

[illegible]

Name	Date modified	Type	Size
SilK	2023-02-25 6:12 AM	File folder	
01_exploit	2023-02-22 5:04 AM	Wireshark capture...	2 KB
application	2023-02-27 2:57 AM	Microsoft Edge P...	641 KB
CR450H23 - Devoir2 - Frederic Perron - G...	2023-02-27 2:59 AM	Microsoft Word D...	367 KB
CR450H23 - Devoir2 - Question - Repons...	2023-02-22 5:04 AM	Microsoft Word D...	85 KB
exercice2	2023-02-22 5:04 AM	Wireshark capture...	2,006 KB
Exercice1	2023-02-22 5:04 AM	Wireshark capture...	1,614 KB

Bro logs

Version 2.6

corelight

conn.log | IP, TCP, UDP, ICMP connection details

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp of first packet
uid	string	Unique identifier of connection
id	record	Connection's 4-tuple of endpoint addresses
proto	enum	Transport layer protocol of connection
service	string	Application protocol ID sent over connection
duration	interval	How long connection lasted
orig_bytes	count	Number of payload bytes originator sent
resp_bytes	count	Number of payload bytes responder sent
conn_state	string	Connection state (see conn.log - conn_state)
local_orig	bool	Value=1 if connection originated locally
local_resp	bool	Value=1 if connection responded locally
missed_bytes	count	Number of bytes missing (packet loss)
history	string	Connection state history (see conn.log - history)
orig_pkts	count	Number of packets originator sent
orig_ip_bytes	count	Number of originator IP bytes (via IP total_length header field)
resp_pkts	count	Number of packets responder sent
resp_ip_bytes	count	Number of responder IP bytes (via IP total_length header field)
tunnel_parents	table	If tunneled, connection UID of encapsulating parents
orig_l2_addr	string	Link-layer address of originator
resp_l2_addr	string	Link-layer address of responder
vlan	int	Outer VLAN for connection
inner_vlan	int	Inner VLAN for connection

conn_state

A summarized state for each connection

- S0 Connection attempt seen, no reply
- S1 Connection established, not terminated (0 byte counts)
- SF Normal establish & termination (0 byte counts)
- R0 Connection attempt rejected
- S2 Established, Orig attempts close, no reply from Resp
- S3 Established, Resp attempts close, no reply from Orig
- R0TO Established, Orig aborted (RST)
- R0TR Established, Resp aborted (RST)
- R0TOSS Orig sent SYN then RST; no Resp SYN-ACK
- R0TRH Resp sent SYN-ACK then RST; no Orig SYN
- SH Orig sent SYN then FIN; no Resp SYN-ACK (half-open)
- SHR Resp sent SYN-ACK then FIN; no Orig SYN
- OTH No SYN, not closed. Midstream traffic. Partial connection


history

Orig UPPERCASE, Resp lowercase, compressed

- S A SYN without the ACK bit set
- H A SYN-ACK ("handshake")
- A A pure ACK
- D Packet with payload ("data")
- F Packet with FIN bit set
- R Packet with RST bit set
- C Packet with a bad checksum
- I Inconsistent packet (Both SYN & RST)
- Q Multi-flag packet (SYN & FIN or SYN & RST)
- T Retransmitted packet
- W Packet with zero window advertisement
- X Flipped connection

dhcp.log | DHCP lease activity

FIELD	TYPE	DESCRIPTION
ts	time	Earliest time DHCP message observed
uids	table	Unique identifiers of DHCP connections
client_addr	addr	IP address of client
server_addr	addr	IP address of server handing out lease
mac	string	Client's hardware address
host_name	string	Name given by client in Hostname option 12
client_fqdn	string	FQDN given by client in Client FQDN option 81
domain	string	Domain given by server in option 15
requested_addr	addr	IP address requested by client
assigned_addr	addr	IP address assigned by server
lease_time	interval	IP address lease interval
client_message	string	Message with DHCP_DECLINE so client can tell server why address was rejected
server_message	string	Message with DHCP_NAK to let client know why request was rejected
msg_types	vector	DHCP message types seen by transaction



AP 3000 Sensor (25 Gbps)

Designed by the creators of open-source Bro (now known as Zeek), Corelight Sensors provide comprehensive network security monitoring. Available as physical or virtual appliances. www.corelight.com

For the most recent version of this document, visit: <https://github.com/corelight/bro-cheatsheets>

Page couverture du fichier PDF extrait du flux http

b) Réalisez l'extraction du fichier joint au trafic SMTP. (5p)

Pour cette partie de l'exercice, j'ai repéré le paquet qui contenait le protocole SMTP/IMF (email). J'ai aperçu sur ce dernier l'application.pdf et des informations attrayantes à son extrait. J'ai extrait le fichier en faisant Export Objects > IMF... Cela étant fait, un nouveau fichier .eml est apparu dans mon dossier Devoir 2. Après l'avoir ouvert, une fenêtre est apparue pour me demander si je voulais l'ouvrir à partir de Mail, ce que j'ai accepté. Lorsque le courriel est ouvert, un fichier PDF s'y trouvait et le celui-ci étant le même qu'à l'exercice précédente. Voir les images ci-dessous dans la prochaine page pour les démarches complètes de l'exercice b) ...

509	0.861269	209.85.215.180	142.93.70.167	SMTP	1484 C: DATA fragment, 1418 bytes
510	0.861331	209.85.215.180	142.93.70.167	SMTP/IMF	943 from: Jose Flavor <joseflavor102030@gmail.com>, subject: Fwd: Soup Du Jour, (text/plain) (text/html) (application/pdf)
511	0.861333	142.93.70.167	209.85.215.180	TCP	66 25 → 38445 [ACK] Seq=278 Ack=901437 Win=1507200 Len=0

8445 [ACK] Seq=278 Ack=899142 Win=1504384 Len=0 TSval=2122185812 TSecr=2749130397

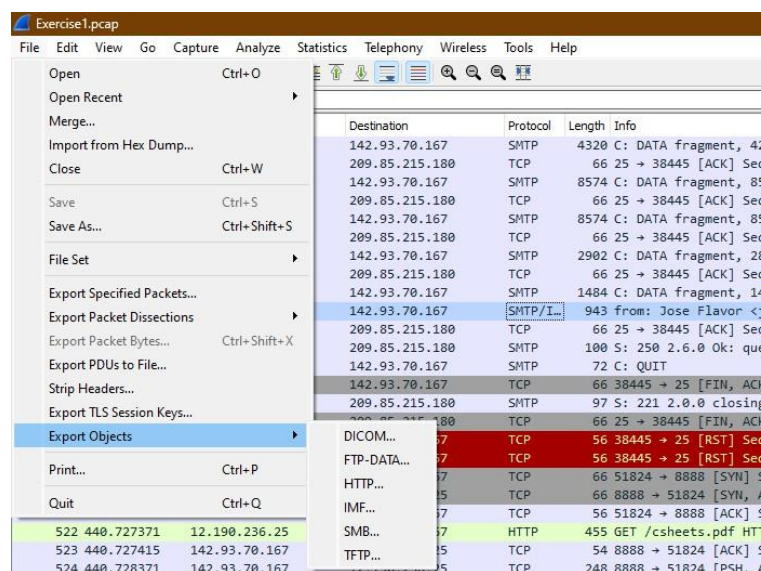
A fragment, 1418 bytes

Jose Flavor <joseflavor102030@gmail.com>, subject: Fwd: Soup Du Jour, (text/plain) (text/html) (application/pdf)

