Hunting beacons

Bartosz Jerzman

agenda

Part I: HTTP beacon detection

Part II: HTTPS beacon detection

Part III: Let’s hunt them early – C2 scanning

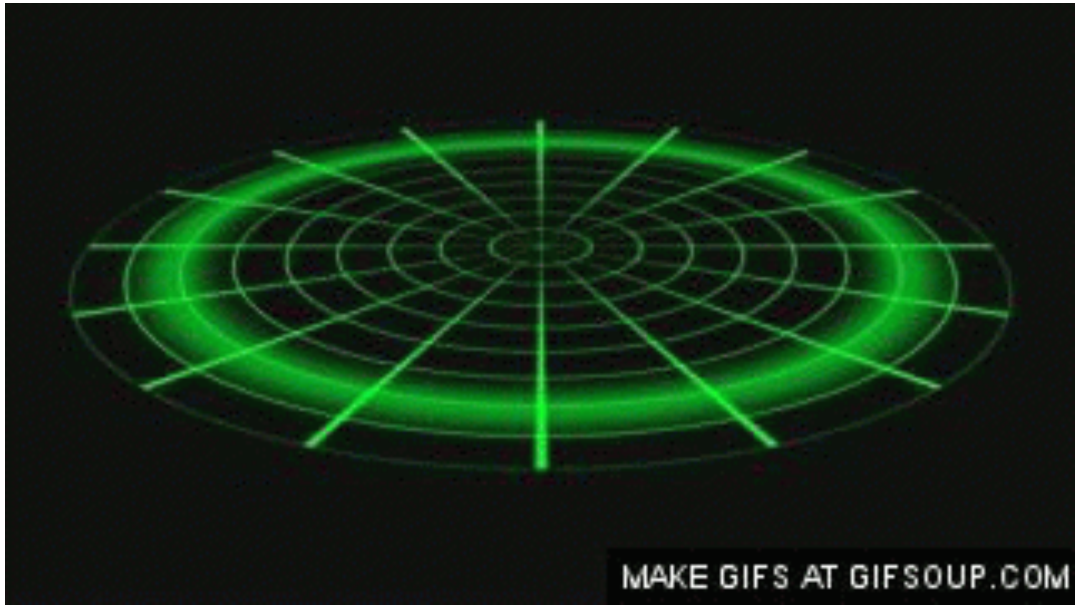
whoami

- Sysadmin and network defender for the Polish Navy
- Incident responder
- Pentester
- Cyber threat intelligence analyst & adversary hunter
- @secman_pl

PART I
Beaconing over **HTTP**

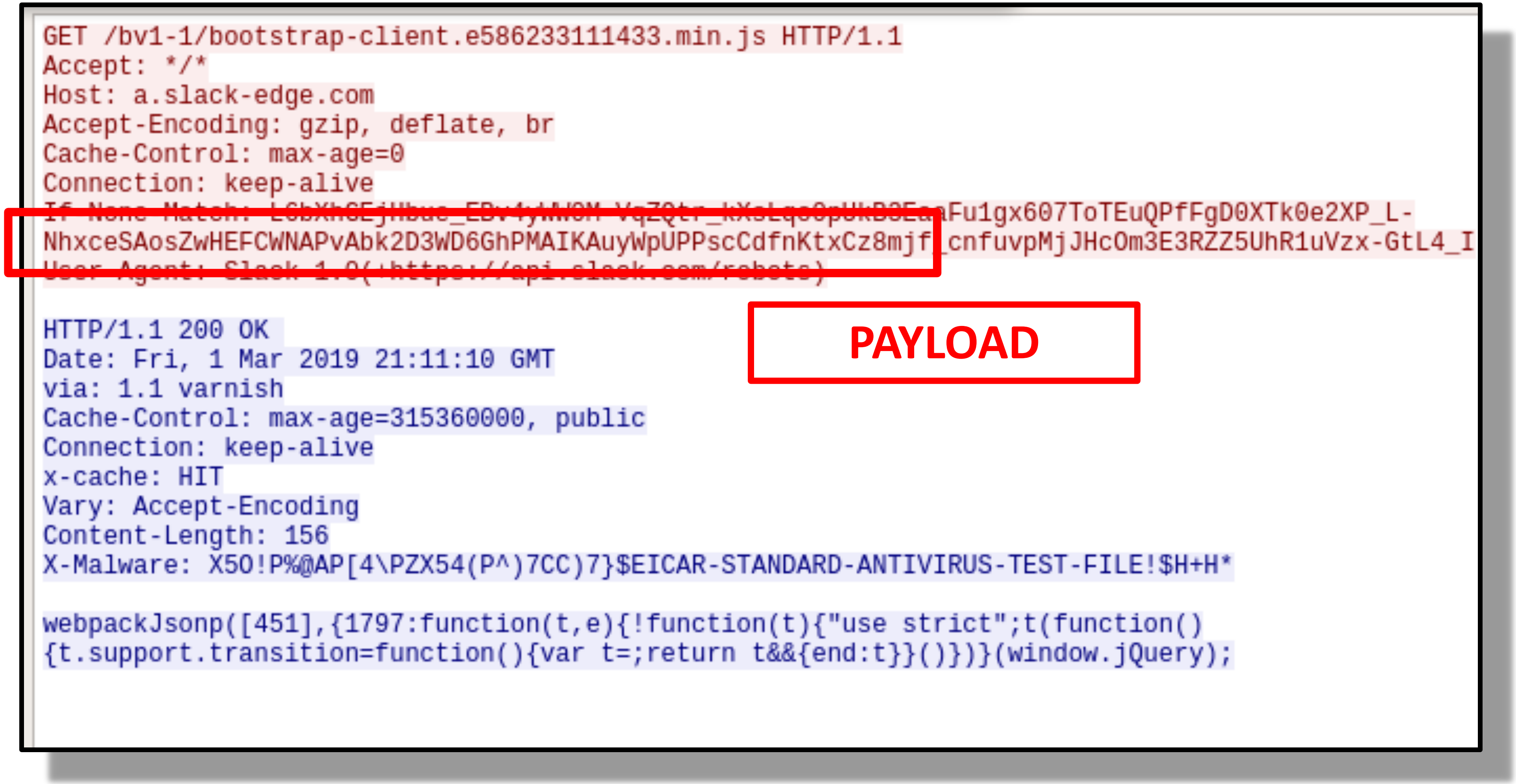
What is beaconing?

- Malware does **not keep long connection** to C2
- Malware connects to C2 **periodically**
- Beaconing can occur regularly at **constant intervals**
- Or it can occur at **pseudorandom** moments of time



Time for x33fcon 2019 most popular meme

Signature matching for beaconing?



