THE UNIVERSITY OF
MELBOURNE

COMP90073- Security Analytics

Assignment 1

Student Name:	Lihua Wang
Student ID:	1164051
Tutor:	Mark Jiang
Semester:	2020/S2

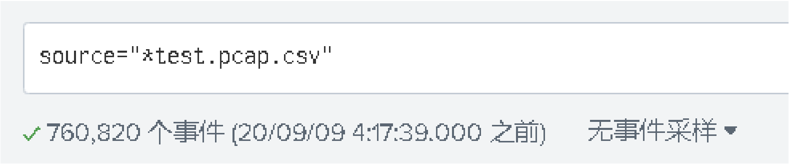
School of Computing and Information Systems

Table of Contents

Technical Report

1. Data Description and Summary (Q1, Q2)

The .pcap file has been ingested into Splunk and converted to (. pcap.csv) format successfully. I renamed the source file as “test.pacp.csv”. The total items collected in Splunk are 760,820 events.



```
source="*test.pcap.csv"
```

✓760,820 个事件 (20/09/09 4:17:39.000 之前)   无事件采样 ▾

I used PCAP Analyzer to search the evidences of following attack scenarios.

1) Botnet Command & Control (C2)

Attackers	202.166.84.165
Victim	31.192.109.167
Attack type	C&C Attack based on HTTP

a. 54 events of C2 HTTP-based in total

URIs: */snapbn/gate.php* for POST (53), and */snapbn/ip.php* for GET (1)

b. Start time: Jun 19, 2020 00:55:21.316470000 AEST

End time: Jun 19, 2020 00:58:00.085710000 AEST

2) SPAM

Attackers	202.166.84.165
Victim	214 emails
Attack type	SPAM attack

a. 214 emails were targeted by this spam and all email address are different.

b. Start time: Jun 19, 2020 00:55:23.278082000 AEST

First recipient: <*nickandsonia@comcast.net*>

End time: Jun 19, 2020 01:01:12.900452000 AEST

Last recipient: <*jberman1@gmail.com*>

3) ClickFraud

Attackers	202.166.84.165
Victim	98.126.71.122
Attack type	ClickFraud attack

a. 38 events of ClickFraud requests in total

URIs: */gen.php*

b. Start time: Jun 19, 2020 00:55:22.906077000 AEST

End time: Jun 19, 2020 01:01:08.208700000 AEST

4) IRC

Attackers	17 IP address of IRC
Victim	Infected machine
Attack type	C&C attack based on IRC

a. 17 IRC servers (IP addresses), 31 POST requests in total made by the infected machine.

b. Start time: Jun 19, 2020 00:55:21.813824000 AEST

End time: Jun 19, 2020 01:01:59.756180000 AEST

Data Summary:

Based on the findings of the four types of attack, they are happened between 00:55:21 to 01:01:59. All the attacks requests from the same compromised machine which IP address is 202.166.84.165. I search this IP by LookInfo website and showed that it located Moscow, the victim networks were almost located in United States which were supported by companies with hardware and software business. it seems that a Russian attacker floods mad requests to American companies, start from asking the DNS server for resolving the IP of the target machine, after TCP 3-way handshake occurred, the attacks started.

I also find some suspicious IP address regarding 212.117.171.138, and 174.133.57.141; 173.192.170.88 which made several transactions to over 200 different ports, but not sure what kind of attacks they are doing.

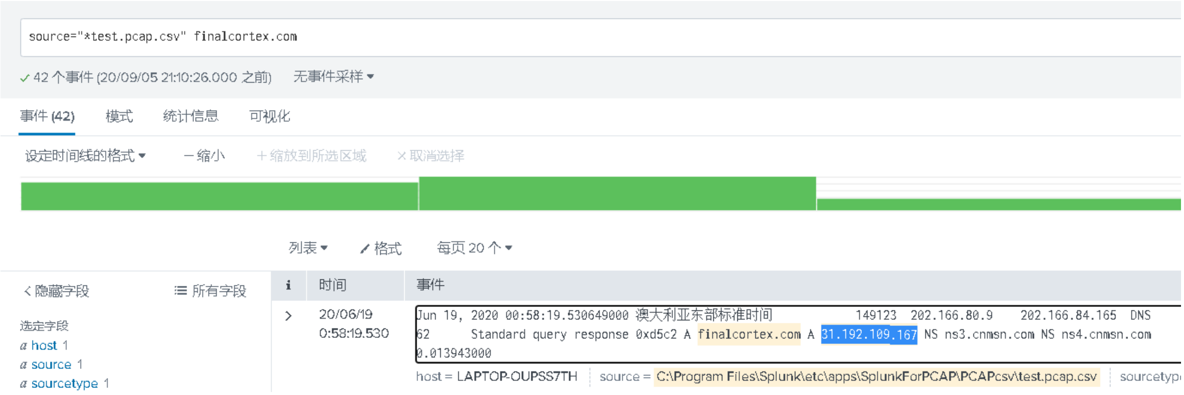
2. Methodology of Analysis (Q2)

The above results are all extracted by PCAP Analyzer and a few evidences got from Wireshark. The following are my process of analysis.

1) Botnet Command & Control (C2)

First, to get the HTTP requests, I searched for the IP address of C2 server “finalcortex.com”.

- **Command**: source="*test.pcap.csv" finalcortex.com
- **Result:** IP address is 31.192.109.167



```
source="*test.pcap.csv" finalcortex.com
```

✓ 42 个事件 (20/09/05 21:10:26.000 之前)

```
Jun 19, 2020 00:58:19.530649000 澳大利亚东部标准时间          149123  202.166.80.9    202.166.84.165  DNS
62      Standard query response 0xd5c2 A finalcortex.com A 31.192.109.167 NS ns3.cnmsn.com NS ns4.cnmsn.com
0.013943000
```

host = LAPTOP-OUPSS7TH | source = C:\Program Files\Splunk\etc\apps\SplunkForPCAP\PCAPcsv\test.pcap.csv

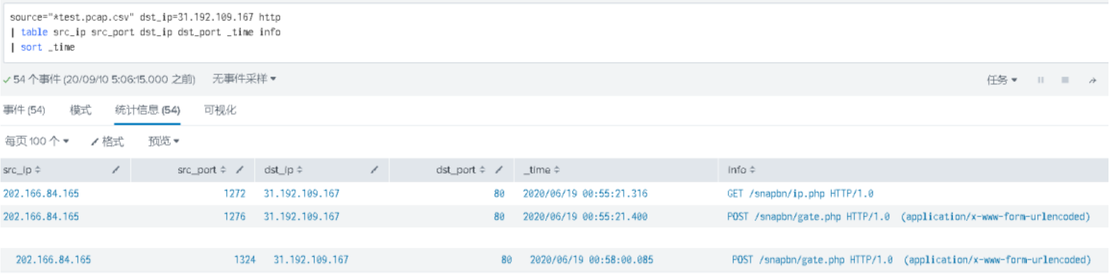
Second, I search the requests via IP address as destination IP because the C2 server should be receiver. Also add http conditions due to Http based C2 channel.

For visualization clearly, I set the data as table to show the results.

- **Command**: source="*test.pcap.csv" dst_ip=31.192.109.167 http

  | table src_ip src_port dst_ip dst_port _time info

  | sort _time
- **Result:** 54 events: 53 for POST; 1 for GET



```
source="*test.pcap.csv" dst_ip=31.192.109.167 http
| table src_ip src_port dst_ip dst_port _time info
| sort _time
```

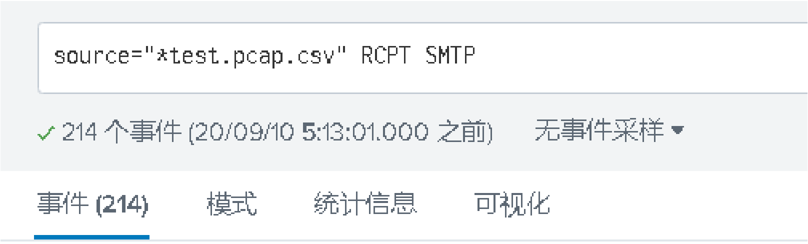
✓ 54 个事件 (20/09/10 5:06:15.000 之前)

src_ip	src_port	dst_ip	dst_port	_time	info
202.166.84.165	1272	31.192.109.167	80	2020/06/19 00:55:21.316	GET /snapbn/ip.php HTTP/1.0
202.166.84.165	1276	31.192.109.167	80	2020/06/19 00:55:21.400	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	1324	31.192.109.167	80	2020/06/19 00:58:00.085	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)

2) SPAM

First, in order to find all the infected email address, I search the key words of email protocol – SMTP and “RCPT”.

