Challenge Writeup: Masterminds

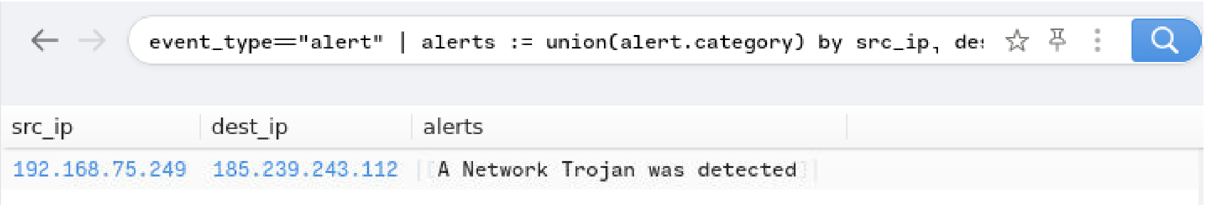
The following is a challenge writeup for the Masterminds room hosted on TryHackMe. This room covers using the tool Brim to investigate a series of pcaps. Brim is simply an open-source desktop application that processes pcap files and log files.

Scenario: Three machines in the Finance department at Pfeffer PLC were compromised. We suspect the initial source of the compromise happened through a phishing attempt and by an infected USB drive. The Incident Response team managed to pull the network traffic logs from the endpoints. Use Brim to investigate the network traffic for any indicators of an attack and determine who stands behind the attack.

Infection 1

Provide the victim's IP address.

There would be several ways to identify the victim's IP address, however, I went the low effort route by checking out the Suricata logs using the 'Suricata Alerts by Source and Destination' query:



```
event_type=="alert" | alerts := union(alert.category) by src_ip, de:
```

src_ip	dest_ip	alerts
192.168.75.249	185.239.243.112	A Network Trojan was detected

The whole query is as follows:

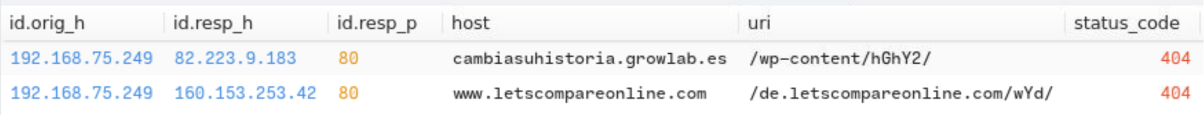
- event_type=="alert" | alerts := union(alert.category) by src_ip, dest_ip

The source IP under the src_ip category is the answer.

The victim attempted to make HTTP connections to two suspicious domains with the status '404 Not Found. Provide the hosts/domains requested.

To answer this question, we can investigate the HTTP logs produced by Zeek, I used the following query to drill down on HTTP logs that have the status code 404:

- _path=="http" status_code==404 | cut id.orig_h, id.resp_h, id.resp_p, host, uri, status_code



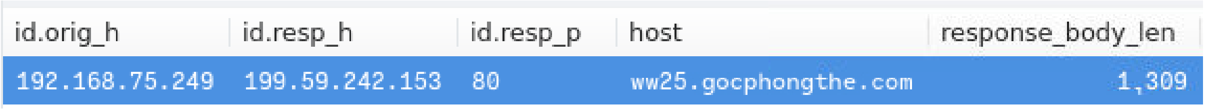
id.orig_h	id.resp_h	id.resp_p	host	uri	status_code
192.168.75.249	82.223.9.183	80	cambiasuhistoria.growlab.es	/wp-content/hGhY2/	404
192.168.75.249	160.153.253.42	80	www.letscompareonline.com	/de.letscompareonline.com/wYd/	404

The answer is simply the first host (entire domain),second host.

The victim made a successful HTTP connection to one of the domains and received the response_body_len of 1,309 (uncompressed content size of the data transferred from the server). Provide the domain and the destination IP address.

We can simply enter the following query which displays all the responses with a response_body_len that equals to 1,309:

- _path=="http" response_body_len==1309 | cut id.orig_h, id.resp_h, id.resp_p, host, response_body_len



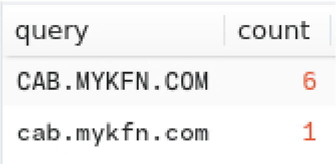
id.orig_h	id.resp_h	id.resp_p	host	response_body_len
192.168.75.249	199.59.242.153	80	ww25.gocphongthe.com	1,309

The answer is host,id.resp_h.

How many unique DNS requests were made to cab[.]myfkn[.]com domain (including the capitalised domain)?

