

Masterminds

Practice analyzing malicious traffic using Brim.

- *Andres Valdivieso Pinilla - Líder de Ciberseguridad (Consultor)*
- www.linkedin.com/in/andres-valdivieso-pinilla

Introducción

Brim es una herramienta de código abierto utilizada para la exploración y análisis forense de tráfico de red, diseñada para trabajar con grandes volúmenes de datos de captura de paquetes (PCAP) y registros de Zeek. Ofrece una interfaz intuitiva para filtrar, buscar y visualizar eventos de red, facilitando la detección de amenazas y el análisis de incidentes de seguridad. Además, permite exportar datos a Zeek y Splunk, integrándose en flujos de trabajo de ciberseguridad.

Planteamiento del compromiso



Tres equipos del departamento de Finanzas de Pfeffer PLC se vieron comprometidos. Sospechamos que la causa inicial del ataque fue un intento de phishing y una unidad USB infectada. El equipo de Respuesta a Incidentes logró extraer los registros de tráfico de red de los endpoints. Utilice Brim para investigar el tráfico de red en busca de indicios de un ataque y determinar quién está detrás de los ataques.

NOTA: NO interactúe directamente con ningún dominio ni dirección IP en este desafío.

[Infección 1]

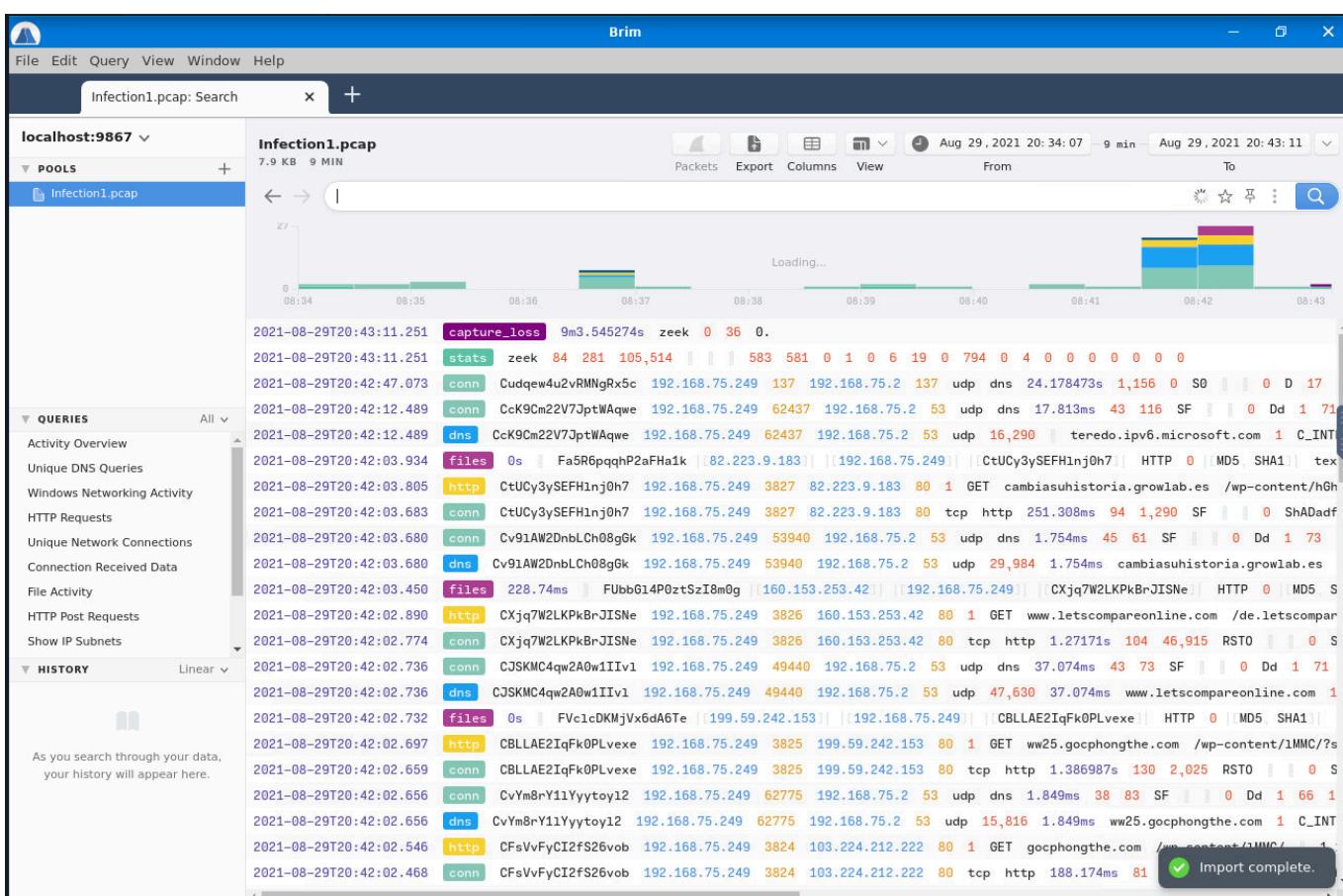
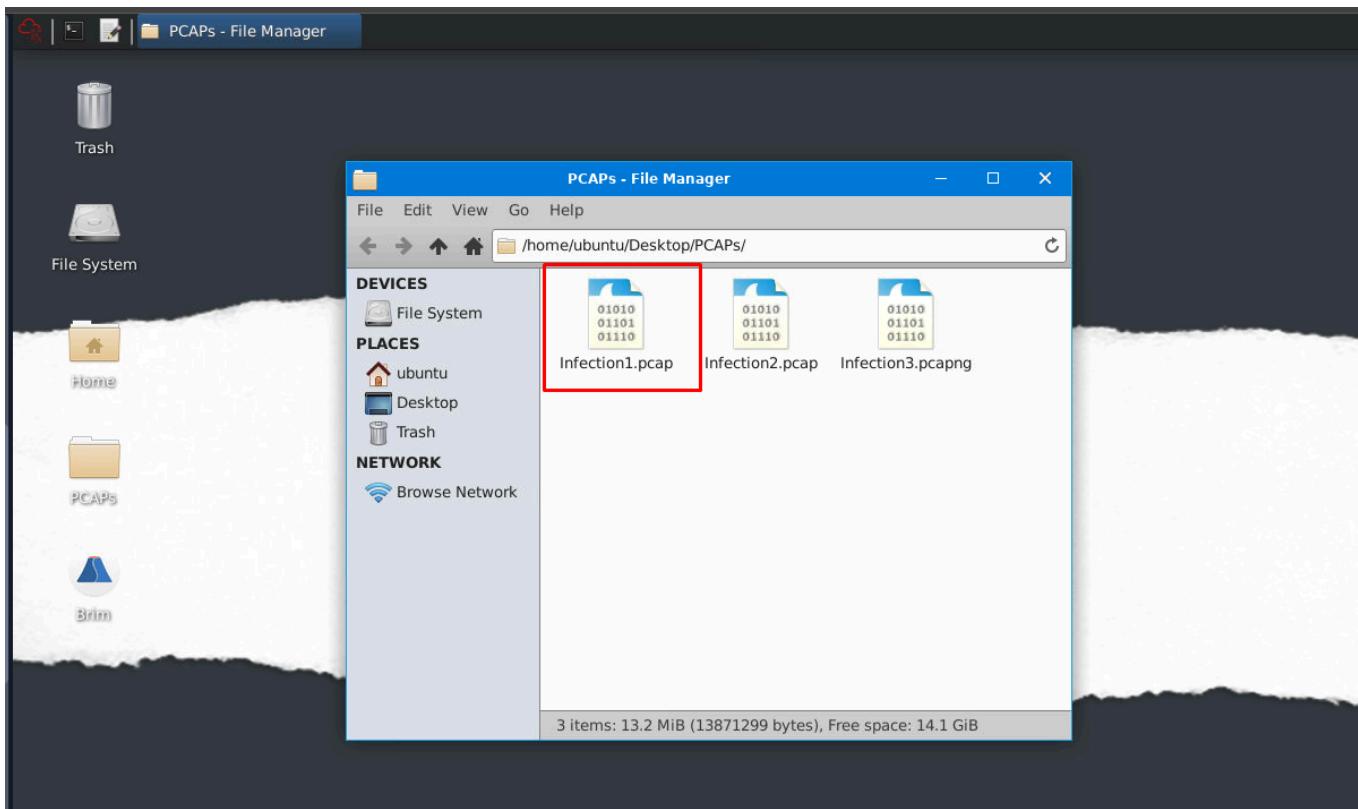


Comience cargando la captura de paquetes de Infection1 en Brim para investigar el evento de vulneración en la primera máquina. Todos los PCAP se pueden encontrar aquí: </home/ubuntu/Desktop/PCAPs>

Nota : Para preguntas que requieren múltiples respuestas, sepárelas con una coma.

Responda las preguntas a continuación

Lo primero es abrir los pcap para ver la información con la que cuento



Proporcione la dirección IP de la víctima.

Lo primero que valide fueron los path que tengo a dispocicion en este pecap cpn
esto voy a ver las conecciones de red

```
_path=="conn" | cut id.orig_h, id.resp_p, id.resp_h | sort | uniq
```

Explicación de la consulta:

1. `_path=="conn"`

- Filtra los registros para trabajar solo con datos de conexiones (`conn.log`), que contienen información sobre direcciones IP, puertos, duración de sesión, bytes transferidos, etc.

2. `cut id.orig_h, id.resp_p, id.resp_h`

- Extrae solo los siguientes campos relevantes:
 - `id.orig_h`: IP de origen (quién inició la conexión).
 - `id.resp_p`: Puerto de destino al que se conectó la IP de origen.
 - `id.resp_h`: IP de destino (el servidor o sistema remoto al que se conectó la IP de origen).

3. `sort`

- Ordena los resultados alfabéticamente o numéricamente, dependiendo del tipo de dato.

4. `uniq`

- Elimina registros duplicados, dejando solo valores únicos.

Lo que busco con este comando es:

- Extrae las direcciones IP de origen, los puertos de destino y las direcciones IP de destino.
- Ordena los resultados.
- Muestra solo combinaciones únicas de estos tres valores.