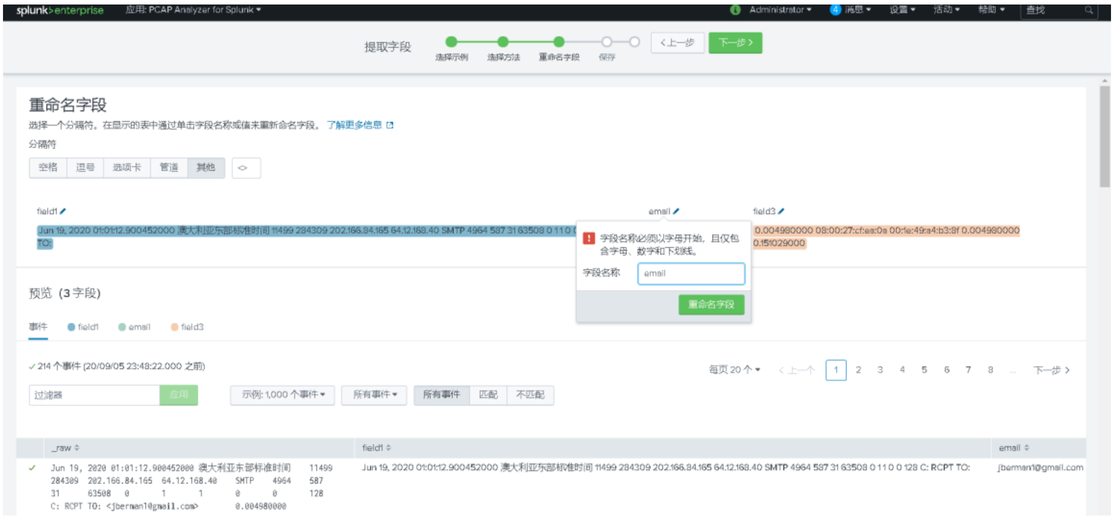
- **Command**: source="*test.pcap.csv" RCPT SMTP
- **Result:** 214 events



```
source="*test.pcap.csv" RCPT SMTP
```

✓ 214 个事件 (20/09/10 5:13:01.000 之前)

In order to visualize clearly, I also extract a new string "email" from "info" field of items. The process is as follow:



重命名字段

选择一个分隔符。在显示的表中通过单击字段名称或值来重新命名字段。 了解更多信息

分隔符: 其他

field1: Jun 19, 2020 01:01:12.900452000 澳大利亚东部标准时间 11499 284309 202.166.84.165 64.12.168.40 SMTP 4964 587 31 63508 0 1 1 0 0 128 C: RCPT TO:

field3: 0.004980000 08:00:27:cf:ea:0a 00:1e:49:a4:b3:8f 0.004980000 0.151029000

! 字段名称必须以字母开始，且仅包含字母、数字和下划线。

字段名称: email

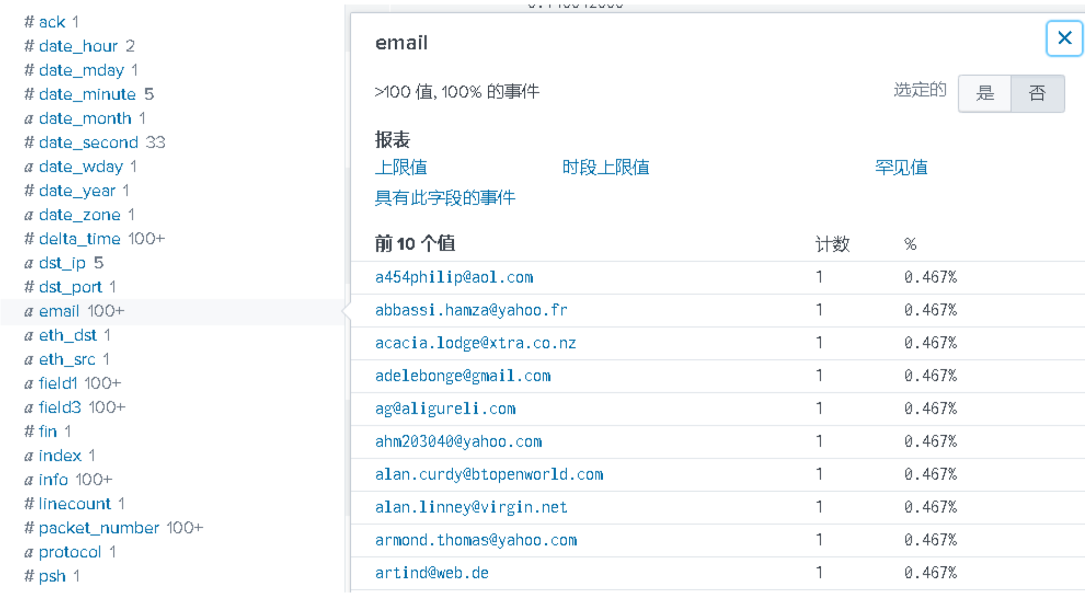
预览 (3 字段)

✓ 214 个事件 (20/09/05 23:48:22.000 之前)

_raw	field1	email
Jun 19, 2020 01:01:12.900452000 澳大利亚东部标准时间 11499 284309 202.166.84.165 64.12.168.40 SMTP 4964 587 31 63508 0 1 1 0 0 128 C: RCPT TO: <jberman1@gmail.com> 0.004980000	Jun 19, 2020 01:01:12.900452000 澳大利亚东部标准时间 11499 284309 202.166.84.165 64.12.168.40 SMTP 4964 587 31 63508 0 1 1 0 0 128 C: RCPT TO:	jberman1@gmail.com

Then the new command and results shown below:

- **Command**: source="*test.pcap.csv" RCPT email="*"
- **Result:** 214 events



ack 1
date_hour 2
date_mday 1
date_minute 5
a date_month 1
# date_second 33
a date_wday 1
# date_year 1
a date_zone 1
# delta_time 100+
a dst_ip 5
# dst_port 1
a email 100+
a eth_dst 1
a eth_src 1
a field1 100+
a field3 100+
# fin 1
a index 1
a info 100+
# linecount 1
# packet_number 100+
a protocol 1
# psh 1

email

>100 值, 100% 的事件

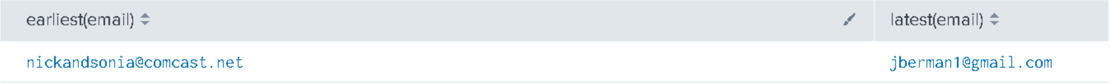
前 10 个值	计数	%
a454philip@aol.com	1	0.467%
abbassi.hamza@yahoo.fr	1	0.467%
acacia.lodge@xtra.co.nz	1	0.467%
adelebonge@gmail.com	1	0.467%
ag@aligureli.com	1	0.467%
ahm203040@yahoo.com	1	0.467%
alan.curdy@btopenworld.com	1	0.467%
alan.linney@virgin.net	1	0.467%
armond.thomas@yahoo.com	1	0.467%
artind@web.de	1	0.467%

Second, to find the first and last recipients and their time, I use some command for easily to find.

- **Command**: source="*test.pcap.csv" email="*" RCPT

```
| table _time email
| sort _time
| stats earliest latest
```

- **Result:**

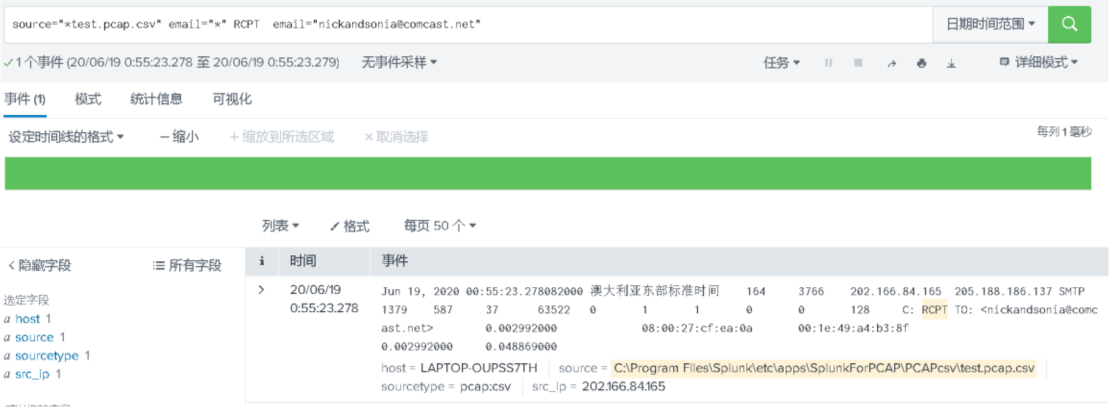


earliest(email)	latest(email)
nickandsonia@comcast.net	jberman1@gmail.com

For look up the details of each items, using following commands.

The first email:

- **Command**: source="*test.pcap.csv" email="*" RCPT email="nickandsonia@comcast.net"
- **Result:**



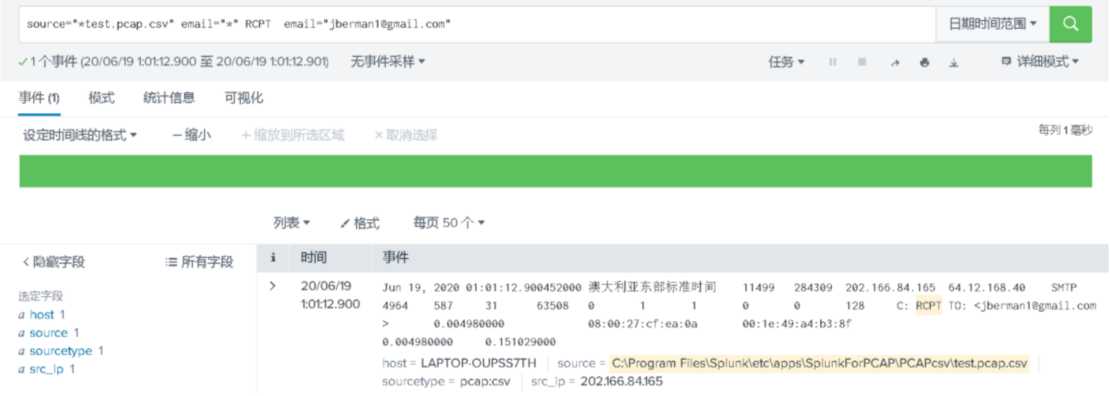
```
source="*test.pcap.csv" email="*" RCPT  email="nickandsonia@comcast.net"
```