Luckily for us, there is a premade query called 'Unique DNS Queries' that we can slightly modify to find the answer. The query I used to find the answer is as follows:

- _path=="dns" CAB.MYKFN.COM cab.mykfn.com | count() by query | sort -r



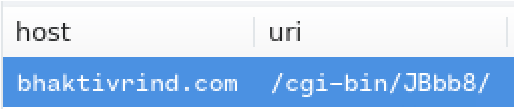
query	count
CAB.MYKFN.COM	6
cab.mykfn.com	1

The answer is 7 (6+1).

Provide the URI of the domain bhaktivrind[.]com that the victim reached out over HTTP.

We can simply search for the http path and the host bhaktivrind[.]com to find the URI in question, the following query allows us to do this:

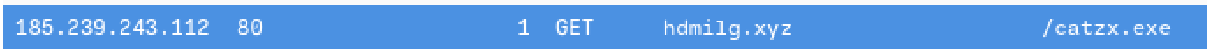
- _path=="http" host=="bhaktivrind.com" | cut host, uri



host	uri
bhaktivrind.com	/cgi-bin/JBbb8/

Provide the IP address of the malicious server and the executable that the victim downloaded from the server.

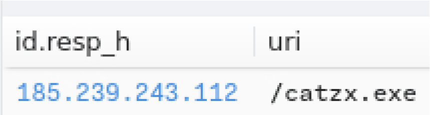
There are multiple ways to find the answer for this question, I did this by searching the http path where I noticed a GET request with the uri /catzx.exe:



185.239.243.112	80	1	GET	hdmilg.xyz	/catzx.exe

Luckily for us there is only one executable file downloaded so we don’t need to determine what file is malicious. I then created the following query which extracts all the information we need for the answer:

- _path=="http" host=="hdmilg.xyz" | cut id.resp_h, uri

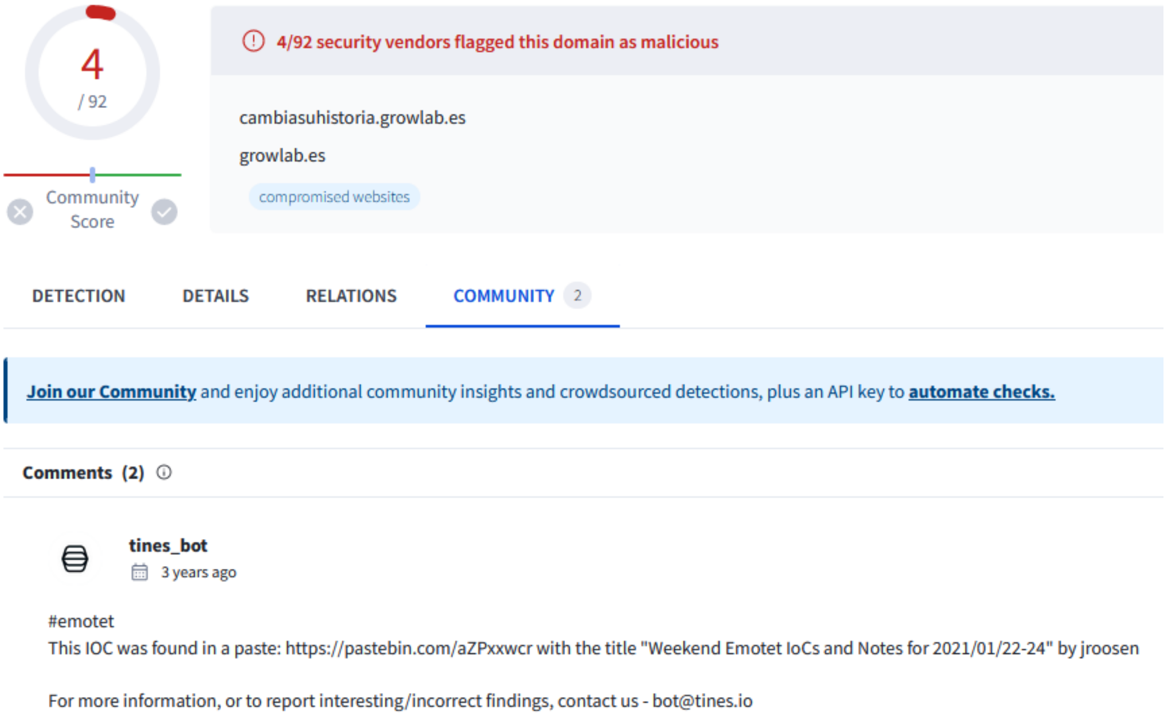


id.resp_h	uri
185.239.243.112	/catzx.exe

The anseer is 185.239.234.112,/catzx.exe.

Based on the information gathered from the second question, provide the name of the malware using VirusTotal.

The answer to the second question was: ‘cambiasuhistoria.growlab.es,www.letscompareonline.com’. Let’s now enter these into VirusTotal to determine the name of the malware.



