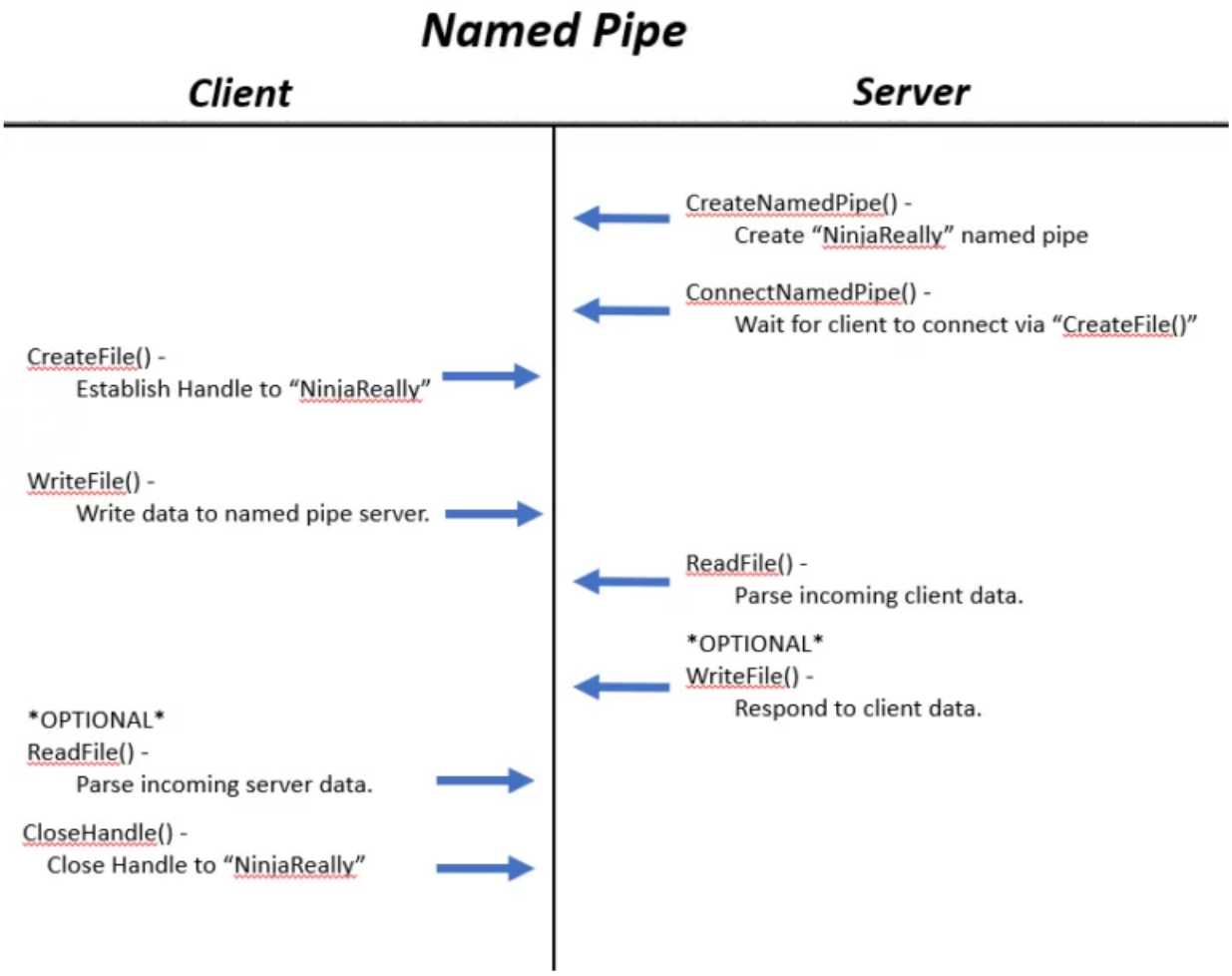
Labs - Working with IDS and IPS

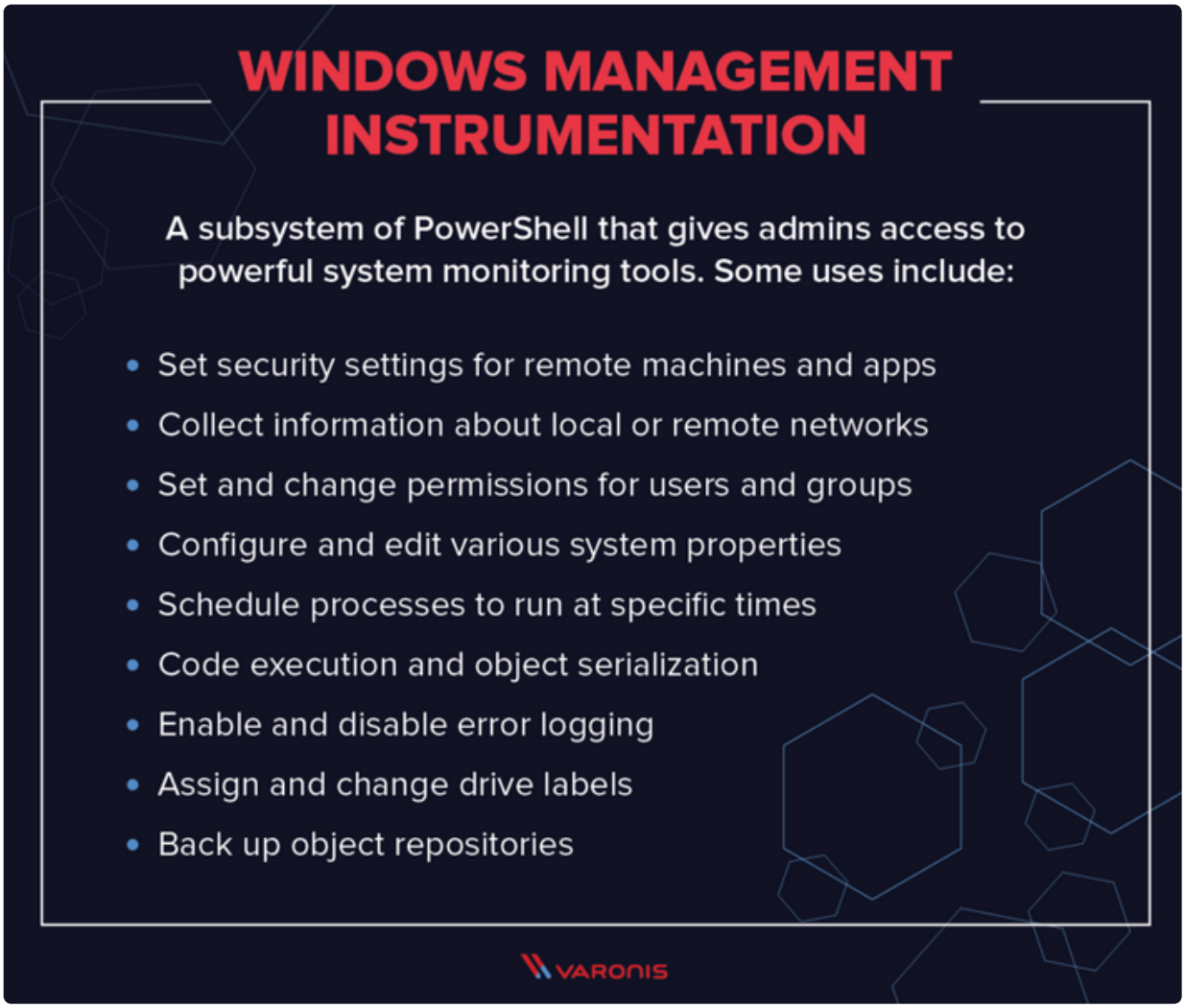
Preliminary knowledge

- DNS exfilitration: Extracting data through DNS requests (data is exfiltrated through the dns request) from the victim machine to an DNS server that attacker controls.
- Pipe: A shared section of memory that processes use for communication. Named pipes can be used to provide communication between the same computer or processes on different computers across a network.



-> Handle, pointer to some underlying type/memory location
https://versprite.com/vs-labs/vulnerable-named-pipe-application/
-> Three part series on named pipe.

- WMI (windows management Instrumentation) is a subsystem of Powershell that gives admin access powerful system monitoring tools
https://www.varonis.com/blog/wmi-windows-management-instrumentation



-> WMIC is an command line interace for WMI.

-> Wmiprvse.exe: A process that runs to enable WMI

https://www.lifewire.com/what-is-the-wmiprvse-exe-process-4775428

What Is the wmiprvse.exe Process

The wmiprvse.exe process is a process that runs alongside the WMI core process, WinMgmt.exe.

Wmiprvse.exe is a normal Windows OS file that's located in %systemroot%\Windows\System32\Wbem. If you find and right-click the file, then select **Properties**, on the details tab you'll see that the file name is: "WMI Provider Host."

-> Win32_ProcessStartup class: An WMI class that represents how a windows-based process is configured (used to set the property for starting up a windows process)

Win32_ProcessStartup class

Article • 04/02/2021 • 6 contributors

Feedback

In this article

The **Win32_ProcessStartup** abstract WMI class represents the startup configuration of a Windows-based process. The class is defined as a method type definition, which means that it is only used for passing information to the Create method of the Win32_Process class.

The following syntax is simplified from Managed Object Format (MOF) code and includes all inherited properties.

Syntax

syntax Copy

```
[Abstract, UUID("{8502C4DB-5FBB-11D2-AAC1-006008C78BC7}"), AMENDMENT]
class Win32_ProcessStartup : Win32_MethodParameterClass
{
```

-> This is related to the Create method, which creates a new process:

Create method of the Win32_Process class

Article • 01/06/2021 • 6 contributors

Feedback

In this article

The **Create** WMI class method creates a new process.

This topic uses Managed Object Format (MOF) syntax. For more information about using this method, see Calling a Method.

- CIM_Process class: Represents a single instance of a running program
  https://learn.microsoft.com/en-us/windows/win32/cimwin32prov/cim-process#syntax

CIM_Process class

Article • 01/06/2021 • 6 contributors

Feedback

In this article

The **CIM_Process** class represents a single instance of a running program. A user typically sees a process as an application or task. A process is defined by a workspace of memory resources and environmental settings that are allocated to it. On a multitasking system, the workspace prevents intrusion of resources by other processes. Additionally, a process can execute as multiple threads, all which run within the same workspace.

Important

The DMTF (Distributed Management Task Force) CIM (Common Information Model) classes are the parent classes upon which WMI classes are built. WMI currently supports only the CIM 2.x version schemas.

The following syntax is simplified from Managed Object Format (MOF) code and includes all of its inherited properties. Properties are listed in alphabetic order, not MOF order.

-> CIM: Common Information model, a universally accepted way of describing and representing data in IT environment:
https://www.splunk.com/en_us/blog/learn/cim-common-information-model.html

What is CIM?

The Common Information Model (CIM) represents a universally accepted standard that allows consistent description and representation of data in IT environments.

The CIM provides a conceptual framework that encompasses how objects in an IT environment can be represented, and it lays out the relationships between these objects. It ensures that information from different sources, systems, or vendors can be integrated and understood consistently.

The Distributed Management Task Force (DMTF) is a crucial industry organization behind the promotion of the CIM. DMTF is dedicated to developing, adopting, and promoting interoperable management initiatives and standards. The CIM, under DMTF's guidance, has been crucial in enabling effective data management in heterogeneous IT environments, ensuring that disparate systems can be managed in a unified manner regardless of their originating vendors or underlying technologies.

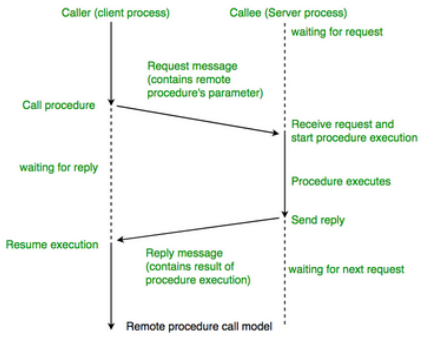
- RPC: Remote procedural call: We can also call it as Remote function call. The called function/procedure does not need to be on the same machine.
https://www.geeksforgeeks.org/remote-procedure-call-rpc-in-operating-system/

Remote Procedure Call (RPC) in Operating System

Last Updated : 22 Apr, 2023

Remote Procedure Call (RPC) is a powerful technique for constructing **distributed, client-server based applications**. It is based on extending the conventional local procedure calling so that the **called procedure need not exist in the same address space as the calling procedure**. The two processes may be on the same system, or they may be on different systems with a network connecting them.

When making a Remote Procedure Call:



-> DCE/RPC is an implementation of RPC and is often used by tools like impacket for calling remote windows functions/procedures on Linux during post-exploitation.

- runas: A command line tool that allows user to run specific tools and program under a certain user

https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/cc771525(v=ws.11)

Runas

Article • 08/31/2016

In this article

Applies To: Windows Server 2003, Windows Vista, Windows XP, Windows HPC Server 2008 R2, Windows Server 2008, Windows 7, Windows Server 2003 R2, Windows Server 2000, Windows Server 2012, Windows Server 2003 with SP1, Windows 8

Allows a user to run specific tools and programs with different permissions than the user's current logon provides.

Runas is a command-line tool that is built into Windows Vista. To use **runas** at the command line, open a command prompt, type **runas** with the appropriate parameters, and then press ENTER.

In the user interface for Windows Vista, the **Run as...** command has been changed to **Run as administrator**. However, you should rarely have to use the **Run as administrator** command because Windows Vista will automatically prompt you for an administrator password when it is needed.

For examples of how this command can be used, see Examples.

- KDC (Key distribution Service): A domain service typically located on a domain controller, it is used for Authentication service (AS) and Ticket granting service (TGS) for the Kerberos protocol.

Key Distribution Center

Article • 01/07/2021 • 5 contributors

Feedback

The Key Distribution Center (KDC) is implemented as a domain service. It uses the Active Directory as its account database and the Global Catalog for directing referrals to KDCs in other domains.

As in other implementations of the *Kerberos protocol*, the KDC is a single process that provides two services:

- Authentication Service (AS)

  This service issues ticket-granting tickets (TGTs) for connection to the ticket-granting service in its own domain or in any trusted domain. Before a client can ask for a ticket to another computer, it must request a TGT from the authentication service in the client's account domain. The authentication service returns a TGT for the ticket-granting service in the target computer's domain. The TGT can be reused until it expires, but the first access to any domain's ticket-granting service always requires a trip to the authentication service in the client's account domain.

- Ticket-Granting Service (TGS)

  This service issues tickets for connection to computers in its own domain. When clients want access to a computer, they contact the ticket-granting service in the target computer's domain, present a TGT, and ask for a ticket to the computer. The ticket can be reused until it expires, but the first access to any computer always requires a trip to the ticket-granting service in the target computer's account domain.