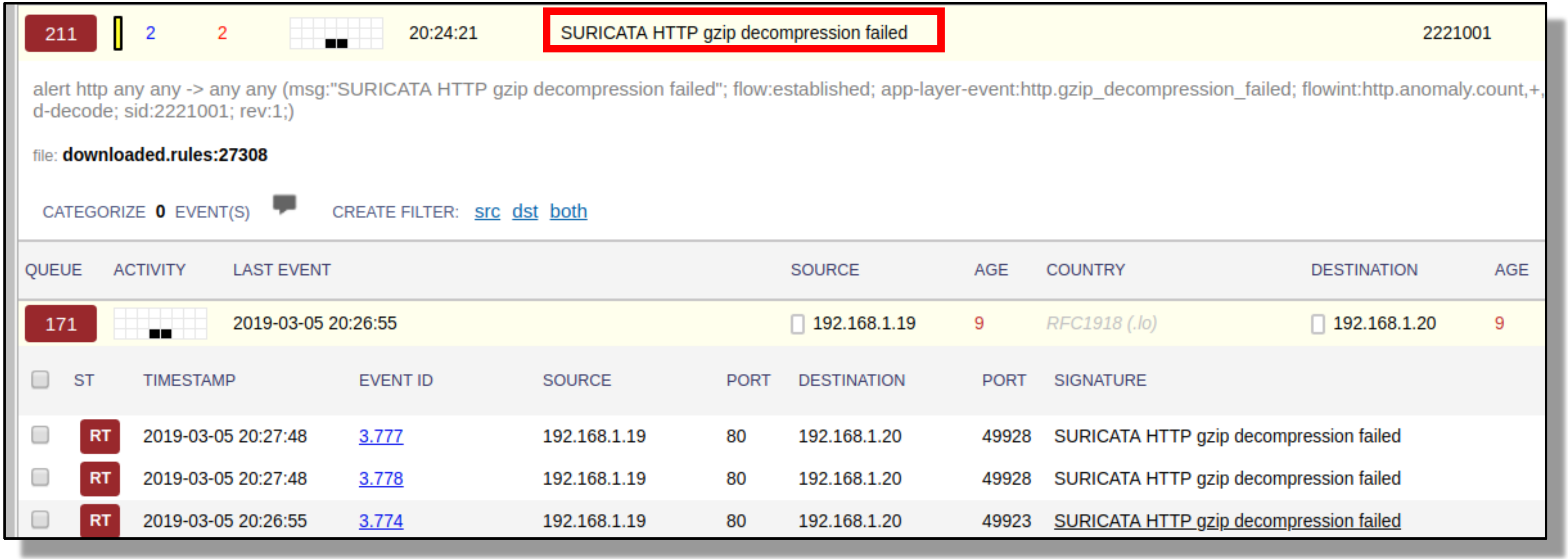
```
GET /bv1-1/bootstrap-client.e586233111433.min.js HTTP/1.1
Accept: */*
Host: a.slack-edge.com
Accept-Encoding: gzip, deflate, br
Cache-Control: max-age=0
Connection: keep-alive
If-None-Match: L6bXhGEjHbue_EBv4yWWOM-VqZQtr_kXsLqs0pUkB3EaaFu1gx607ToTEuQPfFgD0XTk0e2XP_L-
NhxceSAosZwHEFCWNAPvAbk2D3WD6GhPMAIKAuyWpUPPscCdfnKtxCz8mjf_cnfuvpMjJHcOm3E3RZZ5UhR1uVzx-GtL4_I
User-Agent: Slack 1.0(+https://api.slack.com/robots)

HTTP/1.1 200 OK
Date: Fri, 1 Mar 2019 21:11:10 GMT
via: 1.1 varnish
Cache-Control: max-age=315360000, public
Connection: keep-alive
x-cache: HIT
Vary: Accept-Encoding
Content-Length: 156
X-Malware: X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*

webpackJsonp([451],{1797:function(t,e){!function(t){"use strict";t(function()
{t.support.transition=function(){var t=;return t&&{end:t}}()})}(window.jQuery);
```

Cobalt Strike beacon traffic simulating Slack communication

Would your SOC escalate on this?



211	2	2	20:24:21	SURICATA HTTP gzip decompression failed	2221001

```
alert http any any -> any any (msg:"SURICATA HTTP gzip decompression failed"; flow:established; app-layer-event:http.gzip_decompression_failed; flowint:http.anomaly.count,+,
d-decode; sid:2221001; rev:1;)
```

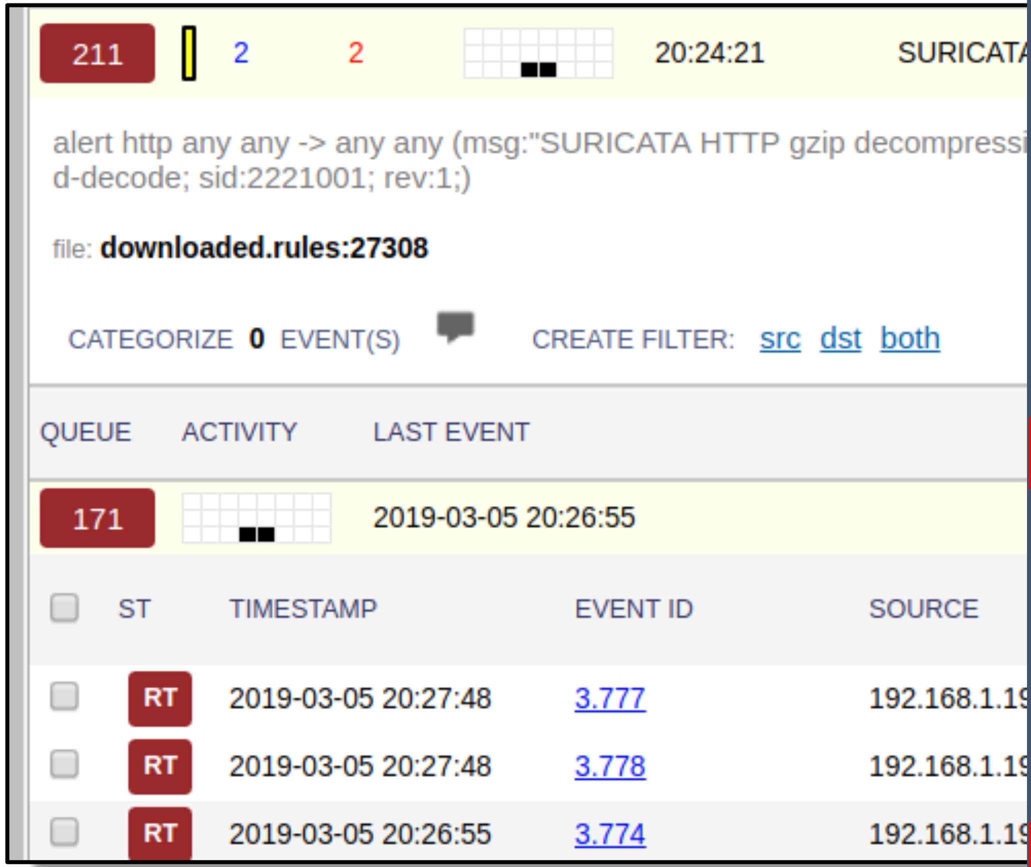
file: **downloaded.rules:27308**

CATEGORIZE **0** EVENT(S) CREATE FILTER: src dst both

QUEUE	ACTIVITY	LAST EVENT	SOURCE	AGE	COUNTRY	DESTINATION	AGE
171		2019-03-05 20:26:55	□ 192.168.1.19	9	*RFC1918 (.lo)*	□ 192.168.1.20	9

□	ST	TIMESTAMP	EVENT ID	SOURCE	PORT	DESTINATION	PORT	SIGNATURE
□	RT	2019-03-05 20:27:48	3.777	192.168.1.19	80	192.168.1.20	49928	SURICATA HTTP gzip decompression failed
□	RT	2019-03-05 20:27:48	3.778	192.168.1.19	80	192.168.1.20	49928	SURICATA HTTP gzip decompression failed
□	RT	2019-03-05 20:26:55	3.774	192.168.1.19	80	192.168.1.20	49923	SURICATA HTTP gzip decompression failed