4/92 security vendors flagged this domain as malicious

cambiasuhistoria.growlab.es

growlab.es

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to **automate checks.**

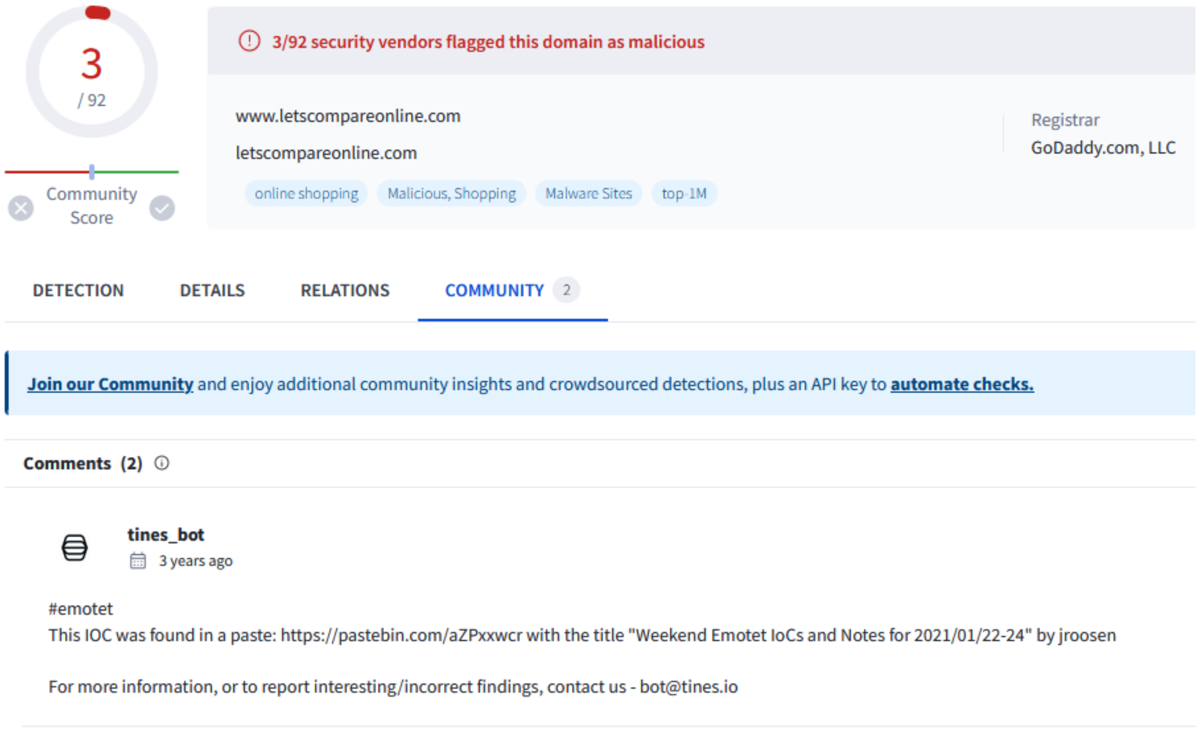
Comments (2)

tines_bot

3 years ago

#emotet
This IOC was found in a paste: https://pastebin.com/aZPxxwcr 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



3/92 security vendors flagged this domain as malicious

www.letscompareonline.com
letscompareonline.com

Registrar
GoDaddy.com, LLC

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to **automate checks.**

Comments (2)

tines_bot
3 years ago

#emotet
This IOC was found in a paste: https://pastebin.com/aZPxxwcr with the title "Weekend Emotet IoCs and Notes for 2021/01/22-24" by jroosen

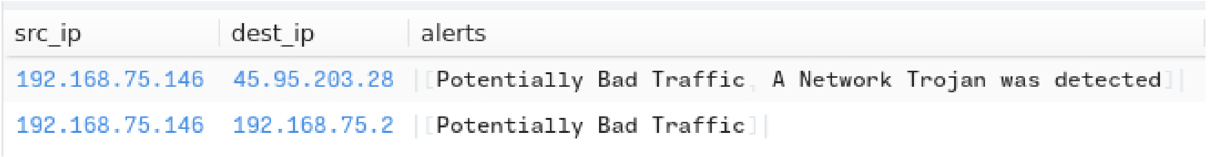
For more information, or to report interesting/incorrect findings, contact us - bot@tines.io

From this, we can clearly determine the malware to be Emotet.

Infection 2

Provide the IP address of the victim machine.

