exercise has a folder. Ensure you are in the right directory to find the pcap file and accompanying files. Desktop/Exercise-Files/TASK-5

Investigate the http.pcap file. Create the HTTP signature shown in the task and investigate the pcap. What is the source IP of the first event?

Ans: 10.10.57.178

Let's create the signature.

```
nano http-password.sig
```



The screenshot shows a terminal window with the title "GNU nano 4.8". The content of the editor is a custom HTTP signature:

```
signature http-password {
    ip-proto == tcp
    dst-port == 80
    payload /.*password.*/
    event "Cleartext Password Found!"
}
```

The signature will match when a “password” phrase is detected in the packet payload.

Let's read the pcap file with the signature created.

```
zeek -C -r http.pcap -s http.password.sig
```

An alert will be generated and two log files will be created, “signatures.log”, and “notice.log”

```
root@ip-10-10-57-178:~/home/ubuntu/Desktop/Exercise-Files/TASK-5/http# zeek -C -r http.pcap -s http-password.sig
root@ip-10-10-57-178:~/home/ubuntu/Desktop/Exercise-Files/TASK-5/http# ls
clear-logs.sh  conn.log  files.log  http-password.sig  http.log  http.pcap  notice.log  packet_filter.log  signatures.log
```

From the “signatures/log”, we will use “zeek-cut” to select the field “src_addr”, then sort it in reverse. This will give use the first source IP address.

```
cat signatures.log | zeek-cut src_addr | sort -r
```

```
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-5/http# cat signatures.log | zeek-cut src_addr | sort -r
10.10.57.178
10.10.57.178
```

What is the source port of the second event?

Ans: 38712

We will modify the command above to select the field name “src_port”. The source port numbers will be displayed.

```
cat signatures.log | zeek-cut src_port | sort
```

```
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-5/http# cat notice.log | zeek-cut id.orig_p | sort
38706
38712
```

Investigate the conn.log.

What is the total number of the sent and received packets from source port 38706?

Ans: 20

We will select the field names “id.orig_p” and “orig_pkts resp_pkts” and grep the source port. The two values are then added to get the total number of bytes sent and received from the source port.

```
cat conn.log
```

```
cat conn.log |zeek-cut id.orig_p orig_pkts resp_pkts | grep 38706
```

```
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-5/http# cat conn.log
#separator "\x09"
#delim "\t"
#empty ""
unset_field ''
push conn
pop conn
#date "2023-06-27-11:58:44"
#fields "id.orig_h id.orig_p id.resp_h id.resp_p proto service duration orig_bytes resp_bytes conn_state local_orig local_resp missed_bytes history orig_pkts orig_tp_bytes resp_pkts resp_tp_bytes"
#times time string addr port addr port enum string interval count count string bool bool count string count count count count setstring
```

```
root@ip-10-10-170-196:~/home/ubuntu/Desktop/Exercise-Files/TASK-5/http# cat conn.log |zeek-cut id.orig_p orig_pkts resp_pkts | grep 38766
38766 11 9
```

Create the global rule shown in the task and investigate the ftp.pcap file.

Let's create first the global rule. The rule will match any payload when an FTP username is used to authenticate and any payload attempting to brute-force the FTP server.

```
nano ftp-bruteforce.sig
```

```
GNU nano 4.8
signature ftp-username {
    ip-proto == tcp
    ftp ./.*USER.*/
    event "FTP Username Input Found!"
}

signature ftp-brute {
    ip-proto == tcp
    payload ./.*530.*Login.*incorrect.*/
    event "FTP Brute-force Attempt!"
}
```

Let's now read the pcap file with the signature applied.

```
zeek -C -r ftp.pcap -s ftp-bruteforce.sig
```

Similarly, an alert will be generated and two log files will be created.

```
root@ip-10-10-170-196:~/home/ubuntu/Desktop/Exercise-Files/TASK-5/ftp# zeek -C -r ftp.pcap -s ftp-bruteforce.sig
root@ip-10-10-170-196:~/home/ubuntu/Desktop/Exercise-Files/TASK-5/ftp# ls
clear-logs.sh  conn.log  ftp-bruteforce.sig  ftp.pcap  notice.log  packet_filter.log  signatures.log  weird.log
root@ip-10-10-170-196:~/home/ubuntu/Desktop/Exercise-Files/TASK-5/ftp#
```

```
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-5/ftp# cat notice.log | head
#separator \x09
#set_separator ,
#empty_field  (empty)
#unset_field -
#path  notice
#open 2023-06-27-12-24-18
#fields ts      uid      id.orig_h      id.orig_p      id.resp_h      id.resp_p      fuid      file_mime_type  file_desc
mote_location.region  remote_location.city  remote_location.latitude  remote_location.longitude
#types time      string     addr      port      addr      port      string     string     enum      enum      string     string     addr      addr
```

Investigate the notice.log. What is the number of unique events?

Ans:1413

We will be counting the number of unique events based off from the “uid” field.

```
cat notice.log |zeek-cut uid | sort | uniq | wc -l
```

```
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-5/ftp# cat notice.log |zeek-cut uid | sort | uniq | wc -l
1413
```

What is the number of ftp-brute signature matches?

Ans: 1410

If you will notice, I always read, “cat”, the log files. That is simply to identify the field names correctly.

```
cat signatures.log | head -n 20
```

```
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-5/ftp# cat signatures.log | head -n 20
#separator \x09
#set_separator ,
#empty_field  (empty)
#unset_field -
#path  signatures
#open 2023-06-27-12-24-18
#fields ts      uid      src_addr      src_port      dst_addr      dst_port      note      sig_id      event_msg      sub_msg      sig_count      host_count
#types time      string     addr      port      addr      port      enum      string      string      count
1024380731.015090 CtTQ5g4LH6NAbXLI64  10.121.70.151  21  10.234.125.254  2217  Signatures::Sensitive_Signature ftp-brute  10.121.70.151: F
1024380731.043248 CfqqOM2ceBj3Xbquz1  10.121.70.151  21  10.234.125.254  2220  Signatures::Sensitive_Signature ftp-brute  10.121.70.151: F
1024380731.110895 CTGR5D2zoNwMIHnsPb  10.121.70.151  21  10.234.125.254  2222  Signatures::Sensitive_Signature ftp-brute  10.121.70.151: F
1024380731.122359 CtdREG3LMRnB3cht09  10.121.70.151  21  10.234.125.254  2221  Signatures::Sensitive_Signature ftp-brute  10.121.70.151: F
1024380731.148231 Coakjp2rLVLPyNsxah  10.121.70.151  21  10.234.125.254  2223  Signatures::Sensitive_Signature ftp-brute  10.121.70.151: F
1024380731.218890 CngRrl3yMME1n9Ky1e  10.234.125.254  2228  10.121.70.151  21  Signatures::Sensitive_Signature ftp-username  10.234.125.254: F
1024380731.251481 CcelR8111gsgCR7v08  10.121.70.151  21  10.234.125.254  2224  Signatures::Sensitive_Signature ftp-brute  10.121.70.151: F
1024380731.267148 C7yle03DQ5mtz6U88  10.234.125.254  2225  10.121.70.151  21  Signatures::Sensitive_Signature ftp-username  10.234.125.254: F
```