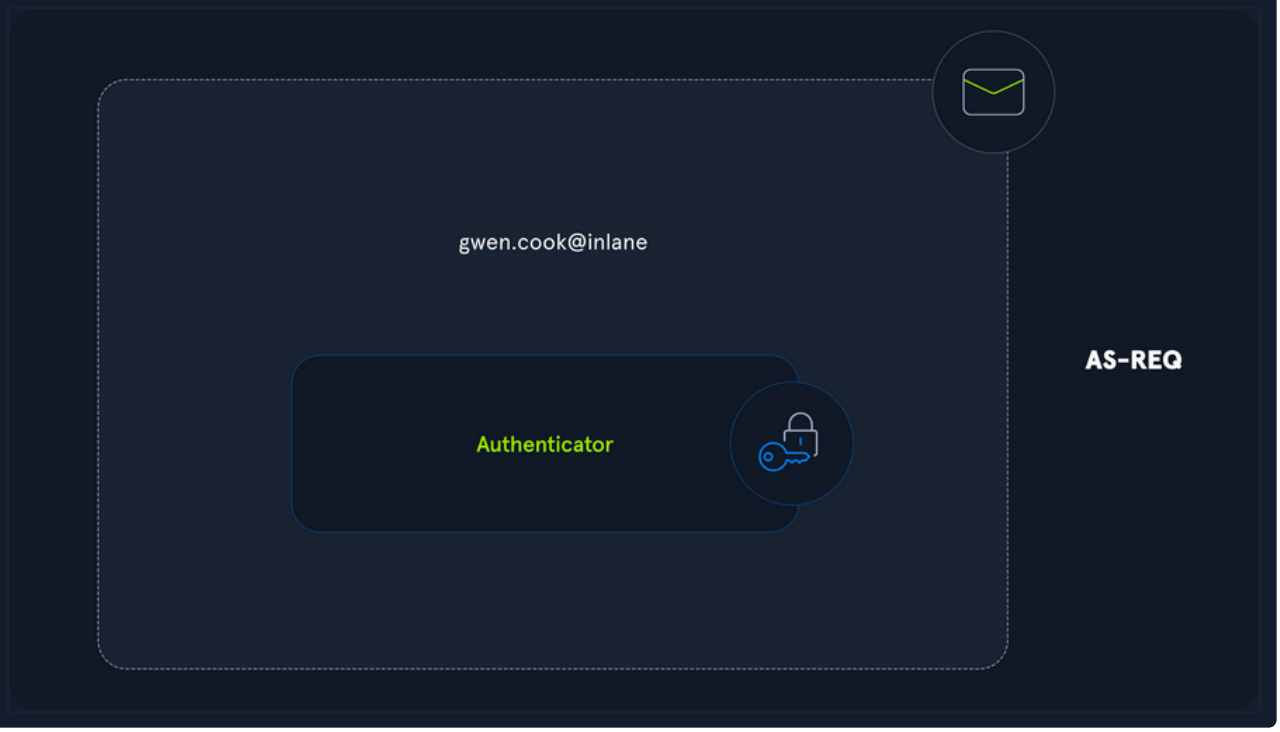
The KDC for a domain is located on a domain controller, as is the Active Directory for the domain. Both services are started automatically by the domain controller's *Local Security Authority* (LSA) and run as part of the LSA's process. Neither service can be stopped. If the KDC is unavailable to network clients, then the Active Directory is also unavailable—and the domain controller is no longer controlling the domain. The system ensures availability of these and other domain services by allowing each domain to have several domain controllers, all peers. Any domain controller can accept authentication requests and ticket-granting requests addressed to the domain's KDC.

-> AS-REQ: A request that an user makes to obtain it TGT for Kerberos process. The timerstap is encrypted with the user's password hash.
(ref for Kerberos Attack module on HTB academy).

Request (AS-REQ)

First, the user makes a TGT (or identity card) request. This request is called `AS-REQ`. But to receive the TGT, they must be able to prove their identity. This request is made to the KDC (which is the Domain Controller in an Active Directory environment). The KDC holds all user keys.

To prove their identity, the user will send an `authenticator`. It's the current timestamp that the user will encrypt with their key. The username is also sent in cleartext so the KDC can know whom it is dealing with.



-> Time-stamp: Current time of an event recorded.

https://www.techtarget.com › whatis › definition › timestamp

What is a timestamp? - TechTarget

timestamp: **A timestamp is** the current **time** of an event that is recorded by a computer. Through mechanisms such as the Network **Time** Protocol (NTP), a computer maintains accurate current **time**, calibrated to minute fractions of a second. Such precision makes it possible for networked computers...

- Exploit kit: Tools that automate the process of exploitation
  https://www.malwarebytes.com/blog/threats/neutrino

Neutrino

Short bio

The Neutrino exploit kit is a malicious tool kit, which can be used by attackers who are not experts on computer security. Threat actors can have zero coding experience and still use exploit kits like Neutrino to conduct their illegal activity.

History

Exploit kits, sometimes referred to as exploit packs, are toolkits that automate the exploitation of client-side vulnerabilities, often targeting browsers and applications that a website can invoke through the browser. Known exploit targets have been vulnerabilities in Adobe Reader, Java Runtime Environment, and Adobe Flash Player.

Neutrino began targeting CVE-2012-1723, CVE-2013-0431, and, CVE-2013-0422, all exploiting vulnerabilities in the Java Runtime Environment (JRE) component. It was marketed as a simple-to-use kit with a nicely user friendly control panel.

Suricata

Suricata Fundamentals

Question

- Filter out only HTTP events from /var/log/suricata/old_eve.json using the the jq command-line JSON processor. Enter the flow_id that you will come across as your answer.
  -> We filter for http events, as follows

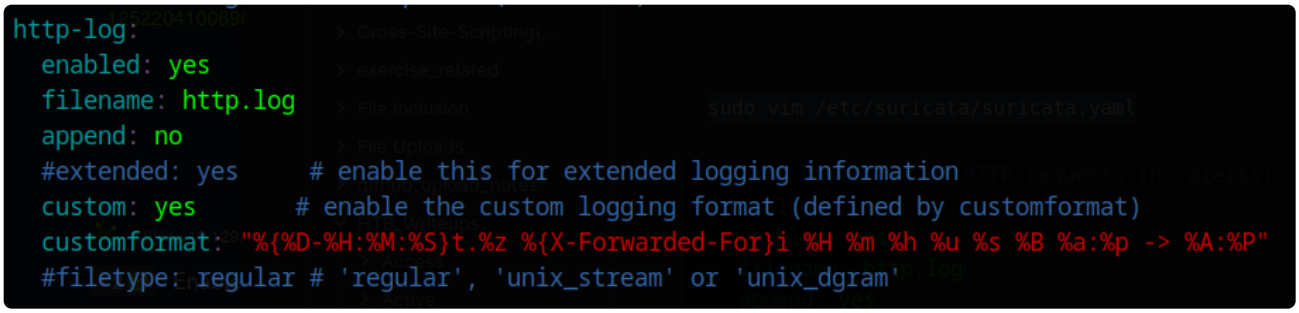
```
cat /var/log/suricata/old_eve.json | jq -c 'select(.event_type == "http")' | jq
```

-> Hence we have our flow_id

- Enable the http-log output in suricata.yaml and run Suricata against /home/htb-student/pcaps/suspicious.pcap. Enter the requested PHP page as your answer. Answer format: _.php
  -> We first go to edit the yaml file

```
sudo vim /etc/suricata/suricata.yaml
```

-> We edit as following the documentation reference



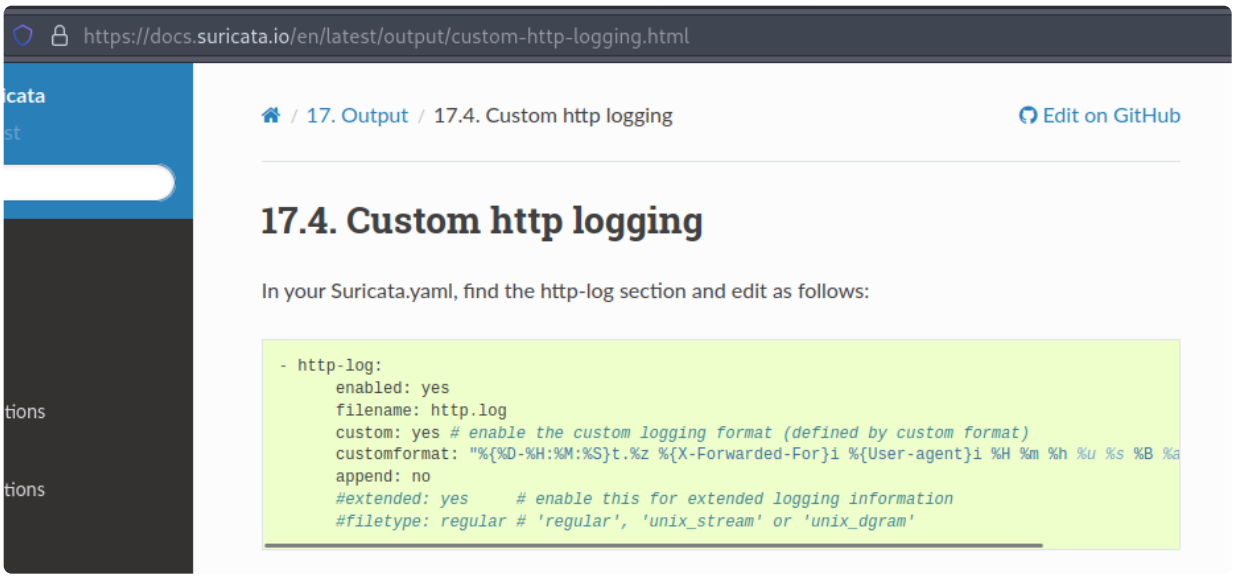
-> We read the pcap file as required

```
suricata -r /home/htb-student/pcaps/suspicious.pcap
```

-> In the same directory, we obtain the answer:

```
ls

cat http.log
```

Reference:

1. 

Suricata Rule Development Part 1

Question

- In the /home/htb-student directory of this section's target, there is a file called local.rules. Within this file, there is a rule with sid 2024217, which is associated with the MS17-010 exploit. Additionally, there is a PCAP file named eternalblue.pcap in the /home/htb-student/pcaps directory, which contains network traffic related to MS17-010. What is the minimum offset value that can be set to trigger an alert?

-> We look at the initial rule file and know that the thing we have to change is the offset.

```
alert smb any any -> $HOME_NET any (msg:"ETERNALBLUE MS17-010"; flow:to_server,e
stablished; content:"|ff|SMB|33 00 00 00 00 18 07 c0 00 00 00 00 00 00 00 00 00
00 00 00 00 08 ff fe 00 08|"; offset:9; depth:30; fast_pattern:10,20; content:"|
00 09 00 00 00 10|"; distance:1; within:6; content:"|00 00 00 00 00 00 00 10|";
within:8; content:"|00 00 00 10|"; distance:4; within:4; pcre:"/^[a-zA-Z0-9+/]{1
000,}/R"; threshold: type both, track by_src, count 3, seconds 30; sid:2024217;
rev:3;)
```

-> Looking at the reference, we know that eternal blue relies on BoF as one of its step, so it has to overflow the buffer first, which requires large amount of bytes.
-> Hence the characteristics of the exploit is large length in packet, which is also shown in the rule above.

-> We analyse the eternalblue.pcap and analyse the smb protocol, filtering out by length as follws:

smb

No.	Time	Source	Destination	Protocol	Length	Info
35465	472.997242	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35460	472.997230	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35455	472.997219	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35450	472.995735	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35445	472.995564	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35440	472.995093	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35430	472.965373	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35425	472.965363	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35420	472.965350	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35415	472.965339	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35410	472.965329	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35405	472.965318	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35400	472.965306	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35395	472.965296	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35390	472.965284	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35385	472.965272	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35380	472.965261	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35375	472.965250	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35370	472.965238	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP
35365	472.965227	192.168.116.172	192.168.116.149	SMB	1312	Trans2 Request, SESSION_SETUP

-> We see packets with length of 1312, which are very large and suspicious.
-> We'll look into one of them.

-> Examining the Packet bytes window, we see that the it seems to match our rule set of content, but the ff starts at the 5th byte, hence the offset should be 4 if we were to catch eternal blue.

```
0000  00 00 10 4e ff 53 4d 42  32 00 00 00 00 18 07 c0   ···N·SMB 2·····
0010  00 00 00 00 00 00 00 00  00 00 00 00 00 08 ff fe   ···· ···· ····
0020  00 08 42 00 0f 0c 00 00  10 01 00 00 00 00 00 00   ··B· ···· ····
0030  00 25 89 1a 00 00 00 0c  00 42 00 00 10 4e 00 01   ·%·· ···· ·B···N··
0040  00 0e 00 0d 10 00 d6 f3  10 43 de 90 40 43 de d0   ···· ···· ·C··@C··
0050  4e 43 d8 b6 3a 92 10 ac  51 00 11 ed 74 e3 ea 0f   NC··:··· Q···t···
0060  bb 22 00 59 06 5e 11 99  6f 53 61 cc 8a 75 75 d3   ·"·Y·^·· oSa··uu·
0070  34 65 28 21 f7 df e9 0b  d0 d3 cf 99 de 26 bc 54   4e(!···· ·····&·T
0080  13 6b 23 74 e1 c1 07 82  50 28 65 f3 07 47 50 09   ·k#t···· P(e··GP·
0090  13 2e 9a 21 27 82 fc 9a  f2 ab f7 4b 61 24 c4 84   ···!'··· ···Ka$··
00a0  ce 86 18 f1 2c 96 df 02  35 e5 b1 5f d8 44 d1 9a   ····,··· 5··_·D··
00b0  5e 47 f7 e1 98 f5 45 76  ea ec 95 df 6a 58 75 dd   ^G····Ev ····jXu·
00c0  1b 77 b5 da d5 db d6 46  5d bf 00 3a ab e3 50 28   ·w·····F ]··:··P(
00d0  7e 1d be c8 09 31 b5 2f  5d ac 73 61 0a ea c8 e5   ~····1·/ ]·sa····
00e0  d8 7e fc a5 ca b3 95 e9  b2 59 ec e2 f7 18 8d f7   ·~····· ·Y·····
00f0  a3 d2 a5 54 61 f0 fd e4  85 a5 7b 9e da b9 8b bd   ···Ta··· ··{·····
0100  12 a2 7d 9a 6d ef 6c 76  74 0a 18 95 85 7a f0 20   ··}·m·lv t····z·
0110  d8 65 33 61 72 5d 59 7c  24 4c c6 f5 67 62 06 80   ·e3ar]Y| $L··gb··
0120  a6 40 44 b1 81 ba e7 8b  26 09 49 29 8a 76 50 31   ·@D····· &·I)·vP1
0130  28 2e 41 be 2d eb 64 3f  c6 bc ee bf 6c c0 8c d8   (.A·-·d? ····l···
```

-> Hence, we change to offset from 9 to 4

```
alert smb any any -> $HOME_NET any (msg:"ETERNALBLUE MS17-010"; flow:to_server,e
stablished; content:"|ff|SMB|33 00 00 00 00 18 07 c0 00 00 00 00 00 00 00 00 00
00 00 00 00 08 ff fe 00 08|"; offset:4; depth:30; fast_pattern:10,20; content:"|
00 09 00 00 00 10|"; distance:1; within:6; content:"|00 00 00 00 00 00 00 10|";
within:8; content:"|00 00 00 10|"; distance:4; within:4; pcre:"/^[a-zA-Z0-9+/]{1
000,}/R"; threshold: type both, track by_src, count 3, seconds 30; sid:2024217;
rev:3;)
```

-> We execute suricata on the pcap file as follows

```
sudo suricata -r /home/htb-student/pcaps/eternalblue.pcap -l . -k none
```