Similar to the first infection, there would be several ways to identify the victim's IP address, however, I went the low effort route by checking out the Suricata logs using the 'Suricata Alerts by Source and Destination' query:



src_ip	dest_ip	alerts
192.168.75.146	45.95.203.28	[Potentially Bad Traffic, A Network Trojan was detected]
192.168.75.146	192.168.75.2	[Potentially Bad Traffic]

The whole query is as follows:

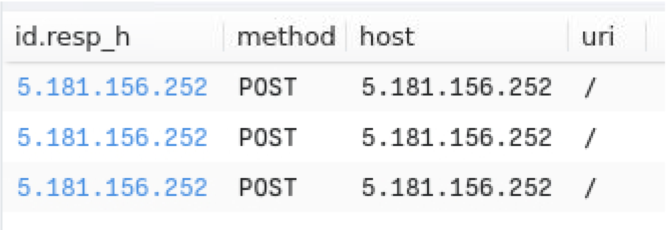
- event_type=="alert" | alerts := union(alert.category) by src_ip, dest_ip

The victim machine is likely 192.168.75.146. In a real world scenario you would obviously want to investigate this further, however, the IP address found is the answer.

Provide the IP address the victim made the POST connections to.

To find this answer, I simply used the HTTP logs and filtered for the victims IP address we found previously. The entire query is as follows:

- _path=="http" 192.168.75.146 method=="POST" | cut id.resp_h, method, host, uri

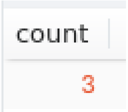


id.resp_h	method	host	uri
5.181.156.252	POST	5.181.156.252	/
5.181.156.252	POST	5.181.156.252	/
5.181.156.252	POST	5.181.156.252	/

The resp_h aka the destination IP is the answer.

How many POST connections were made to the IP address in the previous question?

You can go the simple router and count the POST requests as there are only 3, however, you can also use the count function like as follows:



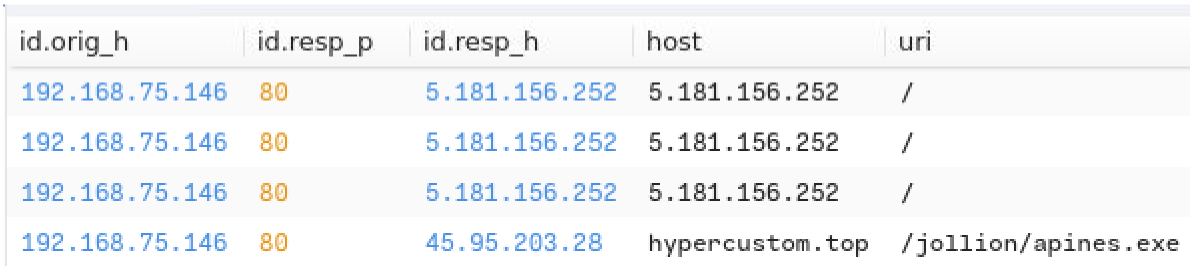
count
3

Query used:

- _path=="http" 192.168.75.146 method=="POST" | cut id.resp_h, method, host | count()

Provide the domain where the binary was downloaded from.

I started off by drilling down on downloaded files over HTTP:



id.orig_h	id.resp_p	id.resp_h	host	uri
192.168.75.146	80	5.181.156.252	5.181.156.252	/
192.168.75.146	80	5.181.156.252	5.181.156.252	/
192.168.75.146	80	5.181.156.252	5.181.156.252	/
192.168.75.146	80	45.95.203.28	hypercustom.top	/jollion/apines.exe

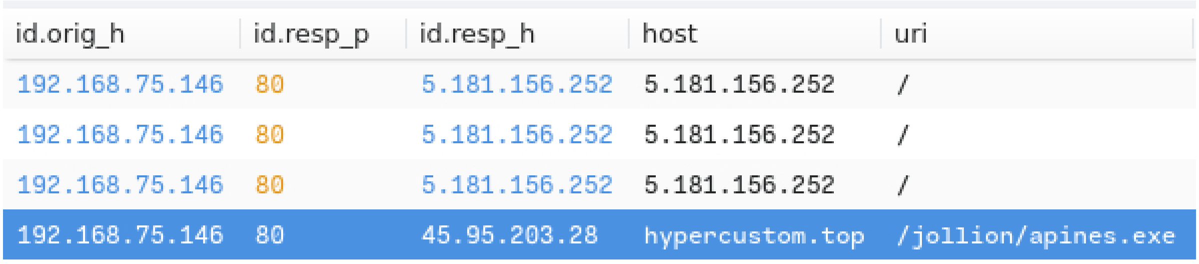
Query used:

- _path=="http" _path=="http" | cut id.orig_h, id.resp_p, id.resp_h, host, uri

This lets us know that an executable called apines.exe was downloaded from hypercustom.top.

Provide the name of the binary including the full URI.

The previously used query tells us the answer which is /jollion/apines.exe:



id.orig_h	id.resp_p	id.resp_h	host	uri
192.168.75.146	80	5.181.156.252	5.181.156.252	/
192.168.75.146	80	5.181.156.252	5.181.156.252	/
192.168.75.146	80	5.181.156.252	5.181.156.252	/
192.168.75.146	80	45.95.203.28	hypercustom.top	/jollion/apines.exe

Provide the IP address of the domain that hosts the binary.