This time, we will investigate the “signatures.log”. From the rule we created, if there was a brute-force attempt, it will create an event, “FTP Brute-force Attempt!”, which is logged in “signatures.log”. This event will be logged as “ftp-brute”.

Notice that in the field name “sig_id”, it contains the event we are after. So we will select that field and the count the lines generated.

```
cat signatures.log | zeek-cut sig_id | grep "ftp-brute" | wc -l
```

```
root@ip-10-0-10-10:~/Desktop/Exercise-Files/TASK-5/ftp# cat signatures.log | zeek-cut sig_id | grep "ftp-brute" | wc -l
1410
```

Task 6: Zeek Scripts | Fundamentals

Each exercise has a folder. Ensure you are in the right directory to find the pcap file and accompanying files. Desktop/Exercise-Files/TASK-6

Investigate the smallFlows.pcap file. Investigate the dhcp.log file. What is the domain value of the “vinlap01” host?

Ans: astaro_vineyard

```
zeek -C -r smallFlows.pcap
cat dhcp.log
cat dhcp.log | zeek-cut domain
```

```
root@ip-10-0-10-10:~/Desktop/Exercise-Files/TASK-6/smallflow# zeek -C -r smallFlows.pcap
root@ip-10-0-10-10:~/Desktop/Exercise-Files/TASK-6/smallflow# ls
clear-logs.sh  conn.log  dhcp-hostname.zeek  dhcp.log  dns.log  dpd.log  files.log  http.log  packet_filter.log  smallFlows.pcap  snmp.log  ssl.log  weird.log  x509.log
```

```
root@ip-10-0-10-10:~/Desktop/Exercise-Files/TASK-6/smallflow# cat dhcp.log | zeek-cut domain
-
astaro_vineyard
```

Investigate the bigFlows.pcap file. Investigate the dhcp.log file. What is the number of identified unique hostnames?

Ans: 17

```
zeek -C -r bigFlows.pcap
cat dhcp.log | zeek-cut host_name | sort | uniq
cat dhcp.log | zeek-cut host_name | sort | uniq | wc -l
```

```
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-6/bigflow# zeek -C -r bigFlows.pcap
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-6/bigflow# ls
bigFlows.pcap  conn.log  dhcp-hostname.zeek  dns.log  files.log  kerberos.log  ntp.log  packet_filter.log  smb_files.log  snmp.log  ssl.log  weird.log
clear-logs.sh  dce_rpc.log  dhcp.log  dpd.log  http.log  ntln.log  ocsp.log  sip.log  smb_mapping.log  ssh.log  syslog.log  xs09.log
```

```
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-6/bigflow# cat dhcp.log | head
#separator '\x09'
#set_separator ''
#empty_value '(empty)'
#empty_field ''
#path dhcp
#open 2023-06-27-13-00-21
#fields ts      uids    client_addr   server_addr   mac      host_name    client_fqdn   domain   requested_addr  assigned_addr  lease_time   client_message  server_message  msg_types      duration
#types tms     set[string]  addr       string        string    string      string      addr      string      string  vector[string]  interval      -
#types tms     set[string]  addr       string        string    string      string      addr      string      string  vector[string]  interval      -
1361916156.615988  CjLmwIcIw4zJWjn5i,CqnJN43a5uy14MgR52  172.16.133.24  .        00:21:70:67:69:d3  J0T115  .        jaalam.net  .        .        .        .        .        .        INFORM,ACK  0.000142
1361916159.858464  CQWChy4awvoRRw3xE7  172.16.133.38  .        00:99:fb:38:0c:da  m30-sqdesk  .        .        .        .        .        .        .        REQUEST,REQUEST  11.007568
```

```
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-6/bigflow# cat dhcp.log | zeek-cut host_name | sort | uniq
-
JDT081
JDT094
JDT096
JDT100
JDT107
JDT115
JDT120
JDT123
JDT131
JDT134
JDT153
JDT168
JDT80
JDT91
JDT95
JLT108
m30-sqdesk
```

When we include the command to count the lines, the result is 18. But in the image above the first line is empty so the correct answer is 17.

Investigate the dhcp.log file. What is the identified domain value?

Ans: jaalam.net

```
cat dhcp.log | zeek-cut domain | sort | uniq
```

```
root@ip-10-10-10-10:/home/ubuntu/Desktop/Exercise-Files/TASK-6/bigflow# cat dhcp.log | zeek-cut domain | sort | uniq
-
jaalam.net
```

Investigate the dns.log file. What is the number of unique queries?

Ans: 1109

```
cat dns.log| head
```

```
root@ip-10-0-10-11:~/Desktop/Exercise-Files/TASK-6/bigflow# cat dns.log| head
#separator {x09}
#set_separator ,
#empty_field (empty)
#unset_field -
#path dns
#open 2023-06-27-13-19-07
#fields ts      uid      id.orig_h      id.orig_p      id.resp_h      id.resp_p      proto     trans_id      rtt      query      qclass      qclass_name
#types time    string   addr       port      enum      count      interval    string      count      string      count      string      bool
1361916156.058887  CvfyUa4K0unsac7lG1  172.16.133.6  63229  8.8.8.8 53  udp      36336 - 45.66.120.96.in-addr.arpa  1
1361916156.322444  CnYaaY174pQHJCyZy  172.16.133.6  63229  8.8.4.4 53  udp      36336 - 45.66.120.96.in-addr.arpa  1
```

Recall from CLI Kung-fu the command `grep -v -e 'test1' -e 'test2'`, which display lines that don't match one or both “test1” and “test2” strings. The hint also provided us this, “`grep -v -e '*' -e '-'`“.