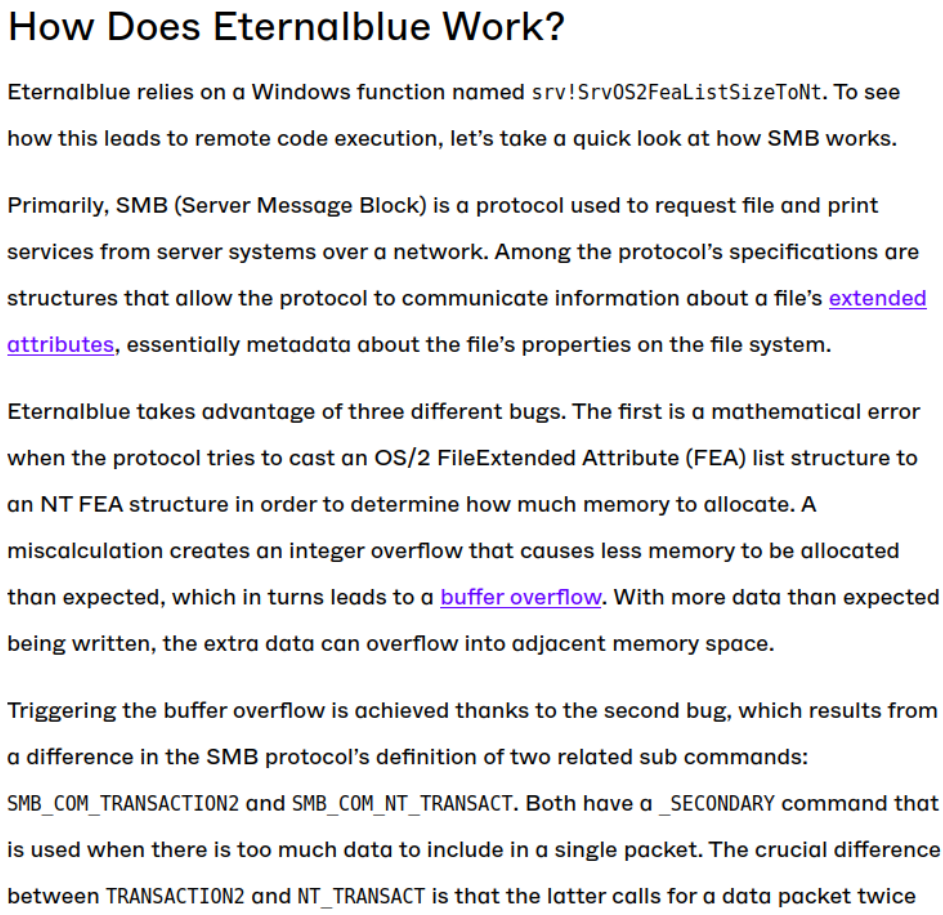
```
$ sudo suricata -r /home/htb-student/pcaps/eternalblue.pcap -l . -k none
16/7/2024 -- 04:08:58 - <Notice> - This is Suricata version 4.0.0-beta1 RELEASE
16/7/2024 -- 04:08:58 - <Notice> - all 5 packet processing threads, 4 management
 threads initialized, engine started.
16/7/2024 -- 04:08:59 - <Notice> - Signal Received.  Stopping engine.
16/7/2024 -- 04:08:59 - <Notice> - Pcap-file module read 46654 packets, 37044839
 bytes
```

-> We see the log with alerts and confirmed that we have the correct offset

```
$ cat fast.log
05/18/2017-01:12:13.428436  [**] [1:2024217:3] ETERNALBLUE MS17-010 [**] [Classi
fication: (null)] [Priority: 3] {TCP} 192.168.116.149:49472 -> 192.168.116.138:4
45
05/18/2017-01:13:02.536176  [**] [1:2024217:3] ETERNALBLUE MS17-010 [**] [Classi
fication: (null)] [Priority: 3] {TCP} 192.168.116.149:50742 -> 192.168.116.172:4
45
05/18/2017-01:13:22.563076  [**] [1:2024217:3] ETERNALBLUE MS17-010 [**] [Classi
fication: (null)] [Priority: 3] {TCP} 192.168.116.149:51495 -> 192.168.116.172:4
45
05/18/2017-01:13:28.061603  [**] [1:2024217:3] ETERNALBLUE MS17-010 [**] [Classi
fication: (null)] [Priority: 3] {TCP} 192.168.116.138:1366 -> 192.168.116.149:44
5
```

Reference:

https://www.sentinelone.com/blog/eternalblue-nsa-developed-exploit-just-wont-die/



How Does Eternalblue Work?

Eternalblue relies on a Windows function named `srv!Srv0S2FeaListSizeToNt`. To see how this leads to remote code execution, let's take a quick look at how SMB works.

Primarily, SMB (Server Message Block) is a protocol used to request file and print services from server systems over a network. Among the protocol's specifications are structures that allow the protocol to communicate information about a file's extended attributes, essentially metadata about the file's properties on the file system.

Eternalblue takes advantage of three different bugs. The first is a mathematical error when the protocol tries to cast an OS/2 FileExtended Attribute (FEA) list structure to an NT FEA structure in order to determine how much memory to allocate. A miscalculation creates an integer overflow that causes less memory to be allocated than expected, which in turns leads to a buffer overflow. With more data than expected being written, the extra data can overflow into adjacent memory space.

Triggering the buffer overflow is achieved thanks to the second bug, which results from a difference in the SMB protocol's definition of two related sub commands: `SMB_COM_TRANSACTION2` and `SMB_COM_NT_TRANSACT`. Both have a `_SECONDARY` command that is used when there is too much data to include in a single packet. The crucial difference between `TRANSACTION2` and `NT_TRANSACT` is that the latter calls for a data packet twice

-> Eternalblue relies on BoF, so it sends large amount of bytes over, as indicated in the rule file, so look for packets large enough.

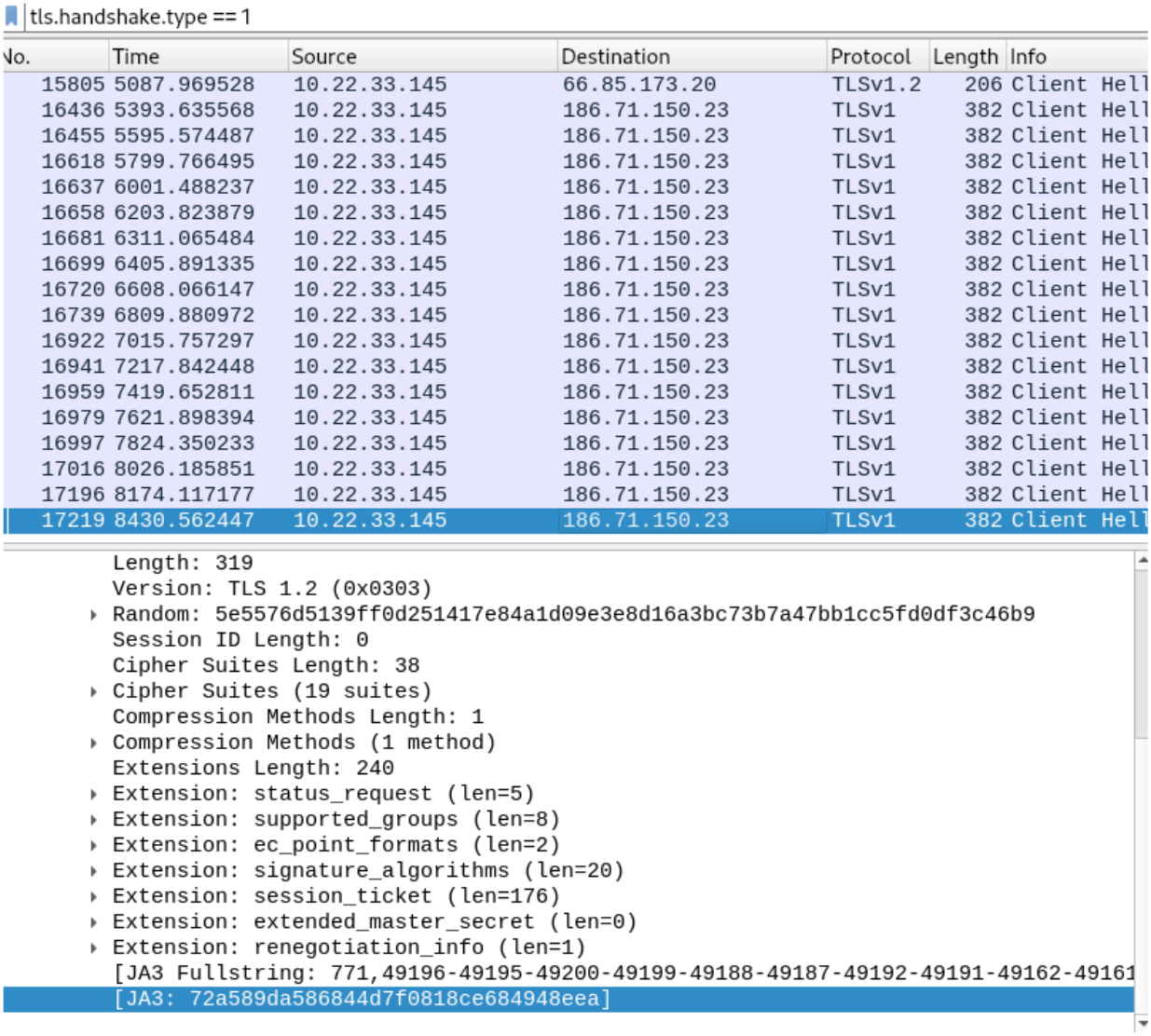
Suricata Rule Development Part 2 (Encrypted Traffic)

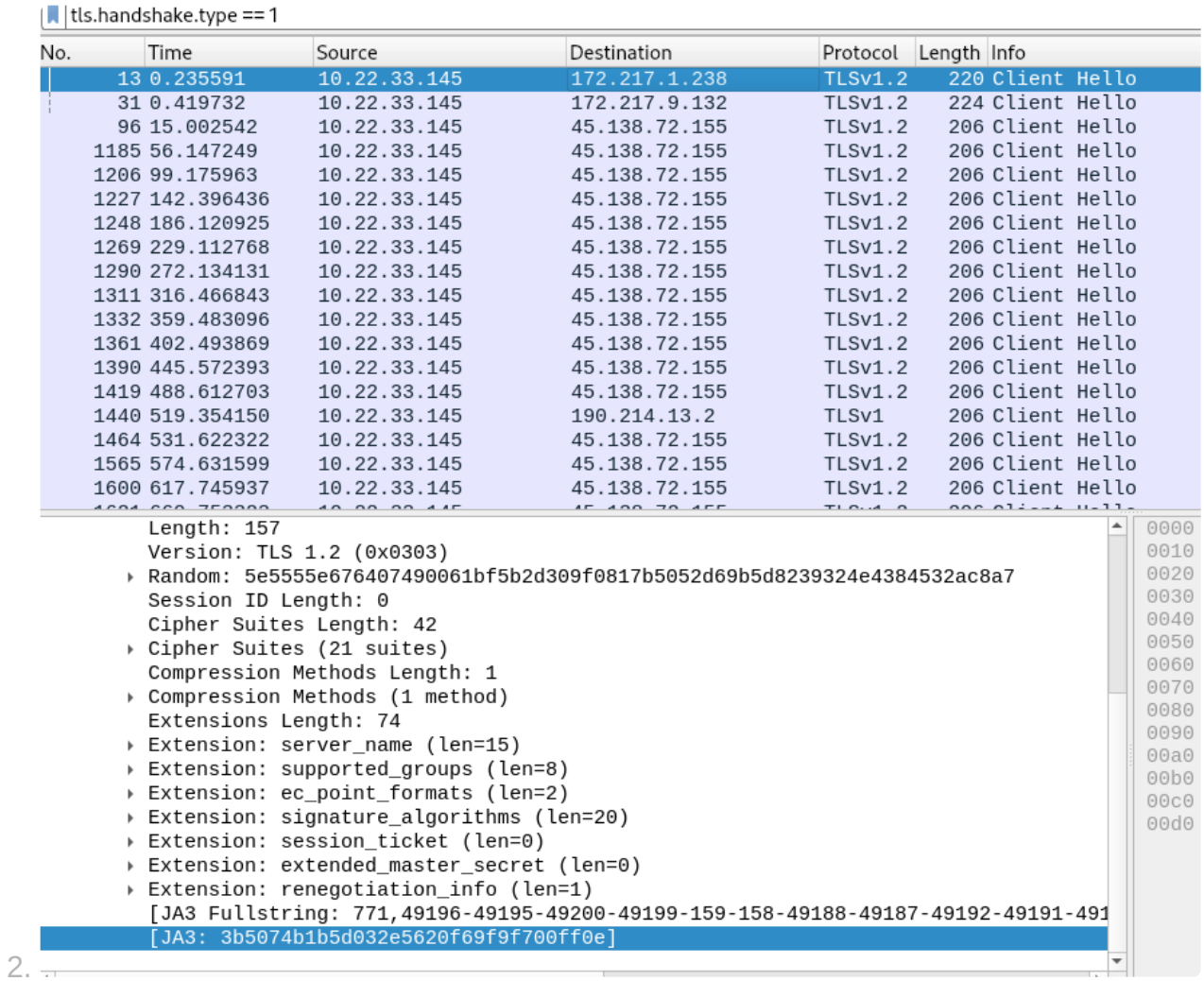
Question

- There is a file named trickbot.pcap in the /home/htb-student/pcaps directory, which contains network traffic related to a certain variation of the Trickbot malware. Enter the precise string that should be specified in the content keyword of the rule with sid 100299 within the local.rules file so that an alert is triggered as your answer.

```
ja3 -a --json /home/htb-student/pcaps/trickbot.pcap
```

-> We looked at the wireshark packets, filtered for JA3 type 3 and we found that there are only 2 JA3 hash available:

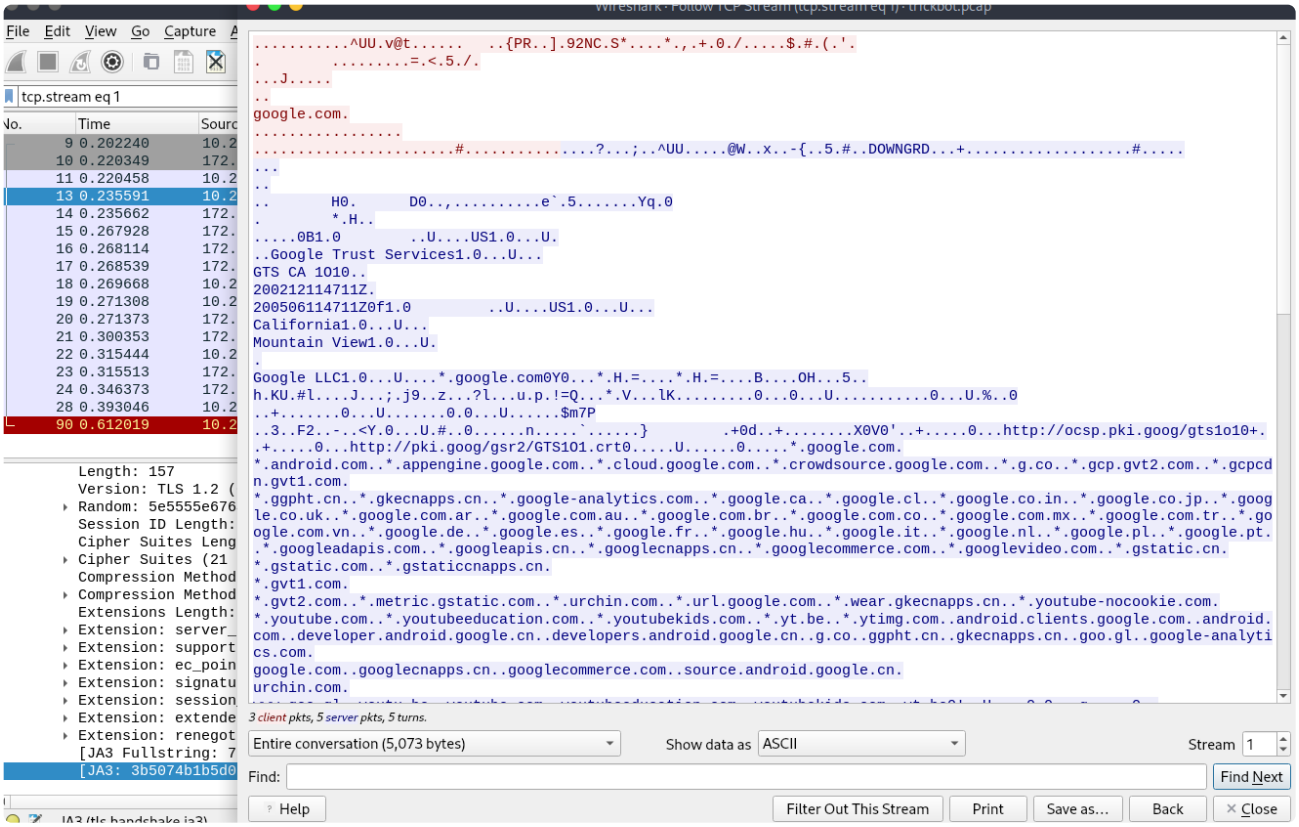




2.

