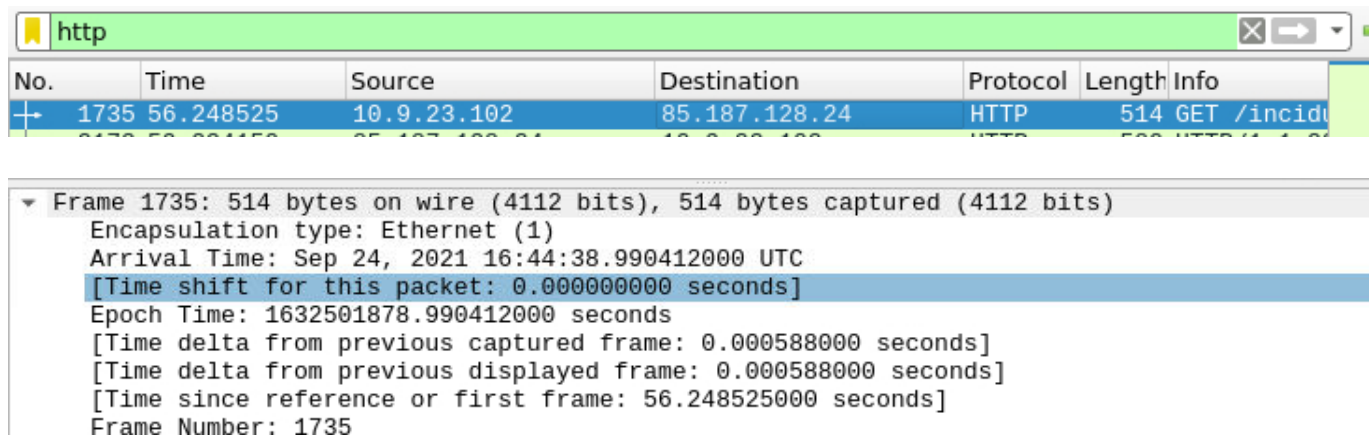


## Overview

Eric Fischer from the Purchasing Department at Bartell Ltd has received an email from a known contact with a Word document attachment. Upon opening the document, he accidentally clicked on "Enable Content." The SOC Department immediately received an alert from the endpoint agent that Eric's workstation was making suspicious connections outbound. The pcap was retrieved from the network sensor and handed to you for analysis.

Our first task is to establish when the first **HTTP** connection was seen, we use http in the filter bar to search for our first http connection then we can follow the frame drop down and we are able to see the arrival of the malicious IP made contact:

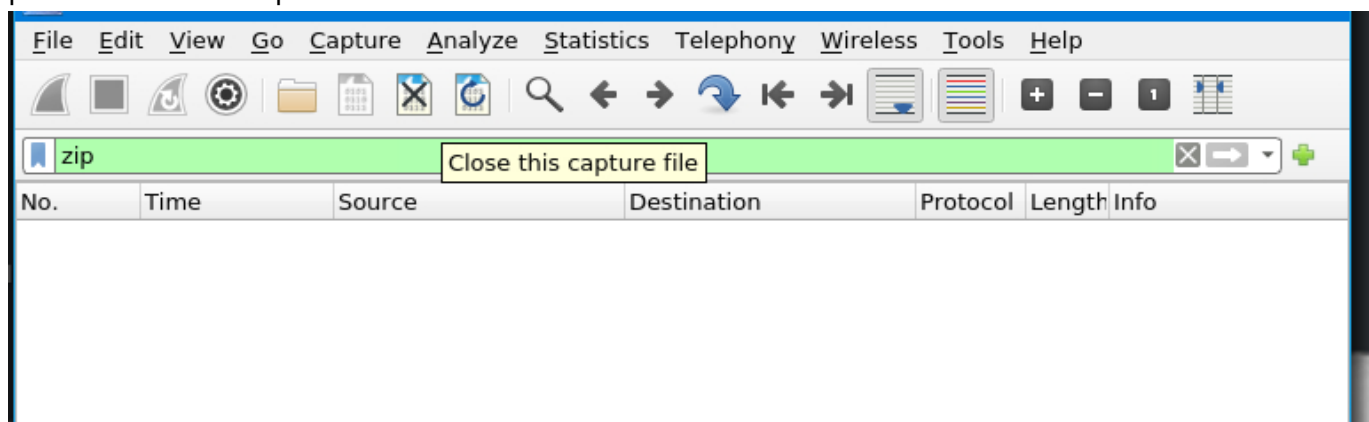


**What was the date and time for the first HTTP connection to the malicious IP?**

**(answer format: yyyy-mm-dd hh:mm:ss)**

**2021-09-24 16:44:38**

Our next task is to identify a zip file that was downloaded, we need to find out what the name of this zip file is, trying to type **zip** in the filter bar shows up green but there is no signs of any packets with the zip file:

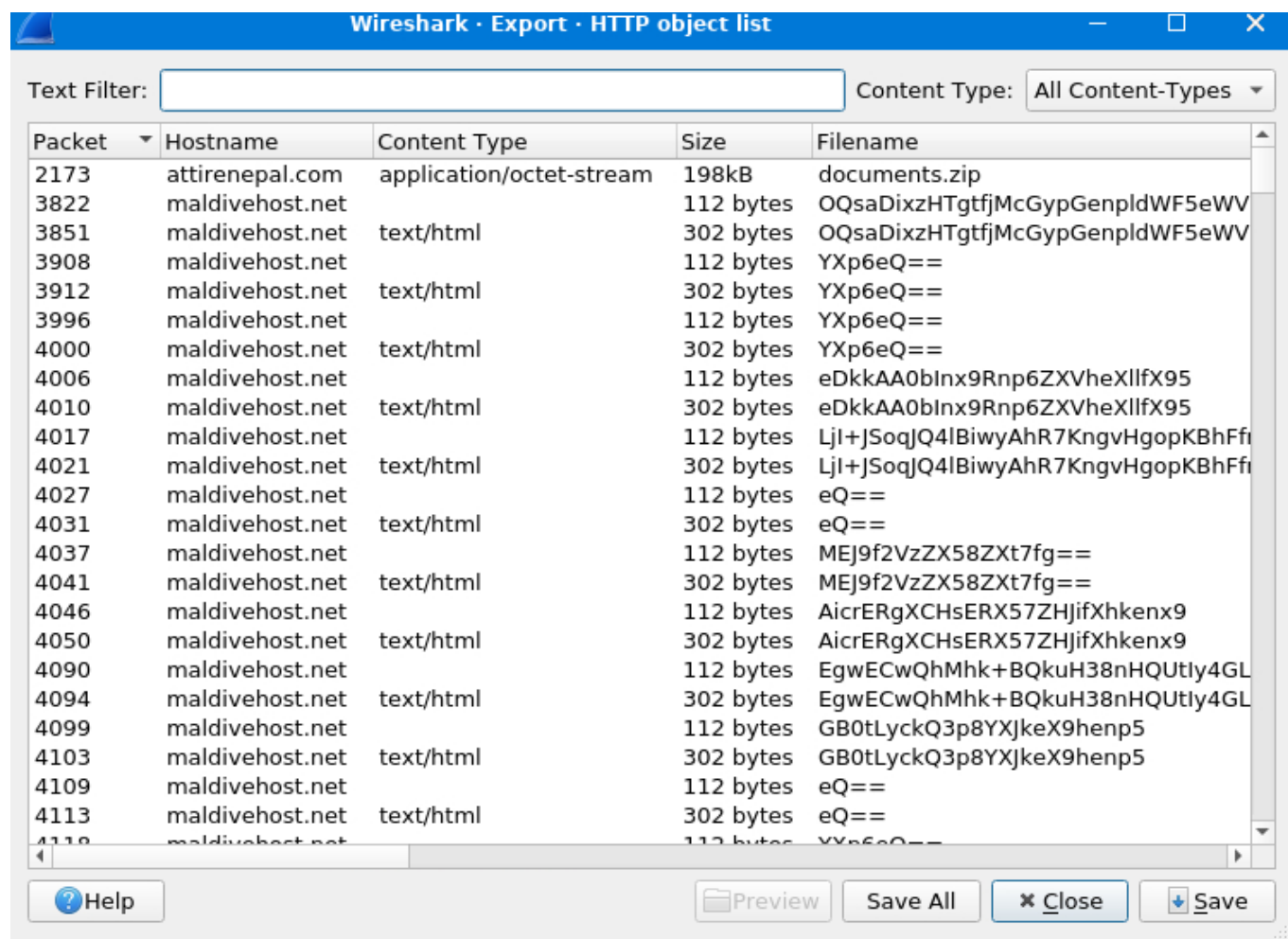


After searching through a few packets, I could not find any zip files in any of them, after googling **how to find a zip file in wireshark**, I found a link (<https://www.rubyguides.com/2012/01/four-ways-to-extract-files-from-pcaps/#:~:text=1,in%20all%20the%20http%20requests.>) that showed me four ways to extract

files from Pcaps. I tried the first solution and managed to find the zip file by going into File > Export > Objects > Http:

## 1. Wireshark: http export

You can find this at **File > Export > Objects > Http**, you will be presented with a list of files found in all the http requests. The bad thing about this feature is that even with the latest version (1.6.5 at the time of this writing) you still can't sort by column or apply any filters which makes finding something specific hard.



We have found the zip file that was downloaded, which is called **documents.zip**

**What is the name of the zip file that was downloaded?**  
**documents.zip**

Now our next step is to find the domain that was hosting this malicious zip file. We can see the packet is **2173**, so we find packet **2173** and we follow the HTTP Stream to find out the domain

that hosted this malicious zip file:

Wireshark packet capture showing an HTTP 200 OK response. The packet list shows frame 2173 (580 bytes) selected. The packet details pane shows the frame structure: Ethernet II, Internet Protocol Version 4, and Hypertext Transfer Protocol. The packet bytes pane shows the raw data. A context menu is open over the selected frame, with 'Follow' > 'HTTP Stream' highlighted.