8445 [ACK] Seq=278 Ack=901437 Win=1507200 Len=0 TSval=2122185812 TSecr=2749130397

2.6.0 Ok: queued as 68B5E2F4

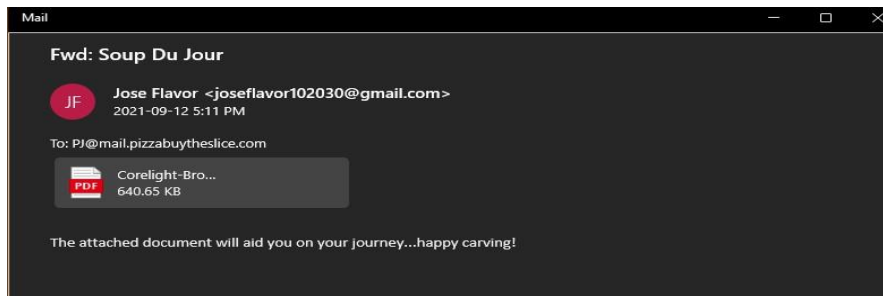
Le paquet SMTP/IMF en question



Extrait du paquet IMF de Wireshark à mon fichier Devoir 2

Analyse et Cybersécurité Opérationnelle > Session Hiver 2023 > CR450 Détection et Réponses aux Incidents > Travaux de session > Devoir 2				
Name	Date modified	Type	Size	
SILK	2023-02-25 6:12 AM	File folder		
01_exploit	2023-02-22 5:04 AM	Wireshark capture...	2 KB	
application	2023-02-27 2:57 AM	Microsoft Edge P...	641 KB	
CR450H23 - Devoir2 - Frederic Perron - G...	2023-02-27 4:06 AM	Microsoft Word D...	833 KB	
CR450H23 - Devoir2 - Question - Repons...	2023-02-22 5:04 AM	Microsoft Word D...	85 KB	
exercice2	2023-02-22 5:04 AM	Wireshark capture...	2,006 KB	
Exercice1	2023-02-22 5:04 AM	Wireshark capture...	1,614 KB	
Fwd%3a Soup Du Jour.eml	2023-02-27 3:49 AM	Outlook.File.eml.15	881 KB	

Le fichier en question dans mon dossier Devoir 2



Le email contenant le fichier PDF voulu

Après avoir ouvert le fichier PDF, voici la page couverture que j'ai eu :

c) Comparez les deux fichiers et exprimez vos conclusions. (2,5p)

Après avoir comparé les 2 fichiers, on peut en conclure qu'ils sont identiques. Les 2 fichiers extraits des 2 différents paquets (SMTP/IMF & HTTP) sont le même guide "Bro logs" de Corelight.

Exercice #2 : Décodage de fichiers (10p)

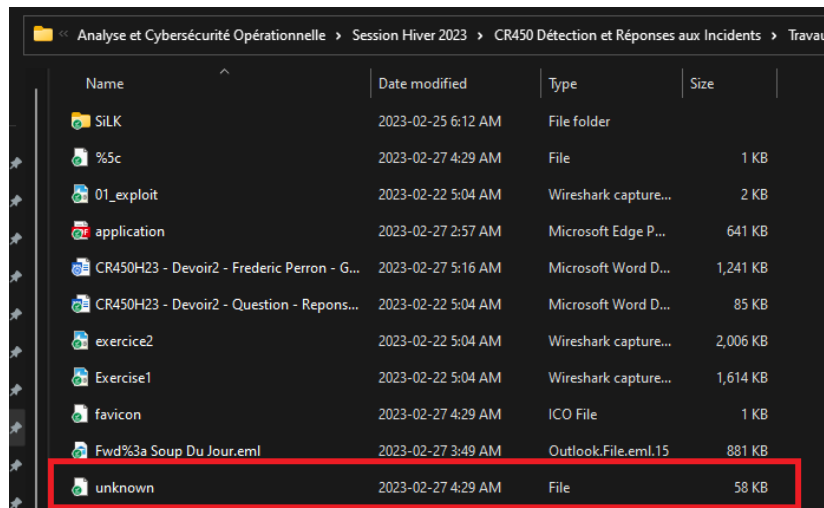
Analysez le fichier exercice2.pcap. La capture contient le transfert de plusieurs fichiers. Procédez à l'extraction du fichier appelé **unknown** et essayer de le convertir dans son format original. Par la suite, répondez à la question suivante :

De quel type de fichier il s'agit? Précisez l'ensemble des détails que vous trouverez.

Conseil : référez-vous au laboratoire du cours 5.

157	62.823762	192.168.126.128	192.168.126.138	TCP	60 24768 → 8000 [ACK] Seq=1 Ack=1 Win=262144 Len=0
158	62.823800	192.168.126.128	192.168.126.138	HTTP	376 GET /unknown HTTP/1.1
159	62.823807	192.168.126.138	192.168.126.128	TCP	54 8000 → 24768 [ACK] Seq=1 Ack=323 Win=64128 Len=0

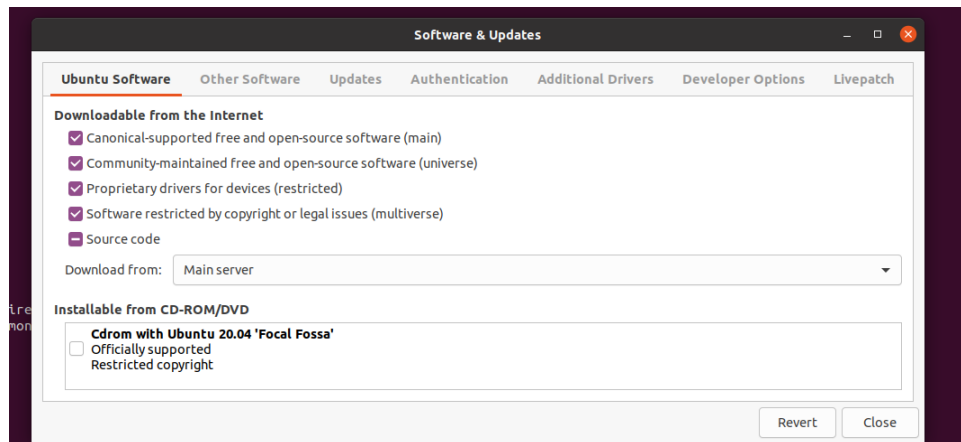
Paquet avec le fichier unknown repéré



Name	Date modified	Type	Size
SILK	2023-02-25 6:12 AM	File folder	
%5c	2023-02-27 4:29 AM	File	1 KB
01_exploit	2023-02-22 5:04 AM	Wireshark capture...	2 KB
application	2023-02-27 2:57 AM	Microsoft Edge P...	641 KB
CR450H23 - Devoir2 - Frederic Perron - G...	2023-02-27 5:16 AM	Microsoft Word D...	1,241 KB
CR450H23 - Devoir2 - Question - Repons...	2023-02-22 5:04 AM	Microsoft Word D...	85 KB
exercice2	2023-02-22 5:04 AM	Wireshark capture...	2,006 KB
Exercise1	2023-02-22 5:04 AM	Wireshark capture...	1,614 KB
favicon	2023-02-27 4:29 AM	ICO File	1 KB
Fwd%3a Soup Du Jour.eml	2023-02-27 3:49 AM	Outlook.File.eml.15	881 KB
unknown	2023-02-27 4:29 AM	File	58 KB