The hint says that there are two values that we should not include in our result. But what if in other situations there could be more values?

So to determine what are the special characters from a file or logs, we can use this command. We will read from the “dns.log” for example.

```
cat dns.log | zeek-cut query | grep -oP "^\w\s+$" | sort -u
```

This command includes a grep command that only output (-o) the matched special character and ‘-P’ option to enable Perl-compatible regular expresssions. The regex expression matches any individual special character. “sort -u” sorts the output in alphabetical order (sort) and ‘-u’ option ensures that duplicate characters are removed, so each special character appears only once in the output.

```
root@ip-10-0-10-11:~/Desktop/Exercise-Files/TASK-6/bigflow# cat dns.log | zeek-cut query | grep -oP "^\w\s+$" | sort -u
*
```

So we know now what special characters not to include in our output.

```
cat dns.log | zeek-cut query |grep -v -e '*' -e '-' | sort | uniq| wc -l
```

```
root@lp-1:~/Desktop/Exercise-Files/TASK-6/bigflow# cat dns.log | zeek-cut query |grep -v -e '*' -e '-' | sort | uniq| wc -l
1109
```

Task 7: Zeek Scripts | Scripts and Signatures

Each exercise has a folder. Ensure you are in the right directory to find the pcap file and accompanying files. Desktop/Exercise-Files/TASK-7

Go to folder TASK-7/101.

Investigate the sample.pcap file with 103.zeek script. Investigate the terminal output. What is the number of the detected new connections?

Ans: 87

let's run the sample.pcap with the script.

```
zeek -C -r sample.pcap 103.zeek
```

We will “cat” the “conn.log” then select the “uid” field, sort the results, and pipe with “uniq” to avoid duplication, and then finally count the lines.

```
cat conn.log | zeek-cut uid | sort | uniq | wc -l
```

```
root@ip-10-0-10-75:~/Desktop/Exercise-Files/TASK-7/101# cat conn.log | head
#separator '\x09'
#set_separator ','
#empty_field '(empty)'
#unset_field '-'
#path conn
#open 2023-06-28-00-42-26
#fields ts uid id.orig_h id.orig_p id.resp_h id.resp_p proto service duration
tes tunnel_parents
#types time string addr port enum string interval count count string bool
1488571051.943250 CnRssBL3MZ8FZTlb 192.168.121.2 51153 192.168.120.22 53 udp dns 0.001265
1488571038.380901 CWZOCm3El3Hj2DUtYc 192.168.121.10 50080 192.168.120.10 514 udp - 0.000505
root@ip-10-0-10-75:~/Desktop/Exercise-Files/TASK-7/101# cat conn.log | zeek-cut uid | sort | uniq | wc -l
87
```

Go to folder TASK-7/201.

Investigate the ftp.pcap file with ftp-admin.sig signature and 201.zeek script.

Investigate the signatures.log file. What is the number of signature hits?

Ans: 1401

```
zeek -C -r ftp.pcap 201.zeek -s ftp-admin.sig | wc -l
```

```
root@ip-10-0-10-75:~/Desktop/Exercise-Files/TASK-7/201# zeek -C -r ftp.pcap 201.zeek -s ftp-admin.sig | wc -l
1401
```

Investigate the signatures.log file. What is the total number of “administrator” username detections?

Ans: 731

```
cat signatures.log | zeek-cut sub_msg | grep "USER administrator" | wc -l
```

```
root@ip-10-0-10-75:~/Desktop/Exercise-Files/TASK-7/201# cat signatures.log | head
#separator '\x09'
#set_separator ','
#empty_field '(empty)'
#unset_field '-'
#path signatures
#open 2023-06-28-00-51-58
#fields ts uid src_addr src_port dst_addr dst_port note sig_id event_msg sub_msg sig_count host_count
#types time string addr port enum string string string count count string bool
1024380731.210890 Cu0abR1suhsuxUEcH 10.234.125.254 2228 10.121.70.151 21 Signatures::Sensitive_Signature ftp-admin 10.234.125.254: FTP Username Input Found! USER admin -
1024380731.267148 CPoERF2X1aPMoupol 10.234.125.254 2228 10.121.70.151 21 Signatures::Sensitive_Signature ftp-admin 10.234.125.254: FTP Username Input Found! USER admin -
```

```
root@ip-10-0-10-75:~/Desktop/Exercise-Files/TASK-7/201# cat signatures.log | zeek-cut sub_msg | grep "USER administrator" | wc -l
731
```

Investigate the ftp.pcap file with all local scripts, and investigate the loaded_scripts.log file. What is the total number of loaded scripts?

Ans: 498

```
zeek -C -r ftp.pcap 201.zeek local
```

```
root@ip-10-0-75-111:/home/ubuntu/Desktop/Exercise-Files/TASK-7/201# zeek -C -r ftp.pcap 201.zeek local
WARNING: No Site::local_nets have been defined. It's usually a good idea to define your local networks.
```

Don't worry if you get a warning.

```
cat loaded_scripts.log | zeek-cut name | wc -l
```

```
root@ip-10-0-75-111:/home/ubuntu/Desktop/Exercise-Files/TASK-7/201# cat loaded_scripts.log | zeek-cut name | wc -l
498
```

Go to folder TASK-7/202.

Investigate the `ftp-brute.pcap` file with

`"/opt/zeek/share/zeek/policy/protocols/ftp/detect-bruteforcing.zeek"` script.

Investigate the `notice.log` file. What is the total number of brute-force detections?