We then see a **GET** Request, which is used for retrieving and requesting data from a specified server, the **GET** request allows us to see the domain of where the malicious zip file was hosted:

```
GET /incidunt-consequatur/documents.zip HTTP/1.1
Host: attirenepal.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/93.0.4577.82 Safari/537.36 Edg/93.0.961.52
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en
```

**What was the domain hosting the malicious zip file?**  
**attirenepal.com**

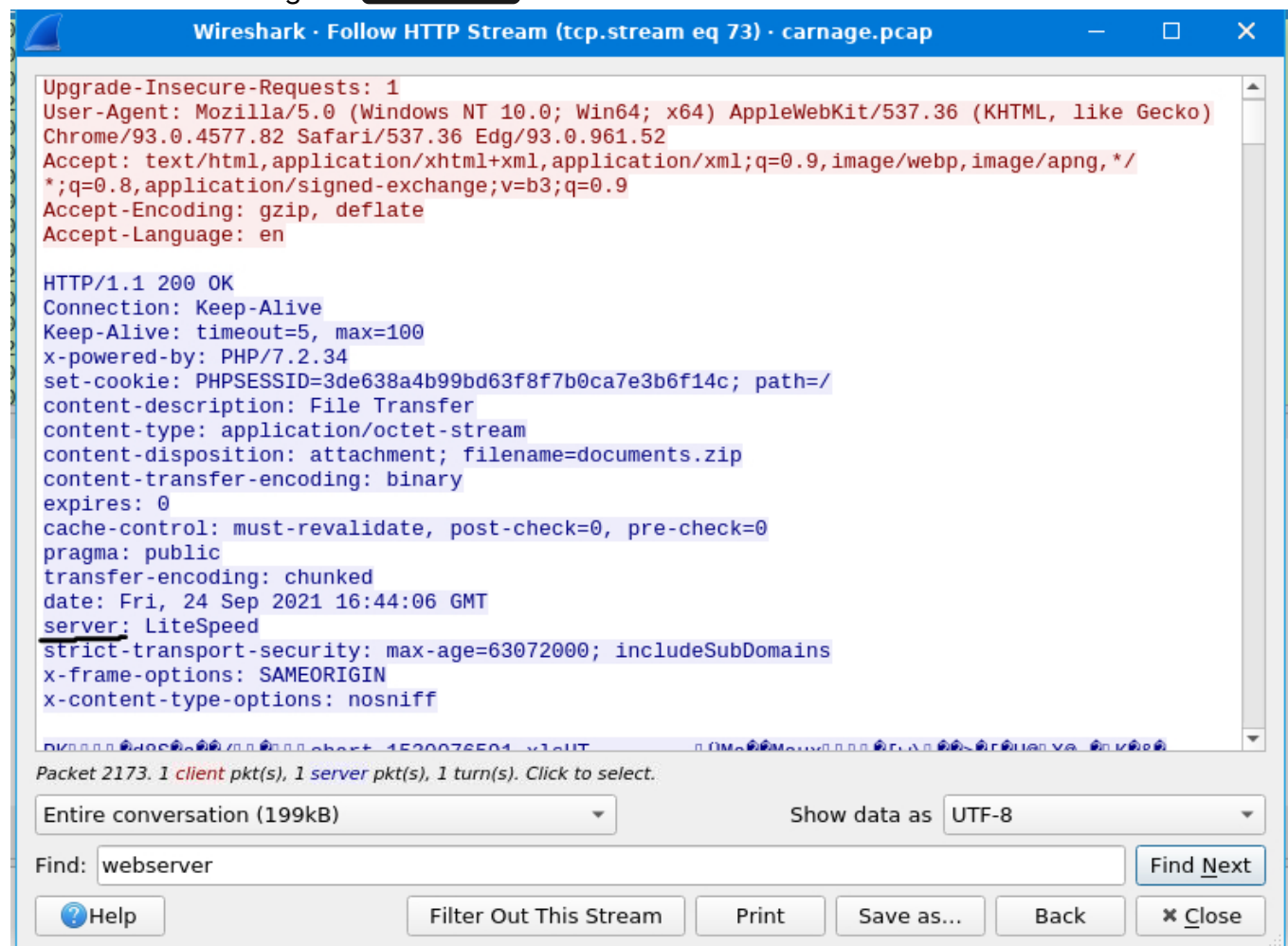
Now we have found the domain and the name of the zip file that was downloaded, let's see what the file is called without trying to download the file. By doing this we go back to packet **2173** and follow with another **HTTP Stream** and on the first line of encoding we are able to see a file called **chart-1530076591.xls**:

```
cache-control: must-revalidate, post-check=0, pre-check=0
pragma: public
transfer-encoding: chunked
date: Fri, 24 Sep 2021 16:44:06 GMT
server: LiteSpeed
strict-transport-security: max-age=63072000; includeSubDomains
x-frame-options: SAMEORIGIN
x-content-type-options: nosniff

PK.....d8S.a../.....chart-1530076591.xlsUT.....Ma..Maux.....
[w\...>...[.U@.X@,...K.&.
..5...c.4.4.g...X..H4..Q1.#...n.I...^.....sfY.....8.s.y..<}.sfgv..j.....
```

**Without downloading the file, what is the name of the file in the zip file?**  
**chart-1530076591.xls**

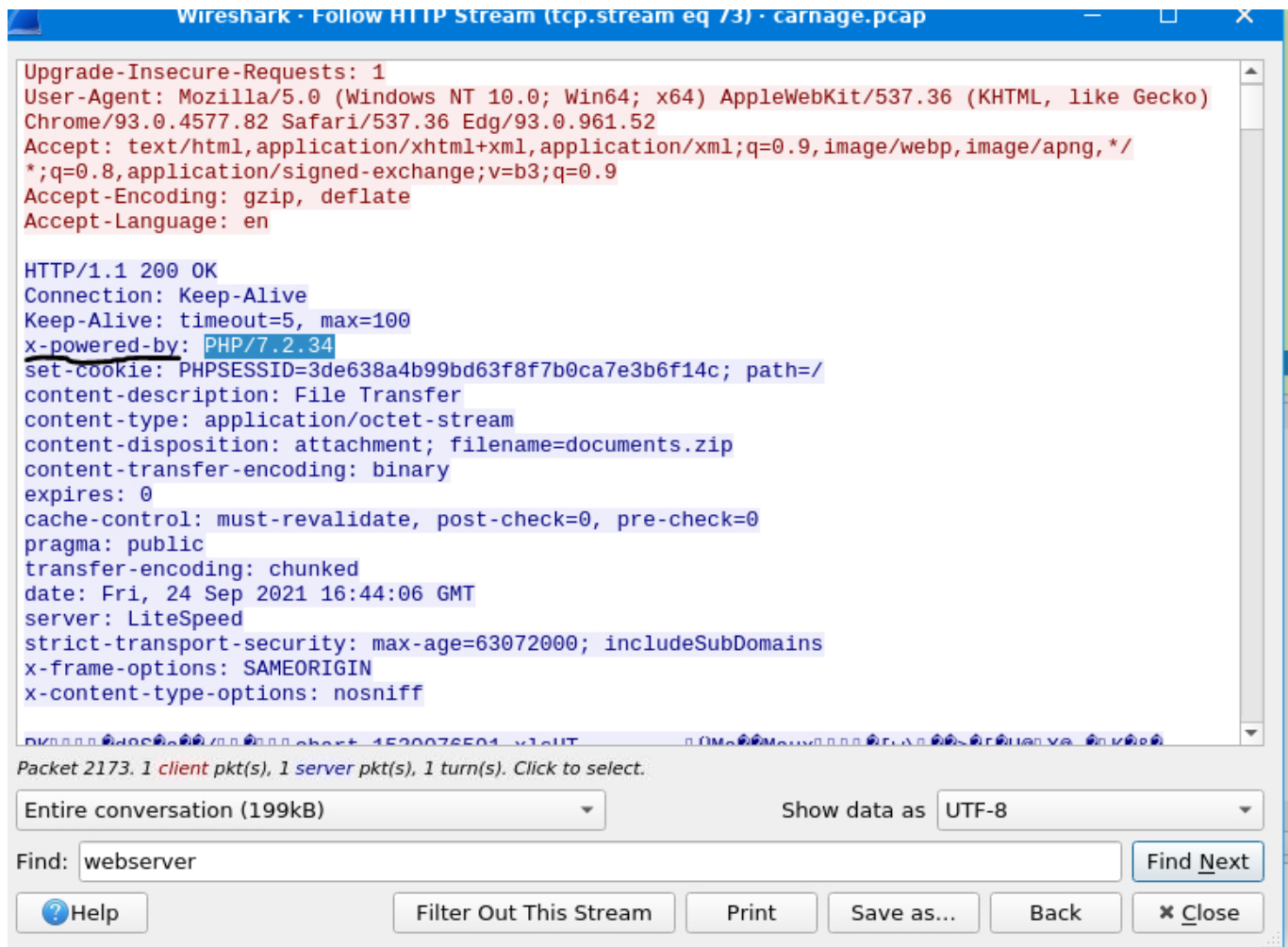
Now we need to identify the webserver, which is pretty easy to spot as it is under the server section when following the **HTTP Stream**:



**What is the name of the webserver of the malicious IP from which the zip file was downloaded?**

**LiteSpeed**

We have also been asked to identify the version of the server, which is also very easy to spot, it is under the **x-powered-by**:





**What is the version of the webserver from the previous question?**

**PHP/7.2.34**

We have now found out that the malicious files were downloaded from multiple domains, so let's track them down

After looking through so many packets, I was unable to find any of the other domains, luckily the room gives us a hint to find these:

 **Question Hint** 

Check HTTPS traffic. Narrow down the timeframe from 16:45:11 to 16:45:30.