Après avoir exporté le fichier unknown dans le même fichier Devoir 2, voici comment j'ai décodé le fichier unknown :

J'ai d'abord commencé par ouvrir ce fichier à partir de ma VM Security Onion et non ma machine principale/personnelle. J'ai installé bless et j'ai par la suite dans Software & Updates mis la configuration source code à "Main Server" au lieu de la configuration par default Server for Canada. Une fois que cela était fait, j'ai été en mesure d'installer yara à partir du terminal. Après avoir installé yara, j'ai fait bless unknown pour ouvrir bless et le fichier à analyser. Pour le décoder, j'ai fait yara xorsigs.yar unknown -s pour obtenir les codes. J'ai obtenu 0x4e:\$ _13_d comme variables, hexacédimal et décimal, etc. J'ai par la suite été dans bless à partir du fichier unknown et j'ai sélectionné le tout avec Ctrl+A, entré la valeur d'as hexadecimal à partir du BitWise Operator. J'ai par la suite obtenu le même résultat, type de fichier et message que dans le Lab du cours 5. (MZ, this program cannot be run in DOS mode [...])



Changement du source code de Server for Canada à Main Server

```
student@cr450:~/labs$ sudo apt install yara
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-scan-plugin libfwupdplugin1 libxmlb1 ubuntu-advantage-desktop-daemon
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libyara3
The following NEW packages will be installed:
  libyara3 yara
0 upgraded, 2 newly installed, 0 to remove and 51 not upgraded.
Need to get 155 kB of archives.
After this operation, 486 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ca.archive.ubuntu.com/ubuntu focal/universe amd64 libyara3 amd64 3.9.0-1 [135 kB]
Get:2 http://ca.archive.ubuntu.com/ubuntu focal/universe amd64 yara amd64 3.9.0-1 [20.4 kB]
Fetched 155 kB in 1s (111 kB/s)
Selecting previously unselected package libyara3:amd64.
(Reading database ... 186479 files and directories currently installed.)
Preparing to unpack .../libyara3_3.9.0-1_amd64.deb ...
Unpacking libyara3:amd64 (3.9.0-1) ...
Selecting previously unselected package yara.
Preparing to unpack .../yara_3.9.0-1_amd64.deb ...
Unpacking yara (3.9.0-1) ...
Setting up libyara3:amd64 (3.9.0-1) ...
Setting up yara (3.9.0-1) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.9) ...
student@cr450:~/labs$ yara -v
3.9.0
```

téléchargement de yara après avoir changé la configuration du code source

téléchargement de bless

Variables et décimales obtenues après la commande ci-dessus

XOR sélectionné, d dans la Hexadecimal et execute (résultats dans la photo ci-dessus)

Exercice #3 : Validation de l'envoi d'un courriel (10p)

Un serveur SMTP ayant comme adresse IP 142.116.34.182 envoie un courriel selon les paramètres suivants :

Détails du courriel

Adresse IP du serveur d'expédition : 142.116.34.182

Adresse courriel de l'expéditeur : dave.munger@polymtl.ca

Adresse courriel du destinataire : dave.munger@cr450-polymtl.ca

Effectuez les analyses requises et répondez aux questions suivantes : une résolution DNS a été nécessaire sur l'adresse du destinataire pour connaître ses hôtes. Après avoir le nom de l'hôte, il est possible d'ensuite effectuer une résolution DNS de type A pour avoir son adresse IP.

- A. Quel sont les noms d'hôtes et les adresses IP des serveurs où le courriel doit être envoyé afin d'être reçu par le destinataire? (2p)
 - a. Nom d'hôte(s) : cr450-polymtl-ca.mail.protection.outlook.com
 - b. Adresse(s) IP : l'adresse IP du serveur de messagerie est 104.47.57.34

- B. Est-ce que le(s) serveur(s) de destination (identifié(s) précédemment) acceptera(ont) le courriel sans restriction? (8p)

Cela peut être dépendant de plusieurs facteurs. Ça peut dépendre des politiques de sécurité mises en place par les administrateurs de messagerie du domaine. Cependant, le fait que le courriel soit envoyé depuis une adresse IP valide et du "même domaine" peut aider à réduire le risque que le courriel soit considéré indésirable ou bloqué. Cela peut aussi dépendre de quels domaines sont considérés comme "trusted" par le domaine polymtl.ca. Cela étant dit, le courriel devrait se rendre à sa destination sans trop de restriction.

Exercice #4 : DKIM (10p)

Pour réaliser cet exercice vous devrez recevoir un courriel provenant de GMAIL. Utilisez des captures d'écran afin de documenter l'ensemble des étapes.

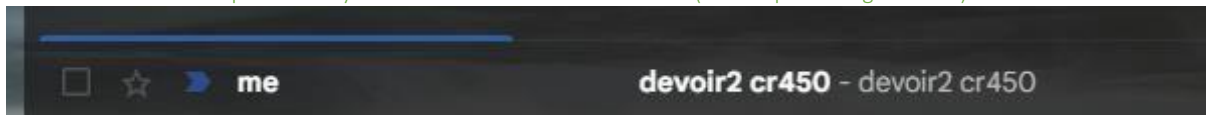
Effectuez les analyses requises et répondez aux questions suivantes :

A. Identifiez le sélecteur utilisé par GMAIL. (5p)

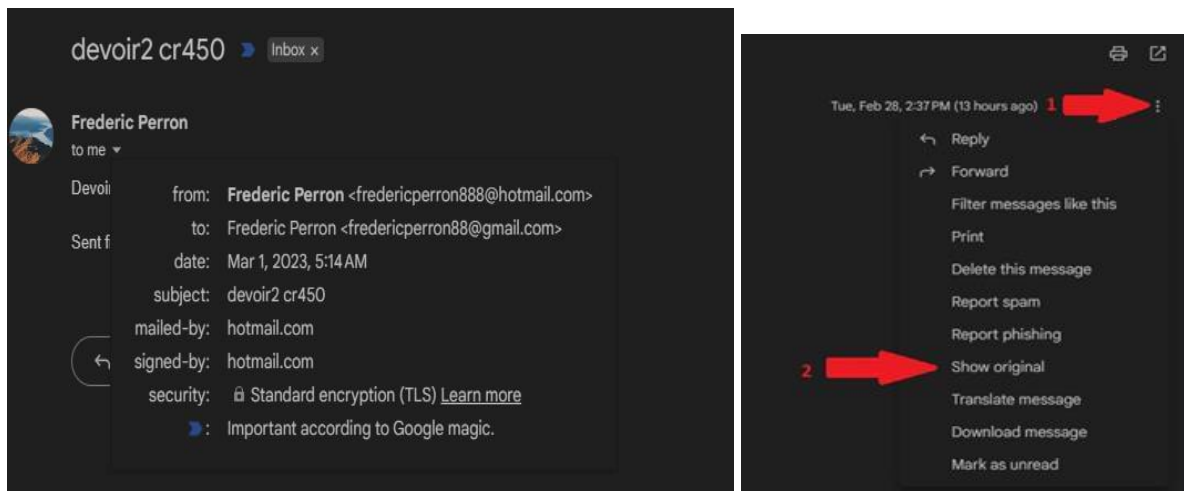
Le sélecteur utilisé par GMAIL est arc-20160816

J'ai commencé par m'envoyer un courriel de mon adresse Gmail. Une fois cela fait, j'ai été en mesure en cliquant sur "show original" de voir le sélecteur dans la section ARC-Message-Signature. Par la suite, j'ai pris en note le domaine (google.com) et le sélecteur (arc-20160816). Pour m'inspirer de la commande, j'ai suivi le word.doc nslookup du Cours 4. Voir démarches dans les captures ci-dessous

J'ai d'abord commencé par m'envoyer un courriel à mon adresse GMAIL (fredericperron88gmail.com)



Boîte de réception Gmail



Démarches pour pouvoir cliquer sur Show Original de mon gmail reçu

B. Documentez les détails de la clé DKIM dans le DNS. (5p)

Arc-20160816._domainkey.google.com.