-> Further examination showed that the second one likely dealing with appropriate website like google.com, as shown in the second packet in the TLS stream, where we can see the

some countries name with google trust in it.



-> While the first one is having weird value of name in its certificate, which likely resembles a certificate from a malware server.

-> Hence, we will use the JA3 hash of 72a589da586844d7f0818ce684948eea, resembling the first JA3 hash, edit the rule file and run suricata as follows

```
alert tls any any -> any any (msg:"Trickbot C2 SSL"; ja3.hash; content:"72a589da586844d7f0818ce684948eea"; sid:100299; rev:1;)
```

```
sudo suricata -r /home/htb-student/pcaps/trickbot.pcap
```

```
$ sudo suricata -r /home/htb-student/pcaps/trickbot.pcap
[sudo] password for htb-student:
17/7/2024 -- 06:11:43 - <Notice> - This is Suricata version 6.0.13 RELEASE running in USER mode
17/7/2024 -- 06:11:43 - <Notice> - all 3 packet processing threads, 4 management threads initialized, engine started.
17/7/2024 -- 06:11:44 - <Notice> - Signal Received.  Stopping engine.
17/7/2024 -- 06:11:44 - <Notice> - Pcap-file module read 1 files, 17227 packets, 16813128 bytes
$
$ ls
eg5  eg6  eve.json  fast.log  local.rules  pcaps  stats.log  suricata.log
$ cat fast.log
02/25/2020-17:14:29.719650  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49796 -> 45.138.72.155:443
02/25/2020-17:15:53.899183  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49798 -> 45.138.72.155:443
02/25/2020-17:16:37.111848  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49799 -> 45.138.72.155:443
02/25/2020-17:17:20.765303  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49800 -> 45.138.72.155:443
02/25/2020-17:18:03.750193  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49801 -> 45.138.72.155:443
02/25/2020-17:18:46.854032  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49802 -> 45.138.72.155:443
02/25/2020-17:20:14.125943  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49804 -> 45.138.72.155:443
02/25/2020-17:20:57.149991  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49806 -> 45.138.72.155:443
02/25/2020-17:22:23.242014  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49810 -> 45.138.72.155:443
02/25/2020-17:23:06.264996  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49813 -> 45.138.72.155:443
02/25/2020-17:15:10.840097  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49797 -> 45.138.72.155:443
02/25/2020-17:25:15.412796  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49820 -> 45.138.72.155:443
02/25/2020-17:19:31.125753  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49803 -> 45.138.72.155:443
02/25/2020-17:21:40.222020  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49808 -> 45.138.72.155:443
02/25/2020-17:28:36.788753  [**] [1:100299:1] Trickbot C2 SSL [**] [Classification: (null)] [Priority: 3] {TCP} 10.22.33.145:49834 -> 5.2.77.18:447
```

Snort

Snort Fundamentals

Question

- There is a file named wannamine.pcap in the /home/htb-student/pcaps directory. Run Snort on this PCAP file and enter how many times the rule with sid 1000001 was triggered as your answer.
  -> We first observe the traffic using snort with the rule file explicitly with alert cmg

```
sudo snort -c /root/snorty/etc/snort/snort.lua --daq-dir
/usr/local/lib/daq -r /home/htb-student/pcaps/wannamine.pcap -R
/home/htb-student/local.rules -A cmg
```

```
03/05-11:20:56.395504 [**] [1:1000001:2] "ICMP test" [**] [Classification: Generic ICMP event] [Priority: 3] {ICMP} 192.168.183
.101 -> 192.168.183.100
```

-> We see that it has caught some traffic with id 1000001

-> We now try to count the number of traffic with this id

```
sudo snort -c /root/snorty/etc/snort/snort.lua --daq-dir
/usr/local/lib/daq -r /home/htb-student/pcaps/wannamine.pcap -R
/home/htb-student/local.rules -A cmg | grep 1000001 | cut -d ":" -f4 |
uniq -c
```

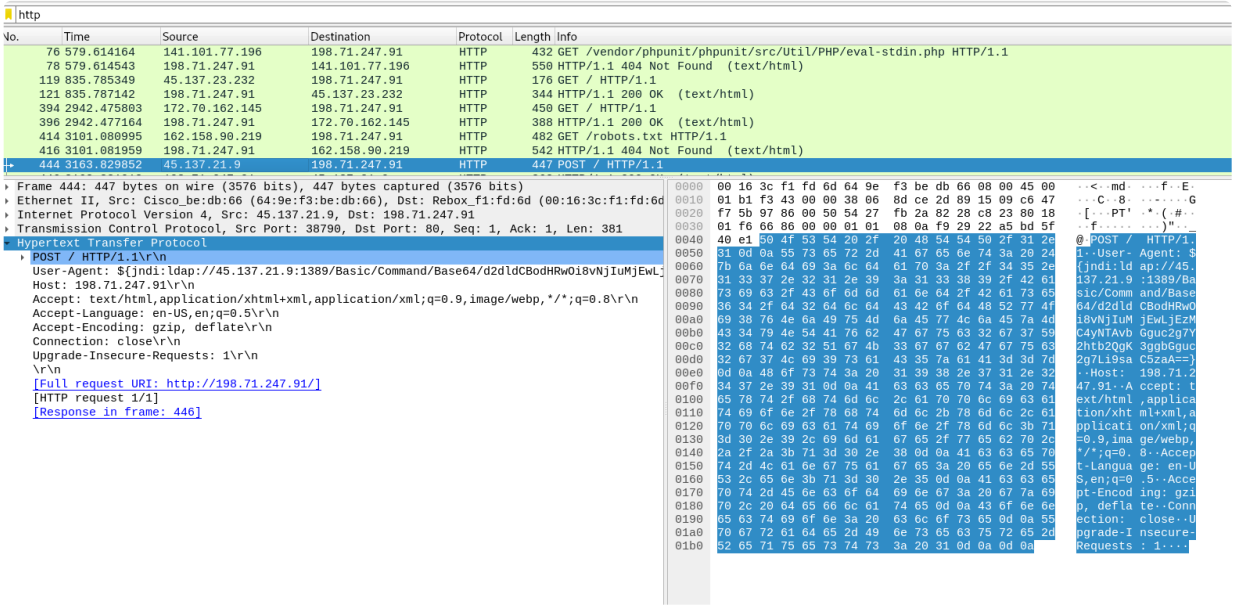
```
$ sudo snort -c /root/snorty/etc/snort/snort.lua --daq-dir /usr/local/l
ib/daq -r /home/htb-student/pcaps/wannamine.pcap -R /home/htb-student/l
ocal.rules -A cmg | grep 1000001 | cut -d ":" -f4 | uniq -c
    234 1000001
```

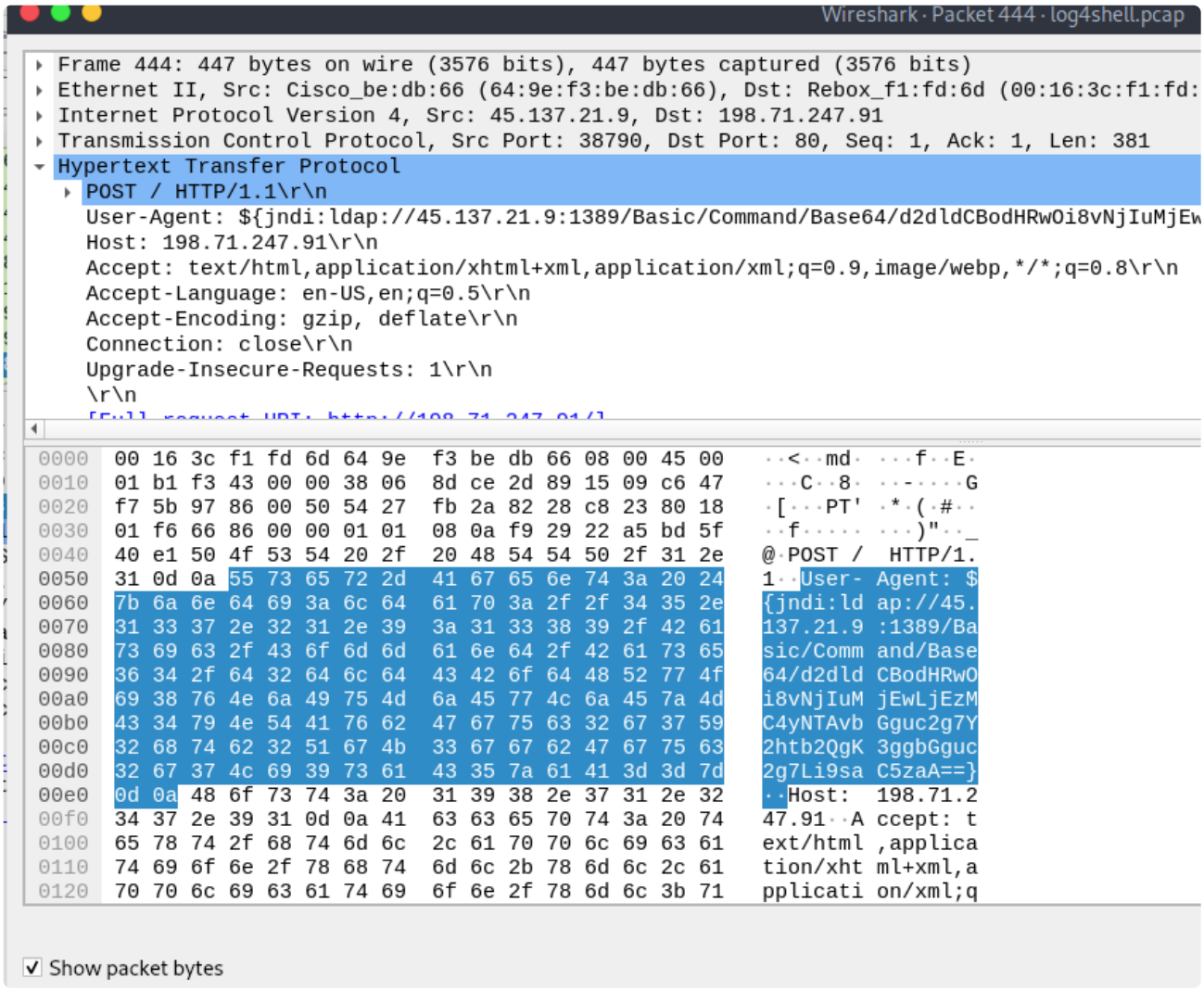
-> Hence, we see that there are 234 alerts generated with sid 1000001

Snort Rule Development

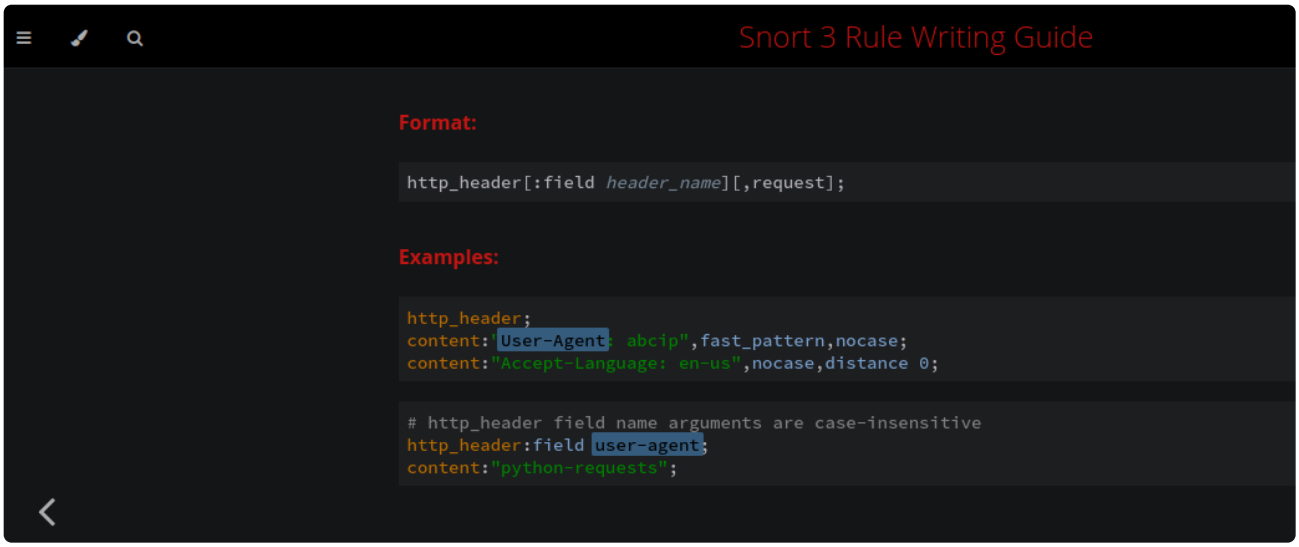
Question

- There is a file named log4shell.pcap in the /home/htb-student/pcaps directory, which contains network traffic related to log4shell exploitation attempts, where the payload is embedded within the user agent. Enter the keyword that should be specified right before the content keyword of the rule with sid 10000098 within the local.rules file so that an alert is triggered as your answer. Answer format: [keyword];
  -> We first analysed the pcap files that matches the rule and we first see that it has jndi in the user-agent section





-> We looked at snort's guide for focusing on user-agent or http header content.