✓1个事件 (20/06/19 0:55:23.278 至 20/06/19 0:55:23.279)

i	时间	事件
>	20/06/19 0:55:23.278	Jun 19, 2020 00:55:23.278082000 澳大利亚东部标准时间 164 3766 202.166.84.165 205.188.186.137 SMTP 1379 587 37 63522 0 1 1 0 0 128 C: RCPT TO: <nickandsonia@comcast.net> 0.002992000 08:00:27:cf:ea:0a 00:1e:49:a4:b3:8f 0.002992000 0.048869000 host = LAPTOP-OUPSS7TH source = C:\Program Files\Splunk\etc\apps\SplunkForPCAP\PCAPcsv\test.pcap.csv sourcetype = pcap:csv src_ip = 202.166.84.165

The second email:

- **Command**: source="*test.pcap.csv" email="*" RCPT email="jberman1@gmail.com"
- **Result:**



```
source="*test.pcap.csv" email="*" RCPT  email="jberman1@gmail.com"
```

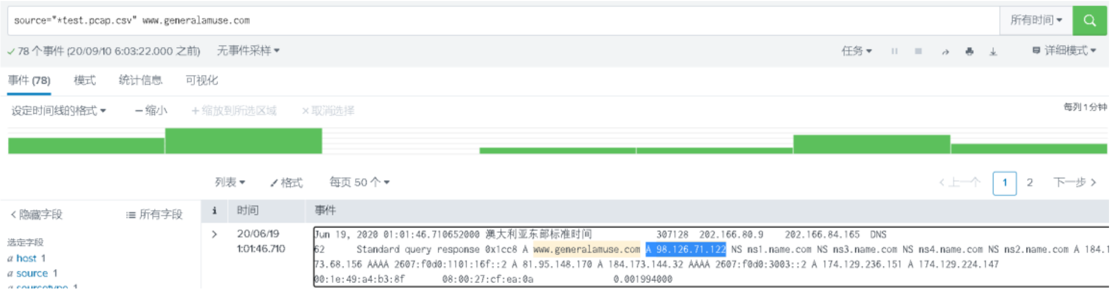
✓1个事件 (20/06/19 1:01:12.900 至 20/06/19 1:01:12.901)

i	时间	事件
>	20/06/19 1:01:12.900	Jun 19, 2020 01:01:12.900452000 澳大利亚东部标准时间 11499 284309 202.166.84.165 64.12.168.40 SMTP 4964 587 31 63508 0 1 1 0 0 128 C: RCPT TO: <jberman1@gmail.com> 0.004980000 08:00:27:cf:ea:0a 00:1e:49:a4:b3:8f 0.004980000 0.151029000 host = LAPTOP-OUPSS7TH source = C:\Program Files\Splunk\etc\apps\SplunkForPCAP\PCAPcsv\test.pcap.csv sourcetype = pcap:csv src_ip = 202.166.84.165

3) ClickFraud

First, to get the HTTP requests, we need to search for the IP address of web site “*www.generalamuse.com*”.

- **Command**: source="*test.pcap.csv" www.generalamuse.com
- **Result:** IP address is 98.126.71.122



```
source="*test.pcap.csv" www.generalamuse.com
```

✓ 78 个事件 (20/09/10 6:03:22.000 之前) 无事件采样

i	时间	事件
>	20/06/19 1:01:46.710	Jun 19, 2020 01:01:46.710652000 澳大利亚东部标准时间 307128 202.166.80.9 202.166.84.165 DNS 62 Standard query response 0x1cc8 A www.generalamuse.com A 98.126.71.122 NS ns1.name.com NS ns3.name.com NS ns4.name.com NS ns2.name.com A 184.1 73.68.156 AAAA 2607:f0d0:1101:16f::2 A 81.95.148.170 A 184.173.144.32 AAAA 2607:f0d0:3003::2 A 174.129.236.151 A 174.129.224.147 00:1e:49:a4:b3:8f 08:00:27:cf:ea:0a 0.001994000

Second, I search the requests via IP address as destination address because the website should be receiver. Also add http conditions due to request based on HTTP GET.

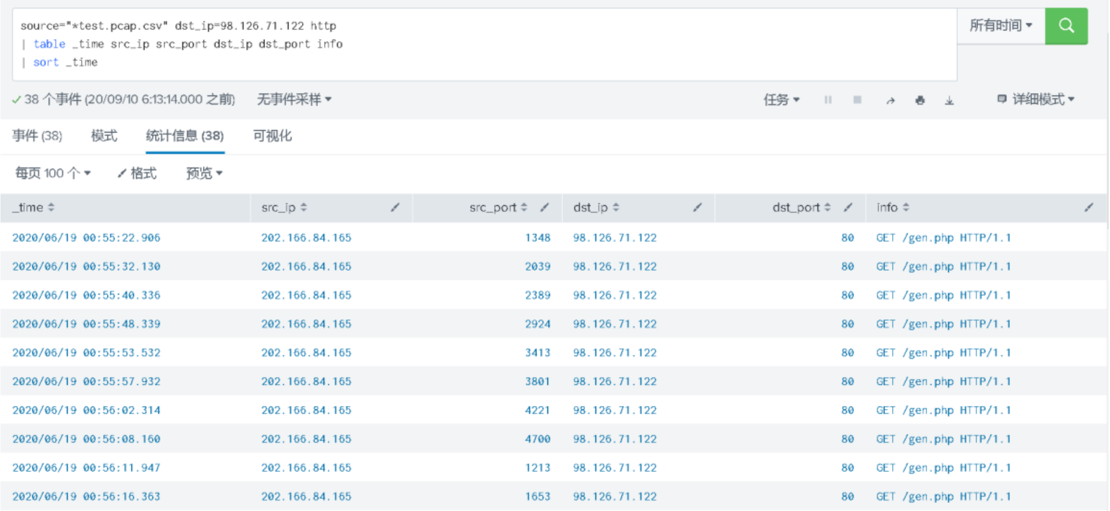
For visualization clearly, I set the data as table to show the results.

- ➢ **Command**: source="*test.pcap.csv" dst_ip=98.126.71.122 http

  | table _time src_ip src_port dst_ip dst_port info

  | sort _time

- ➢ **Result:** 38 events



```
source="*test.pcap.csv" dst_ip=98.126.71.122 http
| table _time src_ip src_port dst_ip dst_port info
| sort _time
```

✓ 38 个事件 (20/09/10 6:13:14.000 之前) 无事件采样

_time	src_ip	src_port	dst_ip	dst_port	info
2020/06/19 00:55:22.906	202.166.84.165	1348	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:55:32.130	202.166.84.165	2039	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:55:40.336	202.166.84.165	2389	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:55:48.339	202.166.84.165	2924	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:55:53.532	202.166.84.165	3413	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:55:57.932	202.166.84.165	3801	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:56:02.314	202.166.84.165	4221	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:56:08.160	202.166.84.165	4700	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:56:11.947	202.166.84.165	1213	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:56:16.363	202.166.84.165	1653	98.126.71.122	80	GET /gen.php HTTP/1.1

Thirdly, in order to find the start time and end time, I made some commands following to visualize clearly.

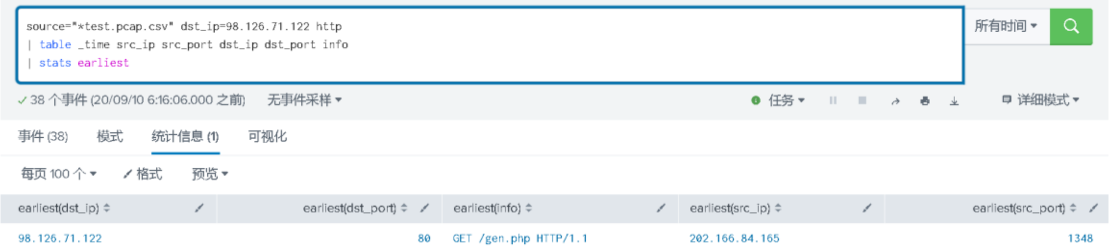
The start event:

- ➢ **Command**: source="*test.pcap.csv" dst_ip=98.126.71.122 http

  | table _time src_ip src_port dst_ip dst_port info

  | stats earliest

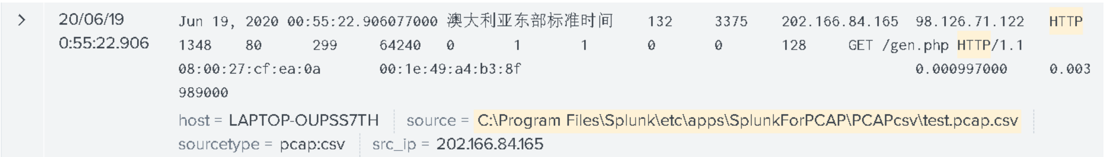
- ➢ **Result:**



```
source="*test.pcap.csv" dst_ip=98.126.71.122 http
| table _time src_ip src_port dst_ip dst_port info
| stats earliest
```