Exercice 5 – SiLK (10p)

ATTENTION! Pour chaque numéro, vous devez ajouter une capture d'écran de votre commande. Pour finir, vous devez réaliser l'ensemble des exercices avec votre utilisateur personnel dans la plateforme Security Onion.

L'exercice 5 repose sur l'utilisation de SiLK. Utilisez les notes de cours ainsi que les laboratoires afin de vous aider.

a) Veuillez interroger les fichiers .silk du devoir(ex `rwfilter *.silk`) afin de déterminer le nombre total de flux observés entre le 5 et 10 septembre 2003 pour l'ensemble des fichiers? (5p)

J'ai commencé par utiliser la commande vue en classe et sélectionné mes critères désirés. J'ai dû ajouter `no columns` et `skip-zeroes` comme vu sur une page trouvée sur google (voir références). Donc le nombre de flux correspond aux nombres de pass, donc **14974**.

```
2003/09/08T17:01:30|3.05|3130.30|30.33|
2003/09/08T17:01:30|2.44|591.31|9.32|
2003/09/08T17:03:00|1.00|144.00|3.00|
fredperron15@cr450:~/devoir2$ rwfilter *.silk --protocol=0-255 --pass=stdout --stime=2003/09/05-2003/09/10 --print-stat | rwcoun --skip-zeroes --no-columns --no-titles
Files      5.  Read      15021.  Pass      14974.  Fail       47.
2003/09/05T02:41:00|0.02|45.16|0.45|
```

Indice

`rwfilter` est le meilleur outil pour répondre à cette question. Il dispose d'une option qui permet de rapporter rapidement le nombre de flux agrégés en fonction de vos critères de sélection.

Indice

Les options `--print-statistics` indiquent le nombre de fichiers qui ont dû être lus à partir du référentiel, le nombre de flux lus, le nombre qui passent votre filtre et le nombre de flux qui échouent votre filtre.

b) Veuillez trouver (interroger) les fichiers .silk pour les flux survenus entre le 5 et 10 septembre 2003. Quel est le total de flux TCP présent pour l'ensemble des fichiers? (5p)

Indice

Vous pouvez répondre à cette question en utilisant presque exactement la même commande que l'exercice précédent, il ne manque que les critères de filtrage pour sélectionner le protocole TCP.

Vous pouvez spécifier un protocole sur lequel filtrer avec l'option `--protocol=`.

La même chose que la dernière sauf que j'ai changé le bin-size à 60 et enlever les `rwcount` filters. J'ai également seulement ajouté le protocole 6 pour TCP et **10935** pass ont été révélés (nombre total de flux pour tcp)

```
Fredperron15@cr450:~/devoir2$ rfilter *.silk --protocol=6 --pass=stdout --stime=2003/09/05-2003/09/10 --print-stat | rcount --bin-size=60
Files      5.  Read      15021.  Pass      10935.  Fail       4086.
 Date|      Records|      Bytes|      Packets|
```

Exercice 6 – SiLK (15p)

Statistiques

SiLK fournit l'outil `rwstats` comme un moyen rapide de traiter les flux sous forme d'ensembles de données. Il peut être utilisé pour rapidement renvoyer des informations statistiques sur les données dans le référentiel et, en fonction de vos paramètres de requête, il peut être regroupé et organisé de n'importe quelle manière vous préférez.

a) Utilisez les outils `rfilter` et `rwstats` pour déterminer quels sont les dix principaux protocoles qui ont été utilisés entre le 5 et 10 septembre 2003. (5p)>

Indice : `rfilter`

Pour répondre à cette question, vous devrez utiliser `pass=stdout` avec un filtre comme arguments de `rfilter`, puis traiter la sortie via `rwstats`. Pouvez-vous penser à un filtre que vous pouvez utiliser pour voir tous les protocoles du référentiel pendant cette période de temps?

Indice : `rfilter`

Un filtre simple que vous pouvez utiliser pour examiner tous les flux au cours d'une période donnée est `-proto=0-255`. Cette option vous permet de sélectionner des listes individuelles séparées par des virgules ou des plages de protocoles. Dans ce cas, il sautera à travers tous les flux IP pour tous les protocoles puisque tous les protocoles correspondront.

Indice : `rwstats`

Pour utiliser `rwstats`, vous devez spécifier un champ ou un ensemble de `--fields` qui doivent être regroupés, à partir desquels, des statistiques seront produites. Vous devez également spécifier un `--count` pour le nombre d'éléments que vous souhaitez signaler. Dans ce cas, nous sommes à la recherche des dix meilleurs articles.

Flux, octets et paquets

L'outil `rwstats` rapportera le nombre de flux par défaut. Vous pouvez remplacer ce comportement en utilisant l'option `--bytes` ou `--packets`. Les statistiques seront désormais rapportées en fonction des critères sélectionnés plutôt que du nombre de flux.

Cela peut être très utile. Par exemple, nous pourrions être plus intéressés par le nombre total d'octets qui sont passés entre deux hôtes que par le nombre de flux qui ont été vus. C'est une bonne façon de trouver l'exfiltration qui est répartie sur plusieurs connexions, ports et protocoles.

Pour commencer, j'ai d'abord pris en notes dans les indices ci-dessus les commandes et été sur google pour en savoir davantage (voir références). La commande utilisée va comme suis :

```
Use 'rfilter --help' for usage
fredperron15@cr450:~/devotr2$ rfilter *.silk --protocol=0-255 --pass=stdout --stime=2003/09/05-2003/09/10 --print-stat | rwstats --top --field=protocol,sip --count=10
Files 5, Read 15021, Pass 14974, Fail 47.
INPUT: 14974 Records for 2876 Bins and 14974 Total Records
OUTPUT: Top 10 Bins by Records
pro|          sip| Records| %Records| cumul_%|
6|          192.168.1.3| 5462| 36.476559| 36.476559|
17|          192.168.1.3| 2062| 13.770536| 50.247095|
17|          145.238.110.68| 260| 1.736343| 51.983438|
17|          134.214.100.6| 253| 1.689595| 53.673033|
17|          193.49.205.19| 247| 1.649526| 55.322559|
6|          200.184.43.197| 172| 1.148658| 56.471217|
1|          192.168.1.3| 164| 1.095232| 57.566449|
6|          63.243.90.10| 135| 0.901563| 58.468011|
6|          78.68.74.84| 93| 0.621077| 59.089088|
6|          203.248.234.10| 67| 0.447442| 59.536530|
```

Nous pouvons voir les top 10 protocoles utilisés dans la rangée de gauche ``pro``.

b) Dans les fichiers .silk, quelle adresse IP source a transféré le plus d'octets entre le 5 et 10 septembre 2003? (5p)

Indice

La seule vraie différence entre cette question et les exercices précédents est que vous devez ajuster les horodatages et regarder le nombre de bytes plutôt que le nombre de flows.