Would your SOC e



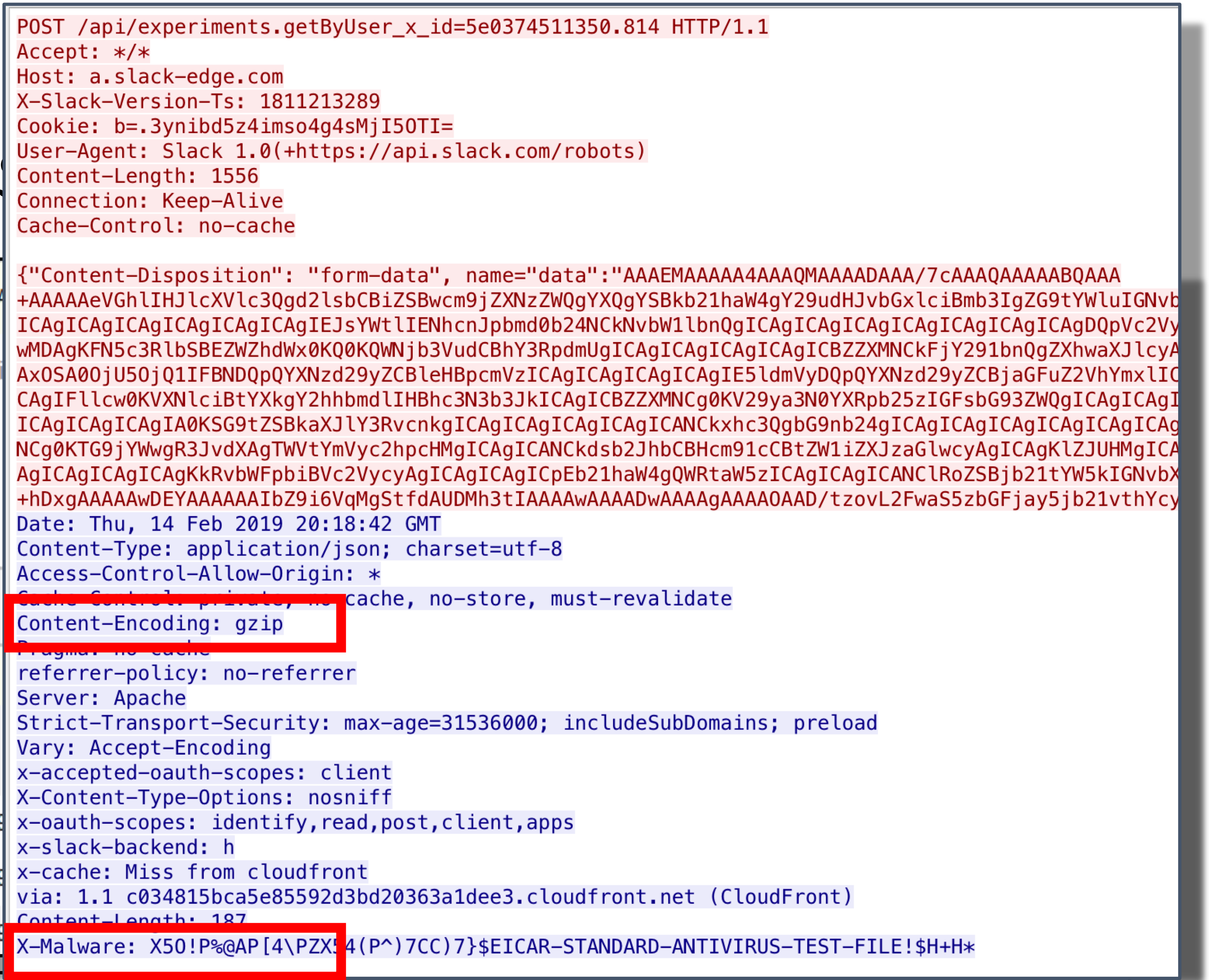
211 | 2 | 2 | 20:24:21 | SURICATA

alert http any any -> any any (msg:"SURICATA HTTP gzip decompressi
d-decode; sid:2221001; rev:1;)