Ans: 2

```
zeek -C -r ftp-brute.pcap /opt/zeek/share/zeek/policy/protocols/ftp/detect-bruteforcing.zeek
```

```
< >
```

```
cat notice.log | zeek-cut note | grep "FTP::Bruteforcing" | wc -l
```

```
root@ip-10-0-75-111:/home/ubuntu/Desktop/Exercise-Files/TASK-7/202# zeek -C -r ftp-brute.pcap /opt/zeek/share/zeek/policy/protocols/ftp/detect-bruteforcing.zeek
root@ip-10-0-75-111:/home/ubuntu/Desktop/Exercise-Files/TASK-7/202# ls
clear-logs.sh  conn.log  ftp-brute.pcap  ftp.pcap  notice.log  packet_filter.log  weird.log
root@ip-10-0-75-111:/home/ubuntu/Desktop/Exercise-Files/TASK-7/202# cat notice.log | head
#separator \x09
#set_separator ,
#empty_field   (empty)
#unset_field   -
#path  notice
#open 2023-06-28-01-16-13
#fields ts      uid      id.orig_h    id.orig_p    id.resp_h    id.resp_p    fuid      file_mime_type  file_desc      proto      note      msg      sub      src
note_location.region  remote_location.city  remote_location.latitude  remote_location.longitude
#types time      string     addr       port       string     string     string     enum       enum       string     string     addr       addr       port       count     string     set[enum]
1024380732.223481    -          -          -          -          -          -          -          -          -          FTP::Bruteforcing  10.234.125.254 had 20 failed logins on 1 F
0                     -          -          -          -          -          -          -          -          -          FTP::Bruteforcing  192.168.56.1 had 20 failed logins on 1 F
0                     -          -          -          -          -          -          -          -          -          FTP::Bruteforcing  192.168.56.1 had 20 failed logins on 1 F
0                     -          -          -          -          -          -          -          -          -          FTP::Bruteforcing  192.168.56.1 had 20 failed logins on 1 F
root@ip-10-0-75-111:/home/ubuntu/Desktop/Exercise-Files/TASK-7/202# cat notice.log | zeek-cut note | grep "FTP::Bruteforcing" | wc -l
2
```

Task 8: Zeek Scripts | Frameworks

Each exercise has a folder. Ensure you are in the right directory to find the pcap file and accompanying files. Desktop/Exercise-Files/TASK-8

Investigate the case1.pcap file with intelligence-demo.zeek script. Investigate the intel.log file. Look at the second finding, where was the intel info found?

Ans: IN_HOST_HEADER

```
zeek -C -r case1.pcap intelligence-demo.zeek
cat intel.log | head
cat intel.log | zeek-cut seen.where
```

```
root@ip-10-0-10-11:~/Desktop/Exercise-Files/TASK-8# zeek -C -r case1.pcap intelligence-demo.zeek
root@ip-10-0-10-11:~/Desktop/Exercise-Files/TASK-8# ls
case1.pcap  clear-logs.sh  conn.log  dhcp.log  dns.log  file-extract-demo.zeek  files.log  hash-demo.zeek  http.log  intel.log  intelligence-demo.zeek  packet_filter.log  pe.log
root@ip-10-0-10-11:~/Desktop/Exercise-Files/TASK-8# cat intel.log | head
#separator '\x09'
#set_separator ','
#empty_field  '(empty)'
#unset_field  ''
#path  intel
#open  2023-06-28-01-39-28
#fields ts      uid      id.orig_h      id.orig_p      id.resp_h      id.resp_p      seen.indicator      seen.indicator_type      seen.where      seen.node      matched sources  fuld
#types time    string    addr      port      addr      port      string    enum      enum      string      set[enum]      set[string]      string      string
1561667898.779213  CVY8CJCJnoyc0vd11  10.6.27.102  53778  10.6.27.1  53  smart-fax.com  Intel::DOMAIN  DNS::IN_REQUEST  zeek  Intel::DOMAIN  TASK-8-Demo
1561667898.911759  CVy5O24L50dSXQFin6  10.6.27.102  49162  107.188.50.162  80  smart-fax.com  Intel::DOMAIN  HTTP::IN_HOST_HEADER  zeek  Intel::DOMAIN  TASK-8
```

```
root@ip-10-0-10-11:~/Desktop/Exercise-Files/TASK-8# cat intel.log | zeek-cut seen.where
DNS::IN_REQUEST
HTTP::IN_HOST_HEADER
```

Investigate the http.log file. What is the name of the downloaded .exe file?

Ans: knr.exe

```
cat intel.log | head
cat http.log | zeek-cut uri | grep '\.exe$'
```

“grep ‘\.exe\$’” searches for lines that contain the “.exe” extension at the end of the line. The backslash (\) before the dot (.) is used to escape it, so that it matches a literal dot. The “\$” matches the end of the line.

```
root@ip-10-10-75-111:/home/ubuntu/Desktop/Exercise-Files/TASK-8# cat http.log | head
#separator \x09
#set_separator ,
#empty_field (empty)
#unset_field -
#path http
#open 2023-06-28-01-39-28
#fields ts uid id.orig_h id.orig_p id.resp_h id.resp_p trans_depth method host url referrer version user_agent
#types time string addr port string string string string string string string string string count count count string count string
ctor[string] vector[string]
1561667874.713411 CuCWXa2J3Ccxr3zVbs 10.6.27.102 49157 23.63.254.163 80 1 GET www.msftncsl.com /ncsi.txt - 1.1
Fpgan59p6uvNzLFja - text/plain
1561667889.643717 CKEkhGxSgeeVnbPjL 10.6.27.102 49159 107.180.50.162 80 1 GET smart-fax.com /Documents/Invoice&MSO-Request.doc
0 OK - (empty) -

```

```
root@ip-10-10-75-111:/home/ubuntu/Desktop/Exercise-Files/TASK-8# cat http.log | zeek-cut uri | grep '\.exe$' /knr.exe
```

Investigate the case1.pcap file with hash-demo.zeek script. Investigate the files.log file. What is the MD5 hash of the downloaded .exe file?