La seule chose qu'il y avait à modifier/ajouter était le compte et la valeur changée en bytes comme vue dans ma page de référence de commandes Silk. J'ai donc seulement ajouté une value=bytes à la fin de rwstats et j'ai pu obtenir en ordre décroissant, les sip avec le plus de Bytes transférés. Voir image ci-dessous pour la commande :

```
fredperron15@cr450:~/devotr2$ rfilter *.silk --protocol=0-255 --pass=stdout --stime=2003/09/05-2003/09/10 --print-stat | rwstats --top --field=sip,sport,dip,dport,protocol --count=10 --value=bytes
Files 5, Read 15021, Pass 14974, Fail 47.
INPUT: 14974 Records for 7839 Bins and 2912498 Total Bytes
OUTPUT: Top 10 Bins by Bytes
sip|sport|          dip|dport|pro|      Bytes|    %Bytes| cumul_%|
65.113.119.134| 80|          192.168.1.3| 1043| 6| 454274| 15.597401| 15.597401|
192.168.1.3| 514|          192.168.1.254| 514| 17| 268588| 7.101825| 22.759226|
192.168.1.3| 130|          192.168.1.255| 130| 17| 190155| 6.528932| 29.288157|
192.168.1.3| 123|          193.49.205.19| 123| 17| 34428| 1.182078| 30.470236|
192.168.1.3| 123|          134.214.100.6| 123| 17| 33212| 1.140327| 31.610562|
192.168.1.3| 123|          145.238.110.68| 123| 17| 32988| 1.129889| 32.740452|
134.214.100.6| 123|          192.168.1.3| 123| 17| 30552| 1.048996| 33.789448|
145.238.110.68| 123|          192.168.1.3| 123| 17| 30172| 1.035949| 34.825397|
193.49.205.19| 123|          192.168.1.3| 123| 17| 30172| 1.035949| 35.861347|
192.168.1.3| 443|          61.61.123.123| 33587| 6| 29286| 1.005529| 36.866875|
```

L'adresse IP qui a donc transféré le plus d'octets (bytes) entre le 5 et 10 septembre 2003 est 65.113.119.134

c) Entre le 5 et 10 septembre 2003, à quelle heure a démarré le flux qui a transféré le plus d'octets ? (5p)

Indice

C'est une question plus difficile mentalement. Lorsque nous passons un ensemble de champs à rwstats, il utilise ces champs pour créer des bacs. Ce sont des regroupements de données. Par exemple, si nous passons ce qui suit :

--fields sip,dip

Cela rapporterait le plus grand nombre de flux où l'adresse IP source et l'adresse IP de destination dans les flux sont utilisées comme clés, quels que soient les ports ou les protocoles. Si nous modifions légèrement cela :

```
--fields sip,dip --bytes
```

Cela signalerait désormais la paire d'adresses source à destination qui avait le plus grand nombre d'octets dans l'agrégat plutôt que dans les flux.

```
--fields sip,dport
```

Cela ressemblerait maintenant à l'agrégation de tous les flux par adresse IP source et port de destination uniquement, en signalant l'appariement qui a le plus grand nombre de flux.

Pour cet exercice, j'ai utilisé le `rwuniq` tout simplement avec le `fields` à `stime,sip,dip` le `value` à `etime` et pour le nombre de bytes. Cela a donné quelque chose comme `rwuniq --bin-time=86400 --fields=stime,sip,dip --value=etime --bytes *.silk`, ce qui m'a donné la liste en bytes. J'ai par la suite scroll down jusqu'à ce que je trouve l'échange avec le plus grand nombre de bytes, **étant 463812, envoyé à 00 :00 et reçu à 00 :17.**

```
2003/09/06T00:00:00|2003/09/06T00:29:59|
fredperron15@cr450:~/devotr2$ rwuniq --bin-time=86400 --fields=stime,sip,dip --value=etime --bytes *.silk
stime|sip|dip|etime-Latest|Bytes|
2003/09/07T00:00:00|192.168.1.3|80.25.252.73|2003/09/07T00:00:00|120|
2003/09/06T00:00:00|192.168.1.3|200.175.33.89|2003/09/06T00:00:01|508|
2003/09/07T00:00:00|62.151.2.8|192.168.1.3|2003/09/07T00:00:00|2780|
2003/09/07T00:00:00|192.168.1.3|80.25.126.106|2003/09/07T00:00:00|120|
2003/09/07T00:00:00|192.168.1.3|80.26.71.12|2003/09/07T00:00:00|120|
2003/09/05T00:00:00|200.164.123.198|192.168.1.3|2003/09/05T00:00:02|463|
2003/09/07T00:00:00|66.156.18.67|192.168.1.3|2003/09/07T00:00:00|463|
2003/09/07T00:00:00|80.26.59.24|192.168.1.3|2003/09/07T00:00:00|144|
2003/09/06T00:00:00|192.168.1.3|172.194.97.130|2003/09/06T00:00:00|40|
2003/09/08T00:00:00|65.92.30.60|192.168.1.3|2003/09/08T00:00:00|479|
2003/09/05T00:00:00|61.146.53.22|192.168.1.3|2003/09/05T00:00:04|551|
```

Commande utilisée

```
2003/09/06T00:00:00|80.25.21.53|192.168.1.3|2003/09/06T00:00:00|144|
2003/09/08T00:00:00|65.113.119.134|192.168.1.3|2003/09/08T00:00:17|463812|
2003/09/05T00:00:00|80.24.71.233|192.168.1.3|2003/09/05T00:00:00|461|
```

Échange avec le plus grand nombre de byte (heure indiquée)

Exercice 7 - Suricata(25p)

ATTENTION! Pour chaque règle demandée, vous devez ajouter une capture d'écran de votre règle ainsi qu'une capture d'écran de l'alerte générée par cette dernière. De plus, votre prénom et nom doit toujours être inclus dans l'alerte. Pour finir, vous devez réaliser l'ensemble des exercices avec votre utilisateur personnel dans la plateforme Security Onion. Suricata ne voulait pas fonctionner sur mon compte personnel. J'ai donc dû utiliser le compte student/student. Désolé pour les inconvénients.

Utilisez le fichier pcap « 01_exploit.pcap » comme entrée pour cet exercice. L'objectif des exercices de cet atelier est de créer une signature fonctionnelle qui détecte avec succès l'exploit utilisé pour compromettre un système basé sur le CVE fictif suivant et le paquet relatif. Considérez la capture de paquet fournie comme un exemple de capture de paquet collecté en exécutant l'exploit de preuve de concept sur un service LamanServer sur le réseau local.

CVE- 2021-0503¶

Description Une condition de dépassement de mémoire tampon entraînant l'exécution de code à distance a été identifiée dans le service Enterprise LamanServer , qui s'exécute généralement sur le port TCP 50503. Bien qu'une preuve de concept ait été démontrée, l'exploitation n'a pas encore été observée dans la nature.

La condition de dépassement de mémoire tampon existe dans l'analyseur de commande de protocole, qui n'est accessible qu'après l'envoi réussi d'un message HELO.