-> This required snort version 3, so we check if we indeed have snort version 3.

```
$ snort -V
--------------------------------------------
o")~   Snort++ 3.1.64.0
--------------------------------------------
```

-> Finally we can modify the alert and run snort against it

```
alert http any any -> any any (msg:"Log4shell Attempt Detected";
http_header:field user-agent; content:"|24 7b|jndi|3a|ldap|3a 2f
2f|",fast_pattern, nocase; sid:10000098; rev:1;)
```

```
alert http any any -> any any (msg:"Log4shell Attempt Detected"; http_header:fie
ld user-agent; content:"|24 7b|jndi|3a|ldap|3a 2f 2f|",fast_pattern, nocase; sid
:10000098; rev:1;)
```

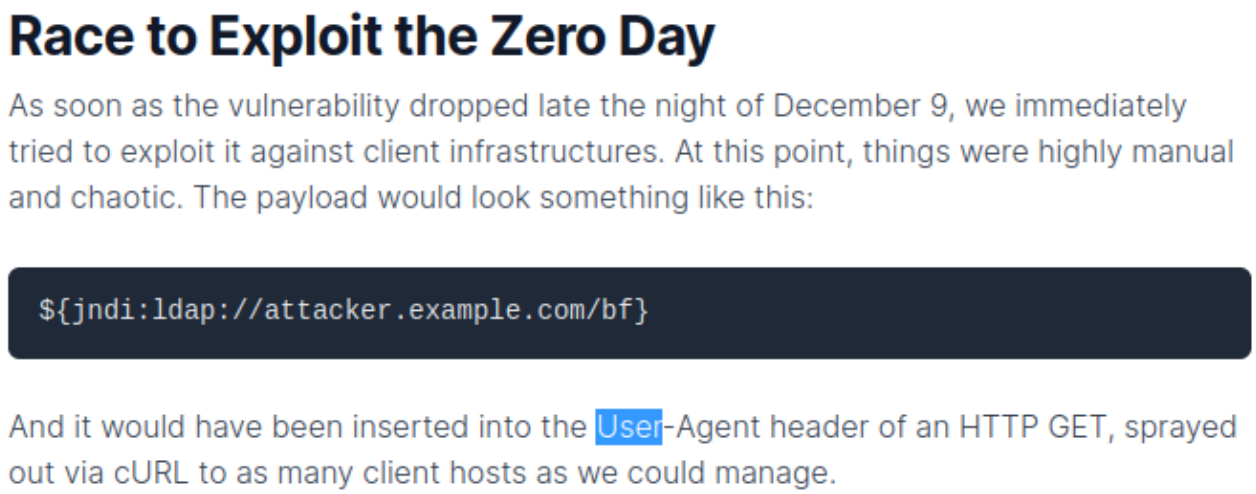
```
sudo snort -c /root/snorty/etc/snort/snort.lua --daq-dir
/usr/local/lib/daq -R /home/htb-student/local.rules -r /home/htb-
student/pcaps/log4shell.pcap -A cmg
```

```
12/12-22:44:32.231645 [**] [1:10000098:1] "Log4shell Attempt Detected" [**] [Pri
ority: 0] {TCP} 175.6.210.66:41158 -> 198.71.247.91:80

http_inspect.http_method[3]:
- - - - - - - - - - - -  - - - - - - - - - - - - - -  - - - - - - - - - -
47 45 54                                            GET
- - - - - - - - - - - -  - - - - - - - - - - - - - -  - - - - - - - - - -

http_inspect.http_version[8]:
- - - - - - - - - - - - - -  - - - - - - - - - - - - - -  - - - - - - - - - -
48 54 54 50 2F 31 2E 31                             HTTP/1.1
- - - - - - - - - - - - - -  - - - - - - - - - - - - - -  - - - - - - - - - -

http_inspect.http_uri[1]:
- - - - - - - - - - - - -  - - - - - - - - - - - - - -  - - - - - - - - - -
2F                                                  /
- - - - - - - - - - - - -  - - - - - - - - - - - - - -  - - - - - - - - - -
```

-> The rule detection worked.

-> Hence, the keyword here is http_header;, the field user-agent we made which sets a sticky buffer to the user-agent header to be more specific.

-> We could also read up on sites like https://bishopfox.com/blog/identify-and-exploit-log4shell to verify and discover that the exploitation does indeed rely on user-agent header on an http request.

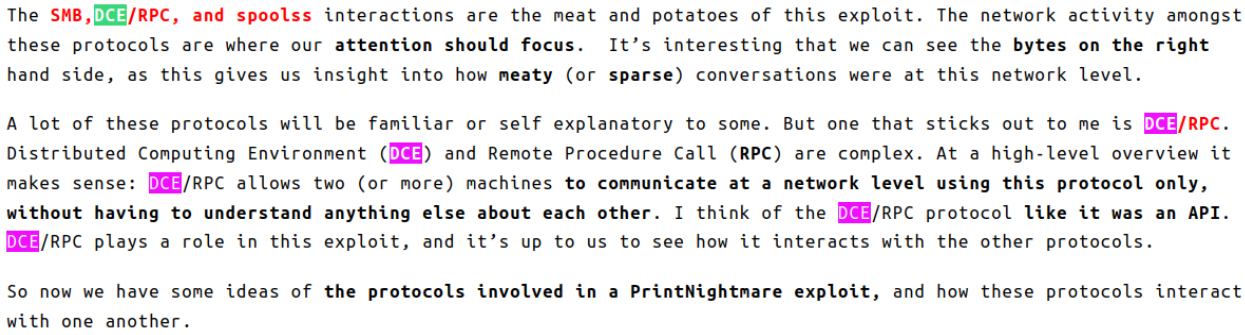


Intrusion Detection With Zeek

Question

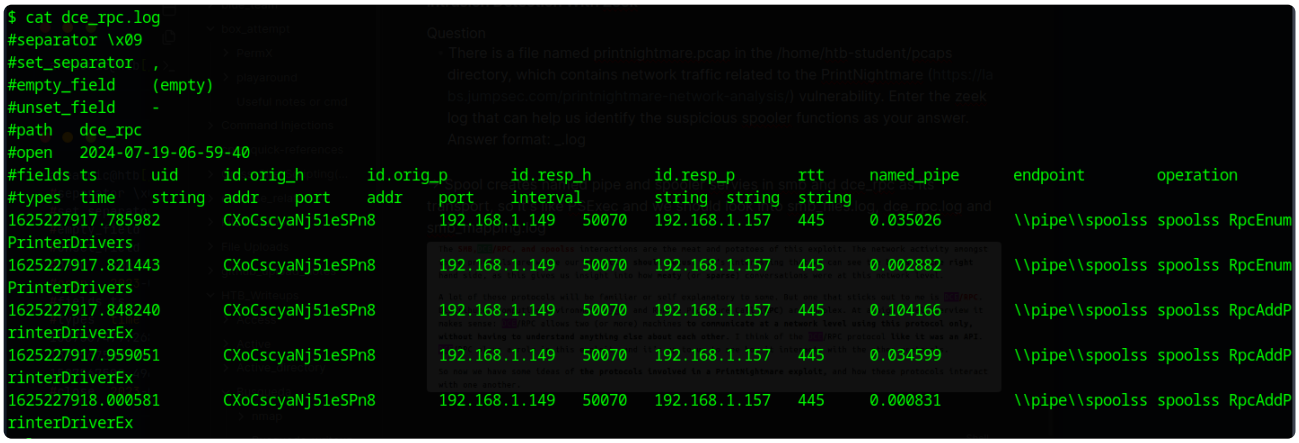
- There is a file named printnightmare.pcap in the /home/htb-student/pcaps directory, which contains network traffic related to the PrintNightmare (https://labs.jumpsec.com/printnightmare-network-analysis/) vulnerability. Enter the zeek log that can help us identify the suspicious spooler functions as your answer. Answer format: _.log

-> Spool creates named pipe and spooler servies in smb and dce_rpc as its transport, so it's like PSExec and we should look into smb_files.log, dce_rpc.log and smb_mapping.log

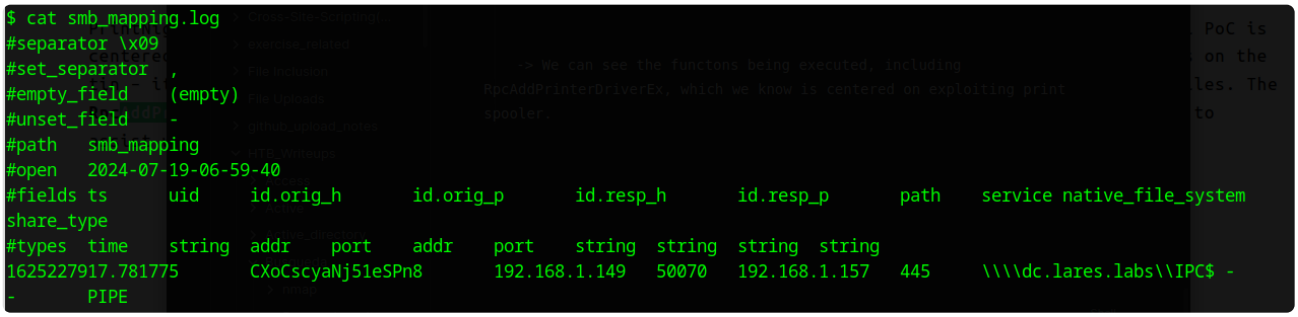


```
/usr/local/zeek/bin/zeek -C -r /home/htb-
student/pcaps/printnightmare.pcap
```

```
cat dce_rpc.log

cat smb_mapping.log
```



```
$ cat dce_rpc.log
#separator \x09
#set_separator  ,
#empty_field    (empty)
#unset_field    -
#path   dce_rpc
#open   2024-07-19-06-59-40
#fields ts      uid     id.orig_h       id.orig_p       id.resp_h       id.resp_p       rtt     named_pipe      endpoint        operation
#types  time    string  addr    port    addr    port    interval        string  string  string
1625227917.785982       CXoCscyaNj51eSPn8       192.168.1.149   50070   192.168.1.157   445     0.035026        \\pipe\\spoolss spoolss RpcEnum
PrinterDrivers
1625227917.821443       CXoCscyaNj51eSPn8       192.168.1.149   50070   192.168.1.157   445     0.002882        \\pipe\\spoolss spoolss RpcEnum
PrinterDrivers
1625227917.848240       CXoCscyaNj51eSPn8       192.168.1.149   50070   192.168.1.157   445     0.104166        \\pipe\\spoolss spoolss RpcAddP
rinterDriverEx
1625227917.959051       CXoCscyaNj51eSPn8       192.168.1.149   50070   192.168.1.157   445     0.034599        \\pipe\\spoolss spoolss RpcAddP
rinterDriverEx
1625227918.000581       CXoCscyaNj51eSPn8       192.168.1.149   50070   192.168.1.157   445     0.000831        \\pipe\\spoolss spoolss RpcAddP
rinterDriverEx
```



```
$ cat smb_mapping.log
#separator \x09
#set_separator  ,
#empty_field    (empty)
#unset_field    -
#path   smb_mapping
#open   2024-07-19-06-59-40
#fields ts      uid     id.orig_h       id.orig_p       id.resp_h       id.resp_p       path    service native_file_system
share_type
#types  time    string  addr    port    addr    port    string  string  string  string
1625227917.781775       CXoCscyaNj51eSPn8       192.168.1.149   50070   192.168.1.157   445     \\\\dc.lares.labs\\IPC$ -
-       PIPE
```

-> We can see the functons being executed, including RpcAddPrinterDriverEx, which we know is centered on exploiting print spooler.
-> We also see an named pipe of IPC being accessed (? not too sure about this).

- There is a file named revilkaseya.pcap in the /home/htb-student/pcaps directory, which contains network traffic related to the REvil ransomware Kaseya supply chain attack. Enter the total number of bytes that the victim has transmitted to the IP address 178.23.155.240 as your answer.
  -> We see that this is an ransomware attack on the cloud service provider Kaseya.

The zero-day vulnerability exploited by REvil gang

For the initial attack vector, REvil operators exploited an authentication bypass in the web interface of the Kaseya VSA server to gain an authenticated session. Then, the attackers uploaded the payload and executed a command via SQL injection to deploy the malicious updates.

The attackers exploited a zero-day vulnerability, tracked as CVE-2021-30116, which was discovered by the Dutch Institute for Vulnerability Disclosure (DIVD). DIVD reported the flaw to Kaseya. The company was validating the patch before they rolled it out to the customers, but REvil ransomware operators exploited the flaw in the massive supply chain ransomware attack.

"During the entire process, Kaseya has shown that they were willing to put in the maximum effort and initiative into this case both to get this issue fixed and their customers patched. They showed a genuine commitment to do the right thing," states an update provided by the Dutch Institute for Vulnerability Disclosure.

"Unfortunately, we were beaten by REvil in the final sprint, as they could exploit the vulnerabilities before customers could even patch."

-> We look first run zeek against it

```
/usr/local/zeek/bin/zeek -C -r /home/htb-student/pcaps/revilkaseya.pcap
```

```
$ /usr/local/zeek/bin/zeek -C -r /home/htb-student/pcaps/revilkaseya.pcap
1623906441.178691 error: connection does not have analyzer specified to disable
1623906441.296930 error: connection does not have analyzer specified to disable
1623906482.449649 error: connection does not have analyzer specified to disable
1623906482.559396 error: connection does not have analyzer specified to disable
1623906491.549771 error: connection does not have analyzer specified to disable
1623906492.154193 error: connection does not have analyzer specified to disable
1623906501.841714 error: connection does not have analyzer specified to disable
1623906501.958803 error: connection does not have analyzer specified to disable
```

-> we now examine the log associated with ip address and look at the sum of bytes send

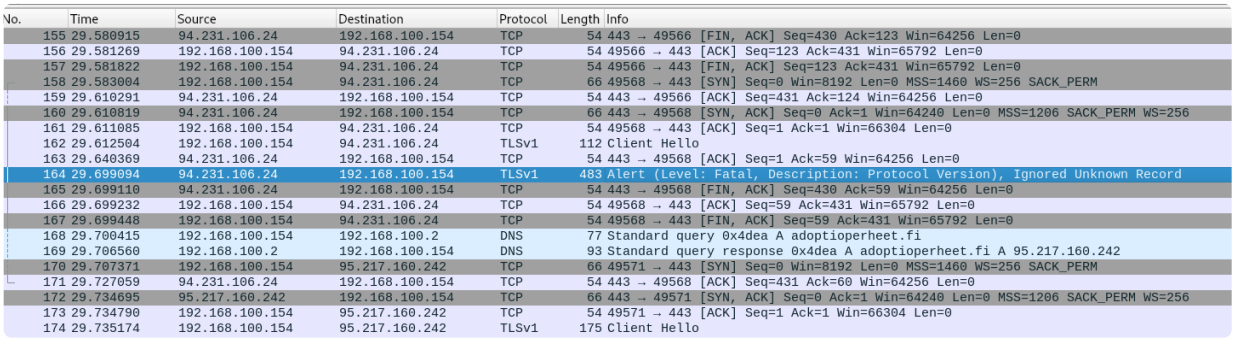
```
cat conn.log | /usr/local/zeek/bin/zeek-cut id.orig_h id.resp_h
orig_bytes | grep 178.23.155.240 | datamash -g 1,2 sum 3
```

```
$ cat conn.log | /usr/local/zeek/bin/zeek-cut id.orig_h id.resp_h orig_bytes | g
rep 178.23.155.240 | datamash -g 1,2 sum 3
192.168.100.154 178.23.155.240  2311
```

-> Hence, we have 2311 bytes send from 192.168.100.154 to 178.23.155.240.