The screenshot shows the Brim interface with a query results table highlighted by a red box. The table has three columns: id.orig_h, id.resp_p, and id.resp_h. The data is as follows:

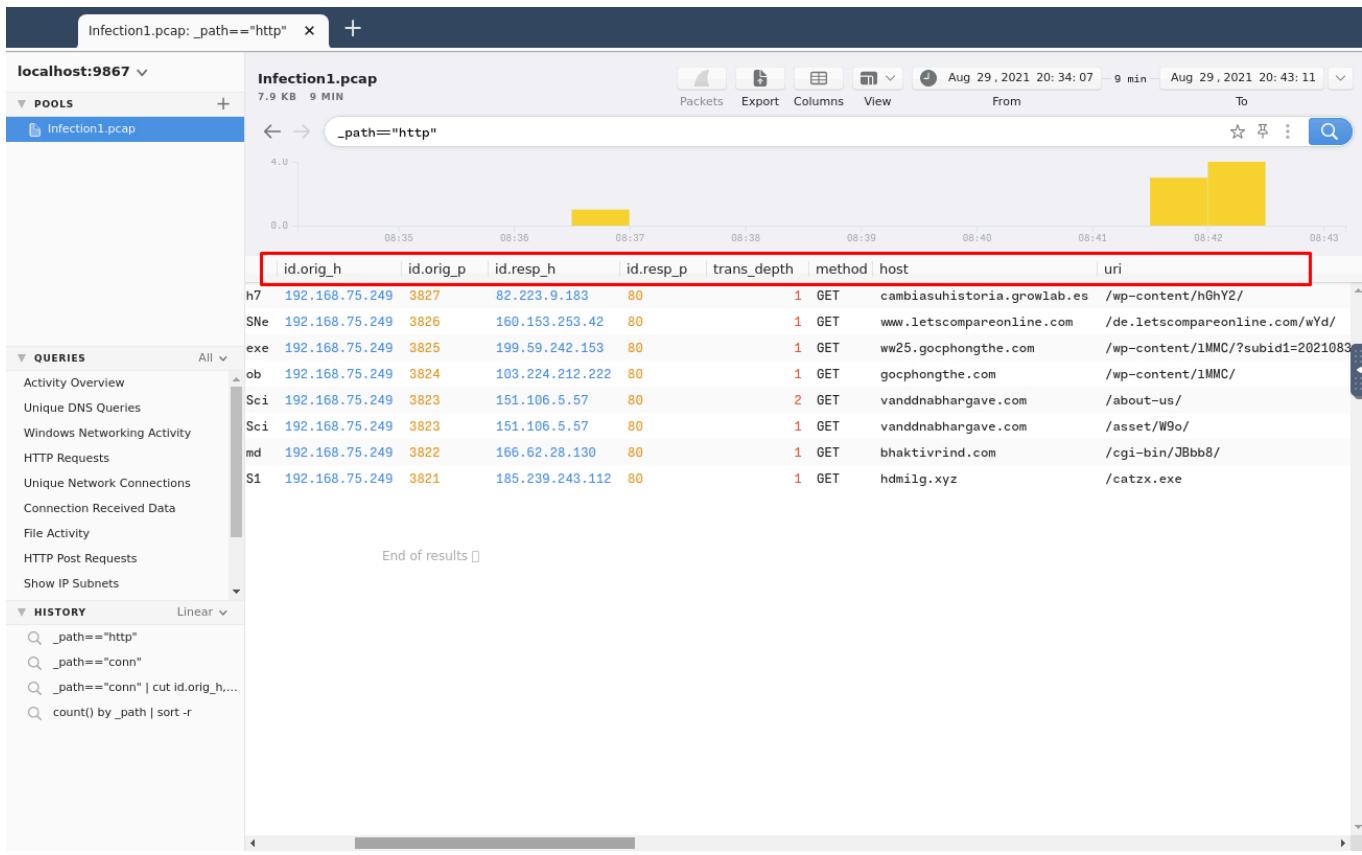
id.orig_h	id.resp_p	id.resp_h
192.168.75.249	53	192.168.75.2
192.168.75.249	67	192.168.75.254
192.168.75.249	67	255.255.255.255
192.168.75.249	80	82.223.9.183
192.168.75.249	80	160.153.253.42
192.168.75.249	80	199.59.242.153
192.168.75.249	80	103.224.212.222
192.168.75.249	80	151.106.5.57
192.168.75.249	80	166.62.28.130
192.168.75.249	80	185.239.243.112
192.168.75.249	137	192.168.75.2
192.168.75.249	137	192.168.75.255
192.168.75.249	137	192.168.75.2
192.168.75.249	138	192.168.75.255
fe80::d8a1:cfa3:ad70:5aa5	547	ff02::1:2
192.168.75.249	1900	239.255.255.250
192.168.75.1	1900	239.255.255.250
fe80::d8a1:cfa3:ad70:5aa5	1900	ff02::c
192.168.75.1	1900	239.255.255.250
192.168.75.249	1900	239.255.255.250

Con esto puedo determinar cual es la única ip interna.

Ahora nos piden validar la conexiones HTTP que la víctima intentó establecer con dos dominios sospechosos con el estado "404 No encontrado". y nos piden proporcionar los hosts/dominios solicitados.

Para esto lo primero es validar los registros en el http

```
_path=="http"
```



Con esto ya saco las casillas que debo usar para la consulta completa

```
_path=="http" | cut host, uri, status_msg
```

Explicación de la consulta:

1. `_path=="http"`

- Filtrar los registros para trabajar únicamente con eventos HTTP (provenientes de `http.log` en Zeek).

2. `cut host, uri, status_msg`

- Extrae solo los siguientes campos:

- `host`: Nombre del servidor o dominio al que se realizó la solicitud HTTP.
- `uri`: La ruta de la solicitud HTTP (por ejemplo, `/index.html`).
- `status_msg`: El mensaje de estado HTTP asociado a la respuesta (por ejemplo, "OK", "Not Found", "Forbidden").

Lo que busco ver con esta consulta es:

- Filtra los registros HTTP.
- Extrae solo el **host**, la **URI** solicitada y el **mensaje de estado** de la respuesta.

The screenshot shows the Wireshark interface with a query results table. The query entered is `_path=="http" | cut host, uri, status_msg`. The results table has columns: host, uri, and status_msg. The data is as follows:

host	uri	status_msg
cambiasuhistoria.growlab.es	/wp-content/hGhY2/	Not Found
www.letscompareonline.com	/de.letscompareonline.com/wYd/	Not Found
ww25.gocphongthe.com	/wp-content/lMMC/?subid1=20210830-0642-0217-8b33-6ab4bf16c29c	OK
gocphongthe.com	/wp-content/lMMC/	Found
vanddnabhargave.com	/about-us/	OK
vanddnabhargave.com	/asset/W9o/	Found
bhaktivrind.com	/cgi-bin/JBbb8/	
hdmlg.xyz	/catzx.exe	Internal Server Error

Podemos identificar que las únicas dos URL que están asociadas al Not Found son las dos primeras url.

La víctima se conectó correctamente a uno de los dominios mediante HTTP y recibió un valor de `response_body_len` de 1309 (**tamaño del contenido sin comprimir de los datos transferidos desde el servidor**). con esto nos piden Indicar el dominio y la dirección IP de destino.

Esta consulta tambien se realiza desde el archivo del http que tiene la consulta ahora lo unico que tenemos que tener en cuenta es el tamaño del contenido ya que nos proporcionan el valor especifico es mas facil encontarr los valores solicitados usando esta consulta:

```
_path=="http" | cut id.orig_h, host, response_body_len
```

Explicación de la consulta:

1. `_path=="http"`

- Filtra los registros para trabajar únicamente con eventos HTTP (provenientes de `http.log` en Zeek).

2. `cut id.orig_h, host, response_body_len`

- Extrae solo los siguientes campos relevantes:

- `id.orig_h` → Dirección IP del cliente que realizó la solicitud HTTP.
- `host` → Nombre del servidor o dominio al que se realizó la solicitud HTTP.
- `response_body_len` → Tamaño (en bytes) del cuerpo de la respuesta HTTP.

Con esta podemos:

- Filtra los eventos HTTP.
- Extrae la IP de origen, el host al que se conectó y el tamaño del contenido recibido en la respuesta.

The screenshot shows the Brim interface. On the left, there's a sidebar with sections like 'POOLS' (localhost:9867), 'QUERIES' (Activity Overview, Unique DNS Queries, etc.), and 'HISTORY' (with two recent queries). The main area has a title bar 'Infection1.pcap: _path=="http"...'. Below it is a table with the following data:

id.resp_h	host	response_body_len
82.223.9.183	cambiasuhistoria.growlab.es	1,020
160.153.253.42	www.letscompareonline.com	46,456
199.59.242.153	ww25.gocphongthe.com	1,309
103.224.212.222	gocphongthe.com	0
151.106.5.57	vanddnabhargave.com	28,308
151.106.5.57	vanddnabhargave.com	0
166.62.28.130	bhaktivrind.com	0
185.239.243.112	hdmlg.xyz	0

The row for '199.59.242.153' is highlighted with a red box. At the bottom right of the table, it says 'End of results'.

Así podemos identificar cual es el dominio y ip a la cual se conecto la victima.

¿Cuántas solicitudes de DNS únicas se realizaron al dominio cab[.]myfkn[.]com (incluido el dominio en mayúsculas)?

```
_path=="dns" | count() by query | sort -r
```

Explicación de la consulta:

1. `_path=="dns"`

Filtrar los registros para trabajar únicamente con eventos DNS.