As I was unaware on how to do this, I searched up [how to search through time frames in wireshark](https://securitronlinux.com/bejiitaswrath/filter-for-a-specific-time-frame-in-wireshark/) - I found this link (<https://securitronlinux.com/bejiitaswrath/filter-for-a-specific-time-frame-in-wireshark/>) that showed how to apply the filter in Wireshark

Going back to Wireshark we apply the filter `(frame.time >= "Sep 24, 2021 16:45:11") && (frame.time <= "Sep 24, 2021 16:45:30")` - This looks for packets between September 24 2021

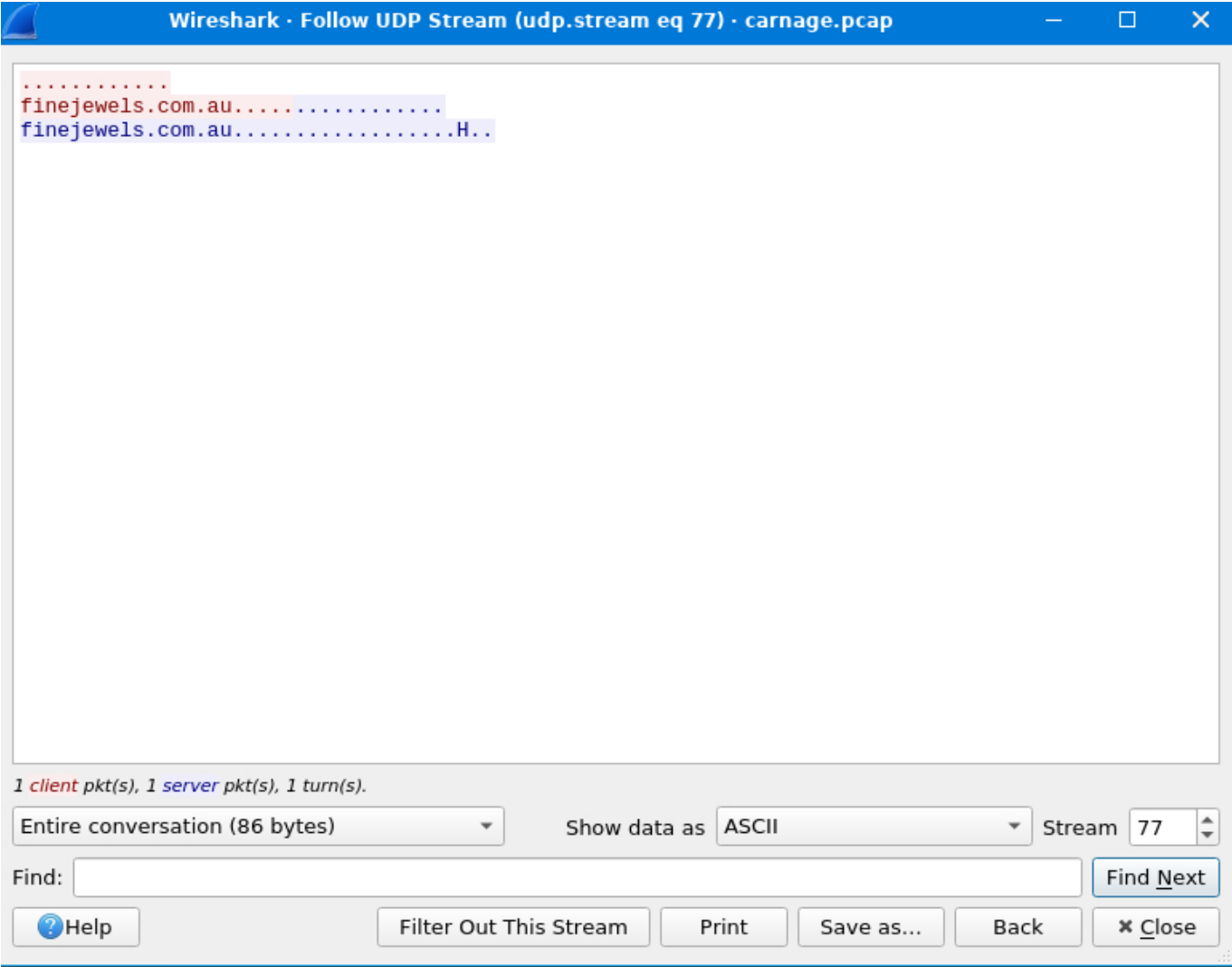


16:45:11 between a 19 second gap as the filter ends at September 24 2021 16:45:30:

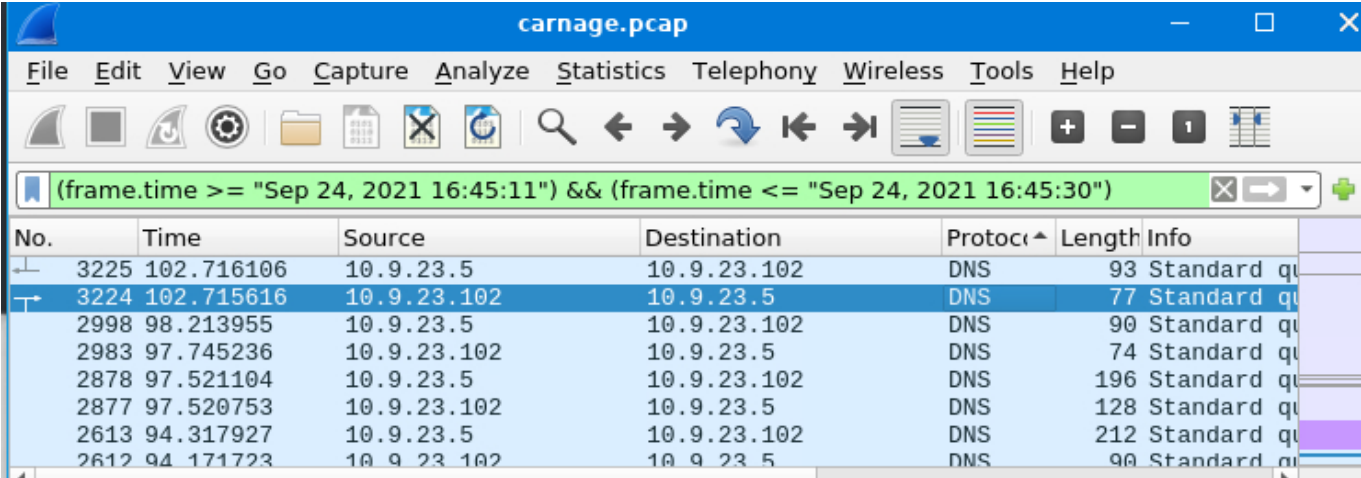
The image shows a Wireshark window titled "carnage.pcap". The filter bar contains the expression: `(frame.time >= "Sep 24, 2021 16:45:11") && (frame.time <= "Sep 24, 2021 16:45:30")`. The packet list shows several entries, with frame 2433 selected. The details pane for frame 2433 shows the following information:

- Frame 2433: 1414 bytes on wire (11312 bits), 1414 bytes captured (11312 bits)
- Ethernet II, Src: Netgear\_b6:93:f1 (20:e5:2a:b6:93:f1), Dst: HewlettP\_1c:47:ae (00:08:02:1c:47:ae)
- Internet Protocol Version 4, Src: 148.72.192.206, Dst: 10.9.23.102
- Transmission Control Protocol, Src Port: 443, Dst Port: 63368, Seq: 1, Ack: 194, Len: 1360
- Transport Layer Security

And by following the UDP stream we successfully find out the first domain out of the three that was involved with the malicious activity:



As we are looking for domains I filtered the packets by protocol to find **DNS** and this tidied up all the packets to keep all the DNS packets in one place:



Here I was able to find the other two domains:  
new.americold.com:

Wireshark · Packet 3224 · carnage.pcap

Authority RRs: 0  
Additional RRs: 0  
Queries  
    new.americold.com: type A, class IN  
        Name: new.americold.com  
        [Name Length: 17]  
        [Label Count: 3]  
        Type: A (Host Address) (1)  
        Class: IN (0x0001)  
        [Response In: 3225]

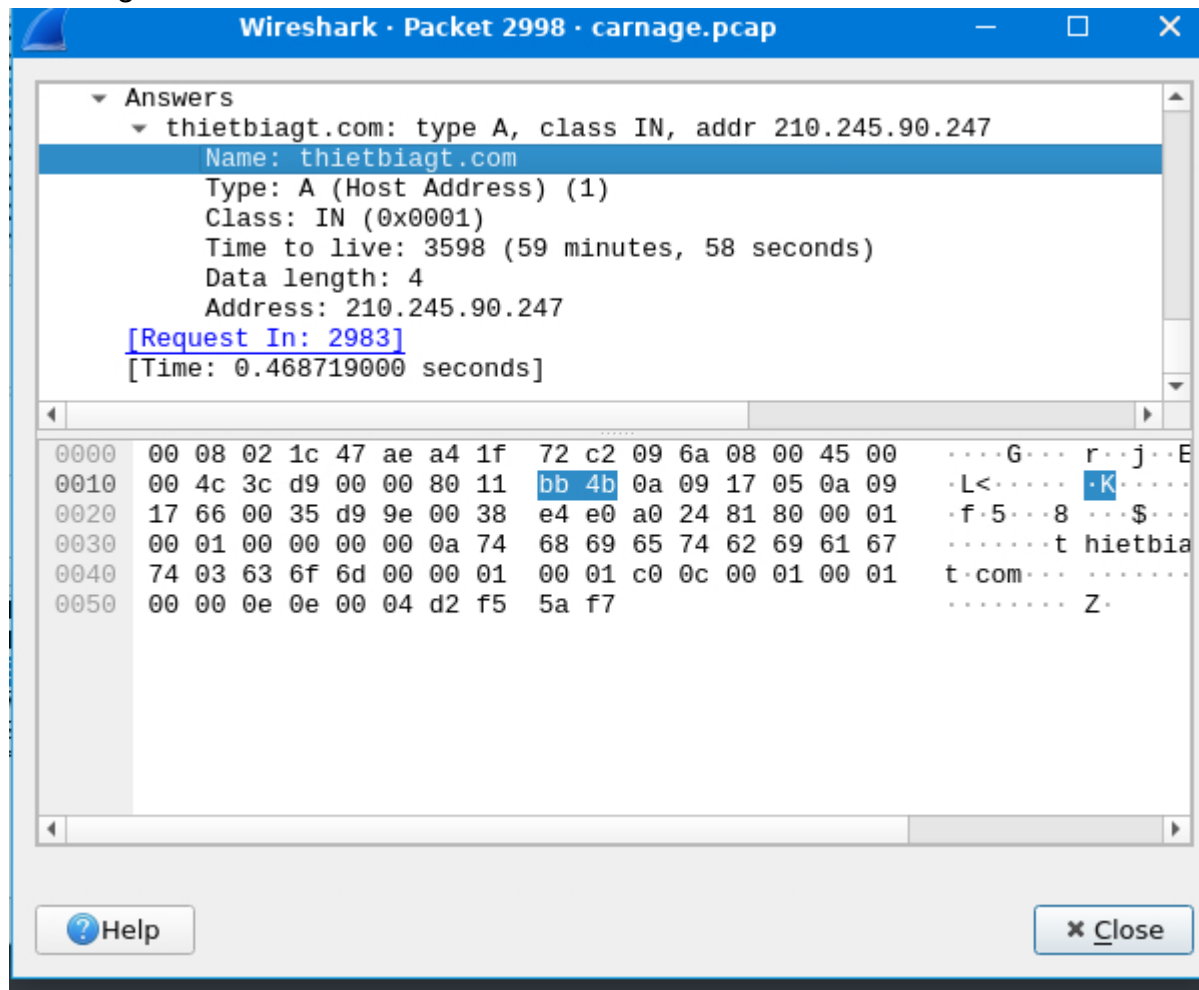
0000	a4 1f 72 c2 09 6a 00 08 02 1c 47 ae 08 00 45 00	..r..j.. ..G...E
0010	00 3f bf 3d 00 00 80 11 38 f4 0a 09 17 66 0a 09	·?·=... 8....f·
0020	17 05 e6 d6 00 35 00 2b d8 a0 65 b4 01 00 00 01	.....5+ ..e....
0030	00 00 00 00 00 00 03 6e 65 77 09 61 6d 65 72 69	.....·n ew·amer
0040	63 6f 6c 64 03 63 6f 6d 00 00 01 00 01	cold·com ·....