file: **downloaded.rules:27308**

QUEUE	ACTIVITY	LAST EVENT
171		2019-03-05 20:26:55

	ST	TIMESTAMP	EVENT ID	SOURCE
□	RT	2019-03-05 20:27:48	3.777	192.168.1.19
□	RT	2019-03-05 20:27:48	3.778	192.168.1.19
□	RT	2019-03-05 20:26:55	3.774	192.168.1.19



```
POST /api/experiments.getByUser_x_id=5e0374511350.814 HTTP/1.1
Accept: */*
Host: a.slack-edge.com
X-Slack-Version-Ts: 1811213289
Cookie: b=.3ynibd5z4imso4g4sMjI5OTI=
User-Agent: Slack 1.0(+https://api.slack.com/robots)
Content-Length: 1556
Connection: Keep-Alive
Cache-Control: no-cache

{"Content-Disposition": "form-data", name="data":"AAAEMAAAAA4AAAQMAAAADAAA/7cAAAQAAAAABQAAA
+AAAAAeVGhlIHJlcXVlc3Qgd2lsbCBiZSBwcm9jZXNzZWQgYXQgYSBkb21haW4gY29udHJvbGxlciBmb3IgZG9tYWluIGNvb
ICAgICAgICAgICAgICAgICAgIEJsYWtlIENhcnJpbmd0b24NCkNvbW1lbnQgICAgICAgICAgICAgICAgICAgICAgDQpVc2Vy
wMDAgKFN5c3RlbSBEZWZhdWx0KQ0KQWNjb3VudCBhY3RpdmUgICAgICAgICAgICAgICBZZXMNCkFjY291bnQgZXhwaXJlcyA
AxOSA0OjU5OjQ1IFBNDQpQYXNzd29yZCBleHBpcmVzICAgICAgICAgICAgIE5ldmVyDQpQYXNzd29yZCBjaGFuZ2VhYmxlIC
CAgIFllcw0KVXNlciBtYXkgY2hhbmdlIHBhc3N3b3JkICAgICBZZXMNCg0KV29ya3N0YXRpb25zIGFsbG93ZWQgICAgICAgI
ICAgICAgICAgIA0KSG9tZSBkaXJlY3RvcnkgICAgICAgICAgICANCkxhc3QgbG9nb24gICAgICAgICAgICAgICAgICAgICAg
NCg0KTG9jYWwgR3JvdXAgTWVtYmVyc2hpcHMgICAgICANCkdsb2JhbCBHcm91cCBtZW1iZXJzaGlwcyAgICAgKlZJUHMgICA
AgICAgICAgICAgKkRvbWFpbiBVc2VycyAgICAgICAgICpEb21haW4gQWRtaW5zICAgICAgICANClRoZSBjb21tYW5kIGNvbX
+hDxgAAAAAwDEYAAAAAAIbZ9i6VqMgStfdAUDMh3tIAAAAwAAAADwAAAAgAAAAOAAD/tzovL2FwaS5zbGFjay5jb21vthYcy
Date: Thu, 14 Feb 2019 20:18:42 GMT
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Cache-Control: private, no-cache, no-store, must-revalidate
Content-Encoding: gzip
Pragma: no-cache
referrer-policy: no-referrer
Server: Apache
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
Vary: Accept-Encoding
x-accepted-oauth-scopes: client
X-Content-Type-Options: nosniff
x-oauth-scopes: identify,read,post,client,apps
x-slack-backend: h
x-cache: Miss from cloudfront
via: 1.1 c034815bca5e85592d3bd20363a1dee3.cloudfront.net (CloudFront)
Content-Length: 187
X-Malware: X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

IDS detected that HTTP response body **is not gzipped** as it has been declared in the response headers.

Set of hipothesis:

#1: analysis of intervals of connections

#2: same URI for different Host names

#3: same or none Referrer to many URIs

#4: different URIs but length is constant

Dataset:

- Data from Cyber Defence Excercise: **„Locked Shields”**
- PCAP -> processed by BRO-IDS/ZEEK -> http.log
- Example of data from **http.log**
- Alternative data sources: flows, webproxy logs

srcIP	srcPort	dstIP	dstPort	method	host	uri	user_agent	Req_body_length	Resp_body_length	cookie
10.18.7.3	50474	39.88.160[.]18	80	POST	test.com	/test.php	Mozilla/5.0 (Windows NT 6.1; WOW64)	0	303	Trackr=eDMzZmNvbg==

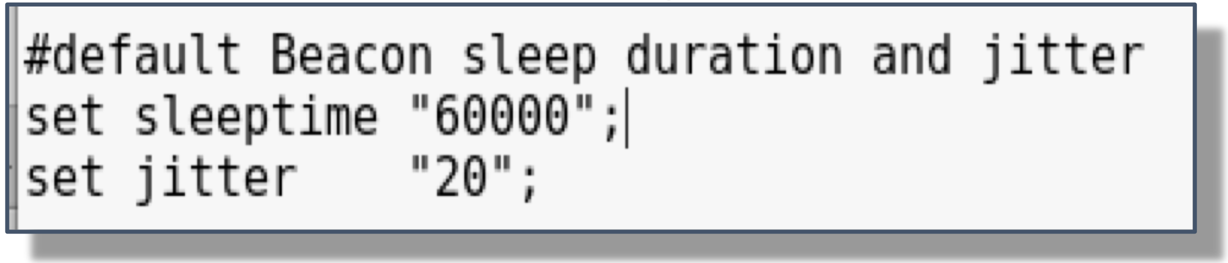
Hipothesis #1: analysis of connections intervals

Assumption: Connection intervals from malware to C2 server are distributed around some average value.

WHY?

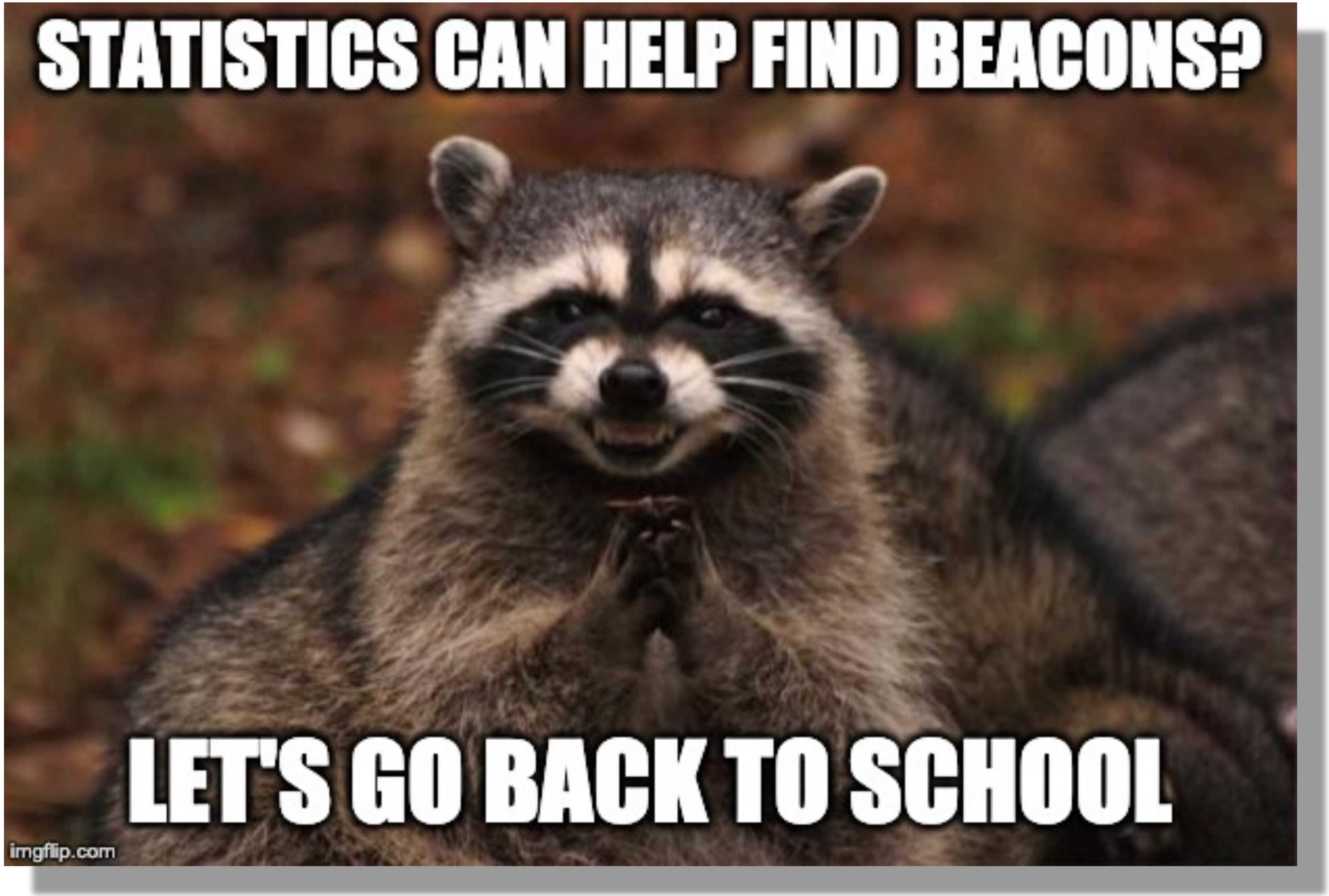
Beaconing malware often has configuration options for setting:

- **sleep** time
- **jitter** (variations from central value)

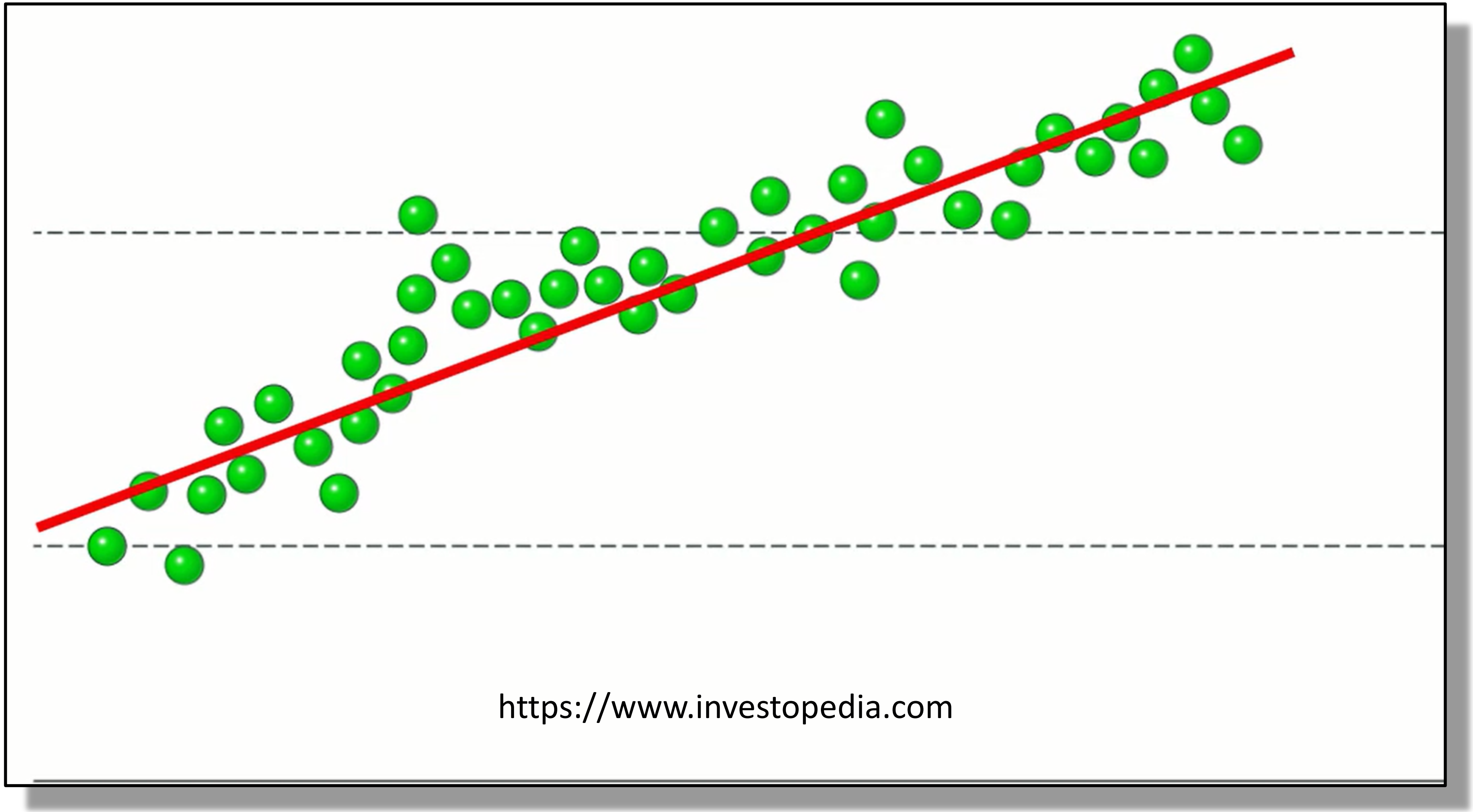