✓ 38 个事件 (20/09/10 6:16:06.000 之前) 无事件采样

earliest(dst_ip)	earliest(dst_port)	earliest(info)	earliest(src_ip)	earliest(src_port)
98.126.71.122	80	GET /gen.php HTTP/1.1	202.166.84.165	1348



```
>  20/06/19      Jun 19, 2020 00:55:22.906077000 澳大利亚东部标准时间    132     3375     202.166.84.165  98.126.71.122   HTTP
   0:55:22.906   1348     80       299      64240    0        1        1        0        0        128      GET /gen.php HTTP/1.1
                 08:00:27:cf:ea:0a        00:1e:49:a4:b3:8f                                         0.000997000      0.003
                 989000
                 host = LAPTOP-OUPSS7TH | source = C:\Program Files\Splunk\etc\apps\SplunkForPCAP\PCAPcsv\test.pcap.csv
                 sourcetype = pcap:csv | src_ip = 202.166.84.165
```

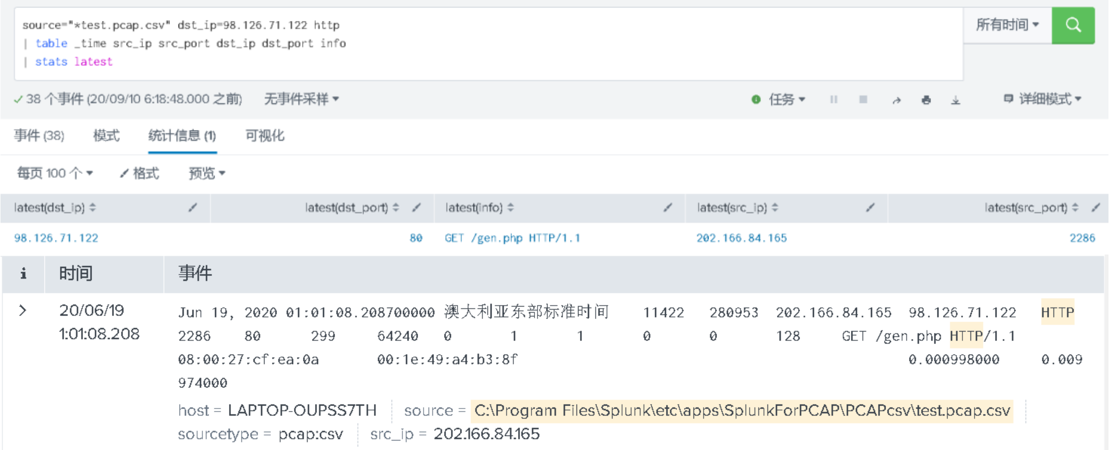
The end event:

- ➢ **Command**: source="*test.pcap.csv" dst_ip=98.126.71.122 http

  | table _time src_ip src_port dst_ip dst_port info

  | stats latest

- ➢ **Result:**



```
source="*test.pcap.csv" dst_ip=98.126.71.122 http
| table _time src_ip src_port dst_ip dst_port info
| stats latest
```

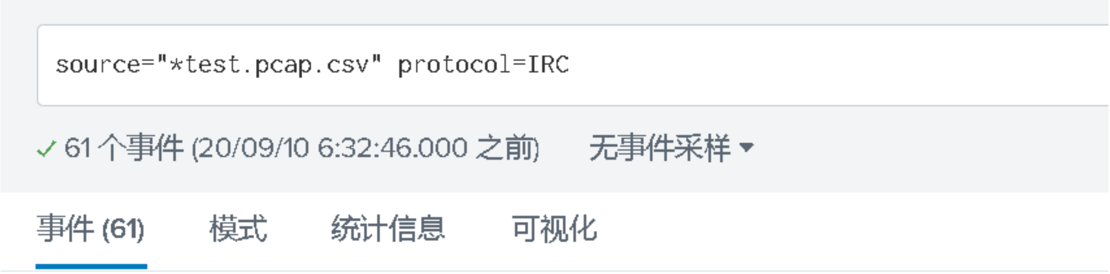
latest(dst_ip)	latest(dst_port)	latest(info)	latest(src_ip)	latest(src_port)
98.126.71.122	80	GET /gen.php HTTP/1.1	202.166.84.165	2286

```
i  时间           事件
>  20/06/19      Jun 19, 2020 01:01:08.208700000 澳大利亚东部标准时间    11422    280953   202.166.84.165  98.126.71.122   HTTP
   1:01:08.208   2286     80       299      64240    0        1        1        0        0        128      GET /gen.php HTTP/1.1
                 08:00:27:cf:ea:0a        00:1e:49:a4:b3:8f                                         0.000998000      0.009
                 974000
                 host = LAPTOP-OUPSS7TH | source = C:\Program Files\Splunk\etc\apps\SplunkForPCAP\PCAPcsv\test.pcap.csv
                 sourcetype = pcap:csv | src_ip = 202.166.84.165
```

4) IRC

To find all the IRC events, we can search events by key words “protocol=IRC”.

- ➢ **Command**: source="*test.pcap.csv" protocol=IRC
- ➢ **Result:** 61 IRC events



```
source="*test.pcap.csv" protocol=IRC
```

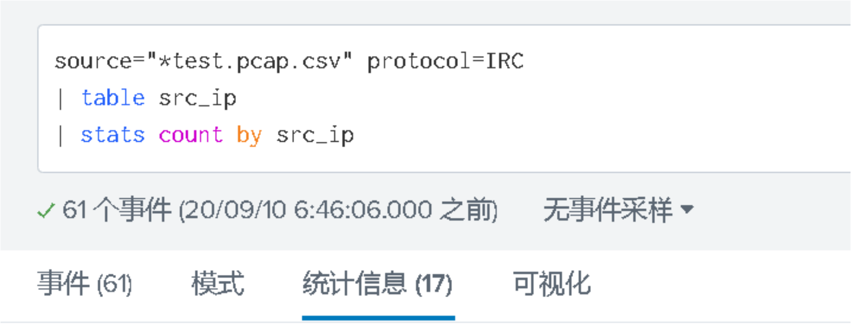
Then, to identify all the IRC servers, we need to find all the unique IP address from IRC servers, so we can search unique source IP of IRC events.

- ➢ **Command**: source="*test.pcap.csv" protocol=IRC

  | table src_ip

  | dedup src_ip

- ➢ **Result:** 17 IRC servers

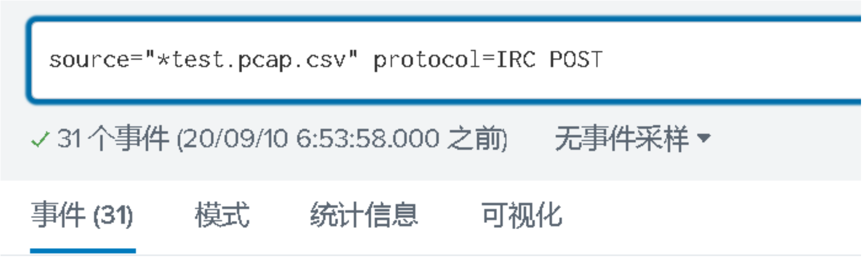


```
source="*test.pcap.csv" protocol=IRC
| table src_ip
| stats count by src_ip
```

✓ 61 个事件 (20/09/10 6:46:06.000 之前) 无事件采样 ▾

Next, to calculate the number of POST requests, we can directly use key words “POST” and “protocol=IRC” to find the results.

- **Command**: source="*test.pcap.csv" protocol=IRC post
- **Result:** 31 servers

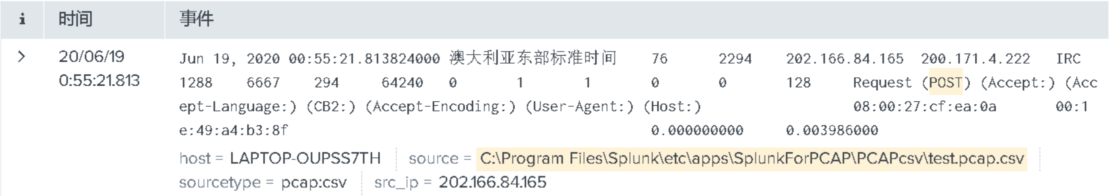


```
source="*test.pcap.csv" protocol=IRC POST
```

✓ 31 个事件 (20/09/10 6:53:58.000 之前) 无事件采样 ▾

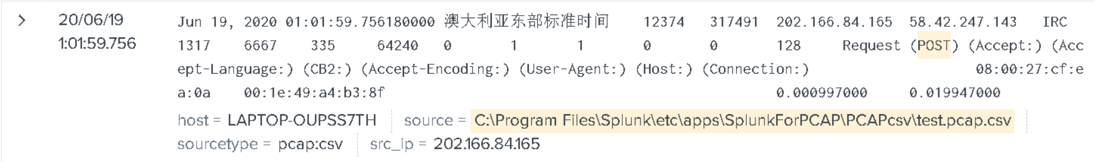
To find the start time and end time, can directly find in the results:

The start event:



```
i   时间            事件
>   20/06/19        Jun 19, 2020 00:55:21.813824000 澳大利亚东部标准时间    76      2294    202.166.84.165  200.171.4.222   IRC
    0:55:21.813     1288    6667    294     64240   0       1       1       0       0       128     Request (POST) (Accept:) (Acc
                    ept-Language:) (CB2:) (Accept-Encoding:) (User-Agent:) (Host:)                  08:00:27:cf:ea:0a       00:1
                    e:49:a4:b3:8f                                                   0.000000000     0.003986000
                    host = LAPTOP-OUPSS7TH | source = C:\Program Files\Splunk\etc\apps\SplunkForPCAP\PCAPcsv\test.pcap.csv
                    sourcetype = pcap:csv | src_ip = 202.166.84.165
```

The end event:



```
>   20/06/19        Jun 19, 2020 01:01:59.756180000 澳大利亚东部标准时间    12374   317491  202.166.84.165  58.42.247.143   IRC
    1:01:59.756     1317    6667    335     64240   0       1       1       0       0       128     Request (POST) (Accept:) (Acc
                    ept-Language:) (CB2:) (Accept-Encoding:) (User-Agent:) (Host:) (Connection:)                    08:00:27:cf:e
                    a:0a    00:1e:49:a4:b3:8f                                       0.000997000     0.019947000
                    host = LAPTOP-OUPSS7TH | source = C:\Program Files\Splunk\etc\apps\SplunkForPCAP\PCAPcsv\test.pcap.csv
                    sourcetype = pcap:csv | src_ip = 202.166.84.165
```

3. Attack Description and Narrative (Q3, Q4)

Based on the filter web logs, we found the 4 attacks almost concentrated on the same time between 00:55:21 to 01:01:59 on the 19 July and it is clear to construct the attack narratives.