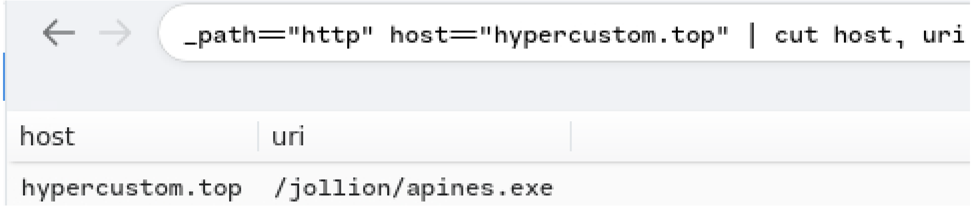
Luckily for us, the query used 2 questions ago also tells us the answer for this question which is 45.95.203.28.

There were 2 Suricata “A Network Trojan was detected” alerts. What were the source and destination IP addresses?

We can use the built in ‘Suricata Alerts by Source and Destination’ to determine the source and destination IP addresses of the 2 Suricata network trojan alerts. The answer is 192.168.75.146,45.95.203.28.

Taking a look at .top domain in HTTP requests, provide the name of the stealer (Trojan that gathers information from a system) involved in this packet capture using URLhaus Database.

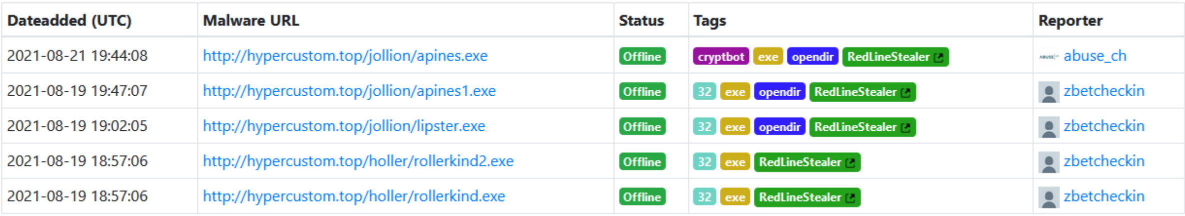
First we need to determine the domain that ends with ,top, To do this, you can analyse the HTTP logs and discover that the domain is hypercustom.top:



```
_path=="http" host=="hypercustom.top" | cut host, uri
```

host	uri
hypercustom.top	/jollion/apines.exe

Now simply navigate to the browse section of URLhause and enter the domain into the search field. You will quickly discover that the Trojan used is called RedLineStealer:

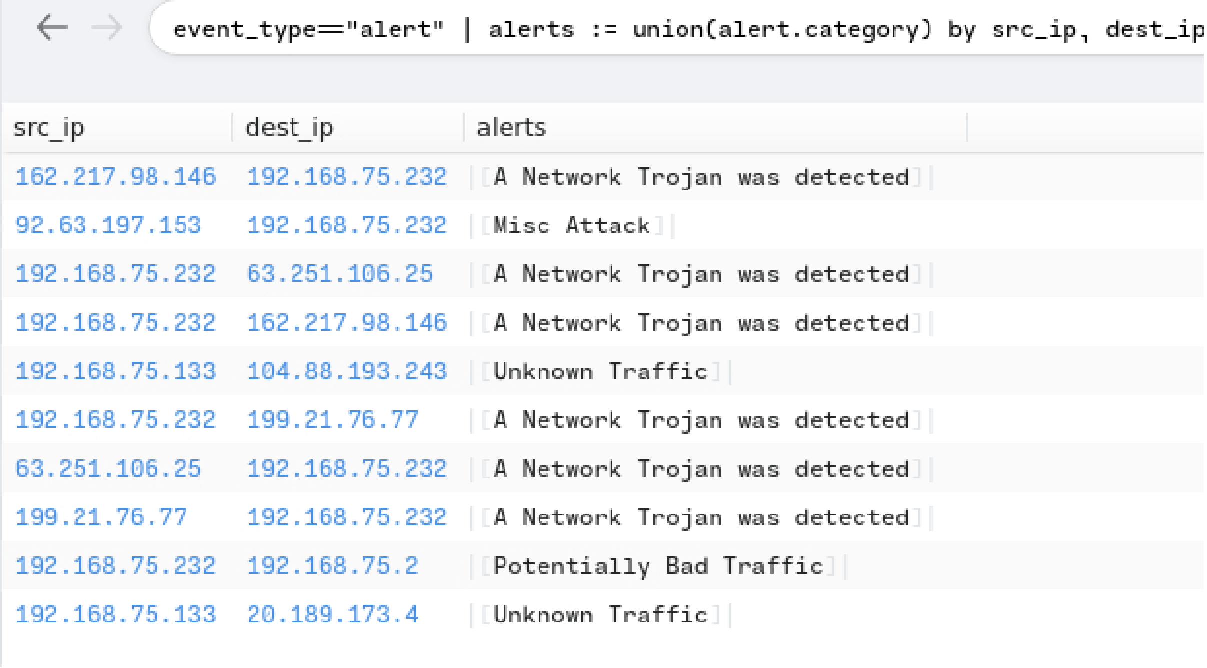


Dateadded (UTC)	Malware URL	Status	Tags	Reporter
2021-08-21 19:44:08	http://hypercustom.top/jollion/apines.exe	Offline	cryptbot exe opendir RedLineStealer	abuse_ch
2021-08-19 19:47:07	http://hypercustom.top/jollion/apines1.exe	Offline	32 exe opendir RedLineStealer	zbetcheckin
2021-08-19 19:02:05	http://hypercustom.top/jollion/lipster.exe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

Infection 3

Provide the IP address of the victim machine.

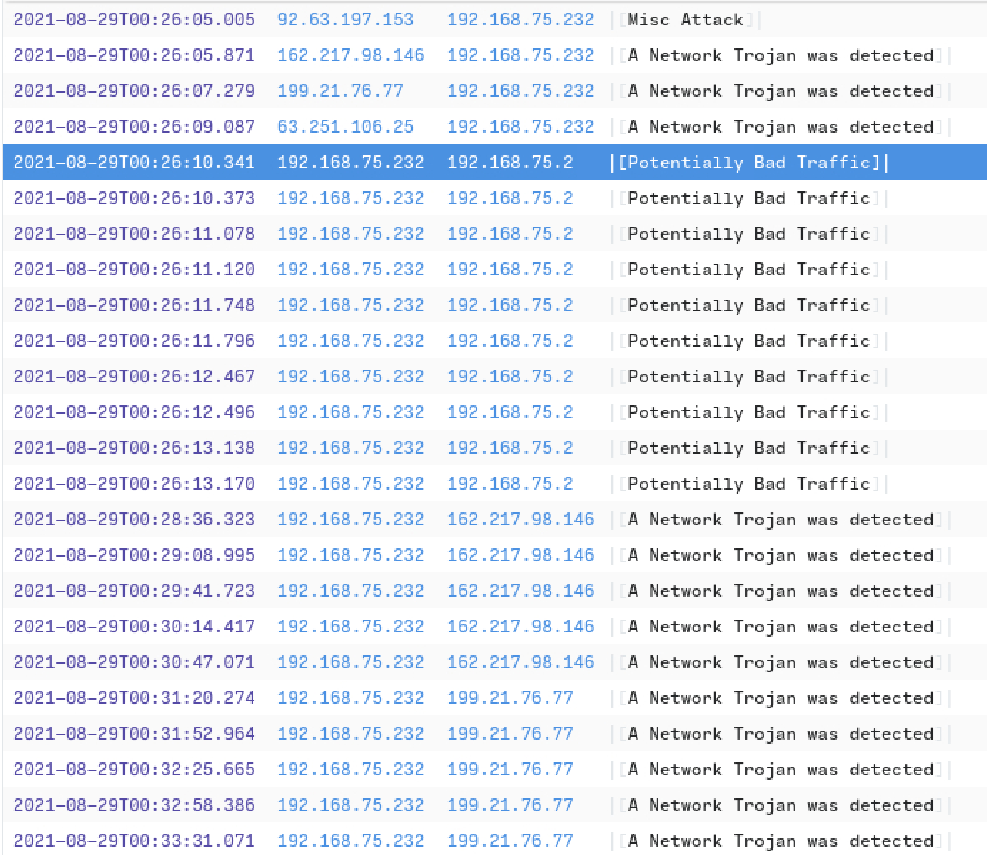
I started off by following the same methodology as the other infections, however, there are a lot of alerts:



```
event_type=="alert" | alerts := union(alert.category) by src_ip, dest_ip
```

src_ip	dest_ip	alerts
162.217.98.146	192.168.75.232	A Network Trojan was detected
92.63.197.153	192.168.75.232	Misc Attack
192.168.75.232	63.251.106.25	A Network Trojan was detected
192.168.75.232	162.217.98.146	A Network Trojan was detected
192.168.75.133	104.88.193.243	Unknown Traffic
192.168.75.232	199.21.76.77	A Network Trojan was detected
63.251.106.25	192.168.75.232	A Network Trojan was detected
199.21.76.77	192.168.75.232	A Network Trojan was detected
192.168.75.232	192.168.75.2	Potentially Bad Traffic
192.168.75.133	20.189.173.4	Unknown Traffic

However, if you filter based off of the timestamp and check the first alerts that were produced for possibly malicious behaviour, you can see that the IP address 192.168.75.232 has produced a significant amount of alerts and therefore is likely to be the victim machine:



2021-08-29T00:26:05.005	92.63.197.153	192.168.75.232	Misc Attack
2021-08-29T00:26:05.871	162.217.98.146	192.168.75.232	A Network Trojan was detected
2021-08-29T00:26:07.279	199.21.76.77	192.168.75.232	A Network Trojan was detected
2021-08-29T00:26:09.087	63.251.106.25	192.168.75.232	A Network Trojan was detected
2021-08-29T00:26:10.341	192.168.75.232	192.168.75.2	[Potentially Bad Traffic]
2021-08-29T00:26:10.373	192.168.75.232	192.168.75.2	Potentially Bad Traffic
2021-08-29T00:26:11.078	192.168.75.232	192.168.75.2	Potentially Bad Traffic
2021-08-29T00:26:11.120	192.168.75.232	192.168.75.2	Potentially Bad Traffic
2021-08-29T00:26:11.748	192.168.75.232	192.168.75.2	Potentially Bad Traffic
2021-08-29T00:26:11.796	192.168.75.232	192.168.75.2	Potentially Bad Traffic
2021-08-29T00:26:12.467	192.168.75.232	192.168.75.2	Potentially Bad Traffic
2021-08-29T00:26:12.496	192.168.75.232	192.168.75.2	Potentially Bad Traffic
2021-08-29T00:26:13.138	192.168.75.232	192.168.75.2	Potentially Bad Traffic
2021-08-29T00:26:13.170	192.168.75.232	192.168.75.2	Potentially Bad Traffic
2021-08-29T00:28:36.323	192.168.75.232	162.217.98.146	A Network Trojan was detected
2021-08-29T00:29:08.995	192.168.75.232	162.217.98.146	A Network Trojan was detected
2021-08-29T00:29:41.723	192.168.75.232	162.217.98.146	A Network Trojan was detected
2021-08-29T00:30:14.417	192.168.75.232	162.217.98.146	A Network Trojan was detected
2021-08-29T00:30:47.071	192.168.75.232	162.217.98.146	A Network Trojan was detected
2021-08-29T00:31:20.274	192.168.75.232	199.21.76.77	A Network Trojan was detected
2021-08-29T00:31:52.964	192.168.75.232	199.21.76.77	A Network Trojan was detected
2021-08-29T00:32:25.665	192.168.75.232	199.21.76.77	A Network Trojan was detected
2021-08-29T00:32:58.386	192.168.75.232	199.21.76.77	A Network Trojan was detected
2021-08-29T00:33:31.071	192.168.75.232	199.21.76.77	A Network Trojan was detected

Query used:

- event_type=="alert" | alerts := union(alert.category) by src_ip, dest_ip, ts | sort ts

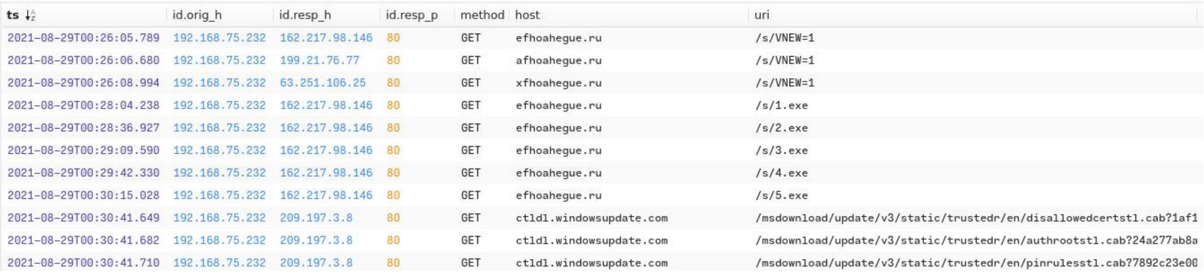
Provide three C2 domains from which the binaries were downloaded (starting from the earliest to the latest in the timestamp)

To answer this, I am going to create a query that looks filters the HTTP log file for the source IP, destination IP and port, http method, domain, and URI:

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

From this, we can sort from the earliest to the latest timestamp by entering:

- _path=="http" | cut ts, id.orig_h, id.resp_h, id.resp_p, method, host, uri | sort ts



ts	id.orig_h	id.resp_h	id.resp_p	method	host	uri
2021-08-29T00:26:05.789	192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/VNEW=1
2021-08-29T00:26:06.680	192.168.75.232	199.21.76.77	80	GET	afhoahegue.ru	/s/VNEW=1
2021-08-29T00:26:08.994	192.168.75.232	63.251.106.25	80	GET	xfhoahegue.ru	/s/VNEW=1
2021-08-29T00:28:04.238	192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/1.exe
2021-08-29T00:28:36.927	192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/2.exe
2021-08-29T00:29:09.590	192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/3.exe
2021-08-29T00:29:42.330	192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/4.exe
2021-08-29T00:30:15.028	192.168.75.232	162.217.98.146	80	GET	efhoahegue.ru	/s/5.exe
2021-08-29T00:30:41.649	192.168.75.232	209.197.3.8	80	GET	ctldl.windowsupdate.com	/msdownload/update/v3/static/trustedr/en/disallowedcertstl.cab?1af1
2021-08-29T00:30:41.682	192.168.75.232	209.197.3.8	80	GET	ctldl.windowsupdate.com	/msdownload/update/v3/static/trustedr/en/authrootstl.cab?24a277ab8a
2021-08-29T00:30:41.710	192.168.75.232	209.197.3.8	80	GET	ctldl.windowsupdate.com	/msdownload/update/v3/static/trustedr/en/pinrulesstl.cab?7892c23e00