Ans: cc28e40b46237ab6d5282199ef78c464

```
zeek -C -r case1.pcap hash-demo.zeek
cat files.log | head
cat files.log | zeek-cut mime_type md5
```

```
root@ip-10-10-75-111:/home/ubuntu/Desktop/Exercise-Files/TASK-8# zeek -C -r case1.pcap hash-demo.zeek
root@ip-10-10-75-111:/home/ubuntu/Desktop/Exercise-Files/TASK-8# ls
case1.pcap clear-logs.sh conn.log dhcp.log dns.log file-extract-demo.zeek files.log hash-demo.zeek http.log int
root@ip-10-10-75-184:/home/ubuntu/Desktop/Exercise-Files/TASK-8# cat files.log | head
#separator \x09
#set_separator ,
#empty_field (empty)
#unset_field -
#path files
#open 2023-06-28-01-53-27
#fields ts fid tx_hosts rx_hosts conn_uids source depth analyzers mime_type

```

We know that it is an executable file so it should be the third md5 value.

```
root@ip-10-10-75-111:/home/ubuntu/Desktop/Exercise-Files/TASK-8# cat files.log | zeek-cut mime_type md5
text/plain cd5a4d3fd5bffc16bf959ef75cf37bc
application/msword b5243ec1df7d1d5304189e7db2744128
application/x-dosexec cc28e40b46237ab6d5282199ef78c464
```

We can also find the correlation of the “.exe” file with the other log files.

First we need a common value of the “.exe” file.

```
cat files.log | zeek-cut fid conn_uids tx_hosts rx_hosts mime_type extracted |
```

```
root@elp-15-01-01-01:/home/ubuntu/Desktop/Exercise-Files/TASK-8# cat files.log | zeek-cut fuid conn_uids tx_hosts rx_hosts mime_type extracted | nl
 1 FpganS9p6uvNzLfja CfmyNM3BzQTMdfkPm6 23.63.254.163 10.6.27.102 text/plain -
 2 FBSo2Hcauv7vpQ8y3 CHPHO63gE5QGtwHT1l 107.180.50.162 10.6.27.102 application/msword -
 3 F0qls3MpIJKpvXaEl CVsnuagu2ZhLnXy91 107.180.50.162 10.6.27.102 application/x-dosexec -
```

We will choose the value of the field “conn_uids”.

So we got the “conn_uids” value of the “.exe” file. Now we will extract all values in the current directory that correlates to the “.exe” file.

```
grep -rin CVsnuagu2ZhLnXy91 * | column -t | nl | less -S
```

- `grep -rin CVsnuagu2ZhLnXy91 *` : This searches for the pattern "CVsnuagu2ZhLnXy91" recursively (-r) in all files (*) within the current directory. The -i option is used for case-insensitive matching, and the -n option displays line numbers.
- `column -t` : This formats the output into multiple columns for better readability. It assumes tabular data with whitespace as the delimiter.
- `nl` : This adds line numbers to the output.
- `less -S` : This command opens the output in the less pager, which allows scrolling through the content. The -S option disables line wrapping for better readability.

From the result, the “.exe” file is found in three logs. We see other information that relates to the file, and in “files.log” we see its MD5 value.

```
1 conn.log:4:1561667898.832680 CVsnuagu2ZhLnXy91 10.6.27.102 49582 187.180.50.162 80 tcp http 169.251.119 307 958721 RSTO - - 0.498764 2437120 6 SHA256: 359 14679 TSI 980765 cc28e40b46237ab6d5182199ef78c464
2 files.log:11:1561667899.000006 F0qls3MpIJKpvXaEl 107.180.50.162 10.6.27.102 CVsnuagu2ZhLnXy91 HTTP 0 PE,MDS,SHA1 smart-fax.com /khr.exe 1.1 Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; MSIE8)
3 http.log:11:1561667899.911759 CVsnuagu2ZhLnXy91 10.6.27.102 49582 187.180.50.162 80 1 GET
```

Investigate the `case1.pcap` file with `file-extract-demo.zeek` script. Investigate the “extract_files” folder. Review the contents of the text file. What is written in the file?

Ans: Microsoft NCSI

```
zeek -C -r case1.pcap file-extract-demo.zeek
file * | nl
```

```
cat extract-1561667874.743959-HTTP-Fpgan59p6uvNzLFja
```

```
root@lp-0:~# ./home/ubuntu/Desktop/Exercise-Files/TASK-8# zeek -C -r case1.pcap file-extract-demo.zeek
root@lp-0:~# ./home/ubuntu/Desktop/Exercise-Files/TASK-8# ls
case1.pcap  clear-logs.sh  conn.log  dhcp.log  dns.log  extract_files  file-extract-demo.zeek  files.log  hash-demo.zeek  http.log  intel.log  intelligence-dem
root@lp-0:~# ./home/ubuntu/Desktop/Exercise-Files/TASK-8# cd extract_files/
root@lp-0:~/Desktop/Exercise-Files/TASK-8# extract_files# ls
extract-1561667874.743959-HTTP-Fpgan59p6uvNzLfja  extract-1561667899.703239-HTTP-FB5o2Hcauv7vpQ8y3  extract-1561667899.060086-HTTP-F0ghls3WpIjKpvXaEl
root@lp-0:~/Desktop/Exercise-Files/TASK-8# extract_files# file * | nl
1 extract-1561667874.743959-HTTP-Fpgan59p6uvNzLfja: ASCII text, with no line terminators
2 extract-1561667899.703239-HTTP-FB5o2Hcauv7vpQ8y3: Composite Document File V2 Document, Little Endian, Os: Windows, Version 6.3, Code page: 1252, Template
ate Time/Date: Thu Jun 27 18:24:08 2019, Last Saved Time/Date: Thu Jun 27 18:24:08 2019, Number of Pages: 1, Number of Words: 0, Number of Characters: 1, Secu
3 extract-1561667899.060086-HTTP-F0ghls3WpIjKpvXaEl: PE32 executable (GUI) Intel 80386, for MS Windows
root@lp-0:~/Desktop/Exercise-Files/TASK-8# extract_files# cat extract-1561667874.743959-HTTP-Fpgan59p6uvNzLfja
Microsoft NCSC root@lp-0:~/Desktop/Exercise-Files/TASK-8# extract_files# cat extract-1561667874.743959-HTTP-Fpgan59p6uvNzLfja
```

Task 9: Zeek Scripts | Packages

Each exercise has a folder. Ensure you are in the right directory to find the pcap file and accompanying files. Desktop/Exercise-Files/TASK-9

Investigate the http.pcap file with the zeek-sniffpass module. Investigate the notice.log file. Which username has more module hits?