Timeline	Narratives
00:55:21.311AM	Attacker (202.166.84.165) start connection to victim network (31.192.109.167) made by the TCP three-way handshake.
00:55:21.316AM	C2 attack started. The unencrypted C&C connection based on HTTP established, 202.166.84.165 sent many flows commands in periodic with

	POST and GET requests.
00:55:21.813AM	Compromised Machine (202.166.84.165) started attack by generating a POST request to victim server (200.171.4.222).
00:55:22.906AM	Compromised machine (202.166.84.165) started ClickFraud attack via different source ports to make GET requests to target website (98.126.71.122).
00:55:23.278AM	SPAM attack start, the compromised machine (202.166.84.165) sent spam to the first recipient <nickandsonia@comcast.net>
00:58:00.085AM	C2 attack based on HTTP ended!
01:01:08.208AM	Compromised machine (202.166.84.165) ended ClickFraud attack.
01:01:12.900AM	SPAM attack ended, the compromised machine (202.166.84.165) sent spam to the last recipient <jberman1@gmail.com>
01:01:59.756AM	Compromised Machine (202.166.84.165) ended C2 attack based on IRC victim server (58.42.247.143).

Table1. Overall Attack Narratives

The following are the details of the 4 attacks narratives.

1) Botnet Command & Control (C2)

Botnet is a network of compromised computers controlled by attackers from remote location via C&C (Command and Control) channels. In the following narratives, the botnet used an HTTP based C&C channel, so the compromised machine started attack from building a C&C channel by TCP three-way handshake.

Time	Narratives
00:55:21.279AM	C2 server (202.166.84.165) request an IP address of *"finalcortex.com"* to domain 202.166.80.9.
00:55:21.309AM	202.166.80.9 responded to 202.166.84.165 the *"finalcortex.com"* IP address is 31.192.109.167
00:55:21.311AM	Attacker (202.166.84.165) start connection to victim network (31.192.109.167) made by the TCP three-way handshake.
00:55:21.316AM	C2 attack started. The unencrypted C&C connection based on HTTP established, 202.166.84.165 sent many flows commands in periodic with POST and GET requests.
00:58:00.085AM	C2 attack based on HTTP ended!

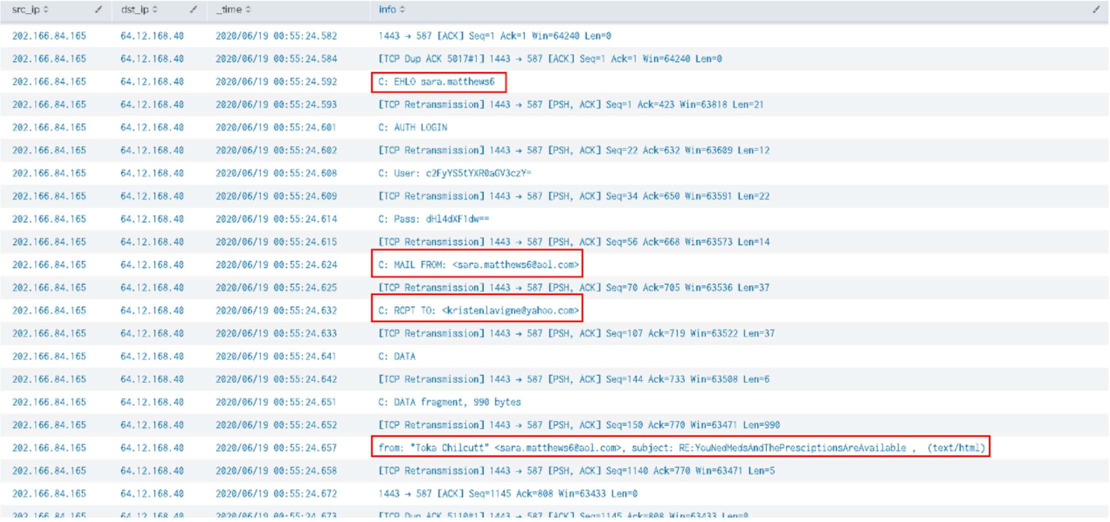
Table1. C2 Narratives

Consequence: Command and control attack will damage data integrity. Once hackers control an organization system remotely and make subtle, stealthy tweaks to data, these subtle modifications could be as crippling to organizations as data breaches. In some scenarios, the hacker didn't change the data space entity framework but will be benefited from the high value of the fraud which has compromised data integrity.

2) SPAM

Spam is any kind of unwanted, unsolicited digital communication, were sent to different users in large quantities in a short period of time. Sometimes, the attacker disguises the real outbox address to send spams. In this scenario, all 214 emails address are different, and each email address were spammed by compromised machine receiving floods emails from fraud email address.

I looked up one targeted email address, and found that the compromised machine has pretended to be other users to send floods emails to the recipients. The screenshot is as follow.



src_ip	dst_ip	_time	info
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.582	1443 → 587 [ACK] Seq=1 Ack=1 Win=64240 Len=0
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.584	[TCP Dup ACK 5017#1] 1443 → 587 [ACK] Seq=1 Ack=1 Win=64240 Len=0
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.592	C: EHLO sara.matthews6
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.593	[TCP Retransmission] 1443 → 587 [PSH, ACK] Seq=1 Ack=423 Win=63818 Len=21
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.601	C: AUTH LOGIN
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.602	[TCP Retransmission] 1443 → 587 [PSH, ACK] Seq=22 Ack=632 Win=63609 Len=12
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.608	C: User: c2FyYS5tYXR0aGV3czY=
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.609	[TCP Retransmission] 1443 → 587 [PSH, ACK] Seq=34 Ack=650 Win=63591 Len=22
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.614	C: Pass: dHl4dXF1dw==
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.615	[TCP Retransmission] 1443 → 587 [PSH, ACK] Seq=56 Ack=668 Win=63573 Len=14
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.624	C: MAIL FROM: <sara.matthews6@aol.com>
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.625	[TCP Retransmission] 1443 → 587 [PSH, ACK] Seq=70 Ack=705 Win=63536 Len=37
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.632	C: RCPT TO: <kristenlavigne@yahoo.com>
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.633	[TCP Retransmission] 1443 → 587 [PSH, ACK] Seq=107 Ack=719 Win=63522 Len=37
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.641	C: DATA
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.642	[TCP Retransmission] 1443 → 587 [PSH, ACK] Seq=144 Ack=733 Win=63508 Len=6
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.651	C: DATA fragment, 990 bytes
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.652	[TCP Retransmission] 1443 → 587 [PSH, ACK] Seq=150 Ack=770 Win=63471 Len=990
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.657	from: "Toka Chilcutt" <sara.matthews6@aol.com>, subject: RE:YouNedMedsAndThePresciptionsAreAvailable , (text/html)
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.658	[TCP Retransmission] 1443 → 587 [PSH, ACK] Seq=1140 Ack=770 Win=63471 Len=5
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.672	1443 → 587 [ACK] Seq=1145 Ack=808 Win=63433 Len=0
202.166.84.165	64.12.168.40	2020/06/19 00:55:24.673	[TCP Dup ACK 5110#1] 1443 → 587 [ACK] Seq=1145 Ack=808 Win=63433 Len=0

Time	Narratives
00:55:23.278AM	SPAM attack start, the compromised machine (202.166.84.165) sent spam to the first recipient <nickandsonia@comcast.net>
01:01:12.900AM	SPAM attack ended, the compromised machine (202.166.84.165) sent spam to the last recipient <jberman1@gmail.com>

Table2. SPAM Narratives

Consequence: A large amount of SPAM will reduce the availability of the mail system. First, SPAMs will take up network bandwidth. Causes mail server congestion, thereby reducing the operating efficiency of the entire network. Secondly, normal emails are overwhelmed by a large number of SPAMs, and users are likely to miss important emails in the process of handling business, which will affect the benefit of the enterprise. Besides, SPAM usually contains viruses or click fraud, posing a threat to the security of the mail system.

3) ClickFraud

Click fraud is when someone or robot pretends to be a legitimate visitor on a web page and then clicks on an advertisement, button or some other type of hyperlink. The purpose of click fraud is to gain more benefit by deceiving platforms or services that actual users are interacting with web pages, advertisements or apps.

The given data we cannot actually distinguish between click fraud and genuine clicks, so we can consider these datasets all clicks as fraud.

Time	Narratives
00:55:22.906AM	Compromised machine (202.166.84.165) started ClickFraud attack via different source ports to make GET requests to target website (98.126.71.122).
01:01:08.208AM	Compromised machine (202.166.84.165) ended ClickFraud attack.