Help

Close



thietbiagt.com:



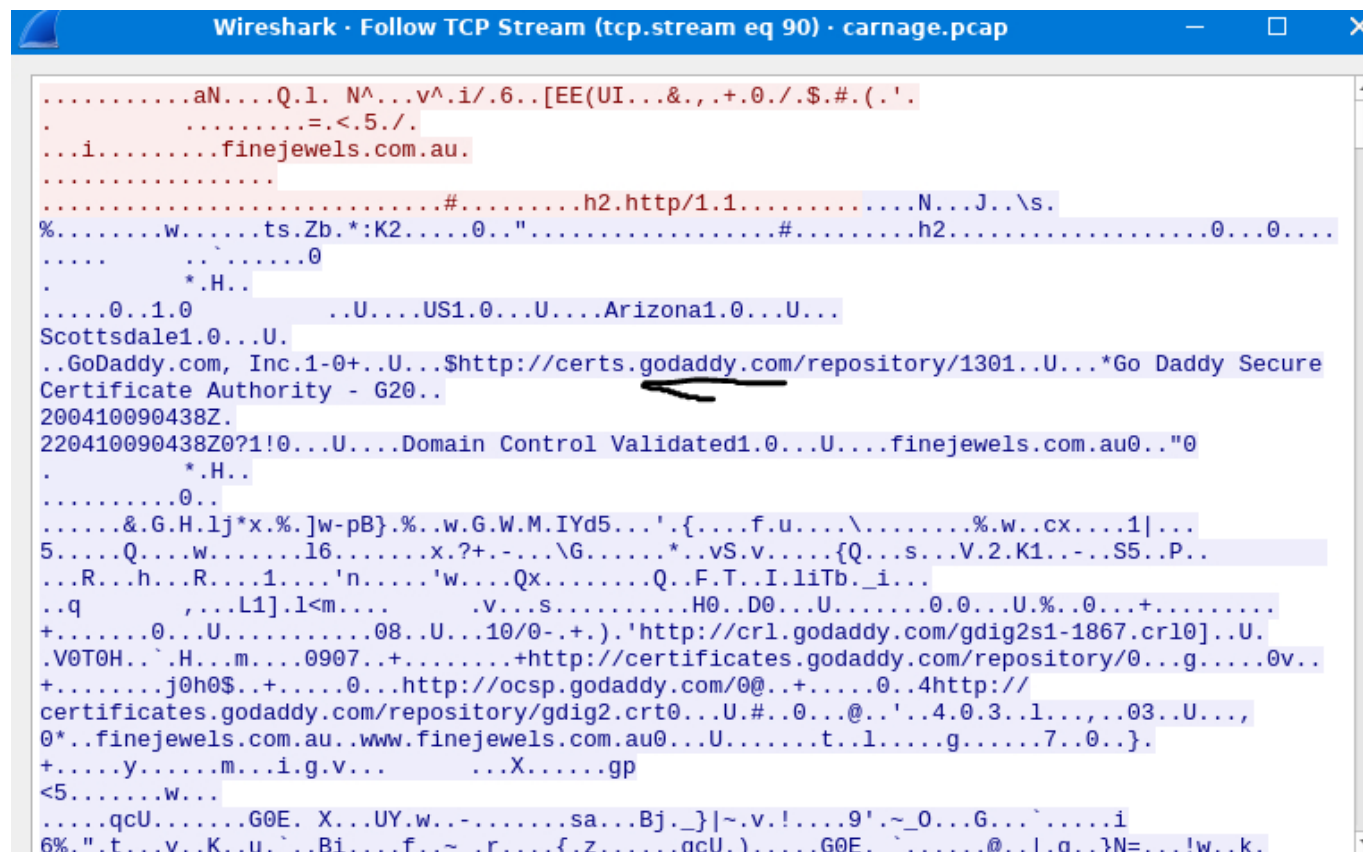
We have successfully captured all 3 domains that were a part of this malicious activity

**Malicious files were downloaded to the victim host from multiple domains. What were the three domains involved with this activity?**

**`finejewels.com.au`, `thietbiagt.com`, `new.americold.com`**

Now we have been asked to identify what CA (Certificate Authority) was issued by the SSL certificate in the first domain - I followed the TCP stream of the first TCP packet within the time

frame filter we used and found out the CA:

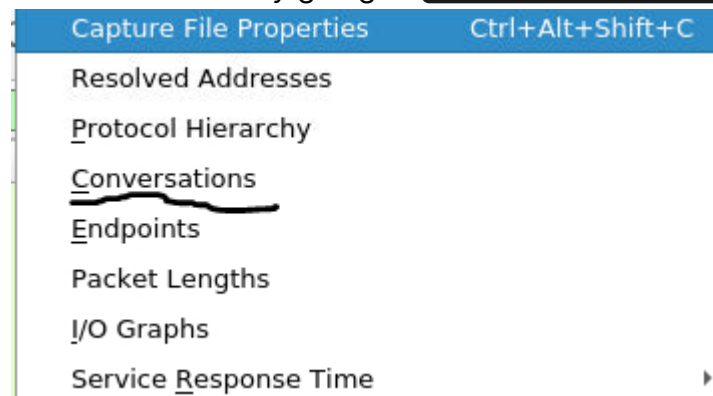


**Which certificate authority issued the SSL certificate to the first domain from the previous question?**

**Godaddy**

Our next task is to find two IP addresses of the cobalt strike servers - We filter all packets to HTTP as we are trying to find the Cobalt Strike C2 servers, however I did not find anything by doing this

Then I learned that we can see multiple IPs due to different conversations being made from different servers by going to **statistics > conversations**:



Opening this up shows us mutple conversations that have been made and we also want to view IPv4 to find the ip addresses:

Wireshark · Conversations · carnage.pcap

Ethernet · 8		IPv4 · 109	IPv6	TCP · 447	UDP · 256														
Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A								
0.0.0.0	255.255.255.255	1	381	1	381	0	0	0.000000	0.0000	—	—								
2.20.156.175	10.9.23.102	50	20k	30	19k	20	1918	762.051358	25.7653	5909	595								
2.20.205.205	10.9.23.102	28	11k	14	6871	14	4514	81.165306	511.1902	107	70								
10.9.23.5	255.255.255.255	1	352	1	352	0	0	0.000681	0.0000	—	—								
10.9.23.5	10.9.23.102	3580	1921k	2081	1577k	1499	344k	0.414394	1267.5760	9953	2174								
10.9.23.102	224.0.0.22	7	386	7	386	0	0	0.023263	38.3465	80	0								
10.9.23.102	224.0.0.251	10	793	10	793	0	0	0.053191	62.7316	101	0								
10.9.23.102	224.0.0.252	7	489	7	489	0	0	0.053559	62.1399	62	0								
10.9.23.102	10.9.23.102	37	4817	37	4817	0	0	1.913013	882.5138	43	0								
10.9.23.102	52.184.217.56	24	5140	11	1282	13	3858	4.800113	1.7329	5918	17k								
10.9.23.102	104.120.111.248	72	26k	33	4126	39	22k	8.541834	719.7008	45	254								
10.9.23.102	13.69.116.104	68	26k	33	11k	35	15k	12.998563	430.7682	218	283								
10.9.23.102	13.89.178.26	25	8001	12	2484	13	5517	38.402132	3.9948	4974	11k								
10.9.23.102	20.54.36.229	64	17k	29	6182	35	11k	38.419408	1207.8286	40	77								
10.9.23.102	131.253.33.203	56	26k	24	3384	32	23k	39.686472	131.0282	206	1432								
10.9.23.102	152.199.19.161	92	27k	45	3812	47	24k	42.234702	526.9979	57	365								
10.9.23.102	131.253.33.200	84	29k	41	12k	43	16k	42.368691	681.0645	148	199								
10.9.23.102	13.107.255.222	23	6746	11	1087	12	5659	42.488711	123.9909	70	365								
10.9.23.102	52.113.196.254	24	6654	12	1140	12	5514	42.559038	126.7598	71	347								
10.9.23.102	13.107.6.254	23	6741	11	1082	12	5659	42.571797	122.1655	70	370								
10.9.23.102	51.141.33.202	23	6131	10	1064	13	5067	42.584735	128.1036	66	316								
10.9.23.102	13.107.42.254	25	6849	12	1136	13	5713	42.608679	125.9566	72	362								
10.9.23.102	13.107.3.254	22	6687	10	1028	12	5659	42.609339	123.8703	66	365								
10.9.23.102	13.107.4.254	24	6526	12	1136	12	5390	42.771681	124.1180	73	347								
10.9.23.102	13.107.136.254	23	6598	11	1084	12	5514	42.811100	123.2329	70	357								
10.9.23.102	13.107.246.254	22	6687	10	1028	12	5659	42.833031	123.7267	66	365								
10.9.23.102	131.253.33.254	22	6696	10	1037	12	5659	42.923436	126.4417	65	358								
10.9.23.102	13.107.246.60	23	6528	11	1085	12	5443	42.955903	126.5642	68	344								
10.9.23.102	13.107.213.60	23	6536	11	1093	12	5443	42.996467	127.1875	68	342								
10.9.23.102	20.82.217.86	34	13k	14	2371	20	11k	43.236483	109.9625	172	804								
10.9.23.102	20.190.159.135	69	49k	27	15k	42	33k	43.295319	109.9827	1145	2422								
10.9.23.102	20.54.232.160	28	12k	12	3968	16	8414	44.532811	110.0136	288	611								
10.9.23.102	23.220.255.44	26	6654	13	1168	13	5486	46.637191	522.5956	17	83								
10.0.0.0	255.255.255.255	1	381	1	381	0	0	56.562535	3.082.3164	70	—								