Références Remarque : Les références sont fournies pour la commodité du lecteur afin d'aider à distinguer les vulnérabilités. La liste n'est pas censée être complète. BID : 503001 URL : <http://www.securityfocus.com/bid/503001> CONFIRM : <https://kb.showmethepackets.com/SMTP503001>

Description : Créez une règle Suricata qui détecte avec succès l'exploit trouvé dans 01_exploit.pcap en utilisant une ou plusieurs options content.

a) Créez une règle d'alerte simple qui utilise uniquement un en-tête de règle et vérifiez sa fonction à l'aide du fichier 01_exploit.pcap , en affichant les alertes sur la console. Votre règle doit alerter sur chaque paquet trouvé dans le dossier. (5p)

Pour créer un en-tête de règle, vous devez d'abord décider de l'action de la règle . Il peut s'agir de l'une des actions suivantes :

- alert Générer une alerte .
- pass Passez ce trafic, sans appliquer d'autres actions.
- log Enregistrez les paquets associés mais ne générez aucune alerte.
- sdrop IPS uniquement. Bloquez le paquet et supprimez-le silencieusement.
- drop IPS uniquement. Bloquez le paquet et enregistrez-le.
- reject Bloquer le paquet, le consigner et générer un TCP RST pour supprimer la connexion.
- activate Activer une règle dynamique qui devrait devenir active si cette règle correspond.
- dynamic Spécifiez qu'il s'agit d'une règle d'alerte dynamique.

Indice

Tous les en-têtes de règle doivent spécifier un protocole, une source, un port source, une direction, une destination et un port de destination. Les protocoles valides incluent :

- IP

- TCP
- UDP
- ICMP

protocoles IP arbitraires ne sont pas pris en charge .

La structure générale d'un en-tête de règle est :

alert IP \$SOURCE \$SPORT -> \$DEST \$DPORT

J'ai commencé par créer un fichier avec le nom devoir2.rules pour mes règles suricata.

```
student@cr450:~/labs/cours6$ nano cours6.rules
```

Par la suite, j'ai entré la règles très basique pour recevoir une alerte pour chaque paquets :

```
GNU nano 4.8
alert tcp any any -> any any ()
```

J'ai par la suite lancé la commande suricata avec mon fichier.rules sélectionné et le fichier wireshark du devoir également et les alertes sont apparues. Voir photo ci-dessous :

```
student@cr450:~/labs/cours6$ sudo so-suricata-testrule devoir2.rules /home/student/labs/cours6/01_exploit.pcap
=====
Running all.rules and devoir2.rules against the following pcap: /home/student/labs/cours6/01_exploit.pcap
===== Begin Suricata Output =====
7/3/2023 -- 00:46:36 - <Notice> - This is Suricata version 6.0.9 RELEASE running in USER mode
7/3/2023 -- 00:46:36 - <Info> - CPUs/cores online: 8
7/3/2023 -- 00:46:36 - <Info> - dropped the caps for main thread
7/3/2023 -- 00:46:36 - <Info> - eve-log output device (regular) initialized: /nsn/eve-%Y-%m-%d-%H:%M.json
7/3/2023 -- 00:46:36 - <Info> - stats output device (regular) initialized: stats.log
7/3/2023 -- 00:46:37 - <Info> - 1 rule files processed, 33043 rules successfully loaded, 0 rules failed
7/3/2023 -- 00:46:37 - <Info> - Threshold config parsed: 0 rule(s) found
7/3/2023 -- 00:46:38 - <Info> - 33046 signatures processed. 1198 are IP-only rules, 5211 are inspecting packet payload, 26612 inspect application layer, 0 are decoder event only
7/3/2023 -- 00:46:39 - <Info> - can't get cpu-affinity node
7/3/2023 -- 00:46:39 - <Notice> - all 1 packet processing threads, 4 management threads initialized, engine started.
7/3/2023 -- 00:46:39 - <Info> - Starting file run for /input.pcap
7/3/2023 -- 00:46:39 - <Info> - pcap file /input.pcap end of file reached (pcap err code 0)
7/3/2023 -- 00:46:39 - <Notice> - Signal Received. Stopping engine.
7/3/2023 -- 00:46:39 - <Info> - time elapsed 0.853s
7/3/2023 -- 00:46:39 - <Notice> - Pcap-file module read 1 files, 19 packets, 1473 bytes
7/3/2023 -- 00:46:39 - <Info> - Alerts: 2
7/3/2023 -- 00:46:39 - <Info> - cleaning up signature grouping structure... complete
===== End Suricata Output =====

If any alerts hit, they will be displayed below:

{
  "timestamp": "2019-11-24T18:52:48.612230+0000",
  "flow_id": 378330004084614,
  "pcap_cnt": 1,
  "event_type": "alert",
  "src_ip": "192.168.2.239",
  "src_port": 58496,
  "dest_ip": "192.168.2.209",
  "dest_port": 50503,
  "proto": "TCP",
  "community_id": "1:rrpKBynUqGUqwPdHco/SeV8yTYo=",
  "alert": {
    "action": "allowed",
    "gid": 1,
    "signature_id": 0,
    "rev": 0,
    "signature": "",
    "category": "",
    "severity": 3,
    "rule": "alert tcp any any -> any any ()"
  },
  "payload_printable": "",
  "stream": 0,
  "packet": "AAwPw/SEAwpQoJyCABFAABvK1AAEAG9v3AqALvwKgC0eSAXUcI6zvNAAAAAKAC+vDHwQAAAGQFtAQCCALi0SYAAAAAAEDAwc=",
  "packet_info": {
    "linktype": 1
  }
}

{
  "timestamp": "2019-11-24T18:52:48.612420+0000",
  "flow_id": 378330004084614,
  "pcap_cnt": 2,
  "event_type": "alert",
  "src_ip": "192.168.2.209",
  "src_port": 50503,
  "dest_ip": "192.168.2.239",
  "dest_port": 58496,
  "proto": "TCP",
```

- b) Ajoutez des métadonnées à votre règle. Utilisez SID 1000000 comme ID de signature. Assurez-vous d'inclure un numéro de révision de 1 et un message d'alerte de votre choix. Les métadonnées BugTraq et CVE doivent correspondre aux informations de la CVE ci-dessus. Après avoir modifié votre règle, vérifiez qu'il alerte toujours correctement à l'aide de Suricata. (5p)

J'ai modifié la règle Suricata pour entrer des métadonnées, un message, un ID de signature, etc. Voici le fichier nano dont je me suis inspiré procéder :

```
GNU nano 4.8
alert tcp any any -> any any (msg: " tcp alert cr450"; sid:1000001; rev:1; classtype:tcp-event;)
```