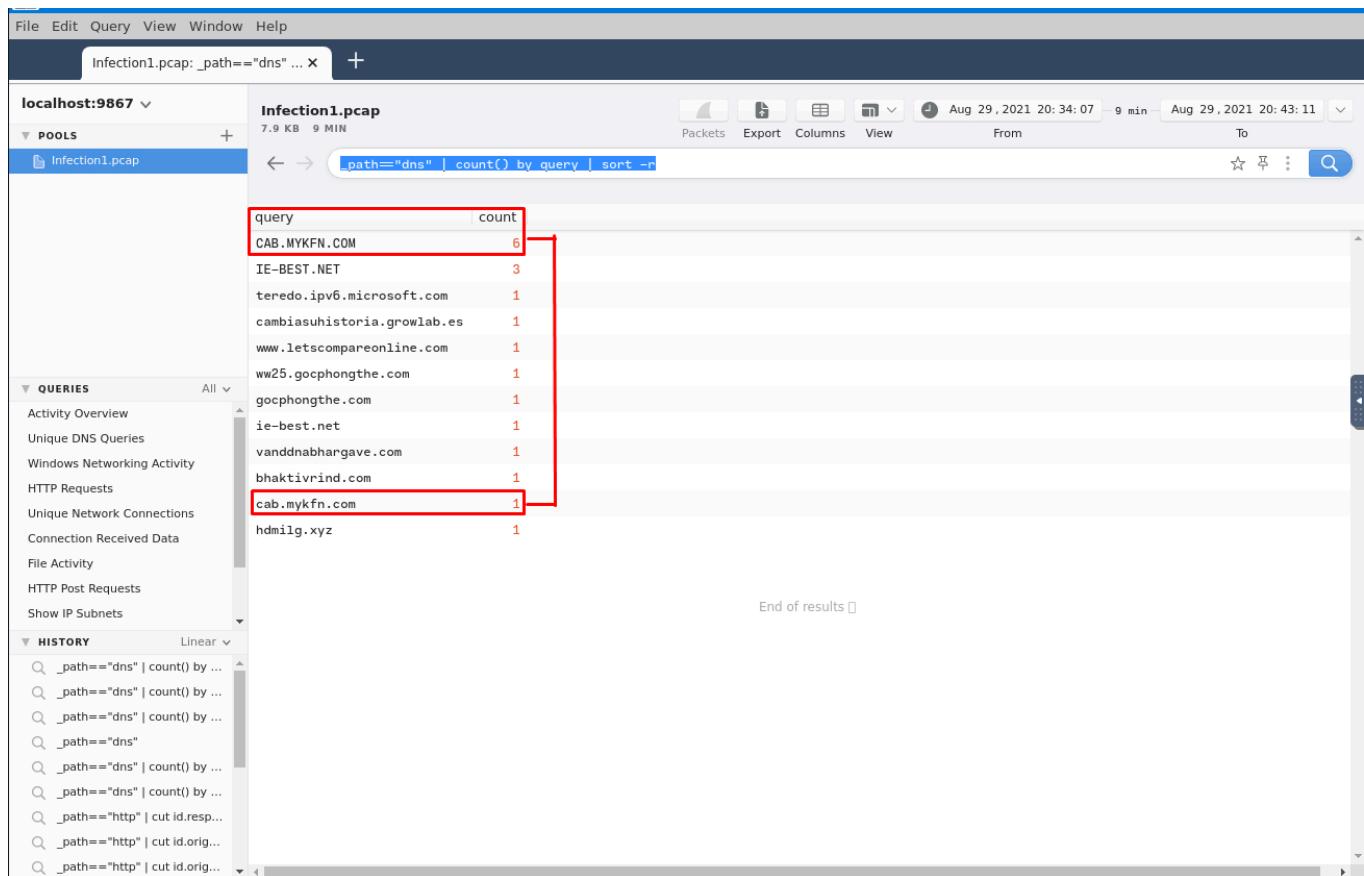
2. `count() by query`

Cuenta cuántas veces aparece cada consulta de dominio (`query`).

3. `sort -r`

Ordena los resultados en orden descendente por la cantidad de consultas.

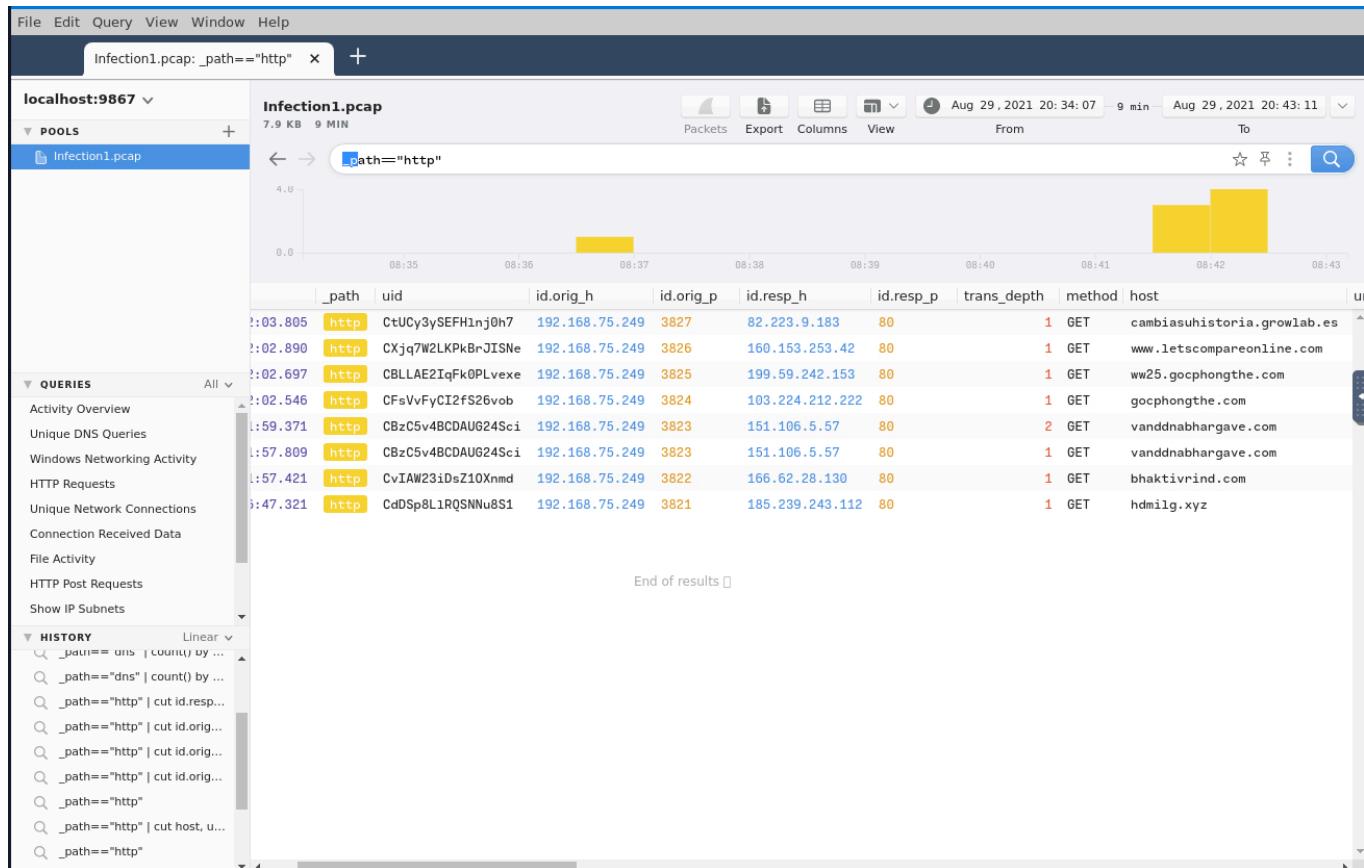
Lo que quiero es que me cuente cuántas veces se ha consultado cada dominio y me ordene los resultados en orden descendente, mostrando los dominios más consultados primero.



Se puede ver que el conteo esta separado en por lo que toca tenerlo encuentra a la hora de responder por o qe se suman los dos resultados.

Proporcione la URI del dominio bhaktivrind[.]com al que la víctima accedió a través de HTTP.

_path=="http"



_path=="http" | cut host, id.orig_h, uri | bhaktivrind.com

Explicación de la consulta:

1. `_path=="http"`

Filtre los registros para trabajar únicamente con eventos HTTP.

2. `cut host, id.orig_h, uri`

Extrae solo los siguientes campos relevantes:

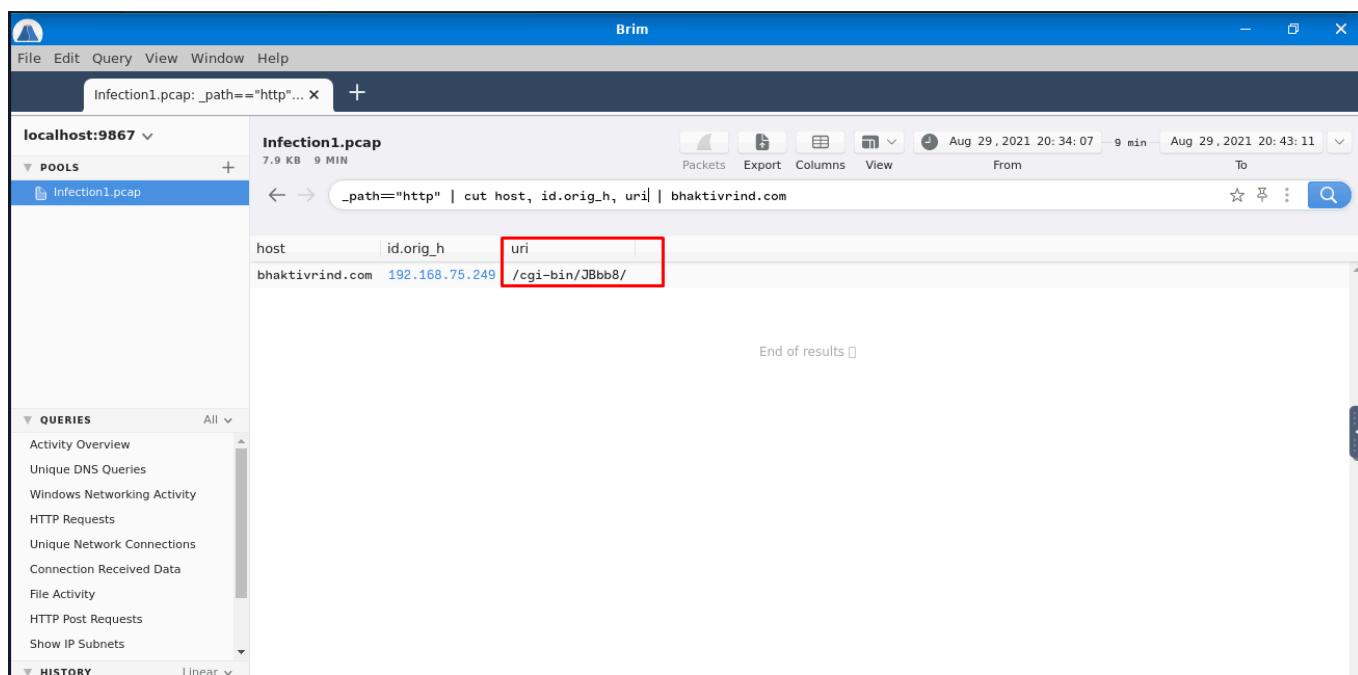
- **host** → Nombre del servidor o dominio al que se realizó la solicitud HTTP.
- **id.orig_h** → Dirección IP del cliente que realizó la solicitud HTTP.

- **uri** → La ruta de la solicitud HTTP.

3. **bhaktivrind.com**

Buscar específicamente solicitudes hacia el dominio **bhaktivrind.com**.

Esta consulta solo filtra eventos HTTP, extrae el host, la IP de origen y la URI de las solicitudes, y luego busca específicamente registros relacionados con el dominio que nos están solicitando que es **bhaktivrind.com**.