Ans: BroZeek

```
zeek -C -r http.pcap zeek-sniffpass  
cat notice.log | head  
cat notice.log | zeek-cut msg | uniq -c  
# "-c"count the number of occurrences for each unique value
```

```
root@ip-172-16-1-10:~# cat notice.log | zeek-cut msg | uniq -c
    3 Password found for user BroZeek
    2 Password found for user ZeekBro
```

Investigate the case2.pcap file with geoip-conn module. Investigate the conn.log file. What is the name of the identified City?

Ans: Chicago

```
zeek -C -r case2.pcap geoip-conn
cat conn.log |head
cat conn.log |zeek-cut id.resp_h geo.resp.city | grep -v -e "-" | uniq -c
```

```
root@ip-10-10-10-10:~/home/ubuntu/Desktop/Exercise-Files/TASK-9/geoip-conn# zeek -C -r case2.pcap geoip-conn
root@ip-10-10-10-10:~/home/ubuntu/Desktop/Exercise-Files/TASK-9/geoip-conn# cat conn.log |head
#separator \x09
#set_separator ,
#empty_field    (empty)
#unset_field   -
#path  conn
#open  2023-06-28-03-58-27
#fields ts      uid     id.orig_h      id.orig_p      id.resp_h      id.resp_p      proto      service duration      orig_bytes      resp_bytes      conn_state      local_orig
tes      tunnel_parents  geo.orig.country_code  geo.orig.region geo.orig.city  geo.orig.latitude  geo.orig.longitude  geo.resp.country_code  geo.resp.region geo.resp.city
12 23.77.86.54      Chicago
```

Which IP address is associated with the identified City?

Ans: 23.77.86.54

Investigate the case2.pcap file with sumstats-counttable.zeek script. How many types of status codes are there in the given traffic capture?

Ans: 4

```
zeek -C -r case2.pcap sumstats-counttable.zeek
```

```
root@ip-10-10-10-10:~/home/ubuntu/Desktop/Exercise-Files/TASK-9/geoip-conn# zeek -C -r case2.pcap sumstats-counttable.zeek
Host: 23.77.86.54
status code: 301, count: 4
Host: 116.203.71.114
status code: 302, count: 4
status code: 404, count: 6
status code: 200, count: 26
status code: 301, count: 4
```

Here is a modified command.

```
zeek -C -r case2.pcap sumstats-counttable.zeek | awk '{print $3}' | grep -v -e
```

In this command, the `-v` option inverts the matching logic, causing `grep` to exclude lines that match the specified pattern. The pattern `^\$` matches empty lines because `^` represents the start of a line, and `$` represents the end of a line. Thus, `^\$` matches lines that contain no characters between the start and end.

```
root@i-0d-0-0-0:/home/ubuntu/Desktop/Exercise-Files/TASK-9/geotp-conn# zeek -C -r case2.pcap sumstats-counttable.zeek | awk '{print $3}' | grep -v -e '^\$' | sort | uniq | wc -l
4
```

Thanks for Reading.

Happy learning.

My write-up for the next Zeek Exercise is here: (coming soon)

Tryhackme

Ctf Writeup

Cybersecurity

Training

Learning



Follow

Written by **igor_sec**

370 Followers · 11 Following

Responses (1)



What are your thoughts?

Respond



Samar
about 2 months ago

...

thank you



Reply

More from [igor_sec](#)



igor_sec

CyberDefenders | Boss Of The SOC v1

Jul 5, 2023 12



...

 igor_sec

TryHackMe | Boogeyman 1

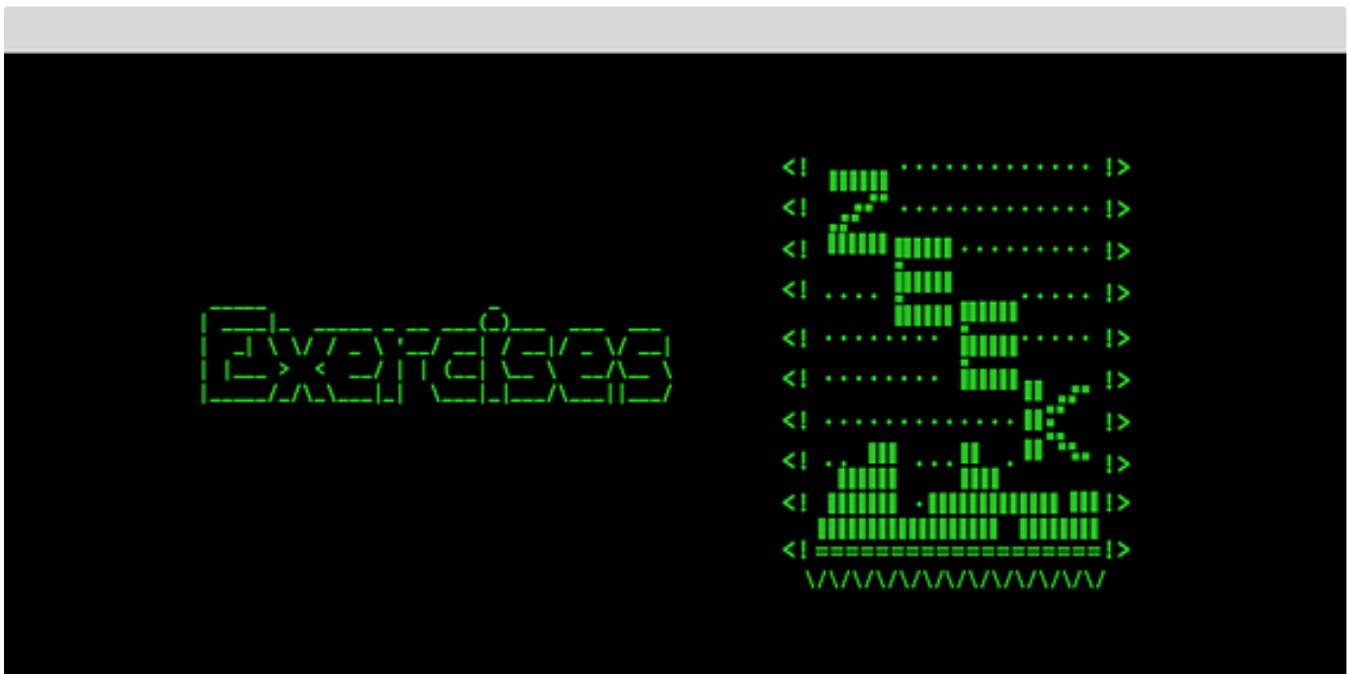
The room provided a phishing email, endpoint logs, and network traffic to analyze. By studying email headers, parsing JSON logs with JQ...