J'ai par la suite lancé la commande suricata pour avoir mes alertes avec mes messages. Voir image ci-dessous :

```
student@cr450:~/labs/course$ sudo so-suricata-testrule devoir2.rules /home/student/labs/course/01_exploit.pcap
[sudo] password for student:
*****
Running all.rules and devoir2.rules against the following pcap: /home/student/labs/course/01_exploit.pcap

==== Begin Suricata Output ====
7/3/2023 -- 01:09:06 - <Notice> - This is Suricata version 0.0.9 RELEASE running in USER mode
7/3/2023 -- 01:09:06 - <Info> - CPUs/cores online: 8
7/3/2023 -- 01:09:06 - <Info> - dropped the caps for main thread
7/3/2023 -- 01:09:06 - <Info> - eve-log output device (regular) initialized: /nsm/eve-%Y-%m-%d-%H:%M.json
7/3/2023 -- 01:09:06 - <Info> - stats output device (regular) initialized: stats.log
7/3/2023 -- 01:09:06 - <Warning> - [ERRCODE: SC_ERR_UNKNOWN_VALUE(129)] - signature sid:1000001 uses unknown classtype: "tcp-event", using default priority 3. This message won't be shown again for this classtype
7/3/2023 -- 01:09:06 - <Info> - 1 rule files processed, 33043 rules successfully loaded, 0 rules failed
7/3/2023 -- 01:09:06 - <Info> - Threshold config parsed: 0 rule(s) found
7/3/2023 -- 01:09:07 - <Info> - 33046 signatures processed. 1198 are IP-only rules, 5211 are inspecting packet payload, 26612 inspect application layer, 0 are decoder event only
7/3/2023 -- 01:09:08 - <Info> - can't get cpu-affinity node
7/3/2023 -- 01:09:08 - <Notice> - all 1 packet processing threads, 4 management threads initialized, engine started.
7/3/2023 -- 01:09:08 - <Info> - Starting file run for /input.pcap
7/3/2023 -- 01:09:08 - <Info> - pcap file /input.pcap end of file reached (pcap err code 0)
7/3/2023 -- 01:09:08 - <Notice> - Signal Received. Stopping engine.
7/3/2023 -- 01:09:08 - <Info> - time elapsed 0.062s
7/3/2023 -- 01:09:08 - <Notice> - Pcap-file module read 1 files, 19 packets, 1473 bytes
7/3/2023 -- 01:09:08 - <Info> - Alerts: 2
7/3/2023 -- 01:09:09 - <Info> - Cleaning up signature grouping structure... complete
==== End Suricata Output ====

If any alerts hit, they will be displayed below:

{
  "timestamp": "2019-11-24T10:52:48.612230+0000",
  "flow_id": 1816409608836998,
  "pcap_cnt": 1,
  "event_type": "alert",
  "src_ip": "192.168.2.239",
  "src_port": 58496,
  "dest_ip": "192.168.2.209",
  "dest_port": 50503,
  "proto": "TCP",
  "community_id": "1:rrpkByMUQgUqWpdhCo/5eY8yTow=",
  "alert": {
    "action": "allowed",
    "gid": 1,
    "signature_id": 1000001,
    "rev": 1,
    "signature": " tcp alert cr450",
    "category": "",
    "severity": 3,
    "rule": "alert tcp any any -> any any (msg: \" tcp alert cr450\"; sid:1000001; rev:1; classtype:tcp-event;)"
  },
  "payload_printable": "",
  "stream": 0,
  "packet": "Aomp/SEAwPQo]yCABFAABVK1AAEAGv3AqALWkgC0eSAXUc16zvNAAAAAKAC+VDHwQAAAgQFtAQCCARLT0SYAAAAAEADAw=",
  "packet_info": {
    "linktype": 1
  }
}

{
  "timestamp": "2019-11-24T10:52:48.612420+0000",
  "flow_id": 1816409608836998,
  "pcap_cnt": 2,
  "event_type": "alert",
  "src_ip": "192.168.2.209",
  "src_port": 50503,
```

Métadonnées de règle

Dès que vous ajoutez des options de règle, vous devez inclure certaines métadonnées. Bien que le SID soit requis, la plupart des métadonnées ne le sont pas. Néanmoins, il est recommandé d'inclure un certain nombre d'éléments :

- **rev** Vous devez incrémenter la valeur des métadonnées de révision chaque fois que vous apportez une modification à votre règle. Cela vous fera économiser beaucoup de temps et d'énergie lors de la tentative de débogage des modifications de règle. Il est courant que la règle que vous modifiez n'est pas celui qui est réellement utilisé. Cette option peut vous aider à le détecter plus tôt.
- **reference** Vous permet de spécifier des métadonnées pour CVE, BugTraq, Nessus, Arachnids, McAfee, OSVDB, MSB ou des URL brutes. Lors de l'utilisation d'une référence prise en charge, la référence sera automatiquement convertie en URL. Plus d'informations peuvent être trouvées dans le manuel Suricata en ligne dans la section 3.4.2.
- **sid** Le SID est un élément obligatoire. Souvenez-vous que, de votre point de vue, tous les nombres < 1 000 000 sont réservés.
- **msg** Une chaîne arbitraire qui sera incluse comme message d'alerte principal.

c) Examinez les paquets fournis dans 01_exploit.pcap. Identifiez le contenu qui pourrait être possiblement utilisé dans une correspondance de signature qui ne serait pas susceptible de générer un faux positif. (5p)

Contenu correspondant

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.2.239	192.168.2.209	TCP	74	58496 → 50503 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=
2	0.000109	192.168.2.209	192.168.2.239	TCP	74	50503 → 58496 [SYN, ACK] Seq=0 Ack=1 Win=14400 Len=0 MSS=1460
3	0.000263	192.168.2.209	192.168.2.239	TCP	74	[TCP Out-Of-Order] 50503 → 58496 [SYN, ACK] Seq=0 Ack=1 Win=1
4	0.000651	192.168.2.239	192.168.2.209	TCP	66	58496 → 50503 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3408086
5	0.000725	192.168.2.239	192.168.2.209	TCP	66	[TCP Dup ACK 4#1] 58496 → 50503 [ACK] Seq=1 Ack=1 Win=64256 L
6	0.000833	192.168.2.209	192.168.2.239	TCP	70	50503 → 58496 [ACK, ACK] Seq=1 Ack=1 Win=14400 Len=0 TSval=1672
7	0.001311	192.168.2.239	192.168.2.209	TCP	66	58496 → 50503 [ACK] Seq=1 Ack=14 Win=64256 Len=0 TSval=3408086
8	0.001357	192.168.2.239	192.168.2.209	TCP	78	[TCP Dup ACK 7#1] 58496 → 50503 [ACK] Seq=1 Ack=14 Win=64256
9	0.002372	192.168.2.239	192.168.2.209	TCP	230	58496 → 50503 [PSH, ACK] Seq=1 Ack=14 Win=64256 Len=164 TSval
10	0.002461	192.168.2.209	192.168.2.239	TCP	66	50503 → 58496 [ACK] Seq=14 Ack=165 Win=15552 Len=0 TSval=2550
11	0.002505	192.168.2.209	192.168.2.239	TCP	66	[TCP Dup ACK 10#1] 50503 → 58496 [ACK] Seq=14 Ack=165 Win=155
12	0.003102	192.168.2.209	192.168.2.239	TCP	66	50503 → 58496 [FIN, ACK] Seq=14 Ack=165 Win=15552 Len=0 TSval
13	0.003194	192.168.2.239	192.168.2.209	TCP	78	58496 → 50503 [ACK] Seq=165 Ack=15 Win=64256 Len=0 TSval=3408
14	0.003271	192.168.2.239	192.168.2.209	TCP	66	58496 → 50503 [FIN, ACK] Seq=165 Ack=15 Win=64256 Len=0 TSval
15	0.003325	192.168.2.209	192.168.2.239	TCP	66	50503 → 58496 [ACK] Seq=15 Ack=166 Win=15552 Len=0 TSval=2550
16	0.003377	192.168.2.209	192.168.2.239	TCP	66	[TCP Dup ACK 15#1] 50503 → 58496 [ACK] Seq=15 Ack=166 Win=155
17	0.003405	192.168.2.239	192.168.2.209	TCP	60	58496 → 50503 [RST] Seq=166 Win=0 Len=0
18	0.003496	192.168.2.209	192.168.2.239	TCP	60	58496 → 50503 [RST] Seq=166 Win=0 Len=0
19	0.003548	192.168.2.239	192.168.2.209	TCP	60	58496 → 50503 [RST] Seq=166 Win=0 Len=0