The screenshot shows the Brim interface with the following details:

- File Bar:** File, Edit, Query, View, Window, Help.
- Toolbar:** Infection1.pcap, 7.9 KB, 9 MIN, Packets, Export, Columns, View, Date range: Aug 29, 2021 20:34:07 - 9 min to Aug 29, 2021 20:43:11, From, To, Star, Filter, Search.
- Left Panel:** POOLS (localhost:9867), Infection1.pcap selected.
- Center Panel:** A search bar contains the query: `_path=="http" | cut host, id.orig_h, uri | bhaktivrind.com`. Below it, a table shows results with columns: host, id.orig_h, and uri. One row is highlighted with a red box: `bhaktivrind.com 192.168.75.249 /cgi-bin/JBbb8/`.
- Bottom Panel:** QUERIES section with various options like Activity Overview, Unique DNS Queries, Windows Networking Activity, etc. HISTORY section with Linear view.

Proporcione la dirección IP del servidor malicioso y el ejecutable que la víctima descargó del servidor.

```
_path=="http" | cut id.orig_h, id.resp_h, id.resp_p, method, host, uri
| uniq -c
```

Explicación por partes:

1. **_path=="http"**

Filtre los registros para trabajar únicamente con eventos HTTP.

2. **cut id.orig_h, id.resp_h, id.resp_p, method, host, uri**

Extrae los siguientes campos relevantes:

- **id.orig_h** → Dirección IP del cliente que realizó la solicitud HTTP.

- **id.resp_h** → Dirección IP del servidor de destino.
- **id.resp_p** → Puerto de destino del servidor.
- **method** → Método HTTP utilizado (GET, POST, etc.).
- **host** → Nombre del servidor o dominio al que se realizó la solicitud HTTP.
- **uri** → La URI (ruta) solicitada en la petición HTTP.

3. **uniq -c**

Cuenta las ocurrencias únicas de las combinaciones de los campos extraídos.

Con este extraemos información sobre las solicitudes (IP de origen, IP de destino, puerto, método HTTP, host y URI), y luego cuenta cuántas veces ocurren combinaciones únicas de esos campos.

id.orig_h	id.resp_h	id.resp_p	method	host	uri
192.168.75.249	82.223.9.183	80	GET	cambiasuhistoria.growlab.es	/wp-content/hGhY2/
192.168.75.249	160.153.253.42	80	GET	www.letscompareonline.com	/de.letscompareonline.com/wYd/
192.168.75.249	199.59.242.153	80	GET	ww25.gocphongthe.com	/wp-content/1MMC/?subid=20210830-0642-0217-8b33-6ab4bf16c29d
192.168.75.249	103.224.212.222	80	GET	gocphongthe.com	/wp-content/1MMC/
192.168.75.249	151.106.5.57	80	GET	vanddnabhangave.com	/about-us/
192.168.75.249	151.106.5.57	80	GET	vanddnabhangave.com	/asset/W9o/
192.168.75.249	166.62.28.130	80	GET	bhaktivrind.com	/cgi-bin/JBbb8/
192.168.75.249	185.239.243.112	80	GET	hdmi1g.xyz	catzx.exe

La respuesta esta asociada al único .exe que se ve dentro de las url.

Con base en la información recopilada en la segunda pregunta, proporcione el nombre del malware que utiliza **VirusTotal**.

En la segunda pregunta nos preguntaban por las conexiones HTTP a dos dominios sospechosos con el estado "404 No encontrado". en este teniamos el

hosts/dominios solicitados estos eran **cambiasuhistoria.growlab.es**, **www.letscompareonline.com** con esto solo vamos a virustotal y ponemos el dominio:

VirusTotal analysis for **cambiasuhistoria.growlab.es**. Community Score: 4 / 94. 4/94 security vendors flagged this domain as malicious. Last Analysis Date: 1 month ago.

Comments (2)

tines_bot 4 years ago
#emotet
This IOC was found in a paste: <https://pastebin.com/aZPxwcr> with the title "Weekend Emotet IoCs and Notes for 2021/01/22-24" by jroosen
For more information, or to report interesting/incorrect findings, contact us - bot@tines.io

VirusTotal analysis for **www.letscompareonline.com**. Community Score: 4 / 94. 4/94 security vendors flagged this domain as malicious. Registrar: GoDaddy.com, LLC. Creation Date: 15 years ago. Last Analysis Date: 25 days ago.

Comments (2)

tines_bot 4 years ago
#emotet
This IOC was found in a paste: <https://pastebin.com/aZPxwcr> with the title "Weekend Emotet IoCs and Notes for 2021/01/22-24" by jroosen
For more information, or to report interesting/incorrect findings, contact us - bot@tines.io

Para los dos esta el mismo nombre de malware que es Emotet.

Nota:

Para los siguientes ejercicios se sigue el mismo principio con las consultas.

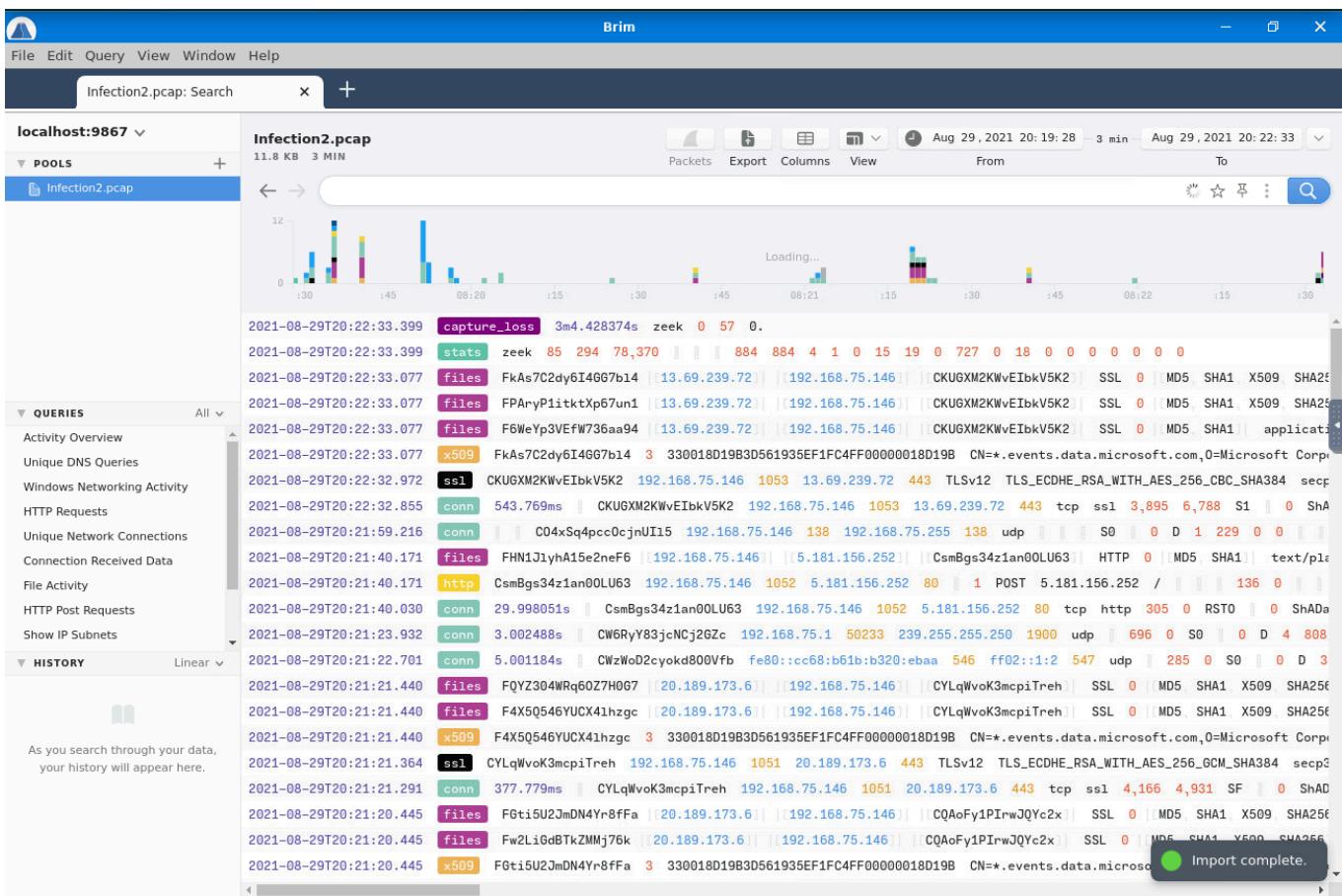
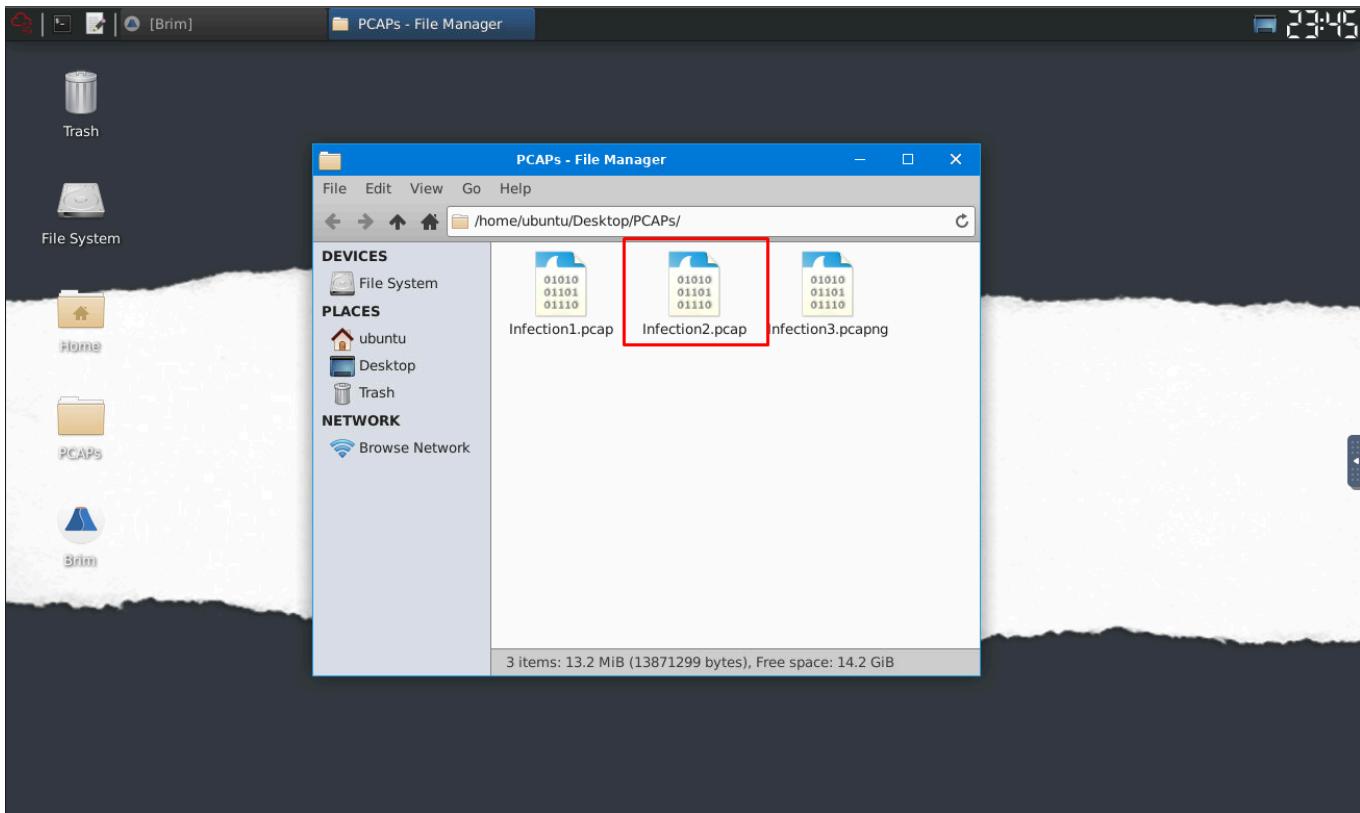
[Infección 2]



Por favor, navegue hasta la captura de paquetes de Infection2 en Brim para investigar el evento de compromiso para la segunda máquina.