```
#default Beacon sleep duration and jitter
set sleeptime "60000";
set jitter    "20";
```

Hipothesis #1: analysis of connections intervals



Hipothesis #1: analysis of connections intervals



https://www.investopedia.com

Hipothesis #1: analysis of connections intervals

Beacon A: Cobalt Strike payload with configuration{ **60 s sleep, 20% jitter** }

Beacon B: Cobalt Strike payload with **manual sleep** commands from operator

Beacon	#1	#2	#3	#4	#5	#6	AVG	STDDEV	Variation Coefficient
A	48s	51s	62s	69s	55s	60s	**57,5s**	**+/- 7,75 s**	**13,4 %**
B	1s	2s	100s	14s	70s	27s	**35,7s**	**+/- 40,5 s**	**113,5 %**

Hipothesis #1: analysis of connections intervals

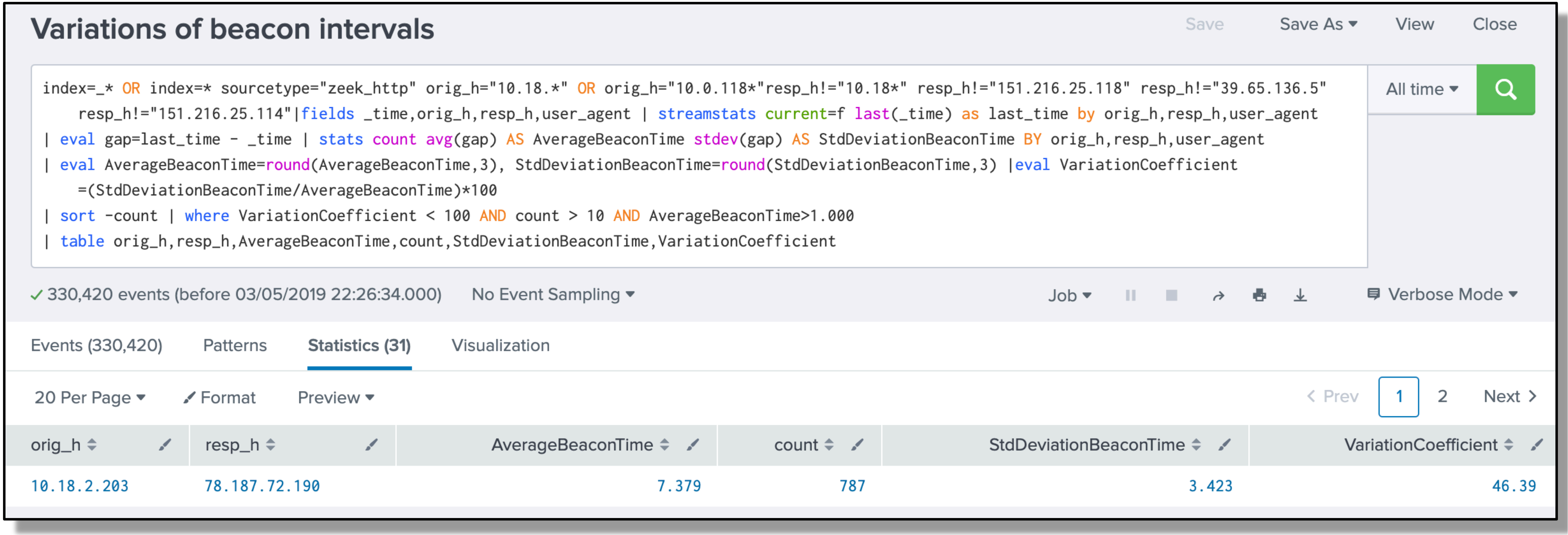
Beacon n{ **60 s sleep, 20% jitter** }

Beacon **ep** commands from operator

$$\textbf{Var. Coeff.} = \frac{STDDEV}{AVG} * 100\%$$

Beacon	#1	#2	#3	#4	#5	#6	AVG	STDDEV	Variation Coefficient
A	48s	51s	62s	69s	55s	60s	**57,5s**	**+/- 7,75 s**	**13,4 %**
B	1s	2s	100s	14s	70s	27s	**35,7s**	**+/- 40,5 s**	**113,5 %**

Hipothesis #1: analysis of connections intervals



Variations of beacon intervals

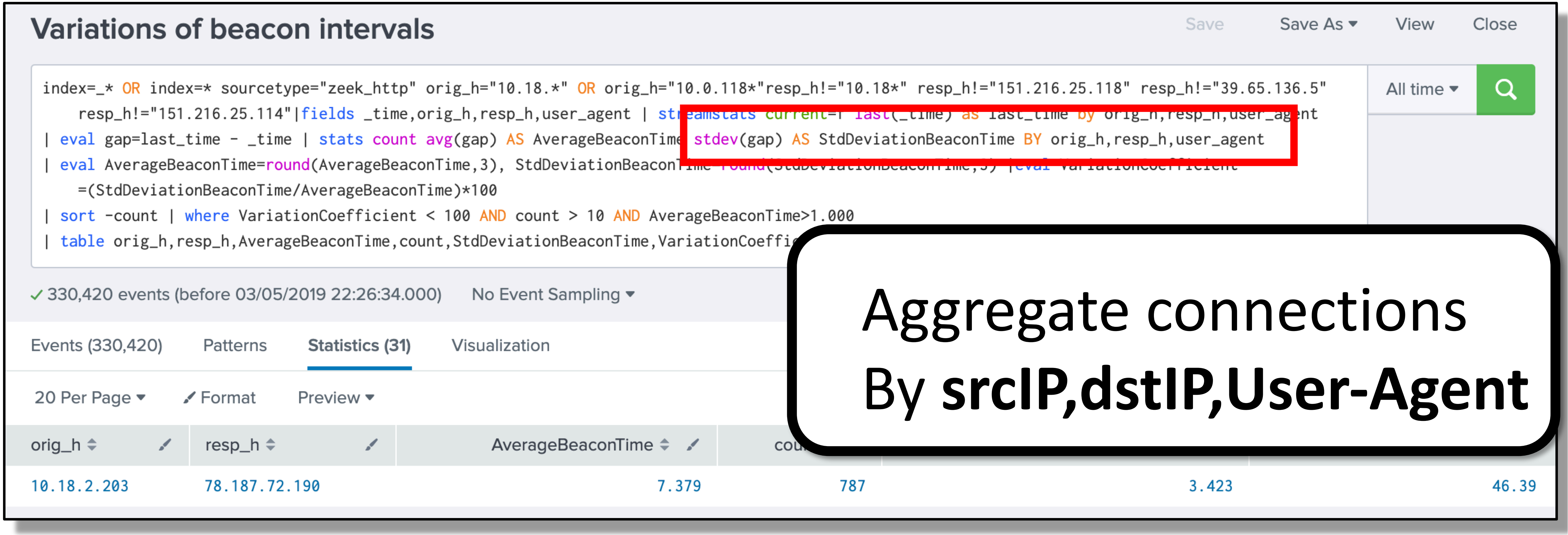
```
index=_* OR index=* sourcetype="zeek_http" orig_h="10.18.*" OR orig_h="10.0.118*"resp_h!="10.18*" resp_h!="151.216.25.118" resp_h!="39.65.136.5"
    resp_h!="151.216.25.114"|fields _time,orig_h,resp_h,user_agent | streamstats current=f last(_time) as last_time by orig_h,resp_h,user_agent
| eval gap=last_time - _time | stats count avg(gap) AS AverageBeaconTime stdev(gap) AS StdDeviationBeaconTime BY orig_h,resp_h,user_agent
| eval AverageBeaconTime=round(AverageBeaconTime,3), StdDeviationBeaconTime=round(StdDeviationBeaconTime,3) |eval VariationCoefficient
    =(StdDeviationBeaconTime/AverageBeaconTime)*100
| sort -count | where VariationCoefficient < 100 AND count > 10 AND AverageBeaconTime>1.000
| table orig_h,resp_h,AverageBeaconTime,count,StdDeviationBeaconTime,VariationCoefficient
```

✓ 330,420 events (before 03/05/2019 22:26:34.000)

orig_h	resp_h	AverageBeaconTime	count	StdDeviationBeaconTime	VariationCoefficient
10.18.2.203	78.187.72.190	7.379	787	3.423	46.39

Query inspired by: https://www.splunk.com/blog/2018/03/20/hunting-your-dns-dragons.html

Hipothesis #1: analysis of connections intervals