Table3. ClickFraud Narratives

Consequence: I think click fraud will cause integrity problem. Since the data is "valid" in that it's a click, but not a click from a real user, which will generate wrong data. However, in reality, some companies apply click fraud attack on internet "pay-per-click" advertisements and obtain profits through falsified data. In this case, all business models based on pay-per-click will be affected, which will cause vicious competition in this industry.

4) IRC

It is another type of C&C attack that botnets communicate via Internet Relay Chat (IRC). The C&C server is centralized, which can provide resources for a single client request. A bot gets some instructions from the IRC channel (controlled by the bot master). In the following narratives, the botnet used an IRC based C&C channel to make requests to different 17 IRC servers.

Time	Narratives
00:55:21.813AM	Compromised Machine (202.166.84.165) started attack by generating a POST request to victim server (200.171.4.222).
01:01:59.756AM	Compromised Machine (202.166.84.165) ended IRC attack to victim server (58.42.247.143).

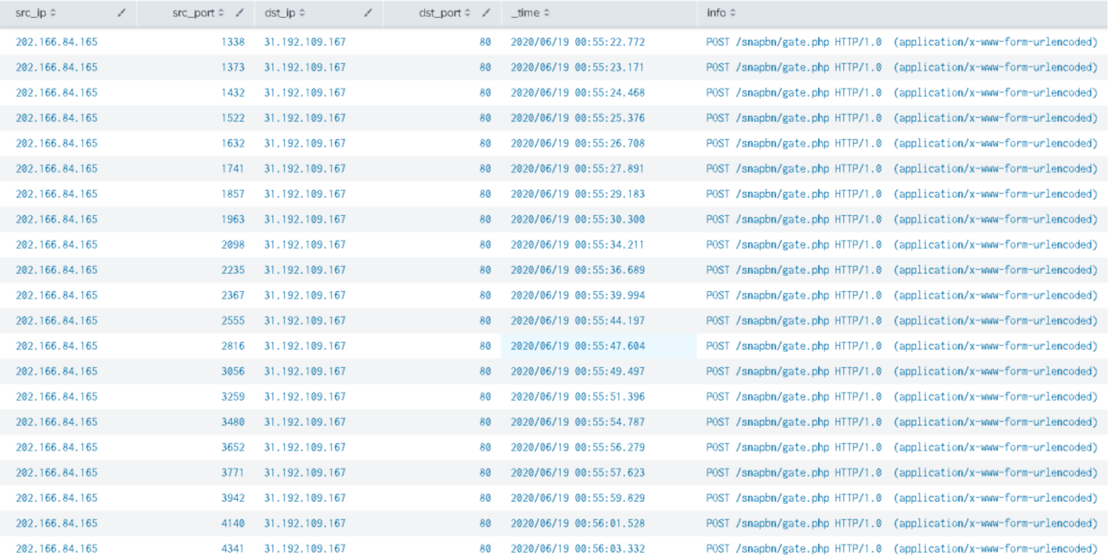
Table4. IRC Narratives

Consequence: Due to this type of botnet attack can also be seen as C2 attack, so it almost has the same consequence with C2 attack based on the HTTP. For example, IRC channels are increasingly being hit with DoS attacks which makes their ISPs to be terminated, and in this case, IRC servers had to shut down and it absolutely caused the unavailable of service.

4. Approaches for Extracting Features (Q5)

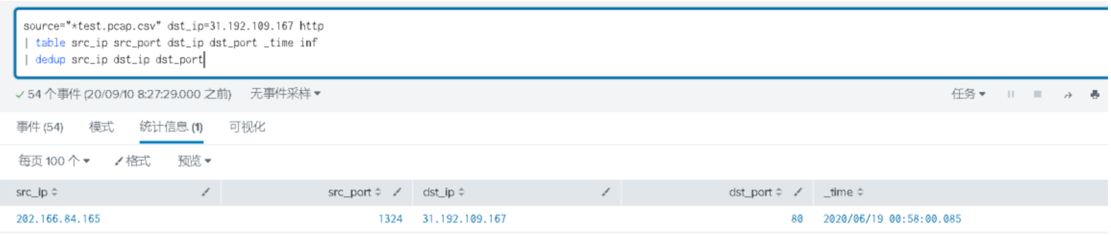
➢ Pattern 1- C2: srcIP + dstIp + dstPrt (80) + URI(*/snapbn/gate.php)*

Based on the data extracted from table, the feature of C2 in this scenario has the same source and destination IP address and destination port, even the same URI. I supposed that it is because when a C2 attack based on HTTP happened, hacker instructs a compromised machine to launch a denial of service attack against a specific target system, and based on port 80 (due to HTTP service).



src_ip	src_port	dst_ip	dst_port	_time	info
202.166.84.165	1338	31.192.109.167	80	2020/06/19 00:55:22.772	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	1373	31.192.109.167	80	2020/06/19 00:55:23.171	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	1432	31.192.109.167	80	2020/06/19 00:55:24.468	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	1522	31.192.109.167	80	2020/06/19 00:55:25.376	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	1632	31.192.109.167	80	2020/06/19 00:55:26.708	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	1741	31.192.109.167	80	2020/06/19 00:55:27.891	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	1857	31.192.109.167	80	2020/06/19 00:55:29.183	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	1963	31.192.109.167	80	2020/06/19 00:55:30.300	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	2098	31.192.109.167	80	2020/06/19 00:55:34.211	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	2235	31.192.109.167	80	2020/06/19 00:55:36.689	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	2367	31.192.109.167	80	2020/06/19 00:55:39.994	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	2555	31.192.109.167	80	2020/06/19 00:55:44.197	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	2816	31.192.109.167	80	2020/06/19 00:55:47.604	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	3056	31.192.109.167	80	2020/06/19 00:55:49.497	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	3259	31.192.109.167	80	2020/06/19 00:55:51.396	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	3480	31.192.109.167	80	2020/06/19 00:55:54.787	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	3652	31.192.109.167	80	2020/06/19 00:55:56.279	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	3771	31.192.109.167	80	2020/06/19 00:55:57.623	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	3942	31.192.109.167	80	2020/06/19 00:55:59.829	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	4140	31.192.109.167	80	2020/06/19 00:56:01.528	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)
202.166.84.165	4341	31.192.109.167	80	2020/06/19 00:56:03.332	POST /snapbn/gate.php HTTP/1.0 (application/x-www-form-urlencoded)

Also, there is an evidence by Splunk filtering, when we delete the duplicate items, the combination of source IP address, destination address and destination port can properly fit to one item. And actually, port 80 has represented the protocol HTTP, so we can consider this pattern as C2 feature.



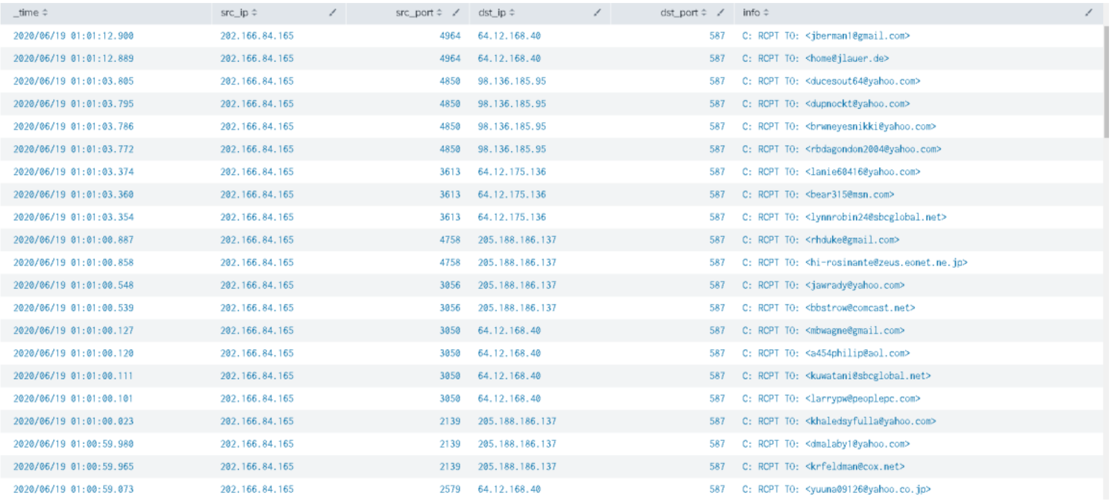
```
source="*test.pcap.csv" dst_ip=31.192.109.167 http
| table src_ip src_port dst_ip dst_port _time inf
| dedup src_ip dst_ip dst_port
```

✓ 54 个事件 (20/09/10 8:27:29.000 之前)

src_ip	src_port	dst_ip	dst_port	_time
202.166.84.165	1324	31.192.109.167	80	2020/06/19 00:58:00.085

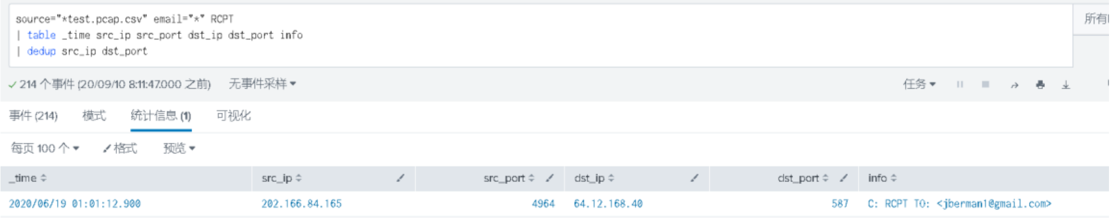
➢ Pattern 2- SPAM: srcIP + dstPrt (587) (+ Protocol (SMTP))

In most SPAM scenarios, a common source IP address can be observed since attackers try to send the emails to the whole network at once. And email service based on the protocol SMTP, which is on a common destination port number 587. So, we can consider the above pattern as SPAM features.