☐ Name resolution
 ☐ Limit to display filter
 ☐ Absolute start time

Conversation Types ▾
 Copy ▾ Follow Stream... Graph... Close

As there is 109 differnt IPs, i decide to check the box at the bottom to limit to display filter, whihc this narrows it down to 5 IP addresses, after checking the top three IPs in virustotal they were not a part of the Cobalt Strike srver:

Wireshark · Conversations · carnage.pcap

Ethernet · 1		IPv4 · 5	IPv6	TCP · 121	UDP														
Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A								
10.9.23.102	85.187.128.24	2	1094	1	514	1	580	56.248525	2.9856	1377	1554								
10.9.23.102	208.91.128.6	52	24k	26	7254	26	17k	153.653113	628.7745	92	218								
10.9.23.102	104.93.124.33	4	806	2	338	2	468	582.206300	1.6207	1668	2310								
10.9.23.102	185.125.204.174	184	254k	0	0	184	254k	584.576793	212.0515	0	9602								
10.9.23.102	185.106.96.158	152	71k	76	43k	76	27k	685.851675	433.7395	801	509								

☐ Name resolution
 ☒ Limit to display filter
 ☐ Absolute start time

Conversation Types ▾
 Copy ▾ Follow Stream... Graph... Close

With the ip address **185.106.96.158** being put into VirusTotal and heading over to th community tab we can see it is a part of the C2 server:

5

/ 90

?

Community Score

5 security vendors flagged this IP address as malicious

185.106.96.158 (185.106.96.0/22)

AS 35913 ( DEDIPATH-LLC )

US

DETECTION

DETAILS

RELATIONS

COMMUNITY 4

Comments

parthmaniar

3 months ago

This IP carried out Apache Log4j RCE attempt(s) (also known as CVE-2021-44228 or Log4Shell). For more information, or to report interesting/incorrect findings, give me a shoutout on @parthmaniar on Twitter.

drb\_ra

6 months ago

Cobalt Strike Server Found  
C2: HTTPS @ 185[.]106[.]96[.]158:8888  
C2 Server: survmeter[.]live[.]gscpl[.]R/185[.]106[.]96[.]158[.]gscpl[.]R/  
POST URI: /supprq/sa/  
Country: United States  
ASN: DediPath  
Host Header: ocsp[.]verisign[.]com  
#c2 #cobaltstrike

The second IP address **185.125.204.174**:

185.125.204.174

Did you intend to search across the file corpus instead? Click here

5

/ 90

?

Community Score

5 security vendors flagged this IP address as malicious

185.125.204.174 (185.125.204.0/22)

AS 25369 ( Hydra Communications Ltd )

GB

DETECTION

DETAILS

RELATIONS

COMMUNITY 6

Contained In Graphs

3xploit

Untitled graphdfasdf

2021-12-02 14:22:45

Comments

parthmaniar

3 months ago

This IP carried out Apache Log4j RCE attempt(s) (also known as CVE-2021-44228 or Log4Shell). For more information, or to report interesting/incorrect findings, give me a shoutout on @parthmaniar on Twitter.

drb\_ra

6 months ago

Cobalt Strike Server Found  
C2: HTTPS @ 185[.]125[.]204[.]174:4444  
C2 Server: securitybusinesspu[.]com[.]query-3[.]3[.]1[.]min[.]js,185[.]125[.]204[.]174[.]query-3[.]3[.]1[.]min[.]js  
POST URI: /query-3[.]3[.]2[.]min[.]js  
Country: N/A  
ASN: Hydra Communications Ltd  
#c2 #cobaltstrike

**What are the two IP addresses of the Cobalt Strike servers? Use VirusTotal (the Community tab) to confirm if IPs are identified as Cobalt Strike C2 servers. (answer format: enter the IP addresses in sequential order)**

**185.106.96.158, 185.125.204.174**

We are then asked to identify the **Host-Header** of the first IP address which can also be seen in the community tab of VirusTotal:



drb\_ra

6 months ago

Cobalt Strike Server Found  
C2: HTTPS @ 185[.]106[.]96[.]158:443  
C2 Server: survmeter[.]live,/gscp[.]R/,185[.]106[.]96[.]158,/gscp[.]R/  
POST URI: /supprq/sa/  
Country: United States  
ASN: DediPath  
Host Header: ocsip[.]verisign[.]com

---

**What is the Host header for the first Cobalt Strike IP address from the previous question?**

**ocsip.verisign.com**

After we need to identify the domain of the first IP address which we can see under **C2 Server**:



drb\_ra

6 months ago

Cobalt Strike Server Found  
C2: HTTPS @ 185[.]106[.]96[.]158:443  
C2 Server: survmeter[.]live,/gscp[.]R/,185[.]106[.]96[.]158,/gscp[.]R/  
POST URI: /supprq/sa/  
Country: United States  
ASN: DediPath  
Host Header: ocsip[.]verisign[.]com

#c2 #cobaltstrike

**What is the domain name for the first IP address of the Cobalt Strike server? You may use VirusTotal to confirm if it's the Cobalt Strike server (check the Community tab).**

**survmeter.live**

After checking a few of the First IP address details we then check the second IP address domain name:



drb\_ra

6 months ago

Cobalt Strike Server Found  
C2: HTTPS @ 185[.]125[.]204[.]174:4444  
C2 Server: securitybusinpuiff[.]com,/jquery-3[.]3[.]1[.]min[.]js,185[.]125[.]204[.]174,/jquery-3[.]3[.]1[.]min[.]js  
POST URI: /jquery-3[.]3[.]2[.]min[.]js  
Country: N/A  
ASN: Hydra Communications Ltd

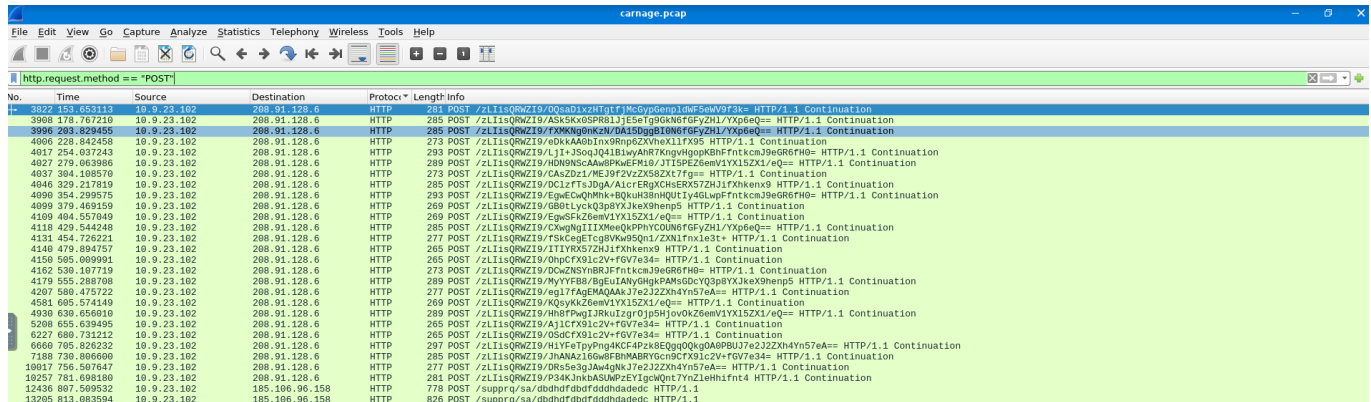
#c2 #cobaltstrike

**What is the domain name of the second Cobalt Strike server IP? You may use VirusTotal to confirm if it's the Cobalt Strike server (check the Community tab).**

**securitybusinpuuff.com**

As we have got all the details from VirusTotal, it is now time to go back to Wireshark, as we now need to find the domain name of the post infection traffic

On stackoverflow I managed to find how to filter **POST** traffic and to filter post traffic we use the following filter **http.request.method == "POST"**:



No.	Time	Source	Destination	Protocol	Length	Info
1002	159.053110	10.0.2.15	208.91.128.6	HTTP	201	POST /zLIisQRWZi9/OQsaDixzHTgtfjMcGypGenpldWF5eWV9f3k= HTTP/1.1 Continuation
3908	178.767210	10.0.2.15	208.91.128.6	HTTP	285	POST /zLIisQRWZi9/ASxxkxds9p41fCoTpGdxN6fGfY2H1/YXp6eQ= HTTP/1.1 Continuation
3908	283.829455	10.0.2.15	208.91.128.6	HTTP	285	POST /zLIisQRWZi9/fXMKngmKzN/DA15DggBION6fGfY2H1/YXp6eQ= HTTP/1.1 Continuation
4006	228.842458	10.0.2.15	208.91.128.6	HTTP	273	POST /zLIisQRWZi9/e0kxAA0BInx9Rnp6ZVXheX1lFX95 HTTP/1.1 Continuation
4017	254.037243	10.0.2.15	208.91.128.6	HTTP	293	POST /zLIisQRWZi9/v1i+35oqJQ4IBuyAHRTKngvvgpR8fFntkcm39eGR6FH= HTTP/1.1 Continuation
4027	279.063986	10.0.2.15	208.91.128.6	HTTP	289	POST /zLIisQRWZi9/H0N9NScAAw6PkeFM18/3T15PEZ6emVYX15ZXL/eQ= HTTP/1.1 Continuation
4037	384.108570	10.0.2.15	208.91.128.6	HTTP	273	POST /zLIisQRWZi9/CaaZdz1/ME39fZvZx58Zkt7fg= HTTP/1.1 Continuation
4046	329.217919	10.0.2.15	208.91.128.6	HTTP	285	POST /zLIisQRWZi9/DCl1fTsBgAAlcrEGgKHEK5ZHL1fRhnkx9 HTTP/1.1 Continuation
4090	354.299575	10.0.2.15	208.91.128.6	HTTP	293	POST /zLIisQRWZi9/EgwECwQMh+BgkuH38nHQutY46LwpFntkcm39eGR6FH= HTTP/1.1 Continuation
4099	379.469159	10.0.2.15	208.91.128.6	HTTP	269	POST /zLIisQRWZi9/6B0tLykQ3p8YXJkeX9hnp5 HTTP/1.1 Continuation
4109	484.557949	10.0.2.15	208.91.128.6	HTTP	269	POST /zLIisQRWZi9/EgSPzZeeW1YX15ZXL/eQ= HTTP/1.1 Continuation
4118	429.544248	10.0.2.15	208.91.128.6	HTTP	285	POST /zLIisQRWZi9/CwXgWgIIXMeedKPHYCOUN6fGfY2H1/YXp6eQ= HTTP/1.1 Continuation
4131	454.726221	10.0.2.15	208.91.128.6	HTTP	277	POST /zLIisQRWZi9/TskCegTcg9Vw95Qn1/ZXN1fmxle3t+ HTTP/1.1 Continuation
4140	479.894757	10.0.2.15	208.91.128.6	HTTP	265	POST /zLIisQRWZi9/ITV8Y5ZHL1fRhnkx9 HTTP/1.1 Continuation
4150	505.069991	10.0.2.15	208.91.128.6	HTTP	265	POST /zLIisQRWZi9/OhpCFX91c2V+f0V7e34= HTTP/1.1 Continuation
4162	530.107719	10.0.2.15	208.91.128.6	HTTP	273	POST /zLIisQRWZi9/DcWZSYnBR3fntkcm39eGR6FH= HTTP/1.1 Continuation
4179	555.288708	10.0.2.15	208.91.128.6	HTTP	289	POST /zLIisQRWZi9/PyYf8B/BgJIAyGhGpAMsDcYQ3p8YXJkeX9hnp5 HTTP/1.1 Continuation
4207	580.475722	10.0.2.15	208.91.128.6	HTTP	277	POST /zLIisQRWZi9/eg1fAgEMQAakJ7e2J2ZxH4Yn57eA= HTTP/1.1 Continuation
4581	605.574149	10.0.2.15	208.91.128.6	HTTP	269	POST /zLIisQRWZi9/Q2sykZ6emVYX15ZXL/eQ= HTTP/1.1 Continuation
4938	630.050010	10.0.2.15	208.91.128.6	HTTP	289	POST /zLIisQRWZi9/Hn8fPwJ2Rnu1nrg0jphJv0vZ6emVYX15ZXL/eQ= HTTP/1.1 Continuation
5208	655.639495	10.0.2.15	208.91.128.6	HTTP	265	POST /zLIisQRWZi9/Aj1CFX91c2V+f0V7e34= HTTP/1.1 Continuation
6227	680.731212	10.0.2.15	208.91.128.6	HTTP	265	POST /zLIisQRWZi9/05dcFX91c2V+f0V7e34= HTTP/1.1 Continuation
6600	705.826232	10.0.2.15	208.91.128.6	HTTP	297	POST /zLIisQRWZi9/H1fYfPpHnCF4PzK8EGgQKQQA0RBUJ7e2J2ZxH4Yn57eA= HTTP/1.1 Continuation
7188	730.806600	10.0.2.15	208.91.128.6	HTTP	285	POST /zLIisQRWZi9/JhANaz16gw8FBMBABRYGcn8CFX91c2V+f0V7e34= HTTP/1.1 Continuation
10017	756.507647	10.0.2.15	208.91.128.6	HTTP	277	POST /zLIisQRWZi9/DrsSe3gJAwgNk37e2J2ZxH4Yn57eA= HTTP/1.1 Continuation
10257	781.698180	10.0.2.15	208.91.128.6	HTTP	281	POST /zLIisQRWZi9/P34k3hKASUWpEY1g0wQn7YmZ1eHm1fnt4 HTTP/1.1 Continuation
12436	887.509532	10.0.2.15	185.106.96.158	HTTP	778	POST /supprq/sa/dbdhdfdbdfddhddadedc HTTP/1.1
13208	813.083594	10.0.2.15	185.106.96.158	HTTP	826	POST /supprq/sa/dbdhdfdbdfddhddadedc HTTP/1.1

We then follow the TCP stream and find the domain name of the POST-infection traffic:



**Wireshark · Follow TCP Stream (tcp.stream eq 104) · carnage.pcap**

**POST /zLIisQRWZi9/OQsaDixzHTgtfjMcGypGenpldWF5eWV9f3k= HTTP/1.1**  
**Host: maldivehost.net**  
**Content-Length: 112**

**Dw8YBxsEGmYFAAEJfR4NQkMmLTYqZDK5KyQmOyRGQglxEB04Lzk/  
EyYrMi1h0T8vIyM7IhcNPzs0KJguFgxgLSIiJCxFRgwFAgIIDQUZGBoFD0JF**

**HTTP/1.1 200 OK**  
**Date: Fri, 24 Sep 2021 16:46:15 GMT**  
**Server: Apache/2.4.49 (cPanel) OpenSSL/1.1.11 mod\_bwlimited/1.4**  
**X-Powered-By: PHP/5.6.40**  
**Content-Length: 302**  
**Strict-Transport-Security: ...max-age=15552000...**  
**Connection: close**  
**Content-Type: text/html; charset=UTF-8**

**eXp7QUVCQ0FBfn15eX1/  
enN8ekJBQ0JGQnpzeWJ+eXtleH1lf3xBRUJDQUELdHkAGAAbZwIDBQh8GQ5GQ1cqNS510D4oICc6I0VGCHAXGTWu0DgQI  
iozKmI9Pi4kID8JFgo8Pw8rPy0TGSUqISY1LUJFCAQDBQsJBBgfGQEQ0QJBRUJDQUEBBAQ0QUVCQ0FBAQQEDKFFqkNBQQ  
EEBA5BRUJDQUFCQUNCRkJGQEJFRUZHQUVGQuDGRkZCQEZBRUJDQUFCQUNCRkJGQEJFRUZHQUVGQuD=**

**What is the domain name of the post-infection traffic?**

**maldivehost.net**

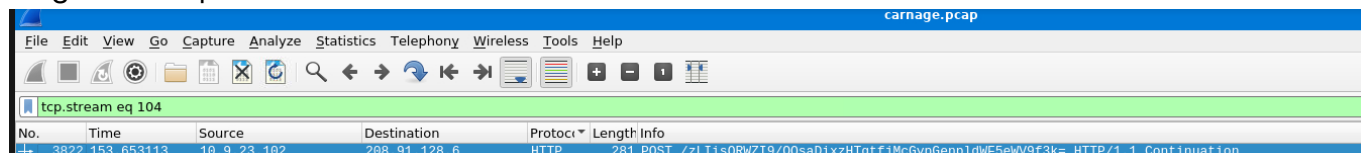


At the **POST** we can see the encoded base64 payload and see the first 11 characters that the victim sends out to the malicious domain

**What are the first eleven characters that the victim host sends out to the malicious domain involved in the post-infection traffic?**

**zLIisQRWZI9**

Wireshark allows us to see the length of packets, which is where we are able to identify the length of the packet sent from the C2 server:

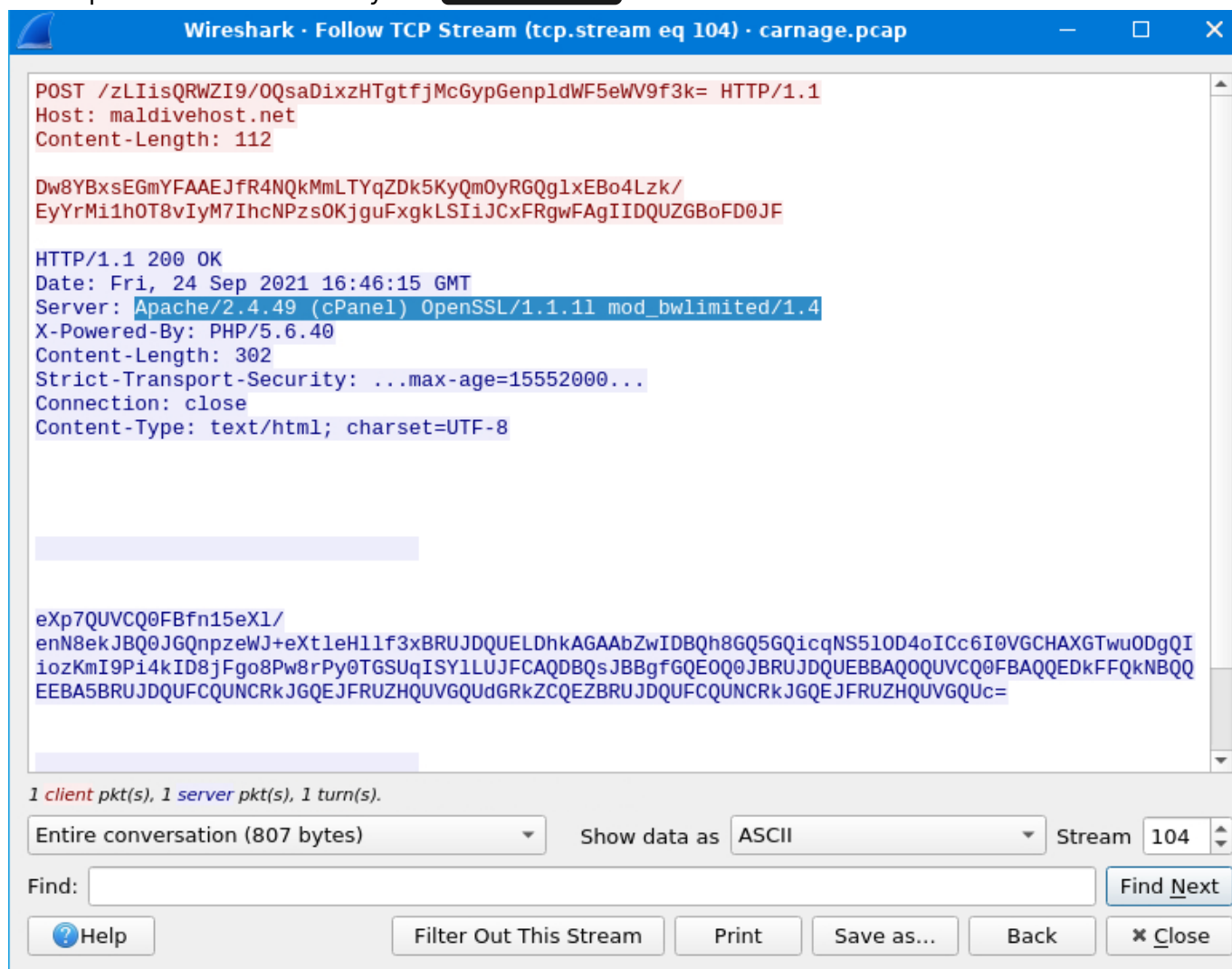


No.	Time	Source	Destination	Protocol	Length	Info
3822	153.653113	10.9.23.102	208.91.128.6	HTTP	281	POST /zLIisQRWZI9/OQsaDixzHTgtfjMcGypGenpldwF5ewV9f3k= HTTP/1.1 Continuation