Nota : Para preguntas que requieren múltiples respuestas, sepárelas con una coma.

Responda las preguntas a continuación



Proporcione la dirección IP de la máquina víctima.

The screenshot shows the Brim interface with the following details:

- File**: Edit, Query, View, Window, Help
- Pools**: localhost:9867, Infection2.pcap
- Infection2.pcap**: 11.8 KB, 3 MIN
- Time Range**: Aug 29, 2021 20:19:28 - Aug 29, 2021 20:22:33
- Search Bar**: count() by _path | sort -r
- Table Results** (highlighted with a red box):

_path	count
conn	34
dns	23
files	18
x509	6
ssl	6
http	4
ntp	2
stats	2
dhcp	1
capture_loss	1
- Queries**: Activity Overview, Unique DNS Queries, Windows Networking Activity, HTTP Requests, Unique Network Connections, Connection Received Data, File Activity, HTTP Post Requests, Show IP Subnets
- History**: Linear, count() by _path | sort -r

```
_path=="conn" | cut id.orig_h, id.resp_p, id.resp_h | sort | uniq
```

The screenshot shows the Brim interface with the following details:

- File**: Edit, Query, View, Window, Help
- Pools**: localhost:9867, Infection2.pcap
- Infection2.pcap**: 11.8 KB, 3 MIN
- Time Range**: Aug 29, 2021 20:19:28 - Aug 29, 2021 20:22:33
- Search Bar**: _path=="conn" | cut id.orig_h, id.resp_p, id.resp_h | sort | uniq
- Table Results** (highlighted with a red box):

id.orig_h	id.resp_p	id.resp_h
192.168.75.146	53	192.168.75.2
192.168.75.146	67	192.168.75.254
192.168.75.146	67	255.255.255.255
192.168.75.146	80	5.181.156.252
192.168.75.146	80	45.95.203.28
192.168.75.146	123	168.61.215.74
192.168.75.146	137	192.168.75.2
192.168.75.1	137	192.168.75.255
192.168.75.146	138	192.168.75.255
192.168.75.146	443	13.69.239.72
192.168.75.146	443	20.189.173.6
192.168.75.146	443	104.88.193.243
192.168.75.146	443	52.113.194.132
192.168.75.146	443	52.109.76.31
192.168.75.146	443	104.88.193.243
192.168.75.146	443	88.99.66.31
192.168.75.146	443	195.201.225.248
fe80::cc68:b61b:b320:ebaa	547	ff02::1:2
192.168.75.1	1900	239.255.255.250
fe80::cc68:b61b:b320:ebaa	5353	ff02::fb
192.168.75.1	5353	224.0.0.251
192.168.75.1	5355	224.0.0.252
fe80::cc68:b61b:b320:ebaa	5355	ff02::1:1
- Queries**: Activity Overview, Unique DNS Queries, Windows Networking Activity, HTTP Requests, Unique Network Connections, Connection Received Data, File Activity, HTTP Post Requests, Show IP Subnets
- History**: Linear, _path=="conn" | cut id.orig_h,..., count() by _path | sort -r

Proporcione la dirección IP a la que la víctima realizó las conexiones POST.

```
_path=="http"
```

The screenshot shows the Brim interface with a search query: `_path=="http"`. The results table is highlighted with a red border. The columns are: ts, _path, uid, id.orig_h, id.orig_p, id.resp_h, id.resp_p, trans_depth, method, host. The data shows four HTTP requests from 192.168.75.146 to 5.181.156.252, with one GET request from hypercustom.top.

ts	_path	uid	id.orig_h	id.orig_p	id.resp_h	id.resp_p	trans_depth	method	host
2021-08-29T20:21:40.171	http	CsmBgs34z1an00LU63	192.168.75.146	1052	5.181.156.252	80	1	POST	5.181.156.252
2021-08-29T20:20:40.187	http	Cng9HJ1go5ARUiF8y2	192.168.75.146	1048	5.181.156.252	80	1	POST	5.181.156.252
2021-08-29T20:19:40.669	http	CZcXYz2Fnj0CvJtPT1	192.168.75.146	1047	5.181.156.252	80	1	POST	5.181.156.252
2021-08-29T20:19:35.686	http	C90w3o4CqizhWWjIrg	192.168.75.146	1046	45.95.203.28	80	1	GET	hypercustom.top

```
_path=="http" | cut id.orig_h, host method | sort | uniq -c
```

The screenshot shows the Brim interface with a search query: `_path=="http" | cut id.orig_h, host, method | sort | uniq -c`. The results table is highlighted with a red border. The columns are: id.orig_h, host, method, _uniq. The data shows two hosts: 192.168.75.146 and hypercustom.top, with their respective POST and GET counts.

id.orig_h	host	method	_uniq
192.168.75.146	5.181.156.252	POST	3
192.168.75.146	hypercustom.top	GET	1

¿Cuántas conexiones POST se realizaron a la dirección IP en la pregunta anterior?

id.orig_h	host	method	_uniq
192.168.75.146	5.181.156.252	POST	3
192.168.75.146	hypercustom.top	GET	1

Proporcione el dominio desde donde se descargó el binario.

```
_path=="http" | cut id.orig_h, host method | sort | uniq -c
```

id.orig_h	host	method	_uniq
192.168.75.146	5.181.156.252	POST	3
192.168.75.146	hypercustom.top	GET	1

Proporcione el nombre del binario incluido el URI completo.

```
_path=="http" | cut id.orig_h, host method, uri | sort | uniq -c
```

The screenshot shows the Brim network analysis tool interface. The main window displays a table of results for a query: `_path=="http" | cut id.orig_h, host, method, uri | sort | uniq -c`. The table has columns: id.orig_h, host, method, uri, and _uniq. Two rows are shown: one for a POST request to 5.181.156.252 and another for a GET request to hypercustom.top. The 'uri' column is highlighted with a red box. The Brim interface includes a sidebar with 'POOLS' and 'QUERIES' sections, and a top bar with file operations and search functions.

id.orig_h	host	method	uri	_uniq
192.168.75.146	5.181.156.252	POST	/	3
192.168.75.146	hypercustom.top	GET	/jollion/apines.exe	1

Proporcione la dirección IP del dominio que aloja el binario.

```
_path=="http" | cut id.resp_h, host method, uri | sort | uniq -c
```

The screenshot shows the Brim network analysis tool interface. The main window displays a table of results for a query: `_path=="http" | cut id.resp_h, host, method, uri | sort | uniq -c`. The table has columns: id.resp_h, host, method, uri, and _uniq. Two rows are shown: one for a POST request from 5.181.156.252 and another for a GET request from 45.95.203.28. The 'id.resp_h' column is highlighted with a red box. The Brim interface includes a sidebar with 'POOLS' and 'QUERIES' sections, and a top bar with file operations and search functions.

id.resp_h	host	method	uri	_uniq
5.181.156.252	5.181.156.252	POST	/	3
45.95.203.28	hypercustom.top	GET	/jollion/apines.exe	1

Se recibieron dos alertas de Suricata sobre la detección de un troyano de red.
¿Cuáles eran las direcciones IP de origen y destino?

Screenshot of Brim interface showing a query results table. A context menu is open over the row "1 A Network Trojan was detected".

alert.severity	alert.category	count
2	Potentially Bad Traffic	2
1	A Network Trojan was detected	2

Context menu options (highlighted with a red box):

- Filter = value
- Filter != value
- New search with this value
- Pivot to logs** (selected)
- Count by field
- Copy
- Sort A...Z
- Sort Z...A
- Use as "start" time
- Use as "end" time
- View in full context
- Open details
- Whois Lookup

Screenshot of Brim interface showing a detailed log table for the alert "A Network Trojan was detected".

ts	event_type	src_ip	src_port	dest_ip	dest_port	vlan	proto	app_proto	alert.severity	alert.signature
2021-08-29T20:20:01.932	alert	192.168.75.146	1046	45.95.203.28	80		TCP	http	1	ET USER_AGENT
2021-08-29T20:20:01.932	alert	192.168.75.146	1046	45.95.203.28	80		TCP	http	1	ET HUNTING SL

Table columns:

- ts
- event_type
- src_ip
- src_port
- dest_ip
- dest_port
- vlan
- proto
- app_proto
- alert.severity
- alert.signature

Al observar el dominio .top en las solicitudes HTTP, proporcione el nombre del ladrón (troyano que recopila información de un sistema) involucrado en esta captura

de paquetes utilizando la base de datos URLhaus .

The screenshot shows the Brim Network Analyzer interface. On the left, a sidebar lists network activity categories like POOLS, QUERIES, and HTTP Requests. The QUERIES section is expanded, with 'Unique DNS Queries' highlighted by a red box. In the main pane, a search bar contains the query `_path=="dns" | count() by query | sort -r`. Below it is a table titled 'Infection2.pcap' showing DNS query counts:

query	count
wpad.local	8
WPAD	6
wpad	4
telete.in	1
mobile.pipe.aria.microsoft.com	1
time.windows.com	1
hypercustom.top	1
iplogger.org	1

At the bottom right of the main pane, there is a link 'End of results'.