```
index=_* OR index=* sourcetype="zeek_http" orig_h="10.18.*" OR orig_h="10.0.118*"resp_h!="10.18*" resp_h!="151.216.25.118" resp_h!="39.65.136.5"
    resp_h!="151.216.25.114"|fields _time,orig_h,resp_h,user_agent | streamstats current=f last(_time) as last_time by orig_h,resp_h,user_agent
| eval gap=last_time - _time | stats count avg(gap) AS AverageBeaconTime stdev(gap) AS StdDeviationBeaconTime BY orig_h,resp_h,user_agent
| eval AverageBeaconTime=round(AverageBeaconTime,3), StdDeviationBeaconTime=round(StdDeviationBeaconTime,3) |eval VariationCoefficient
    =(StdDeviationBeaconTime/AverageBeaconTime)*100
| sort -count | where VariationCoefficient < 100 AND count > 10 AND AverageBeaconTime>1.000
| table orig_h,resp_h,AverageBeaconTime,count,StdDeviationBeaconTime,VariationCoeffi
```

✓ 330,420 events (before 03/05/2019 22:26:34.000)

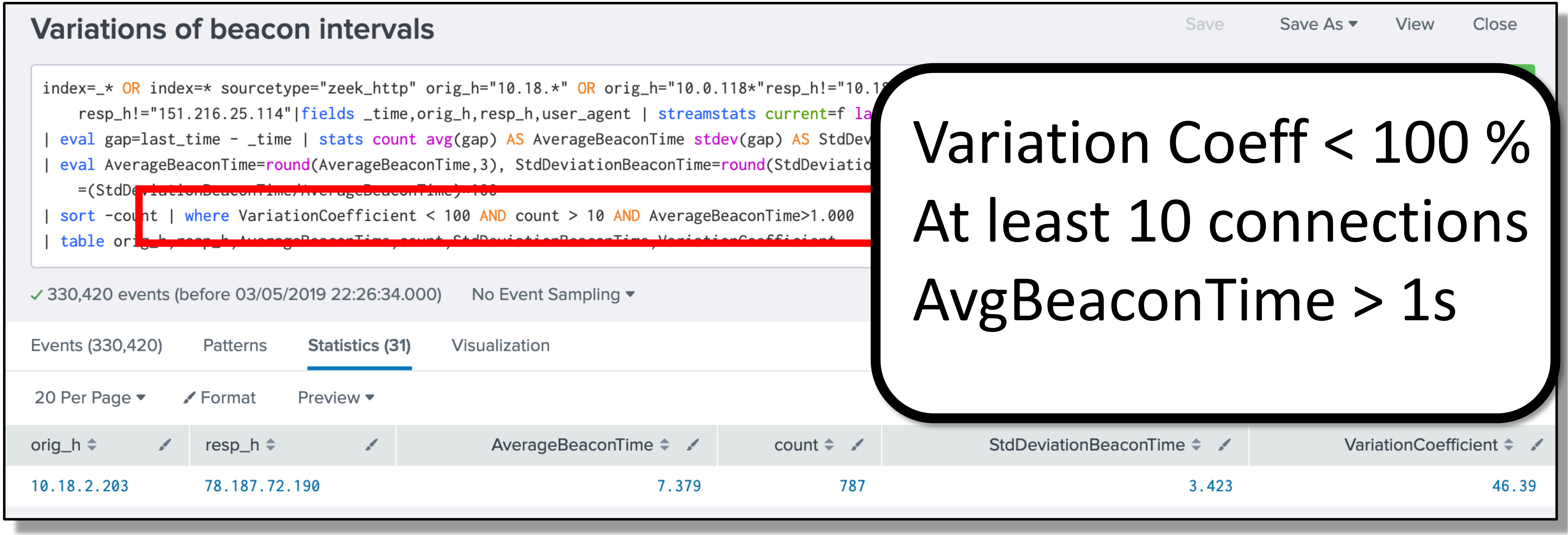
orig_h	resp_h	AverageBeaconTime	cou		
10.18.2.203	78.187.72.190	7.379	787	3.423	46.39

Aggregate connections

By **srcIP,dstIP,User-Agent**

Query inspired by: https://www.splunk.com/blog/2018/03/20/hunting-your-dns-dragons.html

Hipothesis #1: analysis of connections intervals