**What was the length for the first packet sent out to the C2 server?**

**281**

In the packet we also identify the **Server-Header**:



Wireshark · Follow TCP Stream (tcp.stream eq 104) · carnage.pcap

```
POST /zLIisQRWZI9/OQsaDixzHTgtfjMcGypGenpldwF5ewV9f3k= HTTP/1.1
Host: maldivehost.net
Content-Length: 112

Dw8YBxsEGmYFAAEJfR4NqkMmLTyqZDk5KyQm0yRGQg1xEBo4Lzk/
EyYrMi1h0T8vIyM7IhcNPzs0KjguFvgkLSIiJCxFRgwFAgIIDQUZGBoFD0JF

HTTP/1.1 200 OK
Date: Fri, 24 Sep 2021 16:46:15 GMT
Server: Apache/2.4.49 (cPanel) OpenSSL/1.1.1l mod_bwlimited/1.4
X-Powered-By: PHP/5.6.40
Content-Length: 302
Strict-Transport-Security: ...max-age=15552000...
Connection: close
Content-Type: text/html; charset=UTF-8

eXp7QUVCQ0FBfn15eX1/
enN8ekJBQ0JGQnpzeWJ+eXtleH11f3xBRUJDQUELDhkAGAAbZwIDBQh8GQ5GQicqNS510D4oICc6I0VGCHAXGTWu0DgQI
iozKmI9Pi4kID8jFgo8Pw8rPy0TGSUqISY1LUJFCAQDBQsJBBgfGQE0Q0JBRUJDQUEBBAQ0QUVCQ0FBAQQEDkFFQkNBQQ
EEBA5BRUJDQUFCQUNCRkJGQEFJRUZHQUVGQUdGRkZCQEZBRUJDQUFCQUNCRkJGQEFJRUZHQUVGQUc=
```

1 client pkt(s), 1 server pkt(s), 1 turn(s).

Entire conversation (807 bytes)    Show data as ASCII    Stream 104

Find:  Find Next

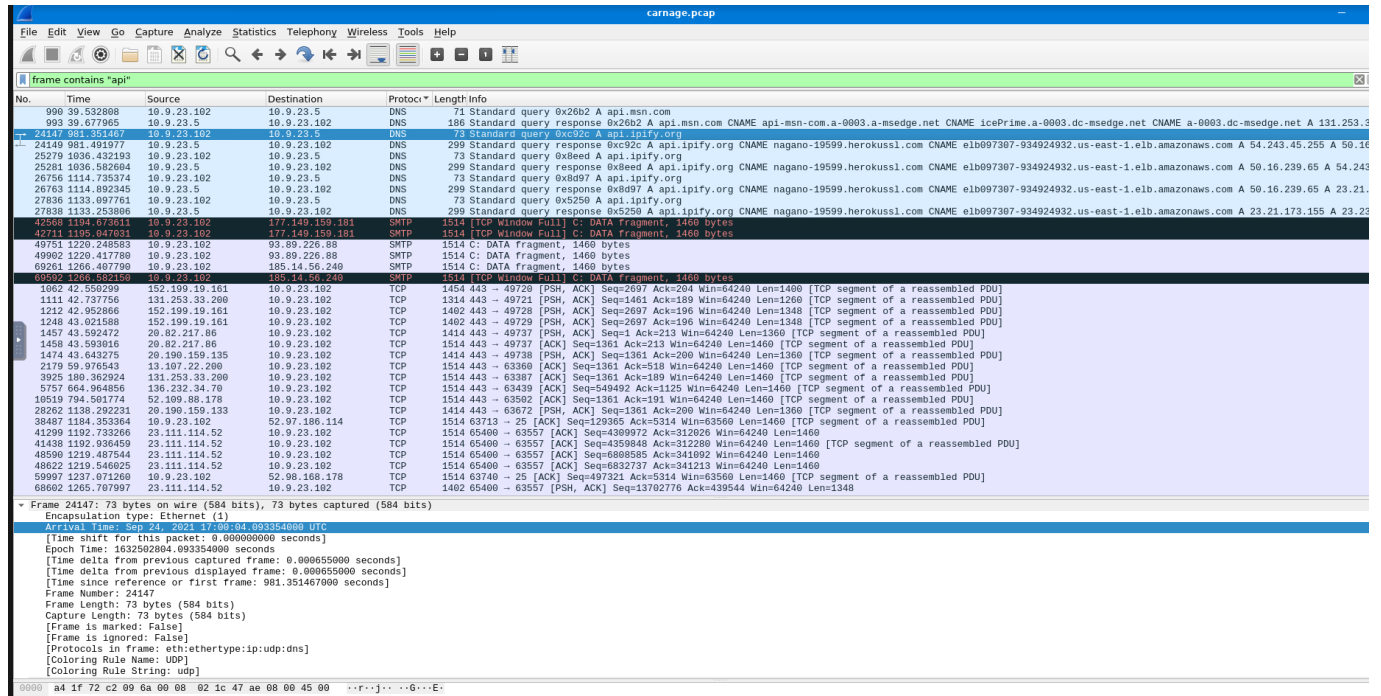
Help Filter Out This Stream Print Save as... Back Close

**What was the Server header for the malicious domain from the previous question?**

**Apache/2.4.49 (cPanel) OpenSSL/1.1.1l mod\_bwlimited/1.4**

We find out that the malware used an API to check for the IP addresses and we need to track down the time this occurred

Using filter `frame contains "api"` this helps track down the frame time and helps us find any packets that contain the word api - The third packet is where we find our answer of when the malware used the API:



No.	Time	Source	Destination	Protocol	Length	Info
996	30.532808	10.9.23.102	10.9.23.5	DNS	71	Standard query 0x26b2 A api.msn.com
993	30.677965	10.9.23.5	10.9.23.102	DNS	186	Standard query response 0x26b2 A api.msn.com CNAME api.msn.com.a-0003.a-msedge.net CNAME icePrime.a-0003.dc-msedge.net CNAME a-0003.dc-msedge.net A 131.253.3
24147	981.351467	10.9.23.102	10.9.23.5	DNS	73	Standard query 0xc92c A api.ipify.org
24149	981.491977	10.9.23.5	10.9.23.102	DNS	299	Standard query response 0xc92c A api.ipify.org CNAME nagano-19599.herokuapp1.com CNAME elb097307-934924932.us-east-1.elb.amazonaws.com A 54.243.45.255 A 50.16
25279	1036.432193	10.9.23.102	10.9.23.5	DNS	73	Standard query 0xc92c A api.ipify.org
25281	1036.582604	10.9.23.5	10.9.23.102	DNS	299	Standard query response 0xc92c A api.ipify.org CNAME nagano-19599.herokuapp1.com CNAME elb097307-934924932.us-east-1.elb.amazonaws.com A 50.16.239.65 A 54.243
26756	1114.735374	10.9.23.102	10.9.23.5	DNS	73	Standard query 0xc92c A api.ipify.org
26763	1114.892345	10.9.23.5	10.9.23.102	DNS	299	Standard query response 0xc92c A api.ipify.org CNAME nagano-19599.herokuapp1.com CNAME elb097307-934924932.us-east-1.elb.amazonaws.com A 50.16.239.65 A 23.21
27836	1133.097761	10.9.23.102	10.9.23.5	DNS	73	Standard query 0xc92c A api.ipify.org
27838	1133.253866	10.9.23.5	10.9.23.102	DNS	299	Standard query response 0xc92c A api.ipify.org CNAME nagano-19599.herokuapp1.com CNAME elb097307-934924932.us-east-1.elb.amazonaws.com A 23.21.173.155 A 23.22
42368	1104.670311	10.9.23.102	177.149.159.181	SMTP	1514	[TCP Window Full] C: DATA Fragment, 1460 bytes
42711	1195.047031	10.9.23.102	177.149.159.181	SMTP	1514	[TCP Window Full] C: DATA Fragment, 1460 bytes
49751	1220.248583	10.9.23.102	93.89.226.88	SMTP	1514	C: DATA Fragment, 1460 bytes
49862	1220.417780	10.9.23.102	93.89.226.88	SMTP	1514	C: DATA Fragment, 1460 bytes
60581	1266.407790	10.9.23.102	185.14.56.240	SMTP	1514	C: DATA Fragment, 1460 bytes
60592	1266.582159	10.9.23.102	185.14.56.240	SMTP	1514	[TCP Window Full] C: DATA Fragment, 1460 bytes
1062	42.550299	152.199.19.161	10.9.23.102	TCP	1454	443 -> 49720 [PSH, ACK] Seq=2697 Ack=204 Win=64240 Len=1400 [TCP segment of a reassembled PDU]
1111	42.787786	131.253.33.200	10.9.23.102	TCP	1514	443 -> 49721 [PSH, ACK] Seq=1461 Ack=189 Win=64240 Len=1260 [TCP segment of a reassembled PDU]
1212	42.952866	152.199.19.161	10.9.23.102	TCP	1402	443 -> 49728 [PSH, ACK] Seq=2697 Ack=196 Win=64240 Len=1348 [TCP segment of a reassembled PDU]
1248	43.021588	152.199.19.161	10.9.23.102	TCP	1402	443 -> 49729 [PSH, ACK] Seq=2697 Ack=196 Win=64240 Len=1348 [TCP segment of a reassembled PDU]
1457	43.582472	20.82.217.86	10.9.23.102	TCP	1414	443 -> 49737 [PSH, ACK] Seq=1 Ack=213 Win=64240 Len=1360 [TCP segment of a reassembled PDU]
1458	43.593016	20.82.217.86	10.9.23.102	TCP	1514	443 -> 49737 [ACK] Seq=1361 Ack=213 Win=64240 Len=1460 [TCP segment of a reassembled PDU]
1474	43.643275	20.190.159.135	10.9.23.102	TCP	1414	443 -> 49738 [PSH, ACK] Seq=1361 Ack=200 Win=64240 Len=1360 [TCP segment of a reassembled PDU]
2179	59.976543	13.107.22.200	10.9.23.102	TCP	1514	443 -> 63369 [ACK] Seq=1361 Ack=518 Win=64240 Len=1460 [TCP segment of a reassembled PDU]
3925	100.362924	131.253.33.200	10.9.23.102	TCP	1514	443 -> 63387 [ACK] Seq=1361 Ack=189 Win=64240 Len=1460 [TCP segment of a reassembled PDU]
5757	664.964856	136.232.34.70	10.9.23.102	TCP	1514	443 -> 63439 [ACK] Seq=549492 Ack=1125 Win=64240 Len=1460 [TCP segment of a reassembled PDU]
10519	794.561774	52.109.88.178	10.9.23.102	TCP	1514	443 -> 63502 [ACK] Seq=1361 Ack=191 Win=64240 Len=1460 [TCP segment of a reassembled PDU]
28262	1138.292231	20.190.159.133	10.9.23.102	TCP	1414	443 -> 63672 [PSH, ACK] Seq=1361 Ack=200 Win=64240 Len=1360 [TCP segment of a reassembled PDU]
38487	1184.353364	10.9.23.102	52.97.186.114	TCP	1514	63713 -> 25 [ACK] Seq=129305 Ack=5314 Win=63560 Len=1460 [TCP segment of a reassembled PDU]
41299	1192.733266	23.111.114.52	10.9.23.102	TCP	1514	65400 -> 63557 [ACK] Seq=4389972 Ack=312026 Win=64240 Len=1460
41438	1192.936450	23.111.114.52	10.9.23.102	TCP	1514	65400 -> 63557 [ACK] Seq=4359848 Ack=312280 Win=64240 Len=1460 [TCP segment of a reassembled PDU]
48590	1219.487544	23.111.114.52	10.9.23.102	TCP	1514	65400 -> 63557 [ACK] Seq=6888585 Ack=341092 Win=64240 Len=1460
48622	1219.546025	23.111.114.52	10.9.23.102	TCP	1514	65400 -> 63557 [ACK] Seq=6832737 Ack=341213 Win=64240 Len=1460
59097	1237.071260	10.9.23.102	52.98.168.178	TCP	1514	63740 -> 25 [ACK] Seq=497321 Ack=5314 Win=63560 Len=1460 [TCP segment of a reassembled PDU]
68602	1265.707997	23.111.114.52	10.9.23.102	TCP	1402	65400 -> 63557 [PSH, ACK] Seq=13702776 Ack=439544 Win=64240 Len=1348