- Informe

The screenshot shows the URLhaus website homepage. At the top, a yellow banner displays the text: 'NEW | Hunt across all abuse.ch platforms with one simple query - discover if an IPv4 address, domain, URL or file hash has been identified on any platform from a centralized search tool. Test it out here [hunting.abuse.ch](#) - and happy hunting!'. Below the banner, the URLhaus logo is visible, along with navigation links: Browse, Hunting Alerts, Access Data, FAQ, About, and Login.

Submit a URL

In order to submit a URL to URLhaus, you need to login with [your abuse.ch account](#)

Browse Database

The screenshot shows the URLhaus database browse page. A search bar at the top contains the query: 'domain, url, md5, sha256, tag:SocGholish, filetype:doc or url_status:online'. Below the search bar, there are two tabs: 'URLs' and 'Payloads', with 'URLs' selected. A table lists malware URLs with their details:

Dateadded (UTC)	Malware URL	Status	Tags	Reporter
2021-08-21 19:44:08	http://hypercustom.top/jollion/apines.exe	Offline	cryptbot.exe opendir RedLineStealer	abuse_ch
2021-08-19 19:47:07	http://hypercustom.top/jollion/apines1.exe	Offline	32.exe opendir RedLineStealer	zbetcheckin
2021-08-19 19:02:05	http://hypercustom.top/jollion/lipster.exe	Offline	32.exe opendir RedLineStealer	zbetcheckin
2021-08-19 18:57:06	http://hypercustom.top/holler/rollerkind2.exe	Offline	32.exe RedLineStealer	zbetcheckin
2021-08-19 18:57:06	http://hypercustom.top/holler/rollerkind.exe	Offline	32.exe RedLineStealer	zbetcheckin

NEW | Hunt across all abuse.ch platforms with one simple query - discover if an IPv4 address, domain, URL or file hash has been identified on any platform from a centralized search tool. Test it out here [hunting.abuse.ch](#) - and happy hunting!

Payload delivery

The table below documents all payloads that URLhaus retrieved from this particular URL.

Firstseen	Filename	File Type	Payload (SHA256)	VT	Bazaar	Signature
2021-09-01	n/a	exe	d6007d59009ba577d65fb1e122dffec2cd0771c34445cd263e1c2699c21368	n/a		RedLineStealer
2021-08-31	n/a	exe	0648b1ab9937a98ee4aba023573e64bfa6b533f5f15d29c42a0cae43844e026d	n/a		RedLineStealer
2021-08-30	n/a	exe	4da095e0a59cecad3fbfc6fa4f33c00e71edc9ffae041a521552390825732f7	n/a		RedLineStealer
2021-08-29	n/a	exe	82b0161335f3e5103150aedc18f94ea8cecbe00871668de002af752e9ffff9dd	n/a		RedLineStealer
2021-08-28	n/a	exe	0c53ee5bbb0e0624fb8281a8b8850e6f190a97aae229fb9f7bfd0e054988f5e8	n/a		RedLineStealer
2021-08-27	n/a	exe	45a87dd4abe07b6c5f296a9ff716f5224b38dd17e4fd8789819a6c9cbcfda55c	n/a		RedLineStealer
2021-08-26	n/a	exe	25a8d52e15f54022a9efdfaf2c3f8bf94e36dc475d428fc880c953f1bbaf2c6	n/a		RedLineStealer
2021-08-25	n/a	exe	cce16e3c61f188234483344b7e95f174df1b6708002a6a0ae63ab197745c8090	n/a		RedLineStealer
2021-08-25	n/a	exe	abc87c7b821bb7bba854958ecd20760e63c9365aff2091edc2b8449040263a67	56.72%		CryptBot
2021-08-24	n/a	exe	8b46a621bdeee821d8688fc152c9ebca9b4d274cba2d4550787a79da83dce04	n/a		RedLineStealer
2021-08-23	n/a	exe	374fe48e2677be33b362d63a5d0f306a29fd7cd0e19302b8bdf4dbbeb46b16b	n/a		RedLineStealer
2021-08-22	n/a	exe	1995d5c5569c14b92d2dd5d849b8a06b5e27280ac6cddaaacc3c00584741a6d0	n/a		RedLineStealer
2021-08-22	n/a	exe	45682ad0823016643d27985aa1d5d0dcffe4be09ddac5419d0447342d9c6f0b	n/a		RedLineStealer
2021-08-21	n/a	exe	fdcce927665c3300db41bbe2c93005756f7da11fef5557ec5c9c5fa7196bf32c	n/a		RedLineStealer
2021-08-21	n/a	exe	c5749c5654d0508326fdf6a51dd8ec091724a6beb711654650c032ab618e9749	53.73%		RedLineStealer

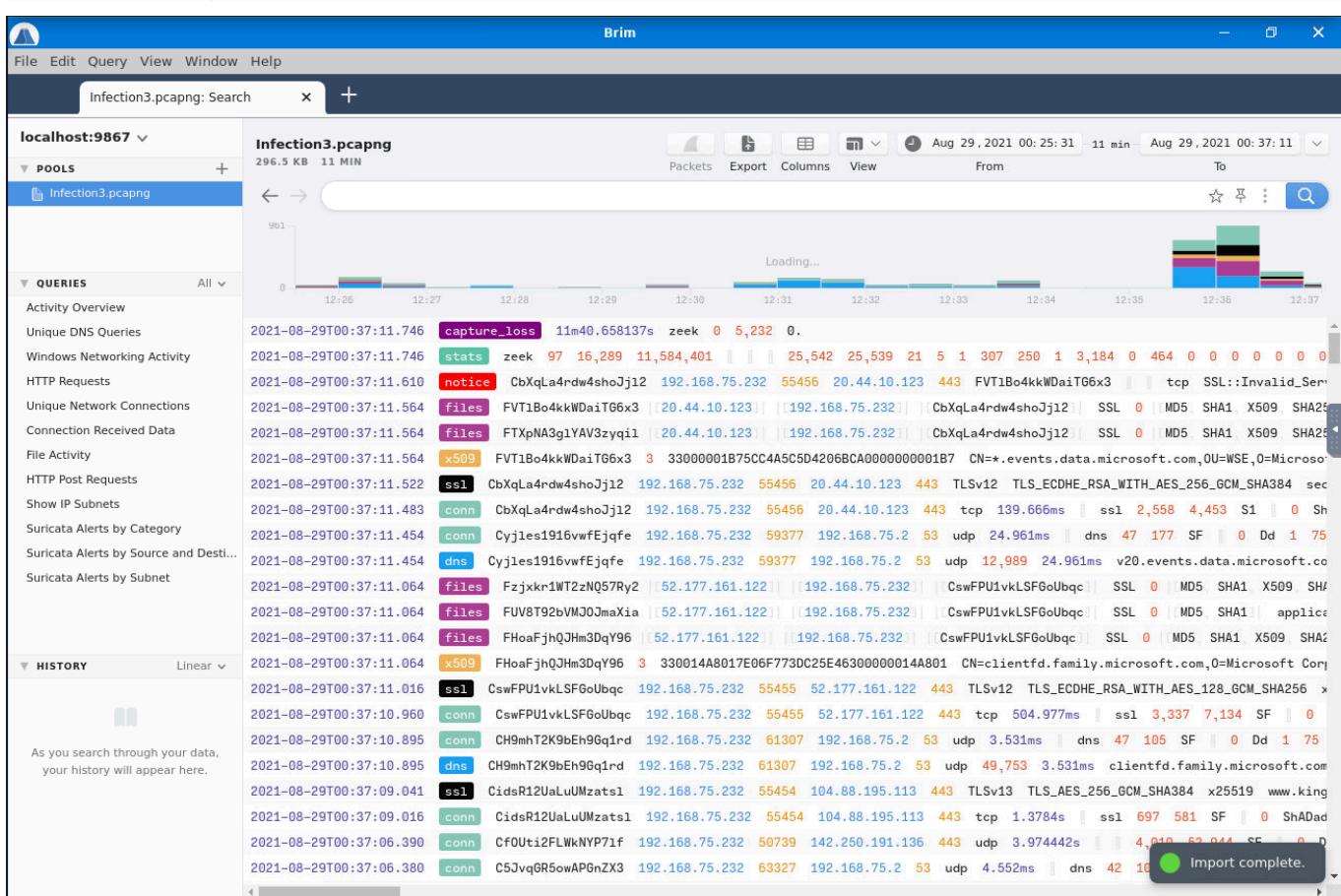
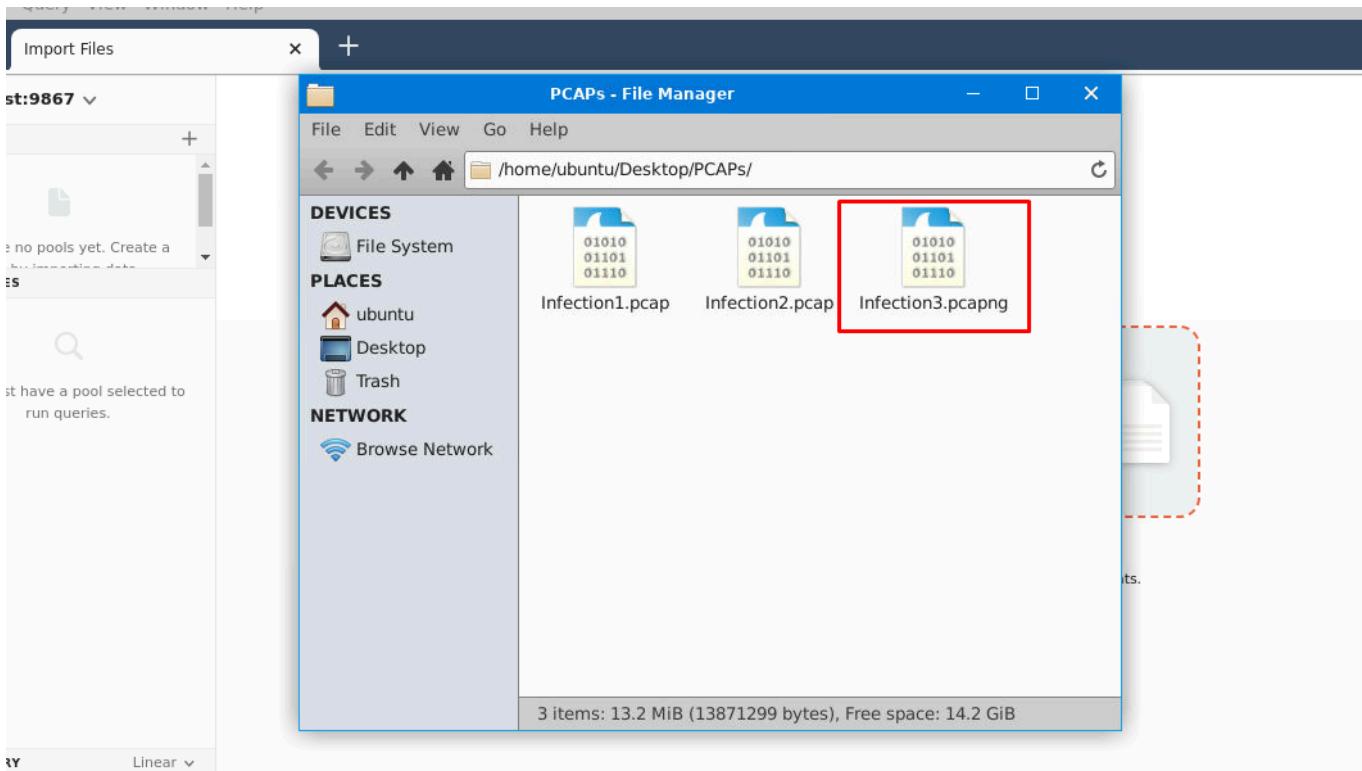
[Infección 3]



Por favor, cargue la captura de paquetes de Infection3 en Brim para investigar el evento de compromiso para la tercera máquina.