```
index=_* OR index=* sourcetype="zeek_http" orig_h="10.18.*" OR orig_h="10.0.118*"resp_h!="10.1
    resp_h!="151.216.25.114"|fields _time,orig_h,resp_h,user_agent | streamstats current=f la
| eval gap=last_time - _time | stats count avg(gap) AS AverageBeaconTime stdev(gap) AS StdDev
| eval AverageBeaconTime=round(AverageBeaconTime,3), StdDeviationBeaconTime=round(StdDeviatio
    =(StdDeviationBeaconTime/AverageBeaconTime)*100
| sort -count | where VariationCoefficient < 100 AND count > 10 AND AverageBeaconTime>1.000
| table orig_h,resp_h,AverageBeaconTime,count,StdDeviationBeaconTime,VariationCoefficient
```

✓ 330,420 events (before 03/05/2019 22:26:34.000)

orig_h	resp_h	AverageBeaconTime	count	StdDeviationBeaconTime	VariationCoefficient
10.18.2.203	78.187.72.190	7.379	787	3.423	46.39

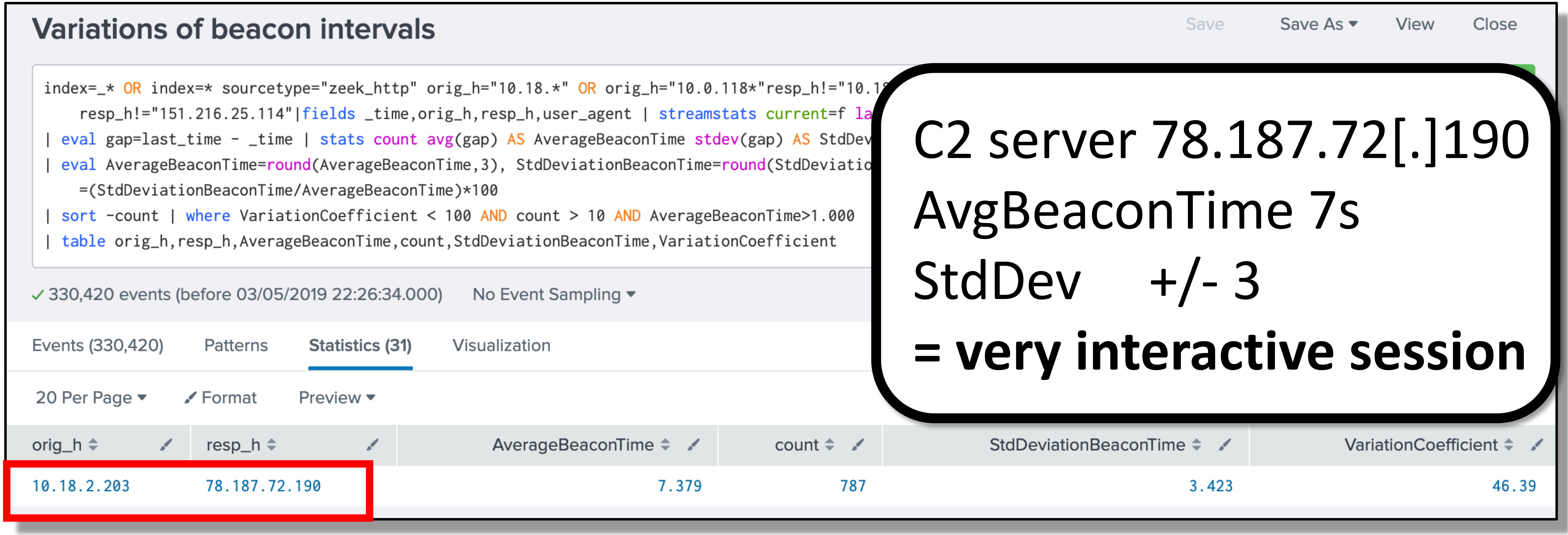
Variation Coeff < 100 %

At least 10 connections

AvgBeaconTime > 1s

Query inspired by: https://www.splunk.com/blog/2018/03/20/hunting-your-dns-dragons.html

Hipothesis #1: analysis of connections intervals



Variations of beacon intervals