_time	src_ip	src_port	dst_ip	dst_port	info
2020/06/19 01:01:12.900	202.166.84.165	4964	64.12.168.40	587	C: RCPT TO: <jberman1@gmail.com>
2020/06/19 01:01:12.889	202.166.84.165	4964	64.12.168.40	587	C: RCPT TO: <home@jlauer.de>
2020/06/19 01:01:03.805	202.166.84.165	4850	98.136.185.95	587	C: RCPT TO: <ducesout64@yahoo.com>
2020/06/19 01:01:03.795	202.166.84.165	4850	98.136.185.95	587	C: RCPT TO: <dupnockt@yahoo.com>
2020/06/19 01:01:03.786	202.166.84.165	4850	98.136.185.95	587	C: RCPT TO: <brwneyesnikki@yahoo.com>
2020/06/19 01:01:03.772	202.166.84.165	4850	98.136.185.95	587	C: RCPT TO: <rbdagondon2004@yahoo.com>
2020/06/19 01:01:03.374	202.166.84.165	3613	64.12.175.136	587	C: RCPT TO: <lanie60416@yahoo.com>
2020/06/19 01:01:03.360	202.166.84.165	3613	64.12.175.136	587	C: RCPT TO: <bear315@msn.com>
2020/06/19 01:01:03.354	202.166.84.165	3613	64.12.175.136	587	C: RCPT TO: <lynnrobin24@sbcglobal.net>
2020/06/19 01:01:00.887	202.166.84.165	4758	205.188.186.137	587	C: RCPT TO: <rhduke@gmail.com>
2020/06/19 01:01:00.858	202.166.84.165	4758	205.188.186.137	587	C: RCPT TO: <hi-rosinante@zeus.eonet.ne.jp>
2020/06/19 01:01:00.548	202.166.84.165	3056	205.188.186.137	587	C: RCPT TO: <jawrady@yahoo.com>
2020/06/19 01:01:00.539	202.166.84.165	3056	205.188.186.137	587	C: RCPT TO: <bbstrow@comcast.net>
2020/06/19 01:01:00.127	202.166.84.165	3050	64.12.168.40	587	C: RCPT TO: <mbwagne@gmail.com>
2020/06/19 01:01:00.120	202.166.84.165	3050	64.12.168.40	587	C: RCPT TO: <a454philip@aol.com>
2020/06/19 01:01:00.111	202.166.84.165	3050	64.12.168.40	587	C: RCPT TO: <kuwatani@sbcglobal.net>
2020/06/19 01:01:00.101	202.166.84.165	3050	64.12.168.40	587	C: RCPT TO: <larrypw@peoplepc.com>
2020/06/19 01:01:00.023	202.166.84.165	2139	205.188.186.137	587	C: RCPT TO: <khaledsyfulla@yahoo.com>
2020/06/19 01:00:59.980	202.166.84.165	2139	205.188.186.137	587	C: RCPT TO: <dmalaby1@yahoo.com>
2020/06/19 01:00:59.965	202.166.84.165	2139	205.188.186.137	587	C: RCPT TO: <krfeldman@cox.net>
2020/06/19 01:00:59.073	202.166.84.165	2579	64.12.168.40	587	C: RCPT TO: <yuuna09126@yahoo.co.jp>

Also, there is an evidence by Splunk filtering, when we delete the duplicate items, the combination of source IP address and destination port can properly fit to one item.

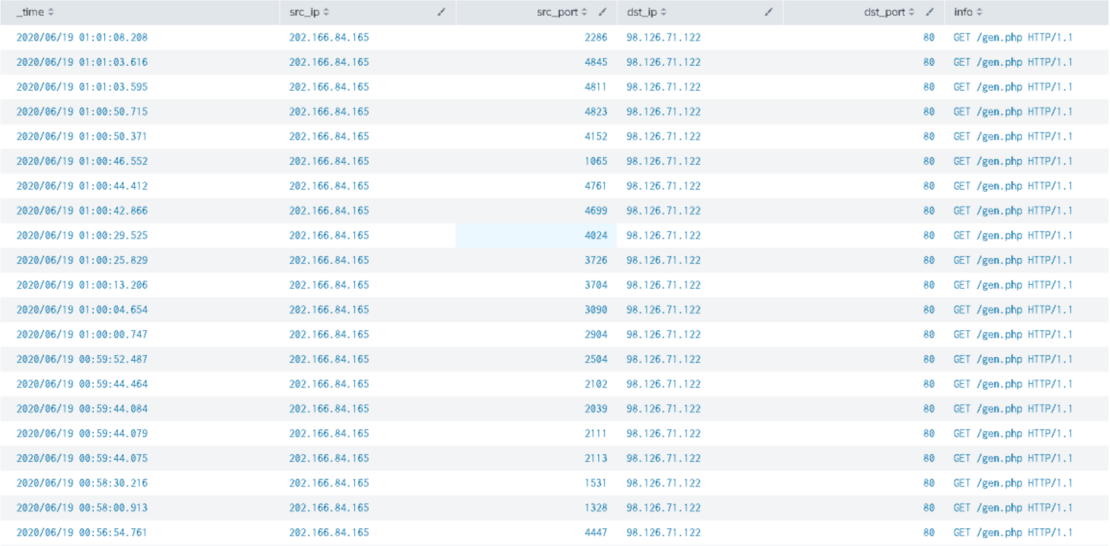


```
source="*test.pcap.csv" email="*" RCPT
| table _time src_ip src_port dst_ip dst_port info
| dedup src_ip dst_port
```

_time	src_ip	src_port	dst_ip	dst_port	info
2020/06/19 01:01:12.900	202.166.84.165	4964	64.12.168.40	587	C: RCPT TO: <jberman1@gmail.com>

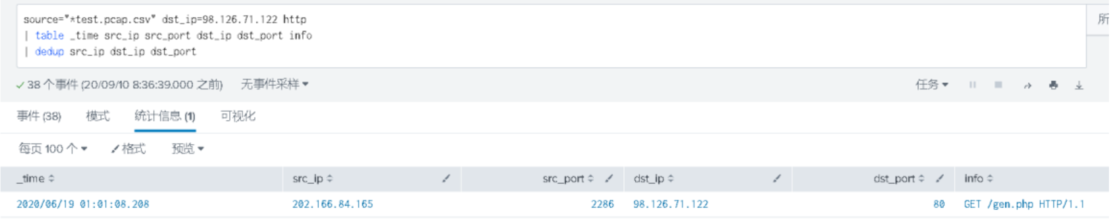
➢ Pattern 3- ClickFraud: srcIP + dstIP + dstPrt (80) (+ Protocol)

In most ClickFraud scenarios, a common source IP address can be observed since attackers try to floods the target website at once via different source ports. And requests service based on the protocol HTTP, which is on a common destination port number 80. So, we can consider the above pattern as ClickFraud features.



_time	src_ip	src_port	dst_ip	dst_port	info
2020/06/19 01:01:08.208	202.166.84.165	2286	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:01:03.616	202.166.84.165	4845	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:01:03.595	202.166.84.165	4811	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:00:50.715	202.166.84.165	4823	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:00:50.371	202.166.84.165	4152	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:00:46.552	202.166.84.165	1065	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:00:44.412	202.166.84.165	4761	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:00:42.866	202.166.84.165	4699	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:00:29.525	202.166.84.165	4024	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:00:25.829	202.166.84.165	3726	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:00:13.206	202.166.84.165	3704	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:00:04.654	202.166.84.165	3090	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 01:00:00.747	202.166.84.165	2904	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:59:52.487	202.166.84.165	2504	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:59:44.464	202.166.84.165	2102	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:59:44.084	202.166.84.165	2039	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:59:44.079	202.166.84.165	2111	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:59:44.075	202.166.84.165	2113	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:58:30.216	202.166.84.165	1531	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:58:00.913	202.166.84.165	1328	98.126.71.122	80	GET /gen.php HTTP/1.1
2020/06/19 00:56:54.761	202.166.84.165	4447	98.126.71.122	80	GET /gen.php HTTP/1.1

Also, there is an evidence by Splunk filtering, when we delete the duplicate items, the combination of source IP address, destination IP and destination port can properly fit to one item.