Nota : Para preguntas que requieren múltiples respuestas, sepárelas con una coma.

Responda las preguntas a continuación



Proporcione la dirección IP de la máquina víctima.

Brim

File Edit Query View Window Help

Infection3.pcapng: count() by _... x +

localhost:9867 v Infection3.pcapng 296.5 KB 11 MIN

Pools Export Columns View Aug 29, 2021 00:25:31 11 min Aug 29, 2021 00:37:11 From To

count() by _path | sort -r

_path	count
dns	986
conn	914
files	574
ssl	331
x509	200
ntlm	40
http	25
notice	11
dhcp	8
stats	4
weird	3
capture_loss	1

End of results □

Queries All

- Activity Overview
- Unique DNS Queries
- Windows Networking Activity
- HTTP Requests
- Unique Network Connections
- Connection Received Data
- File Activity
- HTTP Post Requests
- Show IP Subnets
- Suricata Alerts by Category
- Suricata Alerts by Source and Dest...
- Suricata Alerts by Subnet

History Linear

count() by _path | sort -r

```
_path=="conn" | cut id.orig_h, id.resp_h | sort | uniq -c
```

Saturday 29 March 2025

File Edit Query View Window Help

Infection3.pcapng: _path=="co... x +

localhost:9867 v Infection3.pcapng 296.5 KB 11 MIN

Packets Export Columns View Aug 29, 2021 00:25:31 11 min Aug 29, 2021 00:37:11 From To

_path=="conn" | cut id.orig_h, id.resp_h | sort | uniq -c

id.orig_h	id.resp_h	_uniq
0.0.0.0	255.255.255.255	1
192.168.1.1	192.168.75.232	6
192.168.75.1	192.168.75.232	5
192.168.75.1	192.168.75.254	1
192.168.75.1	192.168.75.255	1
192.168.75.1	224.0.0.251	11
192.168.75.1	224.0.0.252	44
192.168.75.1	239.255.255.250	5
192.168.75.133	13.69.116.104	2
192.168.75.133	20.42.73.25	2
192.168.75.133	20.189.173.4	7
192.168.75.133	20.189.173.6	2
192.168.75.133	52.109.8.30	1
192.168.75.133	52.113.194.132	1
192.168.75.133	104.88.193.162	1
192.168.75.133	104.88.193.243	1
192.168.75.133	192.168.75.2	33
192.168.75.133	192.168.75.232	43
192.168.75.133	192.168.75.254	3
192.168.75.133	192.168.75.255	5
192.168.75.133	255.255.255.255	2
192.168.75.232	3.92.156.8	2
192.168.75.232	3.209.200.144	1
192.168.75.232	3.211.82.118	2

History Linear

_path=="conn" | cut id.orig_h,...
 _path=="conn" | cut id.orig_h,...
 _path=="conn" | cut id.orig_h,...
 count() by _path | sort -r

En esta consulta tuve problemas en la identificación de la IP, por lo que como no

queda claro cual es la ip debido a la cantidad de trafico se puede validar el total de bytes transferidos entre los puntos finales con el siguiente comando

```
_path=="conn" | put total_bytes := orig_bytes + resp_bytes | sort -r  
total_bytes | cut uid, id, orig_bytes, resp_bytes, total_bytes
```

Explicación de la consulta:

1. `_path=="conn"`

- Filtra únicamente los registros que pertenecen a la conexión de red (`conn`).

2. `put total_bytes := orig_bytes + resp_bytes`

- Calcula una nueva columna llamada `total_bytes`, que es la suma de los bytes enviados (`orig_bytes`) y los bytes recibidos (`resp_bytes`).

3. `sort -r total_bytes`

- Ordena los resultados de manera descendente (`-r` significa "reverse"), mostrando primero las conexiones con mayor tráfico total (`total_bytes`).

4. `cut uid, id, orig_bytes, resp_bytes, total_bytes`

- Finalmente, selecciona las columnas clave que queremos visualizar en el resultado final:

- `uid`: Identificador único de la conexión.
- `id`: Contiene los detalles de identificación de la conexión.
- `orig_bytes`: Bytes enviados desde la IP de origen.
- `resp_bytes`: Bytes recibidos por la IP de destino.
- `total_bytes`: Total de bytes intercambiados (calculado previamente).

Con esto lo que buscamos es filtrar el log de conexiones (`conn.log`), extrae las direcciones IP, calcula el total de bytes transferidos, ordenar las conexiones por mayor tráfico y mostrar solo la información relevante.

Brim

File Edit Query View Window Help

Infection3.pcapng: _path=="co... x +

localhost:9867

POOLS

Infection3.pcapng

296.4 KB 11 MIN

Packets Export Columns View From To

_path=="conn" | put total_bytes := orig_bytes + resp_bytes | sort -r total_bytes | cut uid, id, orig_bytes, resp_b

QUERIES

All

Activity Overview
Unique DNS Queries
Windows Networking Activity
HTTP Requests
Unique Network Connections
Connection Received Data
File Activity
HTTP Post Requests
Show IP Subnets

HISTORY

Linear

uid	id.orig_h	id.orig_p	id.resp_h	id.resp_p	orig_bytes	resp_bytes	total_bytes
CD3fkq4ZxzKyVa18d8	192.168.75.232	53289	93.184.216.54	443	6,301	1,047,942	1,054,243
C1cfL1LNQ0bewB14	192.168.75.232	58396	13.107.21.200	443	321,782	484,080	805,862
CNPDyU11kNLM5A67P6	192.168.75.232	59950	93.184.216.54	443	4,333	596,223	600,556
Cdz3sJinhXEPvxDSud	192.168.75.232	64320	205.185.216.42	443	4,008	479,262	483,270
CNKuVfcmakn9yPNy7	192.168.75.232	62184	168.61.170.191	443	35,410	418,675	454,085
CVvXPU102KEpQgJ4Mf	192.168.75.232	55588	13.107.21.200	443	133,200	311,267	444,467
CfxPxe2zrT1tqoKrzg	192.168.75.232	62159	168.61.170.191	443	26,067	344,320	370,387
CCUeP02NVQTd0qIEWb	192.168.75.232	63866	13.226.138.27	443	1,171	337,777	338,948
CWjtxi2WTqvZNDPRc1	192.168.75.232	49310	13.226.138.27	443	5,824	313,892	319,716
C1XSax1nVDOb3Etfw1	192.168.75.232	57475	13.107.21.200	443	101,071	218,515	319,586
CYJHfkmZRy0atJTWg	192.168.75.232	58547	13.107.21.200	443	61,678	173,809	235,487
CISxVT9tKPC9ibBkk	192.168.75.133	1070	104.88.193.162	443	1,001	223,585	224,586
CGGmhj1sVuzBBwe1Zb	192.168.75.232	49185	13.226.138.27	443	5,105	212,661	217,766
CpTf122KvdMDw1SMaj	192.168.75.232	65041	205.185.216.10	443	3,339	198,017	201,356
CHFwKH2YSnJBYYbad	192.168.75.232	51426	13.107.213.51	443	3,381	196,036	199,417
CmpGbpRrTLhK2jw7	192.168.75.133	1067	104.88.193.243	443	583	178,216	178,799
CGnvtf33n98EIQZ7Ad	192.168.75.232	52700	143.204.0.2	443	1,165	162,842	164,007
C2Q04a3UHgaAJsPJ5a	192.168.75.232	60984	13.107.21.200	443	9,835	147,401	157,236
CXieGK3qr0zHRAcCv1	192.168.75.232	57116	13.107.21.200	443	20,370	136,651	157,021
CFYtJv18yWkLknRtkb	192.168.75.232	49528	104.88.195.113	443	1,285	144,611	145,896
CSbC4gpLVEzqEyZwi	192.168.75.232	60227	23.53.113.151	443	1,097	117,264	118,361
CDTn1wRHa720MAkjf	192.168.75.232	50087	13.226.138.27	443	4,520	112,957	117,477
COTWEWM1Rxe0VSQL5	192.168.75.232	49831	168.61.170.191	443	16,596	91,793	108,389
CiIxNftPEpBTTXC1	192.168.75.232	57853	93.184.216.54	443	1,850	104,133	105,983

Así ya podemos ver cual es la ip con mayor transferencia de bytes.