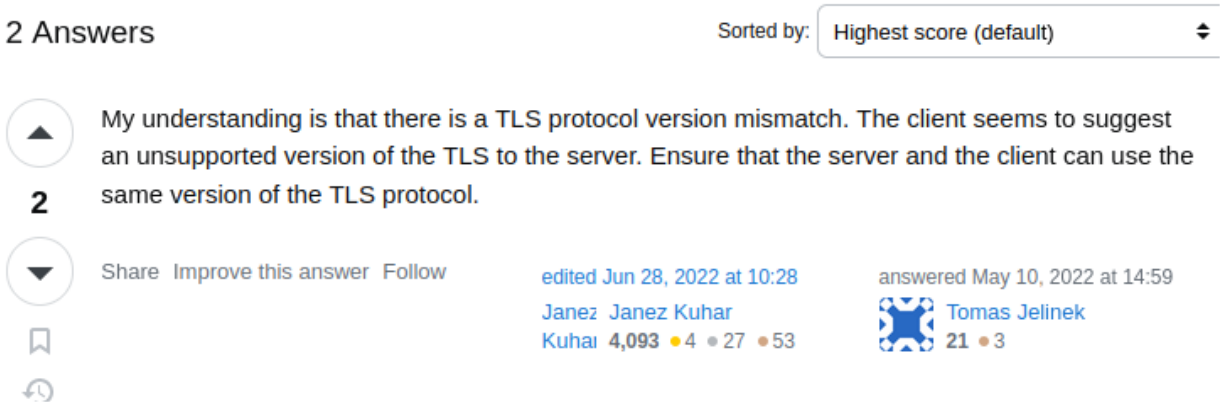
Additional examination of pcap to better understand the context of the ransomware

- Looking at pcap file of the unsure alert



No.	Time	Source	Destination	Protocol	Length	Info
155	29.580915	94.231.106.24	192.168.100.154	TCP	54	443 → 49566 [FIN, ACK] Seq=430 Ack=123 Win=64256 Len=0
156	29.581269	192.168.100.154	94.231.106.24	TCP	54	49566 → 443 [ACK] Seq=123 Ack=431 Win=65792 Len=0
157	29.581822	192.168.100.154	94.231.106.24	TCP	54	49566 → 443 [FIN, ACK] Seq=123 Ack=431 Win=65792 Len=0
158	29.583004	192.168.100.154	94.231.106.24	TCP	66	49568 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
159	29.610291	94.231.106.24	192.168.100.154	TCP	54	443 → 49566 [ACK] Seq=431 Ack=124 Win=64256 Len=0
160	29.610819	94.231.106.24	192.168.100.154	TCP	66	443 → 49568 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1206 SACK_PERM WS=256
161	29.611085	192.168.100.154	94.231.106.24	TCP	54	49568 → 443 [ACK] Seq=1 Ack=1 Win=66304 Len=0
162	29.612504	192.168.100.154	94.231.106.24	TLSv1	112	Client Hello
163	29.640369	94.231.106.24	192.168.100.154	TCP	54	443 → 49568 [ACK] Seq=1 Ack=59 Win=64256 Len=0
164	29.699094	94.231.106.24	192.168.100.154	TLSv1	483	Alert (Level: Fatal, Description: Protocol Version), Ignored Unknown Record
165	29.699110	94.231.106.24	192.168.100.154	TCP	54	443 → 49568 [FIN, ACK] Seq=430 Ack=59 Win=64256 Len=0
166	29.699232	192.168.100.154	94.231.106.24	TCP	54	49568 → 443 [ACK] Seq=59 Ack=431 Win=65792 Len=0
167	29.699448	192.168.100.154	94.231.106.24	TCP	54	49568 → 443 [FIN, ACK] Seq=59 Ack=431 Win=65792 Len=0
168	29.700415	192.168.100.154	192.168.100.2	DNS	77	Standard query 0x4dea A adoptioperheet.fi
169	29.706560	192.168.100.2	192.168.100.154	DNS	93	Standard query response 0x4dea A adoptioperheet.fi A 95.217.160.242
170	29.707371	192.168.100.154	95.217.160.242	TCP	66	49571 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
171	29.727059	94.231.106.24	192.168.100.154	TCP	54	443 → 49568 [ACK] Seq=431 Ack=60 Win=64256 Len=0
172	29.734695	95.217.160.242	192.168.100.154	TCP	66	443 → 49571 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1206 SACK_PERM WS=256
173	29.734790	192.168.100.154	95.217.160.242	TCP	54	49571 → 443 [ACK] Seq=1 Ack=1 Win=66304 Len=0
174	29.735174	192.168.100.154	95.217.160.242	TLSv1	175	Client Hello

-> Client seems to be using unsupported version of TLS protocol.
https://stackoverflow.com/questions/67820241/tls-1-2-alert-level-fatal-description-protocol-version



2 Answers

Sorted by: Highest score (default)

2

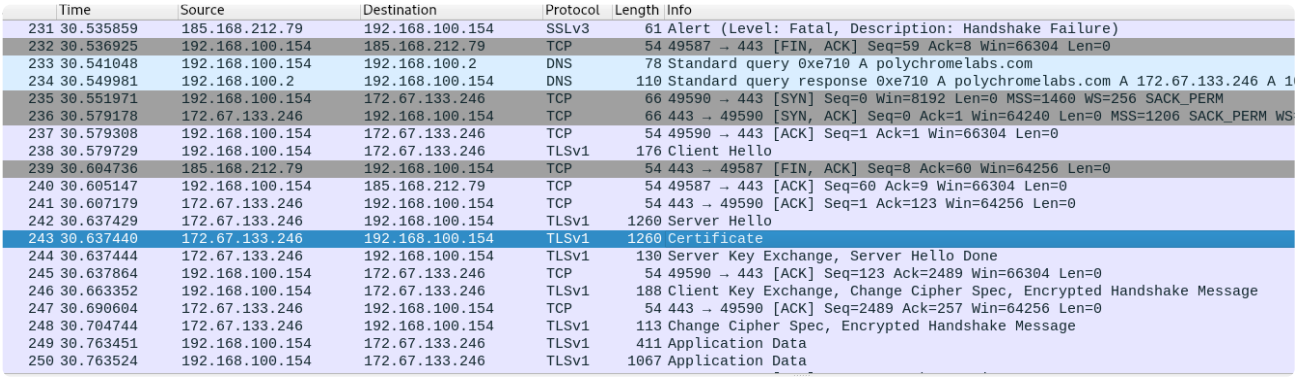
My understanding is that there is a TLS protocol version mismatch. The client seems to suggest an unsupported version of the TLS to the server. Ensure that the server and the client can use the same version of the TLS protocol.

Share Improve this answer Follow

edited Jun 28, 2022 at 10:28
Janez Janez Kuhar
Kuhar 4,093 ● 4 ● 27 ● 53

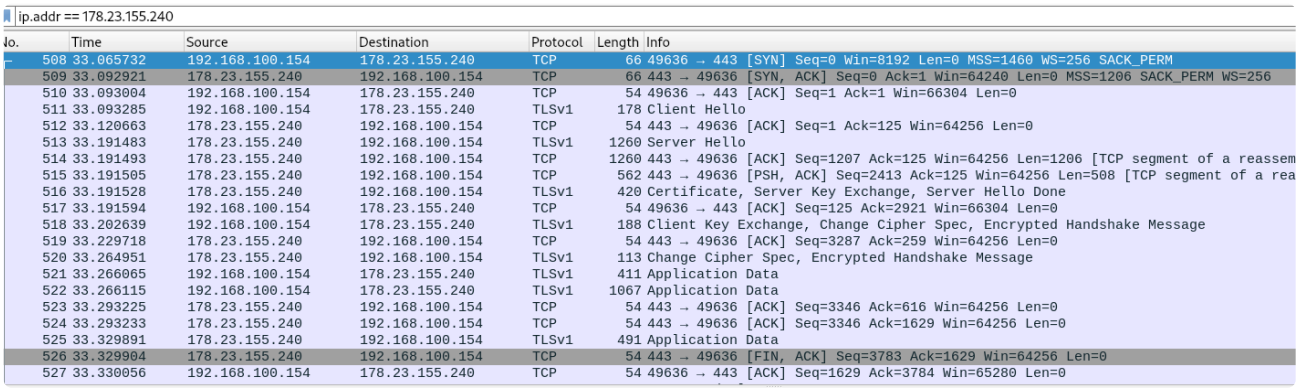
answered May 10, 2022 at 14:59
Tomas Jelinek
21 ● 3

-> Flow of conversation: host accessing lots of website and alot of them not working (SSL/TLS version not supported), maybe some of those are attacker website.
-> The first conversation we see successful:



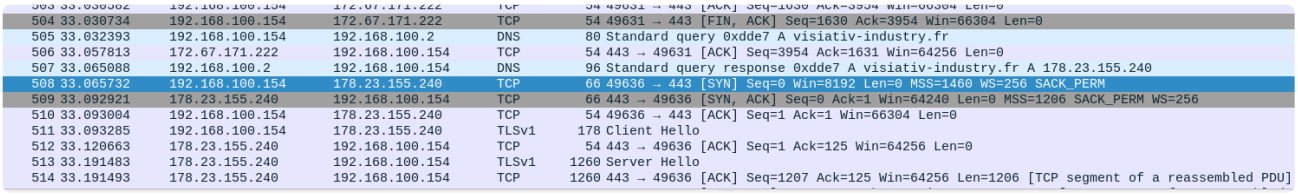
	Time	Source	Destination	Protocol	Length	Info
231	30.535859	185.168.212.79	192.168.100.154	SSLv3	61	Alert (Level: Fatal, Description: Handshake Failure)
232	30.536925	192.168.100.154	185.168.212.79	TCP	54	49587 → 443 [FIN, ACK] Seq=59 Ack=8 Win=66304 Len=0
233	30.541048	192.168.100.154	192.168.100.2	DNS	78	Standard query 0xe710 A polychromelabs.com
234	30.549981	192.168.100.2	192.168.100.154	DNS	110	Standard query response 0xe710 A polychromelabs.com A 172.67.133.246 A 1
235	30.551971	192.168.100.154	172.67.133.246	TCP	66	49590 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
236	30.579178	172.67.133.246	192.168.100.154	TCP	66	443 → 49590 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1206 SACK_PERM WS
237	30.579308	192.168.100.154	172.67.133.246	TCP	54	49590 → 443 [ACK] Seq=1 Ack=1 Win=66304 Len=0
238	30.579729	192.168.100.154	172.67.133.246	TLSv1	176	Client Hello
239	30.604736	185.168.212.79	192.168.100.154	TCP	54	443 → 49587 [FIN, ACK] Seq=8 Ack=60 Win=64256 Len=0
240	30.605147	192.168.100.154	185.168.212.79	TCP	54	49587 → 443 [ACK] Seq=60 Ack=9 Win=66304 Len=0
241	30.607179	172.67.133.246	192.168.100.154	TCP	54	443 → 49590 [ACK] Seq=1 Ack=123 Win=64256 Len=0
242	30.637429	172.67.133.246	192.168.100.154	TLSv1	1260	Server Hello
243	30.637440	172.67.133.246	192.168.100.154	TLSv1	1260	Certificate
244	30.637444	172.67.133.246	192.168.100.154	TLSv1	130	Server Key Exchange, Server Hello Done
245	30.637864	192.168.100.154	172.67.133.246	TCP	54	49590 → 443 [ACK] Seq=123 Ack=2489 Win=66304 Len=0
246	30.663352	192.168.100.154	172.67.133.246	TLSv1	188	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
247	30.690604	172.67.133.246	192.168.100.154	TCP	54	443 → 49590 [ACK] Seq=2489 Ack=257 Win=64256 Len=0
248	30.704744	172.67.133.246	192.168.100.154	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
249	30.763451	192.168.100.154	172.67.133.246	TLSv1	411	Application Data
250	30.763524	192.168.100.154	172.67.133.246	TLSv1	1067	Application Data

For 172.23.155.240:



No.	Time	Source	Destination	Protocol	Length	Info
508	33.065732	192.168.100.154	178.23.155.240	TCP	66	49636 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
509	33.092921	178.23.155.240	192.168.100.154	TCP	66	443 → 49636 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1206 SACK_PERM WS=256
510	33.093004	192.168.100.154	178.23.155.240	TCP	54	49636 → 443 [ACK] Seq=1 Ack=1 Win=66304 Len=0
511	33.093285	192.168.100.154	178.23.155.240	TLSv1	178	Client Hello
512	33.120663	178.23.155.240	192.168.100.154	TCP	54	443 → 49636 [ACK] Seq=1 Ack=125 Win=64256 Len=0
513	33.191483	178.23.155.240	192.168.100.154	TLSv1	1260	Server Hello
514	33.191493	178.23.155.240	192.168.100.154	TCP	1260	443 → 49636 [ACK] Seq=1207 Ack=125 Win=64256 Len=1206 [TCP segment of a reassem
515	33.191505	178.23.155.240	192.168.100.154	TCP	562	443 → 49636 [PSH, ACK] Seq=2413 Ack=125 Win=64256 Len=508 [TCP segment of a rea
516	33.191528	178.23.155.240	192.168.100.154	TLSv1	420	Certificate, Server Key Exchange, Server Hello Done
517	33.191594	192.168.100.154	178.23.155.240	TCP	54	49636 → 443 [ACK] Seq=125 Ack=2921 Win=66304 Len=0
518	33.202639	192.168.100.154	178.23.155.240	TLSv1	188	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
519	33.229718	178.23.155.240	192.168.100.154	TCP	54	443 → 49636 [ACK] Seq=3287 Ack=259 Win=64256 Len=0
520	33.264951	178.23.155.240	192.168.100.154	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
521	33.266065	192.168.100.154	178.23.155.240	TLSv1	411	Application Data
522	33.266115	192.168.100.154	178.23.155.240	TLSv1	1067	Application Data
523	33.293225	178.23.155.240	192.168.100.154	TCP	54	443 → 49636 [ACK] Seq=3346 Ack=616 Win=64256 Len=0
524	33.293233	178.23.155.240	192.168.100.154	TCP	54	443 → 49636 [ACK] Seq=3346 Ack=1629 Win=64256 Len=0
525	33.329891	178.23.155.240	192.168.100.154	TLSv1	491	Application Data
526	33.329904	178.23.155.240	192.168.100.154	TCP	54	443 → 49636 [FIN, ACK] Seq=3783 Ack=1629 Win=64256 Len=0
527	33.330056	192.168.100.154	178.23.155.240	TCP	54	49636 → 443 [ACK] Seq=1629 Ack=3784 Win=65280 Len=0

-> Looking at the dns query for such host



504	33.030734	192.168.100.154	172.67.171.222	TCP	54	49631 → 443 [FIN, ACK] Seq=1630 Ack=3954 Win=66304 Len=0
505	33.032393	192.168.100.154	192.168.100.2	DNS	80	Standard query 0xdde7 A visiativ-industry.fr
506	33.057813	172.67.171.222	192.168.100.154	TCP	54	443 → 49631 [ACK] Seq=3954 Ack=1631 Win=64256 Len=0
507	33.065088	192.168.100.2	192.168.100.154	DNS	96	Standard query response 0xdde7 A visiativ-industry.fr A 178.23.155.240
508	33.065732	192.168.100.154	178.23.155.240	TCP	66	49636 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
509	33.092921	178.23.155.240	192.168.100.154	TCP	66	443 → 49636 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1206 SACK_PERM WS=256
510	33.093004	192.168.100.154	178.23.155.240	TCP	54	49636 → 443 [ACK] Seq=1 Ack=1 Win=66304 Len=0
511	33.093285	192.168.100.154	178.23.155.240	TLSv1	178	Client Hello
512	33.120663	178.23.155.240	192.168.100.154	TCP	54	443 → 49636 [ACK] Seq=1 Ack=125 Win=64256 Len=0
513	33.191483	178.23.155.240	192.168.100.154	TLSv1	1260	Server Hello
514	33.191493	178.23.155.240	192.168.100.154	TCP	1260	443 → 49636 [ACK] Seq=1207 Ack=125 Win=64256 Len=1206 [TCP segment of a reassembled PDU]

-> The host of interest is visiativ-industry.fr.