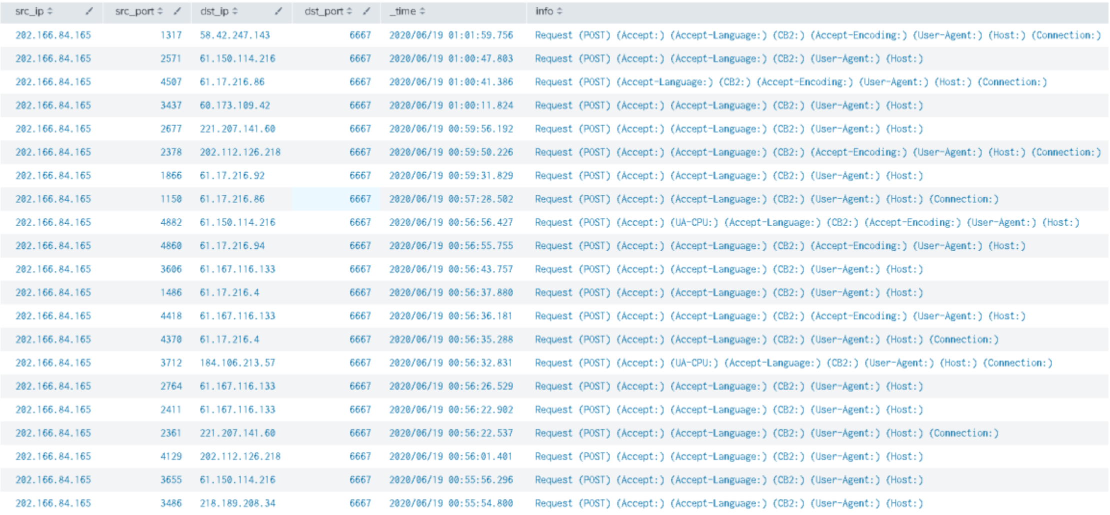
```
source="*test.pcap.csv" dst_ip=98.126.71.122 http
| table _time src_ip src_port dst_ip dst_port info
| dedup src_ip dst_ip dst_port
```

✓38 个事件 (20/09/10 8:36:39.000 之前)

_time	src_ip	src_port	dst_ip	dst_port	info
2020/06/19 01:01:08.208	202.166.84.165	2286	98.126.71.122	80	GET /gen.php HTTP/1.1

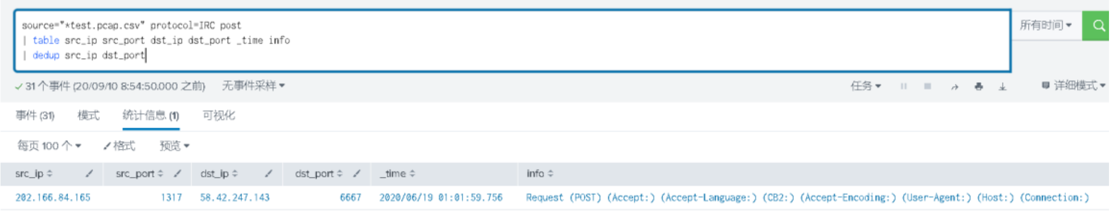
➢ Pattern 4- IRC: srcIP + dstPrt (6667) (+ Protocol)

In most IRC scenarios, a common source IP address can be observed since attackers try to floods the target website at once via different source ports. And POST requests service based on the protocol IRC, which is on a common destination port number 6667. So, we can consider the above pattern as IRC features.



src_ip	src_port	dst_ip	dst_port	_time	info
202.166.84.165	1317	58.42.247.143	6667	2020/06/19 01:01:59.756	Request (POST) (Accept:) (Accept-Language:) (CB2:) (Accept-Encoding:) (User-Agent:) (Host:) (Connection:)
202.166.84.165	2571	61.150.114.216	6667	2020/06/19 01:00:47.803	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:)
202.166.84.165	4507	61.17.216.86	6667	2020/06/19 01:00:41.386	Request (POST) (Accept-Language:) (CB2:) (Accept-Encoding:) (User-Agent:) (Host:) (Connection:)
202.166.84.165	3437	60.173.109.42	6667	2020/06/19 01:00:11.824	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:)
202.166.84.165	2677	221.207.141.60	6667	2020/06/19 00:59:56.192	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:)
202.166.84.165	2378	202.112.126.218	6667	2020/06/19 00:59:50.226	Request (POST) (Accept:) (Accept-Language:) (CB2:) (Accept-Encoding:) (User-Agent:) (Host:) (Connection:)
202.166.84.165	1866	61.17.216.92	6667	2020/06/19 00:59:31.829	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:)
202.166.84.165	1150	61.17.216.86	6667	2020/06/19 00:57:28.502	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:) (Connection:)
202.166.84.165	4882	61.150.114.216	6667	2020/06/19 00:56:56.427	Request (POST) (Accept:) (UA-CPU:) (Accept-Language:) (CB2:) (Accept-Encoding:) (User-Agent:) (Host:)
202.166.84.165	4860	61.17.216.94	6667	2020/06/19 00:56:55.755	Request (POST) (Accept:) (Accept-Language:) (CB2:) (Accept-Encoding:) (User-Agent:) (Host:)
202.166.84.165	3606	61.167.116.133	6667	2020/06/19 00:56:43.757	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:)
202.166.84.165	1486	61.17.216.4	6667	2020/06/19 00:56:37.880	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:)
202.166.84.165	4418	61.167.116.133	6667	2020/06/19 00:56:36.181	Request (POST) (Accept:) (Accept-Language:) (CB2:) (Accept-Encoding:) (User-Agent:) (Host:)
202.166.84.165	4370	61.17.216.4	6667	2020/06/19 00:56:35.288	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:) (Connection:)
202.166.84.165	3712	184.106.213.57	6667	2020/06/19 00:56:32.831	Request (POST) (Accept:) (UA-CPU:) (Accept-Language:) (CB2:) (User-Agent:) (Host:) (Connection:)
202.166.84.165	2764	61.167.116.133	6667	2020/06/19 00:56:26.529	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:)
202.166.84.165	2411	61.167.116.133	6667	2020/06/19 00:56:22.902	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:)
202.166.84.165	2361	221.207.141.60	6667	2020/06/19 00:56:22.537	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:) (Connection:)
202.166.84.165	4129	202.112.126.218	6667	2020/06/19 00:56:01.401	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:)
202.166.84.165	3655	61.150.114.216	6667	2020/06/19 00:55:56.296	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:)
202.166.84.165	3486	218.189.208.34	6667	2020/06/19 00:55:54.800	Request (POST) (Accept:) (Accept-Language:) (CB2:) (User-Agent:) (Host:)

Also, there is an evidence by Splunk filtering, when we delete the duplicate items, the combination of source IP address, destination IP and destination port can properly fit to one item.



```
source="*test.pcap.csv" protocol=IRC post
| table src_ip src_port dst_ip dst_port _time info
| dedup src_ip dst_port
```

✓31 个事件 (20/09/10 8:54:50.000 之前)

src_ip	src_port	dst_ip	dst_port	_time	info
202.166.84.165	1317	58.42.247.143	6667	2020/06/19 01:01:59.756	Request (POST) (Accept:) (Accept-Language:) (CB2:) (Accept-Encoding:) (User-Agent:) (Host:) (Connection:)

Besides, for all above patterns, I think we can also consider a time interval as a feature to detect such attacks, since it is impossible for a human conduct to make more than 200 requests in a definitely short time, like 5 mins. Once we notice this suspicious phenomenon, we can pay attention on this abnormal data. However, I can not make a precious pattern of time interval for there aren't enough dataset to draw a conclusion, and it seems we need to make further analyse by the help of Machine Learning.

5. Proposed Countermeasures (Q6)

To detect or prevent from detecting the above four type attacks, we can take following steps to deal with them:

Firstly, port scanning regularly to existing network master nodes, check out possible security vulnerabilities by applying attack pattern finding in #5. For C2 attacks, the computer of the master node is the best location for hackers to use because of its high bandwidth. Therefore, it is very important to strengthen the security of these hosts. Besides, by interrupting the communication channel with the command and control server and having visibility into suspicious traffic, companies can consider blocking the most advanced malware.

Secondly, an effective way to prevent attack intrusion is to focus on disrupting communications with command and control nodes. When facing spamming and click fraud, it is unrealistic to prevent malware from gaining a foothold in the enterprise because users will inevitably click on email attachments or links, leading to infection. Based on pattern detection, sometimes it is not applicable in rare cases. Security resources should focus on preventing malicious software from communicating with the command and control server, thereby effectively eliminating the kill chain. To this end, all companies should observe basic safety practices。

Thirdly, it is necessary to configure powerful egress firewall rules to restrict all traffic except the external Web traffic of the enterprise. This will stop automatic beacons via non-standard ports and protocols (such as dynamic DNS). Some malware uses a web port to communicate it back to the command and control location.

Besides, enterprises need to be able to trigger alerts for suspicious traffic. This can be done by creating alerts in the security information and event management platform, or it can be done from the tool's management console based on technology. And the network administrator should provide a 24x7x365 response.

6. Conclusion

There still many other types of attacks except above 4 scenarios, for example, the crazy number of DNS requests in the latter stages of the .pcap file out to .ru (Russian) servers, however, I cannot figure out the exactly larger picture of the whole attack due to some missed key field in Splunk, and

I spent much time to dig out the rabbit holes but still have no idea what to do with a fact that an infected machine sent a MAC address to some server with a suspicious domain name, and the MAC transmitted in not compromised machine's one, but somehow associated with it through a series of ARP communication.

I also find some suspicious IP address regarding 212.117.171.138, and 174.133.57.141; 173.192.170.88 which made several transactions to over 200 different ports, but not sure about how does the port scan work.

Maybe in the next stage, I can figure them out with the help of ML.

Reference

1. Eric Chou and Rich Groves, 2016, Distributed Denial of Service, O'Reilly Media, Inc.
2. Joseph Gardiner, Marco Cova, Shishir Nagaraja, 2014, Command & Control Understanding, Denying and Detecting, In collaboration with Lastline, Inc.
3. MICHELLE DELIO, 2001, IRC: Attack From Killer 'HaX0rZ', MICHELLE DELIO, https://www.wired.com/2001/01/irc-attack-from-killer-hax0rz/.
4. FireEye. (2015, July). HAMMERTOSS: Stealthy Tactics Define a Russian Cyber Threat Group. Retrieved March 6, 2017.
5. Adam RiceJames Ringold, 2018, Command-and-control servers: The puppet masters that govern malware, https://searchsecurity.techtarget.com/feature/Command-and-control-servers-The-puppet-masters-that-govern-malware