Nov 20, 2023

👏 13



...



```
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
<! ..... !>
```

 igor_sec

TryHackMe | Zeek Exercises — Write-up

Put your Zeek skills into practice and analyse network traffic.

Jul 12, 2023

38

1



...



igor_sec

TryHackMe | Brim

Learn and practice log investigation, pcap analysis and threat hunting with Brim.

Jul 1, 2023

3

1



...

See all from igor_sec

Recommended from Medium

```
d

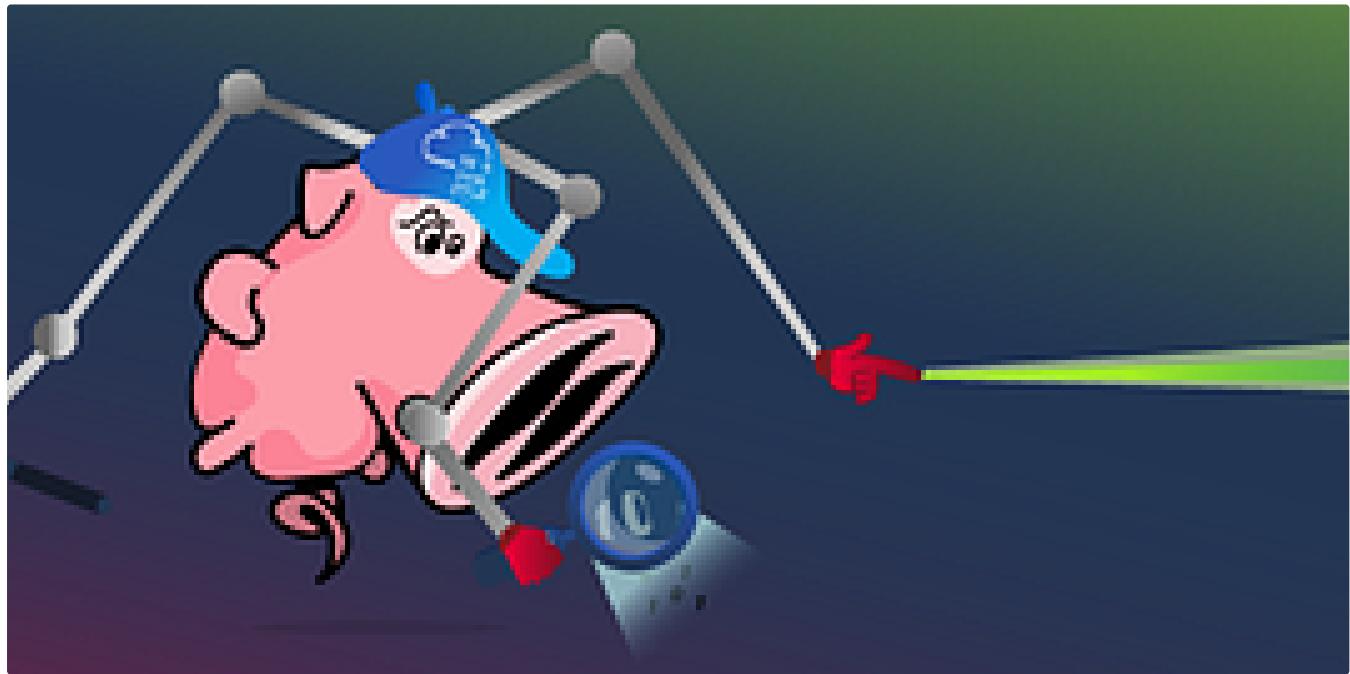
rd.img.old lib64 media opt root sbin srv tmp var vmlinuz.old
          lost+found mnt proc run snap sys usr vmlinuz
var/log
log# ls
cloud-init-output.log dpkg.log      kern.log    lxd       unattended-upgrades
cloud-init.log     fontconfig.log  landscape  syslog    wtmp
dist-upgrade      journal        lastlog    tallylog
log# cat auth.log | grep install
8-55 sudo:  cybert : TTY=pts/0 ; PWD=/home/cybert ; USER=root ; COMMAND=/usr/bin/
8-55 sudo:  cybert : TTY=pts/0 ; PWD=/home/cybert ; USER=root ; COMMAND=/usr/bin/
8-55 sudo:  cybert : TTY=pts/0 ; PWD=/home/cybert ; USER=root ; COMMAND=/bin/chow
hare/dokuwiki/bin /usr/share/dokuwiki/doku.php /usr/share/dokuwiki/feed.php /usr/s
hare/dokuwiki/install.php /usr/share/dokuwiki/lib /usr/share/dokuwiki/vendor -R
log# █
```

T Dan Molina

Disgruntled CTF Walkthrough

This is a great CTF on TryHackMe that can be accessed through this link here:
<https://tryhackme.com/room/disgruntled>

Oct 22, 2024



In T3CH by Axoloth

TryHackMe | Snort Challenge—The Basics | WriteUp

Put your snort skills into practice and write snort rules to analyse live capture network traffic

Nov 9, 2024 100



...