Frame 24147: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface 0  
Encapsulation type: Ethernet (1)  
Capture interface: eth0  
Time shift for this packet: 0.000000000 seconds  
Epoch Time: 1632502804.093354000 seconds  
[Time delta from previous captured frame: 0.000550000 seconds]  
[Time delta from previous displayed frame: 0.000550000 seconds]  
[Time since reference or first frame: 981.351467000 seconds]  
Frame Number: 24147  
Frame Length: 73 bytes (584 bits)  
Capture Length: 73 bytes (584 bits)  
[Frame is marked: False]  
[Frame is ignored: False]  
[Protocols in frame: eth:ethertype:ip:udp:dns]  
[Coloring Rule Name: UDP]  
[Coloring Rule String: udp]

0000 a4 1f 72 c2 99 6a 00 08 02 1c 47 ae 08 00 45 80 ...f...-G...E-

The malware used an API to check for the IP address of the victim's machine. What was the date and time when the DNS query for the IP check domain occurred? (answer format: yyyy-mm-dd hh:mm:ss UTC)

2021-09-24 17:00:04

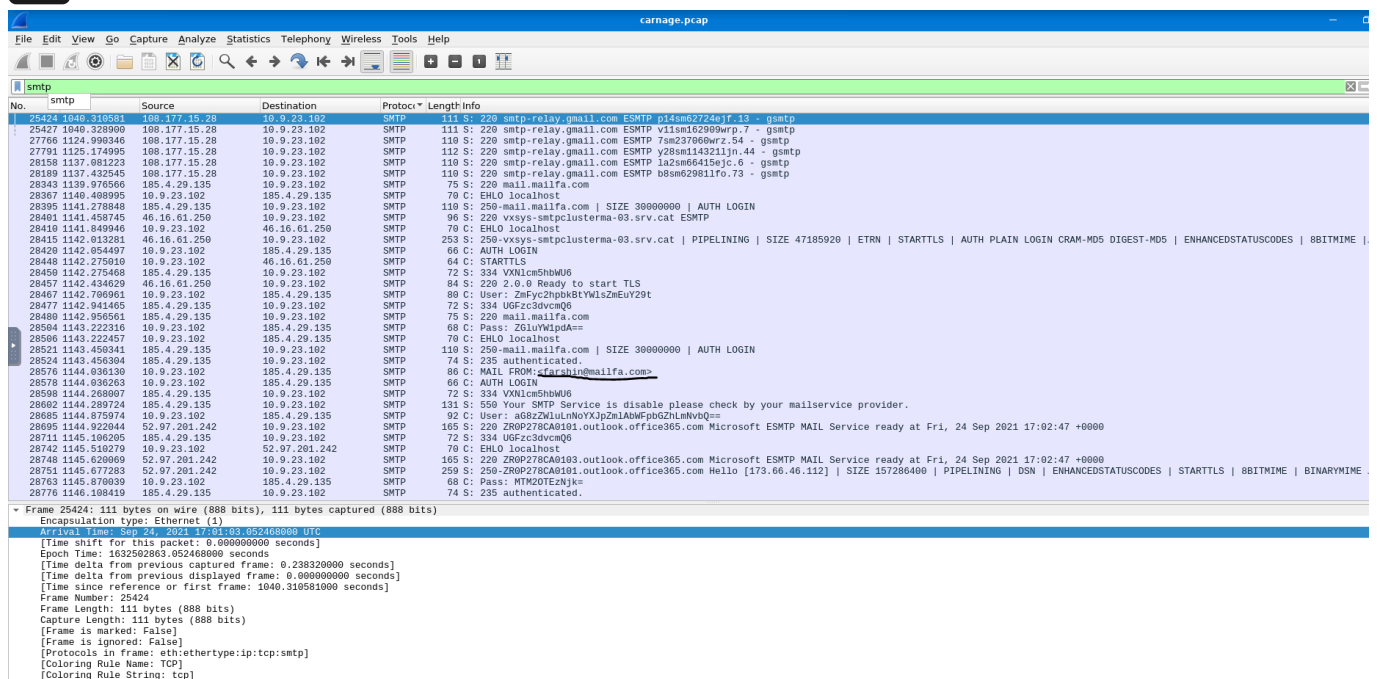
We also manage to find the domain name of the api:



What was the domain in the DNS query from the previous question?

api.ipify.org

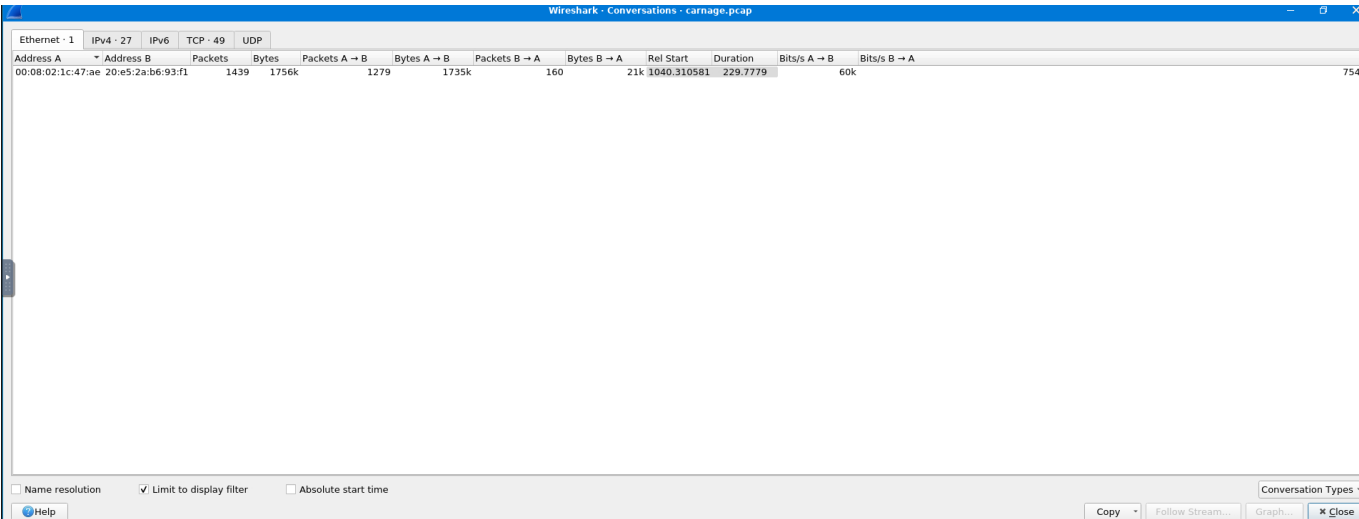
We have almost finished locating everything from the pcap files, the last two things we are asked to identify is a malicious email that was sent - We know that mail servers run from a port called **SMTP** so we filter this in Wireshark and we find out who sent the email:



Looks like there was some malicious spam (malspam) activity going on. What was the first MAIL FROM address observed in the traffic?

[farshin@mailfa.com](mailto:farshin@mailfa.com)

For our last task we need to identify how many packets were observed for the SMTP traffic - We identify this by going to **statistics > conversations** after we select **ethernet** as it is on the network and we click **limit to display filter** and we have completed our final task and have successfully identified every malicious domain/IP:



How many packets were observed for the SMTP traffic?

1439