Skills Assessment - Suricata\

Suricata Rule Development Exercise: Detecting WMI Execution (Through WMIExec)

PCAP source: https://github.com/elcabezzonn/Pcaps

Attack description and possible detection points:
https://labs.withsecure.com/publications/attack-detection-fundamentals-discovery-and-lateral-movement-lab-5

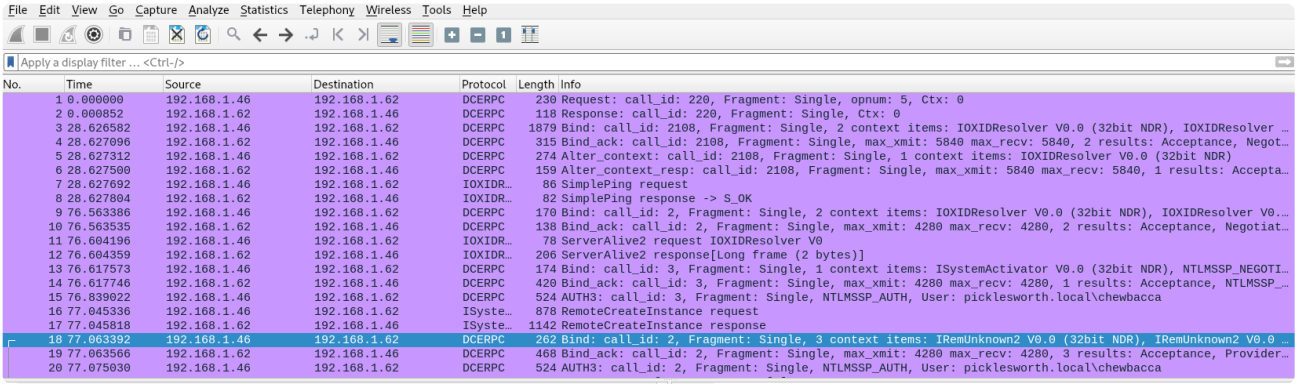
- `Windows Management Instrumentation (WMI)` is a powerful feature in the Windows operating system that allows for management tasks, such as the execution of code or management of devices, both locally and remotely.
  - As you might expect, this can be a very enticing tool for attackers who are seeking to execute malicious activities remotely.
- To detect WMI execution (through `wmiexec`) over the network, we need to focus on the `SMB (Server Message Block)` and `DCOM (Distributed Component Object Model)` protocols, which are the primary means by which remote WMI execution is accomplished.
- One method an attacker might use is to create a `Win32_Process` via the WMI service.
  - In this instance, the attacker would create an instance of `Win32_ProcessStartup`, set its properties to control the environment of the new process, then call the `Create` method to start a new process such as `cmd.exe` or `powershell.exe`.

Question

- There is a file named pipekatposhc2.pcap in the /home/htb-student/pcaps directory, which contains network traffic related to WMI execution.
    - Add yet another content keyword right after the msg part of the rule with sid 2024233 within the local.rules file so that an alert is triggered and enter the specified payload as your answer. Answer format: C____e

-> Refer to the preliminary knowledge for things that you are unsure, and we first examine the network traffic

-> Lots of DCE/RPC protocol found:



-> Following one of the stream, we see Win32 Process with a famiiar UUID that represents CIM_Process class

```
...w.i.n.3.2._.p.r.o.c.e.s.s.....................
.............(Mbf.................. ....P.....................P...P..MEOW.........s...M...K.$...E:.........K.$....eP..xV
4.]P....SNICKLEFRITZ..ROOT\cimv2..  .......W...:....CIM_LogicalElement......CIM_ManagedSystemElement.....)...
...................       ............%.......M...V.....................t.........................*...2.............
......H...N...........0...8..............
.......  ...T...b............WUU.....................................................................................CI
M_Process..Abstract..Locale..UUID..{8502C566-5FBB-11D2-AAC1-006008C78BC7}..Caption..@............)...
...#............"....@......."........string..MaxLen..CreationClassName..............4...
.....................................................string..CIM_Key..MaxLen..CreationDate.e..... .......'...
........N...X.....................datetime..Fixed..CSCreationClassName..............A...
.............................................       .............string..Propagated..CIM_OperatingSystem.CSCreationClas
sName..CIM_Key..MaxLen..CSName..............A...
........h...p........|.....................................string..Propagated..CIM_OperatingSystem.CSName..CIM_Key..Max
Len..Description..@.................
...#............".......string..ExecutionState.......<...........
........".............uint16..Handle.....  .$.......4...
........t.........................|.............string..MaxLen..InstallDate.e@............)...
...#............". ..........".......datetime..MappingStrings..........MIF.DMTF|ComponentID|001.5..KernelModeTime.....
.(............
........@.............uint64..Name..@.................
...#....x........".......string..OSCreationClassName.......0.......A...
.............................................(.............string..Propagated..CIM_OperatingSystem.CreationClassName..C
IM_Key..MaxLen..OSName.......4.......A...
.....................................................................string..Propagated..CIM_OperatingSystem.Name..CIM_Key..MaxLe
n..Priority.....
.8...........
.........................uint32..Status..@............6...
...#....N........".......V...".....
...^...". ..h....string..MaxLen..ValueMap.....................................................OK..Error..Degraded..Unkno
wn..Pred Fail..Starting..Stopping..Service..Stressed..NonRecover..No Contact..Lost Comm..TerminationDate.e.....>.........
```

-> We also see that the following for Win32_ProcessStartup class

-> Default rule looks the following

```
alert tcp any any -> any any (msg:"WMI Execution Detected"; content:"Win32_Proce
ssStartup"; content:"powershell"; sid:2024233; rev:2;)
```

-> We change it to Create, because we know that after Win32_ProcessStartup, it would usually call Create function to create the process.

```
alert tcp any any -> any any (msg:"WMI Execution Detected"; content:"Create"; co
ntent:"Win32_ProcessStartup"; content:"powershell"; sid:2024233; rev:2;)
```

-> This hypothesis is also verified in the network traffic analysis (see CreateProcess)

-> Saving this rule and running suricata, we obtained that:

```
sudo suricata -r /home/htb-student/pcaps/pipekatposhc2.pcap -l . -k none
```

```
$ sudo suricata -r /home/htb-student/pcaps/pipekatposhc2.pcap -l . -k none
21/7/2024 -- 21:25:07 - <Notice> - This is Suricata version 4.0.0-beta1 RELEASE
21/7/2024 -- 21:25:07 - <Notice> - all 5 packet processing threads, 4 management
 threads initialized, engine started.
21/7/2024 -- 21:25:10 - <Notice> - Signal Received.  Stopping engine.
21/7/2024 -- 21:25:10 - <Notice> - Pcap-file module read 84121 packets, 14899668
 bytes
$ ls
eve.json  keyword_perf.log  rule_group_perf.log  stats.log
fast.log  packet_stats.log  rule_perf.log
$ cat fast.log
12/26/2019-08:04:55.353819  [**] [1:2024233:2] WMI Execution Detected [**] [Clas
sification: (null)] [Priority: 3] {TCP} 192.168.1.46:58198 -> 192.168.1.62:49154
```

-> And the rule indeed works, so we confirmed that the content we have to add is create.

Skills Assessment - Snort

Snort Rule Development Exercise: Detecting Overpass-the-Hash

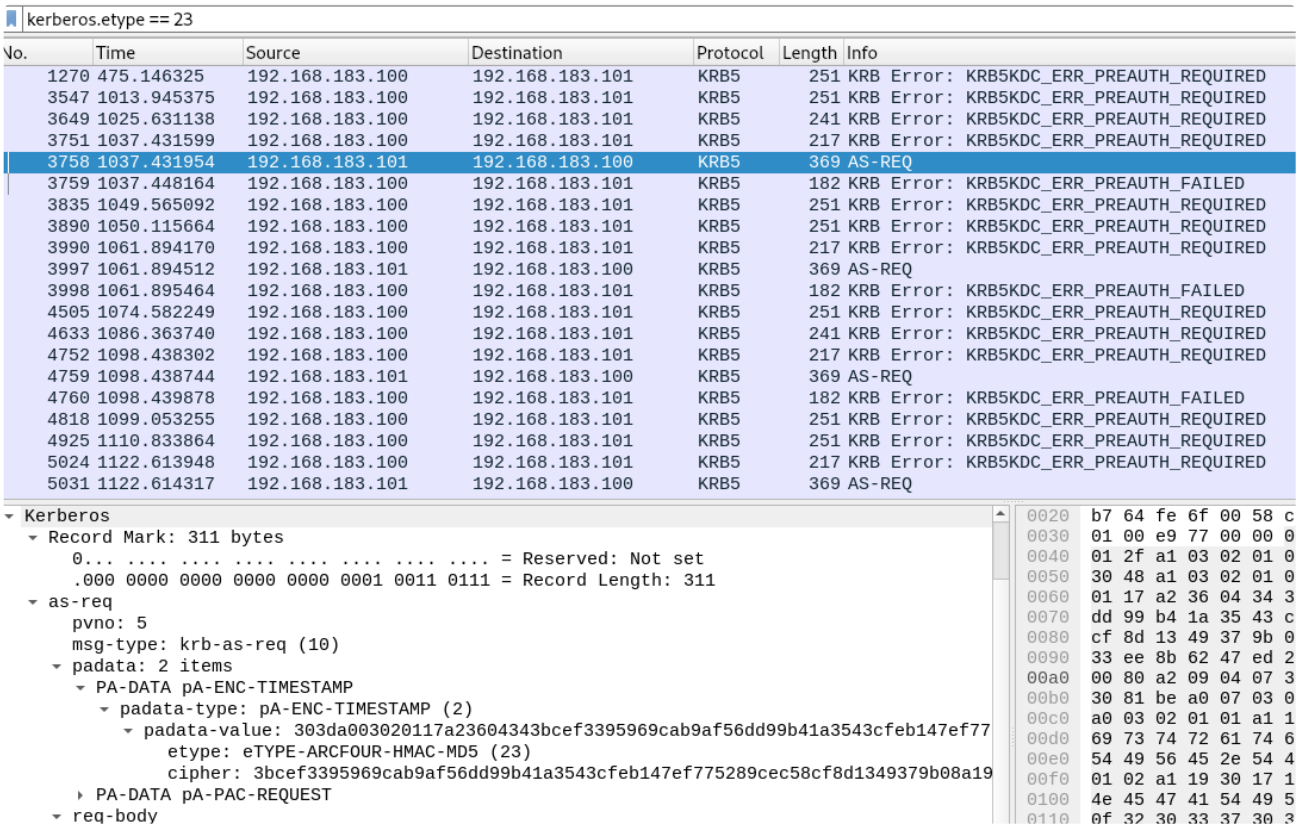
PCAP source: https://github.com/elcabezzonn/Pcaps

Attack description and possible detection points:
http://www.labofapenetrationtester.com/2017/08/week-of-evading-microsoft-ata-day2.html

- `Overpass-the-Hash (Pass-the-Key)` is a type of attack where an adversary gains unauthorized access to resources by using a stolen `NTLM (NT LAN Manager)` hash or Kerberos key, without needing to crack the password from which the hash was derived.
  - The attack involves using the hash to create a `Kerberos TGT (Ticket-Granting Ticket)` to authenticate to Active Directory (AD).
- When the adversary utilizes Overpass-the-Hash, they have the `NTLM` hash of the user's password, which is used to craft an `AS-REQ (Authentication Service Request)` to the `Key Distribution Center (KDC)`. To appear authentic, the `AS-REQ` contains a `PRE-AUTH` field, which contains an encrypted timestamp (`Enc-Timestamp`).
  - This is normally used by a legitimate client to prove knowledge of the user's password, as it is encrypted using the user's password hash.
  - In this attack scenario, the hash used to encrypt the timestamp is not derived from the actual password but rather it is the stolen NTLM hash.
  - More specifically, in an `Overpass-the-Hash` attack the attacker doesn't use this hash to encrypt the `Enc-Timestamp`.
  - Instead, the attacker directly uses the stolen NTLM hash to compute the Kerberos `AS-REQ`, bypassing the usual Kerberos process that would involve the user's password and the `Enc-Timestamp`.
  - The attacker essentially "overpasses" the normal password-based authentication process, hence the name Overpass-the-Hash.
- One key aspect of this type of attack that we can leverage for detection is the `encryption type` used for the `Enc-Timestamp`.
  - A standard `AS-REQ` from a modern Windows client will usually use the `AES256-CTS-HMAC-SHA1-96` encryption type for the Enc-Timestamp, but an `Overpass-the-Hash` attack using the older NTLM hash will use the `RC4-HMAC` encryption type.
  - This discrepancy can be used as an indicator of a potential attack.