Proporcione tres dominios C2 desde los cuales se descargaron los binarios (comenzando desde el más antiguo hasta el más reciente en la marca de tiempo)

```
_path=="http" | cut id.orig_h, method,host, uri | sort | uniq -c
```

The screenshot shows the Brim interface with a packet capture named 'Infection3.pcapng'. The search bar displays the command: `_path=="http" | cut id.orig_h, method, host, uri | sort | uniq -c`. The results table lists network traffic with the following columns: id.orig_h, method, host, uri, and _uniq. The results show multiple requests to 'xfhoahegue.ru' and 'afhoahegue.ru' with various file paths like '/s/5.exe', '/s/4.exe', etc. The row for 'afhoahegue.ru' at index 192.168.75.232 is highlighted with a red box.

id.orig_h	method	host	uri	_uniq
192.168.75.232	GET	x1.i.lencr.org	/	3
192.168.75.232	GET	xfhoahegue.ru	/s/5.exe	1
192.168.75.232	GET	xfhoahegue.ru	/s/4.exe	1
192.168.75.232	GET	xfhoahegue.ru	/s/3.exe	1
192.168.75.232	GET	xfhoahegue.ru	/s/2.exe	1
192.168.75.232	GET	xfhoahegue.ru	/s/1.exe	1
192.168.75.232	GET	afhoahegue.ru	/s/5.exe	1
192.168.75.232	GET	afhoahegue.ru	/s/4.exe	1
192.168.75.232	GET	afhoahegue.ru	/s/3.exe	1
192.168.75.232	GET	afhoahegue.ru	/s/2.exe	1
192.168.75.232	GET	tile-service.weather.microsoft.com	/en-US/livetile/preinstall?region=US&appid=C98EA5B0842DBB9405BBF071	1
192.168.75.232	GET	afhoahegue.ru	/s/1.exe	1
192.168.75.232	GET	ctld1.windowsupdate.com	/msdownload/update/v3/static/trustedr/en/pinrulesst1.cab?7892c23e00	1
192.168.75.232	GET	ctld1.windowsupdate.com	/msdownload/update/v3/static/trustedr/en/authrootsst1.cab?24a277ab8a	1
192.168.75.232	GET	ctld1.windowsupdate.com	/msdownload/update/v3/static/trustedr/en/disallowedcertst1.cab?1af1	1
192.168.75.232	GET	efhoahegue.ru	/s/5.exe	1
192.168.75.232	GET	efhoahegue.ru	/s/4.exe	1
192.168.75.232	GET	efhoahegue.ru	/s/3.exe	1
192.168.75.232	GET	efhoahegue.ru	/s/2.exe	1
192.168.75.232	GET	efhoahegue.ru	/s/1.exe	1
192.168.75.232	GET	xfhoahegue.ru	/s/VNEW=1	1
192.168.75.232	GET	afhoahegue.ru	/s/VNEW=1	1
192.168.75.232	GET	efhoahegue.ru	/s/VNEW=1	1

Proporcione las direcciones IP de los tres dominios en la pregunta anterior.

```
_path=="http" | cut id.resp_h, method, host, uri | sort | uniq -c
```

The screenshot shows the Brim interface with the following details:

- File Edit Query View Window Help**
- Infection3.pcapng: _path=="http" X +**
- localhost:9867** (selected in the POOLS dropdown)
- Infection3.pcapng** (selected in the QUERIES dropdown)
- Packets Export Columns View**
- Aug 29, 2021 00:25:31 11 min Aug 29, 2021 00:37:11**
- From To**
- _path=="http" | cut id.resp_h, method, host, uri | sort | uniq -c**
- id.resp_h method host uri _uniq**

id.resp_h	method	host	uri	_uniq
23.38.133.193	GET	tile-service.weather.microsoft.com	/en-US/livetile/preinstall?region=US&appid=C98EA5B0842DBB9405BBF071	1
63.251.106.25	GET	xfohahegue.ru	/s/5.exe	1
63.251.106.25	GET	xfohahegue.ru	/s/4.exe	1
63.251.106.25	GET	xfohahegue.ru	/s/3.exe	1
63.251.106.25	GET	xfohahegue.ru	/s/2.exe	1
63.251.106.25	GET	xfohahegue.ru	/s/1.exe	1
63.251.106.25	GET	xfohahegue.ru	/s/VNEW=1	1
162.217.98.146	GET	efhoahegue.ru	/s/5.exe	1
162.217.98.146	GET	efhoahegue.ru	/s/4.exe	1
162.217.98.146	GET	efhoahegue.ru	/s/3.exe	1
162.217.98.146	GET	efhoahegue.ru	/s/2.exe	1
162.217.98.146	GET	efhoahegue.ru	/s/1.exe	1
162.217.98.146	GET	efhoahegue.ru	/s/VNEW=1	1
184.85.171.231	GET	x1.i.lencr.org	/	3
199.21.76.77	GET	afhoahegue.ru	/s/5.exe	1
199.21.76.77	GET	afhoahegue.ru	/s/4.exe	1
199.21.76.77	GET	afhoahegue.ru	/s/3.exe	1
199.21.76.77	GET	afhoahegue.ru	/s/2.exe	1
199.21.76.77	GET	afhoahegue.ru	/s/1.exe	1
199.21.76.77	GET	afhoahegue.ru	/s/VNEW=1	1
209.197.3.8	GET	ctld1.windowsupdate.com	/msdownload/update/v3/static/trustedr/en/pinrulesstl.cab?7892c23e00	1
209.197.3.8	GET	ctld1.windowsupdate.com	/msdownload/update/v3/static/trustedr/en/authrootstl.cab?24a277ab8a	1
209.197.3.8	GET	ctld1.windowsupdate.com	/msdownload/update/v3/static/trustedr/en/disallowedcertstl.cab?1af1	1

QUERIES: All

 - Activity Overview
 - Unique DNS Queries
 - Windows Networking Activity
 - HTTP Requests
 - Unique Network Connections
 - Connection Received Data
 - File Activity
 - HTTP Post Requests
 - Show IP Subnets

HISTORY: Linear

 - _path=="http" | cut id.res...
 - _path=="http" | cut host ...
 - _path=="http" | cut hos | ...
 - _path=="http" | cut host, ...
 - _path=="http" | cut id.orl...
 - _path=="http" | cut id.orl...

¿Cuántas consultas DNS únicas se realizaron al dominio asociado desde la primera dirección IP de la respuesta anterior?

```
_path=="dns" | 162.217.98.146 | count() by query | sort -r
```

The screenshot shows the Brim interface with the following details:

- File Menu:** File, Edit, Query, View, Window, Help.
- Title Bar:** Brim
- Current File:** Infection3.pcapng: _path=="dn... X
- Pool List:** localhost:9867, POOLS, Infection3.pcapng (selected).
- Search Bar:** _path=="dns" | 162.217.98.146 | count() by query | sort -r (highlighted with a red box).
- Results Table:** A table showing the count of queries for a specific IP address.

query	count
efhoahegue.ru	2
- Text:** End of results □
- Queries Sidebar:** Activity Overview, Unique DNS Queries, Windows Networking Activity, HTTP Requests, Unique Network Connections, Connection Received Data, File Activity, HTTP Post Requests, Show IP Subnets.

¿Cuántos binarios se descargaron en total del dominio mencionado anteriormente?

```
_path=="http" | efhoahegue.ru | cut id.orig_h, id.resp_h, id.resp_p,  
method,host, uri | uniq -c
```

id.orig_h	id.resp_h	id.resp_p	method	host	uri	_uniq
192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/5.exe	1
192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/4.exe	1
192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/3.exe	1
192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/2.exe	1
192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/1.exe	1
192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/VNEW=1	1

Proporcione el agente de usuario indicado para descargar los binarios.

```
_path=="http" | efhoahegue.ru | cut id.resp_h, id.resp_p, host, uri,  
user_agent | uniq -c
```

id.resp_h	id.resp_p	host	uri	user_agent	_uniq
162.217.98.146	80	efhoahegue.ru	/s/5.exe	Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/2010010	1
162.217.98.146	80	efhoahegue.ru	/s/4.exe	Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/2010010	1
162.217.98.146	80	efhoahegue.ru	/s/3.exe	Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/2010010	1
162.217.98.146	80	efhoahegue.ru	/s/2.exe	Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/2010010	1
162.217.98.146	80	efhoahegue.ru	/s/1.exe	Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/2010010	1
162.217.98.146	80	efhoahegue.ru	/s/VNEW=1	Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/2010010	1

Proporcione la cantidad de conexiones DNS realizadas en total para esta captura de paquetes.

```
_path=="dns" | count() by query | sort -r | sum (count)
```

Explicación de la consulta:

1. `_path=="dns"`

- Filtra solo los registros que pertenecen a eventos de consultas DNS (Zeek los guarda en `dns.log`).

2. * `count() by query`

- Cuenta cuántas veces aparece cada consulta DNS (`query`).
- Agrupa los resultados por el campo `query`, que representa el dominio consultado.

3. `sort -r`

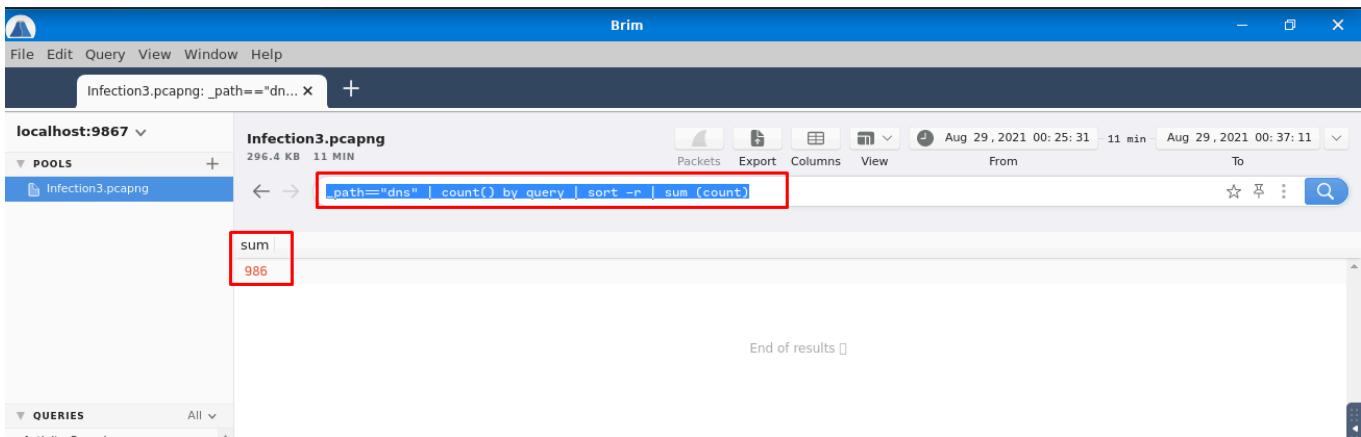
- Ordena los resultados en orden descendente (`-r`), mostrando primero las consultas más frecuentes.

4. `sum(count)`

- Suma el total de consultas DNS registradas en el dataset.

En resumen lo que busco es:

1. Filtra solo los eventos de DNS.
2. Contar cuántas veces se consultó cada dominio.
3. Ordena los dominios más consultados de mayor a menor.
4. Calcula el total de consultas DNS registradas.



Con algunas habilidades de OSINT, proporcione el nombre del gusano utilizando el primer dominio que haya logrado recopilar de la Pregunta 2. (Utilice comillas para las búsquedas de Google, no utilice .ru en su búsqueda y NO interactúe con el dominio directamente).

Lo primero es buscar el dominio lo hice en inicio con virustotal con esto encontramos este **Informe** en el se puede ver en la parte de los detalles algunos resultados de google:

lo que hice fue validar los dos mas relevantes para entender a que hacia referencia y el otro es el triage del malware:

- **Phorpiex/Trik Botnet Campaign Leads to Multiple Infections**
- **Triage Phorpiex**