Lists



Self-Improvement 101

20 stories · 3195 saves



How to Find a Mentor

11 stories · 785 saves



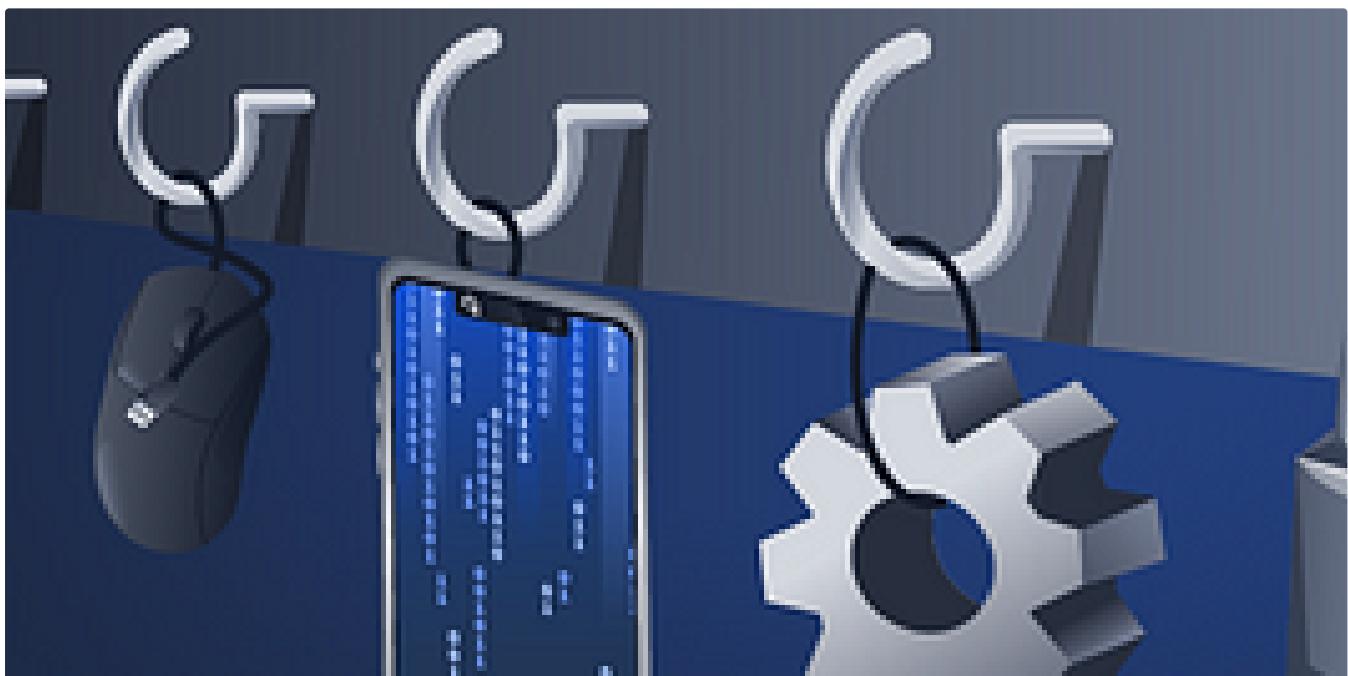
Good Product Thinking

13 stories · 794 saves



Tech & Tools

22 stories · 380 saves



In T3CH by Axoloth

TryHackMe | FlareVM: Arsenal of Tools| WriteUp

Learn the arsenal of investigative tools in FlareVM

Nov 28, 2024 50



...



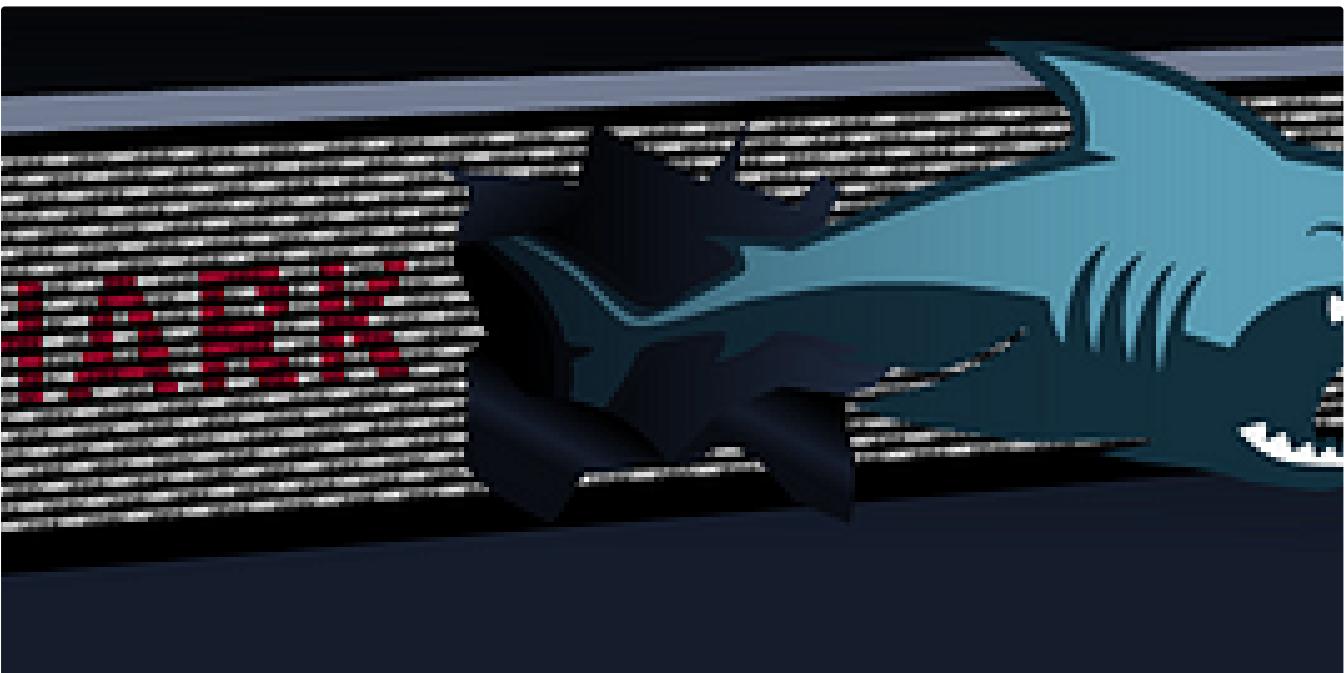
In T3CH by Axoloth

TryHackMe | Training Impact on Teams | WriteUp

Discover the impact of training on teams and organisations

Nov 5, 2024

60



MAGESH

TShark: The Basics— Tryhackme

Learn the basics of TShark and take your protocol and PCAP analysis skills a step further.

Sep 3, 2024





 DevSecOps

[Wi-Fi attacks] Day 11: If you'd like to WPA, press the star key!

[Wi-Fi attacks] Day 11: If you'd like to WPA, press the star key! [TryHackMe THM][Advent of Cyber AoC 2024], [Walktgrrough, Write Up]

Dec 12, 2024  1



...

See more recommendations