The three domains from which the binaries were downloaded are efhoahegue.ru, afhoahegue.ru, xfhoahegue.ru.

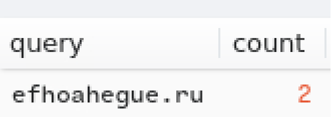
Provide the IP addresses for all three domains in the previous question.

The output from the queries entered above gives us the IP addresses for all three domains, which are 162.217.98.146, 199.21.76.77, and 63.251.106.25 respectively.

How many unique DNS queries were made to the domain associated from the first IP address from the previous answer?

We can use the premade 'Unique DNS Queries' query and slightly modify it to find the answer. The query used was:

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

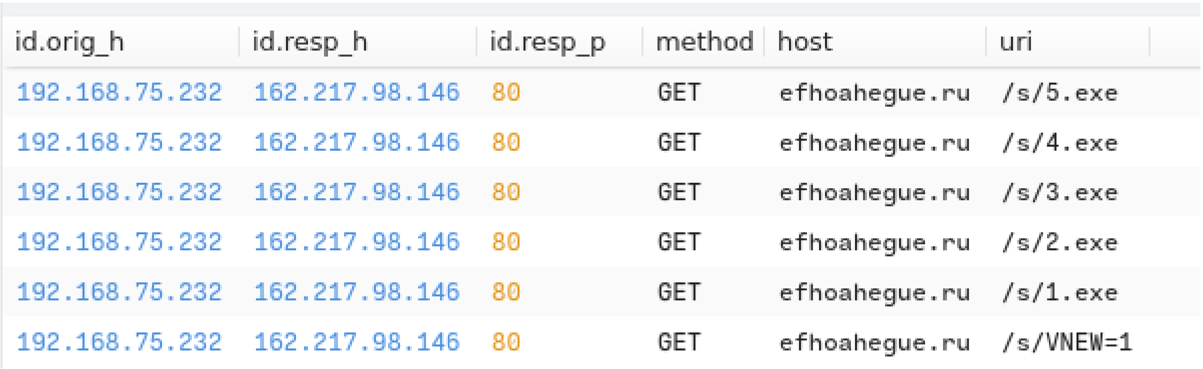


query	count
efhoahegue.ru	2

How many binaries were downloaded from the above domain in total?

To answer this question I simply used the following query which filters the HTTP log traffic to only display logs with the domain 'efhoahegue.ru". Query used is as follows:

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



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

The number of binaries downloaded was 5.

Provide the user-agent listed to download the binaries.

All we need to do is simply modify the above query to include the user_agent column:

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

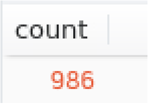
We then need to find the user agent of the C2 domain which is:

- Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/20100101 Firefox/25.0

Provide the amount of DNS connections made in total for this packet capture.

This is very simply to answer, we just need to query for the dns logs and used the count function:

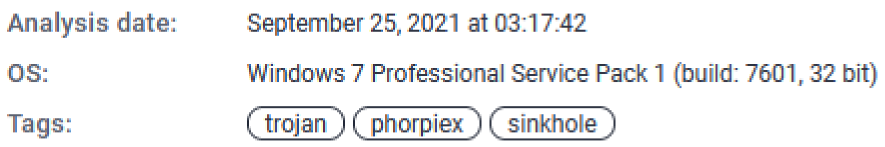
- _path=="dns" | count()



count
986

With some OSINT skills, provide the name of the worm using the first domain you have managed to collect from question 2.

I started by searching for the domain in quotations marks to ensure I don't visit a possibly active malicious domain. I then discovered an any run report which contained the domain and led me to discover that it is associated with the Phorphiex worm.



Analysis date:	September 25, 2021 at 03:17:42
OS:	Windows 7 Professional Service Pack 1 (build: 7601, 32 bit)
Tags:	trojan phorpiex sinkhole

Although it was spelt incorrectly so I had to search around and find the correct name of the worm.

Through detailed analysis and queries, we identified multiple indicators of compromise, suspicious domains, and malware involved in the three incidents. I really enjoyed this challenge as it helped me practice using the Brim tool. I hope this can be of use to someone else out there.