-> Look for kerberos encryption type 23 (RC4/ntlm) encryption for the kerberos protocol in AS-REQ during the Kerberos authentication process in the wireshark packet capture:

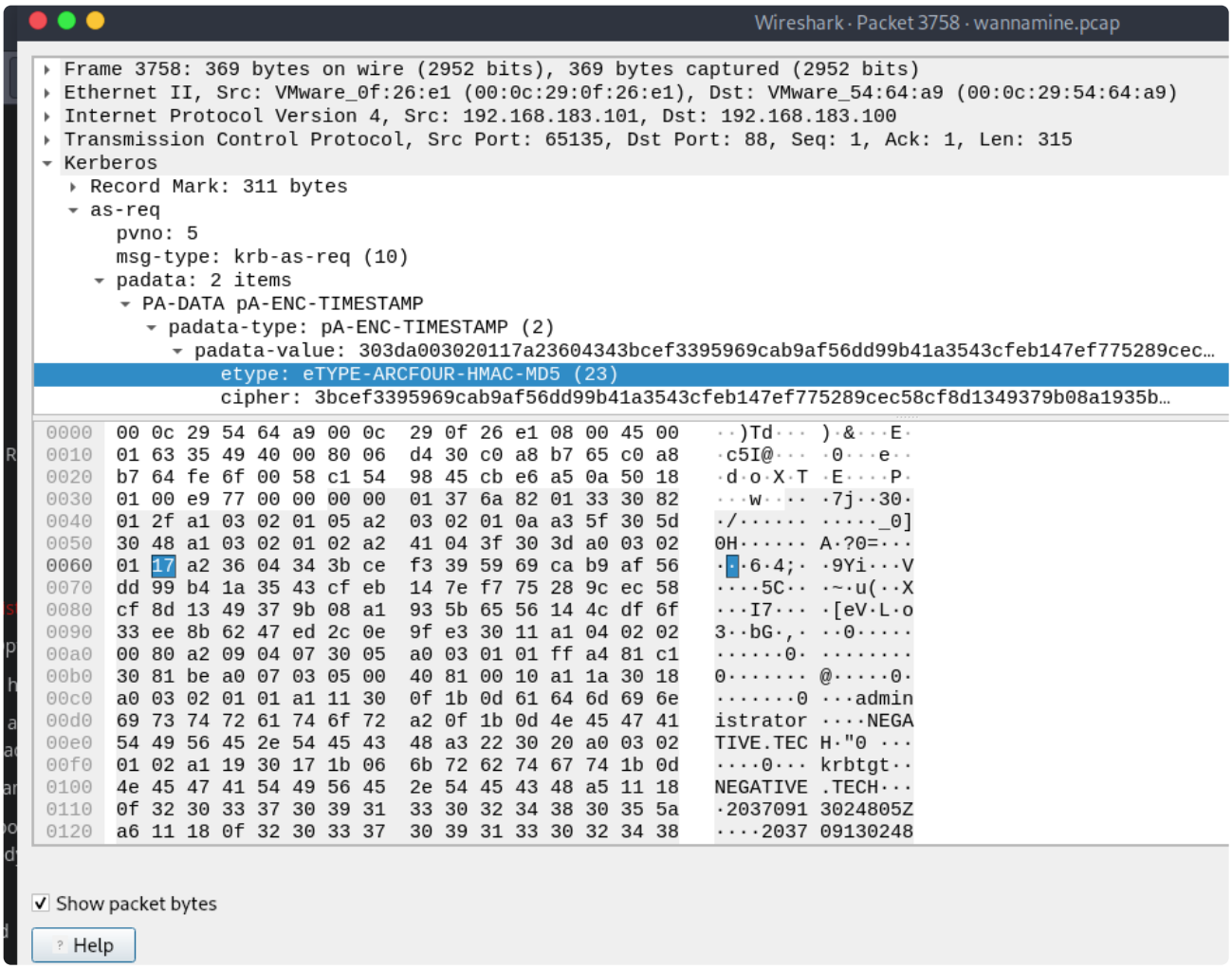
-> We see one example of encrypted time stamp using ntlm after filtering, which could be an attempt on Overpass-the-Hash attack without obtaining the user's password (if given the option, attacker's would choose the AES-256 encryption method).



- Initial rule examainiation

```
alert tcp $HOME_NET any -> any 88 (msg: "Kerberos Ticket Encryption Downgrade to
 RC4 Detected"; flow: no_stream, established, to_server; content: "|A1 03 02 01
05 A2 03 02 01 0A|", offset 12, depth 10; content: "|A1 03 02 01 02|", distance
5,  within 6; content: "|A0 03 02 01 XX|",  distance 6,  within 6; content: "krb
tgt", distance 0; sid:9999999;)
```

-> Taking a closer look at one of the as-req message, we see it is 17, which is a byte in the Kerberos encryption message.



-> Changed the XX to 17.

```
alert tcp $HOME_NET any -> any 88 (msg: "Kerberos Ticket Encryption Downgrade to
 RC4 Detected"; flow: no_stream, established, to_server; content: "|A1 03 02 01
05 A2 03 02 01 0A|", offset 12, depth 10; content: "|A1 03 02 01 02|", distance
5, within 6; content: "|A0 03 02 01 17|", distance 6, within 6; content: "krb
tgt", distance 0; sid:9999999;)
-- INSERT --                                                    19,280        Bot
```

-> Now running snort on the packet capture and examining the result

```
sudo snort -c /root/snorty/etc/snort/snort.lua --daq-dir
/usr/local/lib/daq -R /home/htb-student/local.rules -r /home/htb-
student/pcaps/wannamine.pcap -A cmg
```

```
03/05-11:20:55.492747 [**] [1:9999999:0] "Kerberos Ticket Encryption Downgrade t
o RC4 Detected" [**] [Priority: 0] {TCP} 192.168.183.101:65194 -> 192.168.183.10
0:88
00:0C:29:0F:26:E1 -> 00:0C:29:54:64:A9 type:0x800 len:0x171
192.168.183.101:65194 -> 192.168.183.100:88 TCP TTL:128 TOS:0x0 ID:14379 IpLen:2
0 DgmLen:355 DF
***AP*** Seq: 0x2FB94D57  Ack: 0x51499B4C  Win: 0x100  TcpLen: 20

snort.raw[315]:
```

-> And we see the alert.

- There is a file named neutrinogootkit.pcap in the /home/htb-student/pcaps directory, which contains network traffic related to the Neutrino exploit kit sending Gootkit malware.
    - Enter the x509.log field name that includes the "MyCompany Ltd." trace as your answer.

-> SSL traffic info on related links

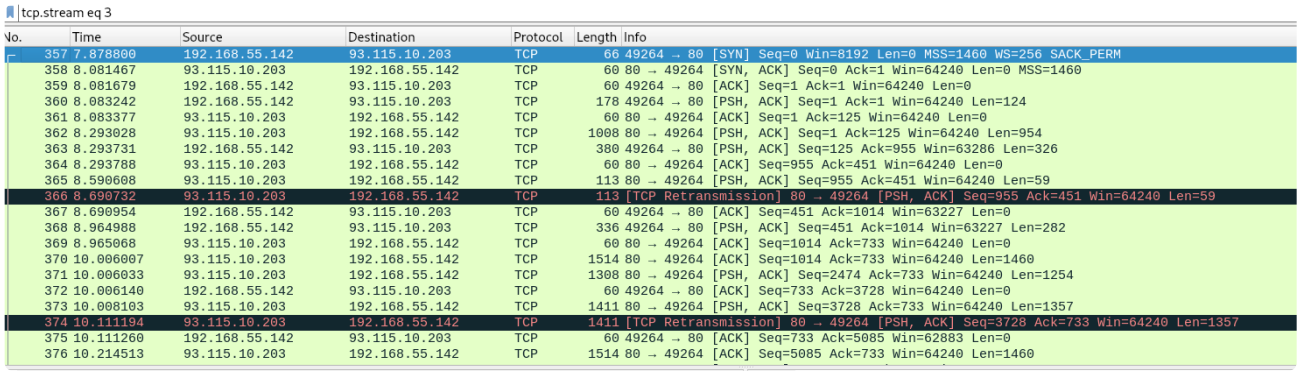
ASSOCIATED DOMAINS:

- 85.93.0.43 port 80 - **fin7.tk** - EITest flash redirect
- 69.162.116.166 port 80 - **primordia.home-office-desk-chair.co.uk** - EITest Neutrino EK
- 69.162.116.166 port 80 - **tnessaarbeitsjubilaeum.home-office-desk-chair.co.uk** - pseudoDarkleech Neutrino EK
- 188.138.125.31 port 80 - **zugeschldorstyd.lightingandmirrors.co.uk** - pseudoDarkleech Neutrino EK
- 188.0.236.9 port 443 - CryptXXX post-infection traffic, custom encoded (not SSL)

- 5.2.72.163 port 80 - **tcptz.iwhateverblue.top** - Other Neutrino EK
- 93.115.10.203 port 80 - **domptorang.com** - SSL traffic with MyCompany Ltd cert (probable Gootkit callback)

- 93.190.140.110 port 80 - **cosmos.furnipict.com** - GET /gsvot2.html - Redirector from malvertising
- 5.61.40.131 port 80 - **wrbktrebtwb.droneboneyard.net** - Rig EK
- 58.64.142.89 port 80 - **laoismacau.com** - Cryptobit post-infection traffic

-> We examine the pcap file with the given ip address and look at some of traffic:



No.	Time	Source	Destination	Protocol	Length	Info
357	7.878800	192.168.55.142	93.115.10.203	TCP	66	49264 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
358	8.081467	93.115.10.203	192.168.55.142	TCP	60	80 → 49264 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
359	8.081679	192.168.55.142	93.115.10.203	TCP	60	49264 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
360	8.083242	192.168.55.142	93.115.10.203	TCP	178	49264 → 80 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=124
361	8.083377	93.115.10.203	192.168.55.142	TCP	60	80 → 49264 [ACK] Seq=1 Ack=125 Win=64240 Len=0
362	8.293028	93.115.10.203	192.168.55.142	TCP	1008	80 → 49264 [PSH, ACK] Seq=1 Ack=125 Win=64240 Len=954
363	8.293731	192.168.55.142	93.115.10.203	TCP	380	49264 → 80 [PSH, ACK] Seq=125 Ack=955 Win=63286 Len=326
364	8.293788	93.115.10.203	192.168.55.142	TCP	60	80 → 49264 [ACK] Seq=955 Ack=451 Win=64240 Len=0
365	8.590608	93.115.10.203	192.168.55.142	TCP	113	80 → 49264 [PSH, ACK] Seq=955 Ack=451 Win=64240 Len=59
366	8.690732	93.115.10.203	192.168.55.142	TCP	113	[TCP Retransmission] 80 → 49264 [PSH, ACK] Seq=955 Ack=451 Win=64240 Len=59
367	8.690954	192.168.55.142	93.115.10.203	TCP	60	49264 → 80 [ACK] Seq=451 Ack=1014 Win=63227 Len=0
368	8.964988	192.168.55.142	93.115.10.203	TCP	336	49264 → 80 [PSH, ACK] Seq=451 Ack=1014 Win=63227 Len=282
369	8.965068	93.115.10.203	192.168.55.142	TCP	60	80 → 49264 [ACK] Seq=1014 Ack=733 Win=64240 Len=0
370	10.006007	93.115.10.203	192.168.55.142	TCP	1514	80 → 49264 [ACK] Seq=1014 Ack=733 Win=64240 Len=1460
371	10.006033	93.115.10.203	192.168.55.142	TCP	1308	80 → 49264 [PSH, ACK] Seq=2474 Ack=733 Win=64240 Len=1254
372	10.006140	192.168.55.142	93.115.10.203	TCP	60	49264 → 80 [ACK] Seq=733 Ack=3728 Win=64240 Len=0
373	10.008103	93.115.10.203	192.168.55.142	TCP	1411	80 → 49264 [PSH, ACK] Seq=3728 Ack=733 Win=64240 Len=1357
374	10.111194	93.115.10.203	192.168.55.142	TCP	1411	[TCP Retransmission] 80 → 49264 [PSH, ACK] Seq=3728 Ack=733 Win=64240 Len=1357
375	10.111260	192.168.55.142	93.115.10.203	TCP	60	49264 → 80 [ACK] Seq=733 Ack=5085 Win=62883 Len=0
376	10.214513	93.115.10.203	192.168.55.142	TCP	1514	80 → 49264 [ACK] Seq=5085 Ack=733 Win=64240 Len=1460

-> We examine some of the messages and we see the name MyCompany Ltd, which could

be indicative of certificate name (given the blue colour denotes server packets)

-> We run zeek against it:

```
/usr/local/zeek/bin/zeek -C -r /home/htb-
student/pcaps/neutrinogootkit.pcap
```

-> We also see from zeek that the possible fields can be certificate.issuer or certificate.subject

x509.log

In the last section we looked at Zeek's `ssl.log`, a source which offered details on TLS connections. In this section we will look an associated source, Zeek's `x509.log`. The `x509.log` captures details on certificates exchanged during certain TLS negotiations. We will compare sessions using TLS 1.2 and TLS 1.3

For details on all of the fields in the `x509.log`, please refer to **`X509::Info`**.

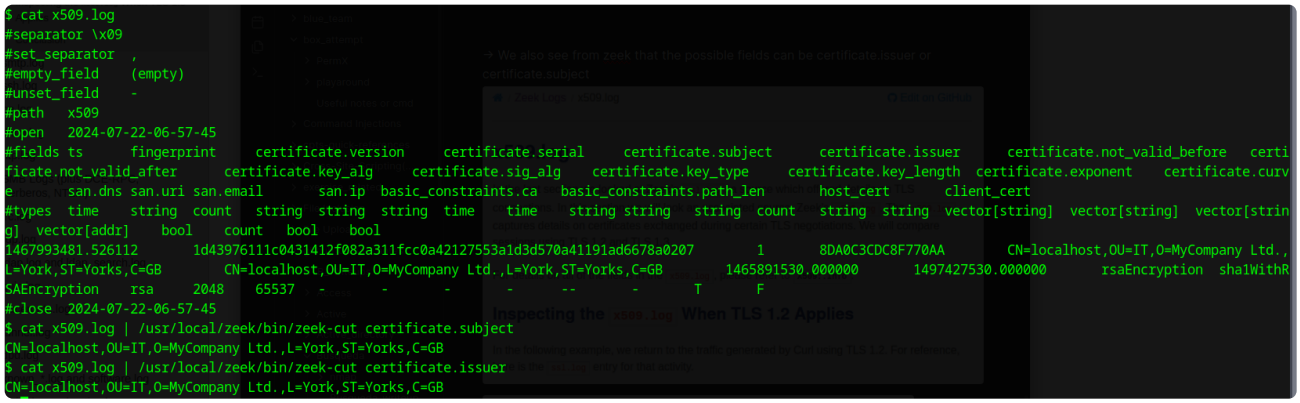
Inspecting the `x509.log` When TLS 1.2 Applies

In the following example, we return to the traffic generated by Curl using TLS 1.2. For reference, here is the `ssl.log` entry for that activity.

```
{
  "ts": 1598377391.938343,
  "id": "F2XEvj1CahhdhtfvT4",
  "certificate.version": 3,
  "certificate.serial": "0B58BC3898391F36592BA1BE1F6B03EF",
  "certificate.subject": "CN=www.taosecurity.com",
  "certificate.issuer": "CN=Amazon,OU=Server CA 1B,O=Amazon,C=US",
  "certificate.not_valid_before": 1590969600,
  "certificate.not_valid_after": 1625140800,
  "certificate.key_alg": "rsaEncryption",
  "certificate.sig_alg": "sha256WithRSAEncryption",
  "certificate.key_type": "rsa",
  "certificate.key_length": 2048,
  "certificate.exponent": "65537",
  "san.dns": [
    "www.taosecurity.com",
    "taosecurity.com",
    "*.taosecurity.com"
  ],
  "basic_constraints.ca": false
}
{
  "ts": 1598377391.938343,
  "id": "FZ7ygD3ERPfEVVohG9",
  "certificate.version": 3,
  "certificate.serial": "067F94578587E8AC77DEB253325BBC998B560D",
  "certificate.subject": "CN=Amazon,OU=Server CA 1B,O=Amazon,C=US",
  "certificate.issuer": "CN=Amazon Root CA 1,O=Amazon,C=US",
  "certificate.not_valid_before": 1445472000,
  "certificate.not_valid_after": 1760832000,
  "certificate.key_alg": "rsaEncryption",
  "certificate.sig_alg": "sha256WithRSAEncryption",
  "certificate.key_type": "rsa",
  "certificate.key_length": 2048,
  "certificate.exponent": "65537",
  "basic_constraints.ca": true,
  "basic_constraints.path_len": 0
}
```


-> This can be verified below

```
cat x509.log | /usr/local/zeek/bin/zeek-cut certificate.subject

cat x509.log | /usr/local/zeek/bin/zeek-cut certificate.issuer
```



-> Now, given that the question looks for the SSL certificate, it is assumed that they are looking for the subject of the communication and not the issuer (looking for subject of certificate), so the answer here would be certificate.subject.