* Frame 6: 79 bytes on wire (632 bits), 79 bytes captured (632 bits)
 * Ethernet II, Src: VMware_c3:f4:84 (00:0c:29:c3:f4:84), Dst: VMware_42:88:f2 (00:0c:29:42:88:f2)
 * Internet Protocol Version 4, Src: 192.168.2.209, Dst: 192.168.2.239
 * Transmission Control Protocol, Src Port: 50503, Dst Port: 58496, Seq: 1, Ack: 1, Len: 13
 * Data (13 bytes)
 data: 40504cf0a33f4d4d41e4d3a
 [Length: 13]

```

0000  00 0c 29 42 88 f2 00 0c 29 c3 f4 84 00 00 45 00  ..JB....).....E
0010  00 41 85 69 48 00 40 86 2e 3d c0 a0 02 d1 c0 a0  A 10 @ . # .....
0020  02 ef c5 47 e4 80 16 3f a1 93 08 eb 3b ce 80 18  G...?.....:
0030  03 89 4b d4 00 00 01 01 00 0a 01 85 24 fa cb 23  K.....$ #
0040  44 98 40 42 4c 4f 9a 43 4f 40 4d 41 4e 44 3a    D HELLO.C ORMAND:
  
```

Lorsque vous identifiez du contenu que vous pourriez utiliser dans le cadre d'une règle, vous recherchez du contenu qui n'apparaîtra probablement pas couramment dans votre trafic réseau général. En règle générale, une chaîne d'au moins quatre octets, mais de préférence plus longue, est préférable. Si elle est

plus petite des chaînes ou des chaînes d'octets doivent être spécifiées, essayez toujours d'inclure une chaîne plus longue qui doit également être présente.

Suricata essaiera toujours de trouver la chaîne la plus longue qui se trouve soit à un décalage connu, soit qui peut être trouvée relativement au début de la charge utile du paquet.

Indice

Utilisez n'importe quel outil avec lequel vous vous sentez à l'aise pour afficher les charges utiles des paquets. Wireshark ou tcpdump sont de bons choix.

- c) Utilisez le contenu que vous avez identifié dans la question précédente ainsi que les informations de protocole pour affiner votre règle. Après avoir fait cela, vérifiez que Suricata alertera correctement sur cette tentative d'exploit. (10p)

Nous pouvons faire cela simplement en ajoutant le contenu :data du paquet dans les options de ma règle suricata :

```
▶ Transmission Control Protocol, Src Port: 443, Seq: 3000000000, Len: 13
▼ Data (13 bytes)
  Data: 48454c4f0a434f4d4d414e443a
  [Length: 13]
```

data string du paquet en question

```
GNU nano 4.8
alert tcp any any -> any any (msg:"CVE ALERTCR450"; sid:1000001; rev:1; content:"48454c4f0a434f4d4d414e443a";)
```

regle suricata

```
student@cr450:~/labs/cours6$ sudo so-suricata-testrule devoir2.rules /home/student/labs/cours6/01_exploit.pcap
=====
Running all.rules and devoir2.rules against the following pcap: /home/student/labs/cours6/01_exploit.pcap
==== Begin Suricata Output ====
7/3/2023 -- 02:18:01 - <Notice> - This is Suricata version 6.0.9 RELEASE running in USER mode
7/3/2023 -- 02:18:01 - <Info> - CPUs/cores online: 8
7/3/2023 -- 02:18:01 - <Info> - dropped the caps for main thread
7/3/2023 -- 02:18:01 - <Info> - eve-log output device (regular) initialized: /nsm/eve-%Y-%m-%d-%H:%M.json
7/3/2023 -- 02:18:01 - <Info> - stats output device (regular) initialized: stats.log
7/3/2023 -- 02:18:02 - <Info> - 1 rule files processed. 33043 rules successfully loaded, 0 rules failed
7/3/2023 -- 02:18:02 - <Info> - Threshold config parsed: 0 rule(s) found
7/3/2023 -- 02:18:02 - <Info> - 33046 signatures processed. 1197 are IP-only rules, 5212 are inspecting packet payload, 26612 inspect application layer, 0 are decoder event only
7/3/2023 -- 02:18:04 - <Info> - can't get cpu-affinity mode
7/3/2023 -- 02:18:04 - <Notice> - all 1 packet processing threads, 4 management threads initialized, engine started.
7/3/2023 -- 02:18:04 - <Info> - Starting file run for /input.pcap
7/3/2023 -- 02:18:04 - <Info> - pcap file /input.pcap end of file reached (pcap err code 0)
7/3/2023 -- 02:18:04 - <Notice> - Signal Received. Stopping engine.
7/3/2023 -- 02:18:04 - <Info> - time elapsed 0.059s
7/3/2023 -- 02:18:04 - <Notice> - Pcap-file module read 1 files, 19 packets, 1473 bytes
7/3/2023 -- 02:18:04 - <Info> - Alerts: 0
7/3/2023 -- 02:18:04 - <Info> - cleaning up signature grouping structure... complete
==== End Suricata Output ====

If any alerts hit, they will be displayed below:

End so-suricata-testrule
=====
```

exécution de la commande

Clarification

Votre solution doit inclure :

- Au moins une (mais probablement plus d'une) option de contenu
- Informations d'en-tête de règle plus spécifiques pour réduire le nombre de paquets qui doivent être comparés à cette règle en production

Suggestion

Nous vous recommandons vivement de prendre l'habitude d'apporter une modification à votre règle à la fois. En d'autres termes, modifiez une option, mettez à jour le numéro de révision, enregistrez la règle, puis vérifiez que la règle change toujours.

Bien que cela puisse sembler plus laborieux et prendre plus de temps, nous pouvons vous dire par expérience que cela rendra le dépannage très simple. Si vous prenez une règle qui fonctionne, apportez une modification et que la règle cesse de fonctionner, quelle modification a enfreint la règle ? La réponse est évidente

Indice

Lors de l'ajout de règles de contenu, vous n'avez pas à vous soucier de l'ordre du contenu dans le paquet par rapport à l'ordre des options de contenu dans les options de règle. Bien qu'il existe certaines options de règle qui nécessitent strictement un ordre spécifique pour les options de contenu, à ce stade l'ordre n'a pas d'importance.

L'inclusion de plusieurs options de contenu rend simplement toutes ces options obligatoires. S'il en manque, il n'y aura pas d'alerte.

Indice

Cette section nécessite que vous mettiez à jour l'en-tête de la règle pour qu'il soit plus spécifique. Bien que vous puissiez être extrêmement précis (y compris les adresses IP, par exemple), cela n'est probablement pas judicieux. Le numéro de port du serveur et le protocole sont les seules choses que vous devriez vraiment changer.

Références :

<https://tools.netsa.cert.org/silk/silk-reference-guide.html#x1-1830001>

<https://tools.netsa.cert.org/silk/silk-reference-guide.html>