```
index=_* OR index=* sourcetype="zeek_http" orig_h="10.18.*" OR orig_h="10.0.118*"resp_h!="10.1
    resp_h!="151.216.25.114"|fields _time,orig_h,resp_h,user_agent | streamstats current=f la
| eval gap=last_time - _time | stats count avg(gap) AS AverageBeaconTime stdev(gap) AS StdDev
| eval AverageBeaconTime=round(AverageBeaconTime,3), StdDeviationBeaconTime=round(StdDeviatio
    =(StdDeviationBeaconTime/AverageBeaconTime)*100
| sort -count | where VariationCoefficient < 100 AND count > 10 AND AverageBeaconTime>1.000
| table orig_h,resp_h,AverageBeaconTime,count,StdDeviationBeaconTime,VariationCoefficient
```

✓ 330,420 events (before 03/05/2019 22:26:34.000) No Event Sampling ▾

orig_h	resp_h	AverageBeaconTime	count	StdDeviationBeaconTime	VariationCoefficient
10.18.2.203	78.187.72.190	7.379	787	3.423	46.39

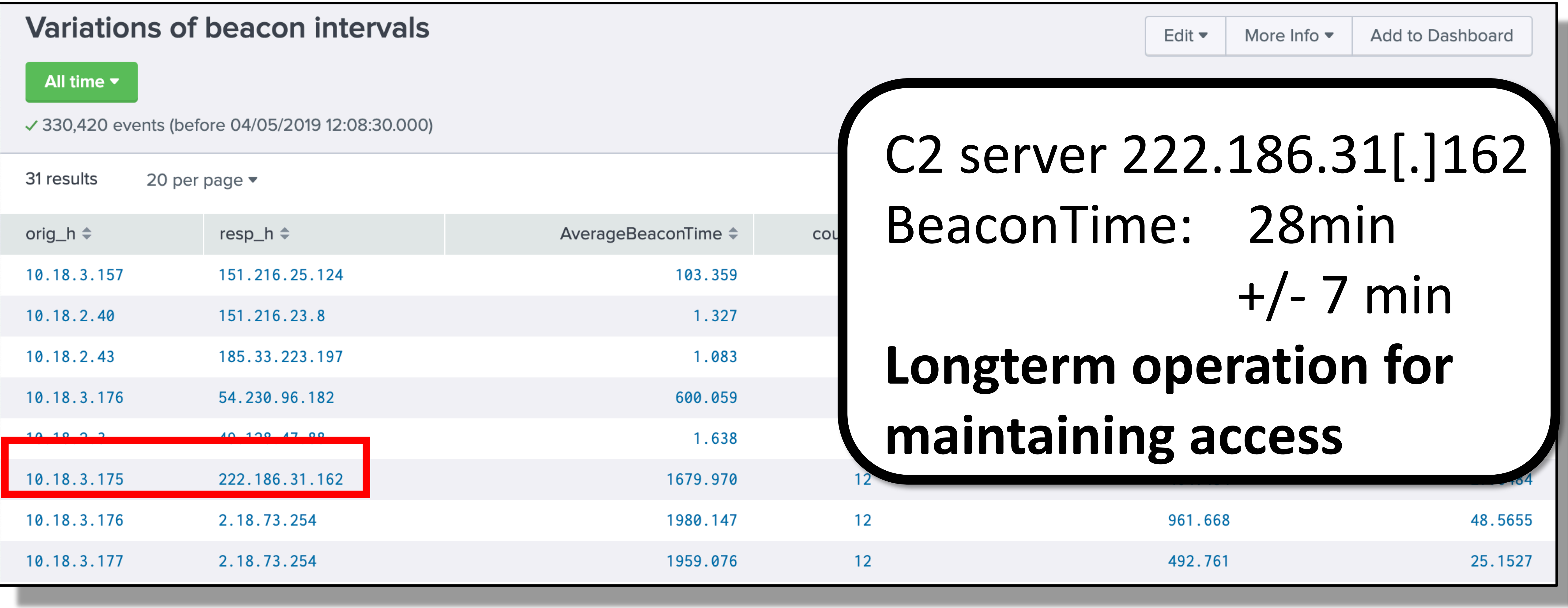
C2 server 78.187.72[.]190

AvgBeaconTime 7s

StdDev +/- 3

= very interactive session

Hipothesis #1: analysis of connections intervals



Variations of beacon intervals

✓ 330,420 events (before 04/05/2019 12:08:30.000)

31 results 20 per page ▾

orig_h	resp_h	AverageBeaconTime	cou		
10.18.3.157	151.216.25.124	103.359			
10.18.2.40	151.216.23.8	1.327			
10.18.2.43	185.33.223.197	1.083			
10.18.3.176	54.230.96.182	600.059			
10.18.2.3	40.128.47.88	1.638			
10.18.3.175	222.186.31.162	1679.970	12		[illegible]
10.18.3.176	2.18.73.254	1980.147	12	961.668	48.5655
10.18.3.177	2.18.73.254	1959.076	12	492.761	25.1527

C2 server 222.186.31[.]162

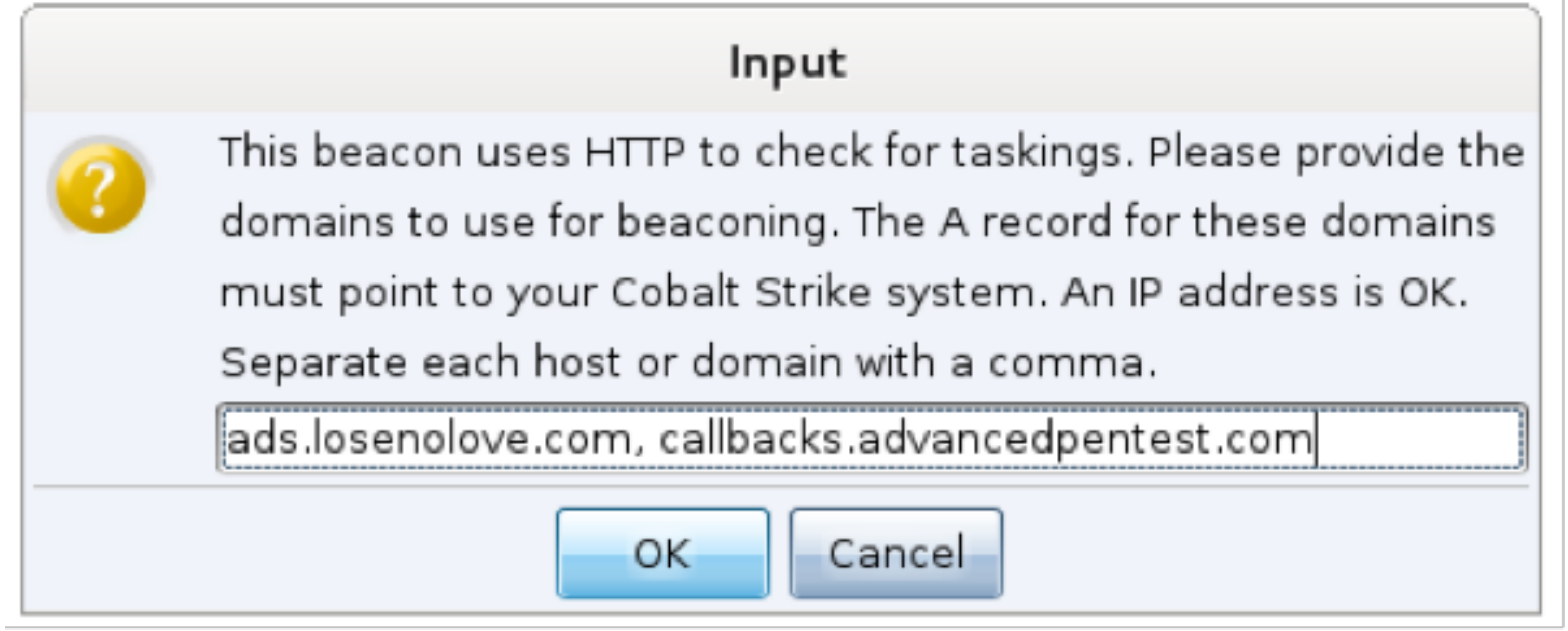
BeaconTime: 28min
+/- 7 min

Longterm operation for maintaining access

Hipothesis #2: same URI for different Host names

Hipothesis is based on the assumption that:

Adversary is using backdoor that has **several C2 backup domains included** in the configuration.

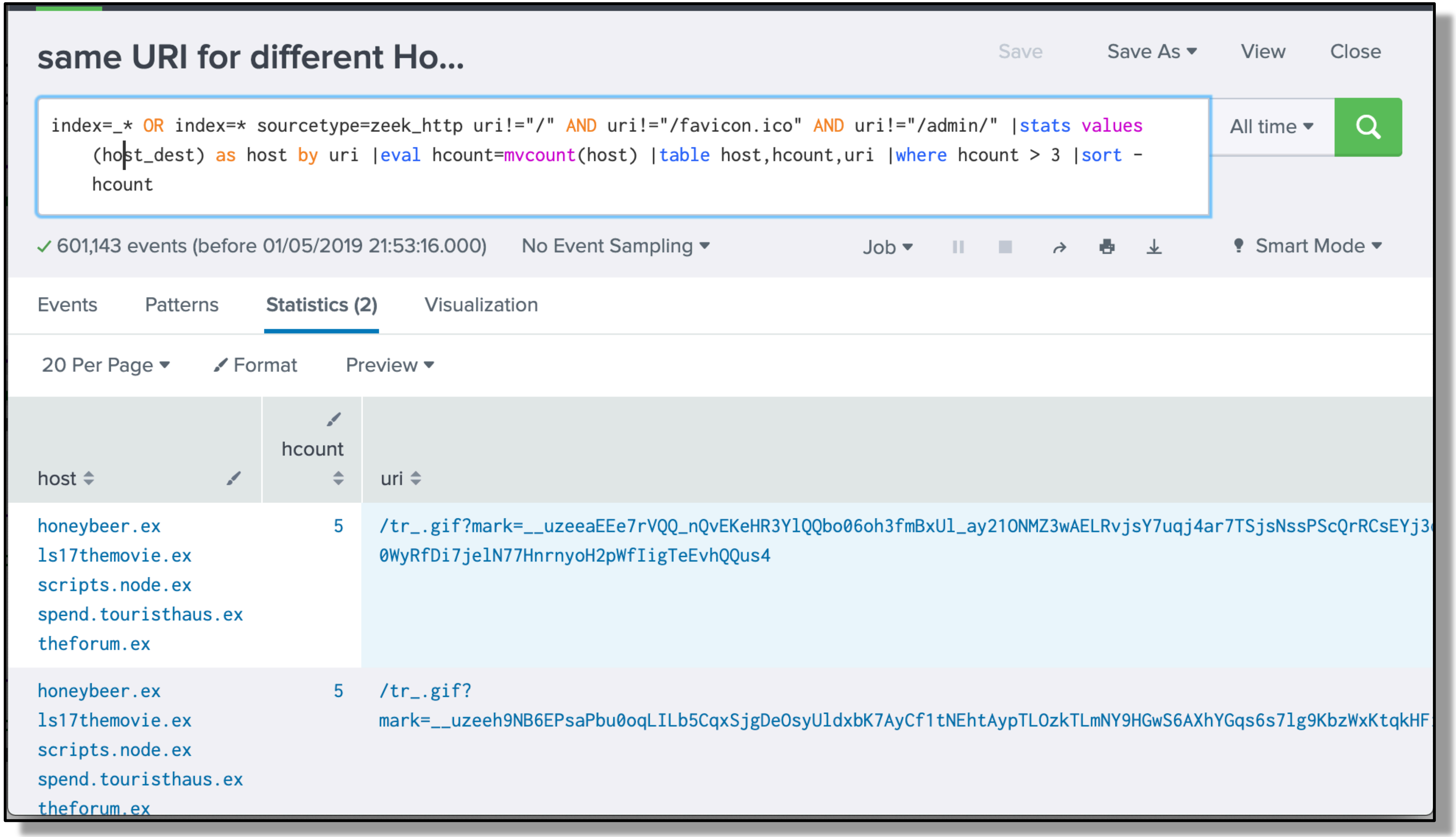


This beacon uses HTTP to check for taskings. Please provide the domains to use for beaconing. The A record for these domains must point to your Cobalt Strike system. An IP address is OK. Separate each host or domain with a comma.

ads.losenolove.com, callbacks.advancedpentest.com

https://www.cobaltstrike.com/help-http-beacon

Hipothesis #2: same URI for different Host names

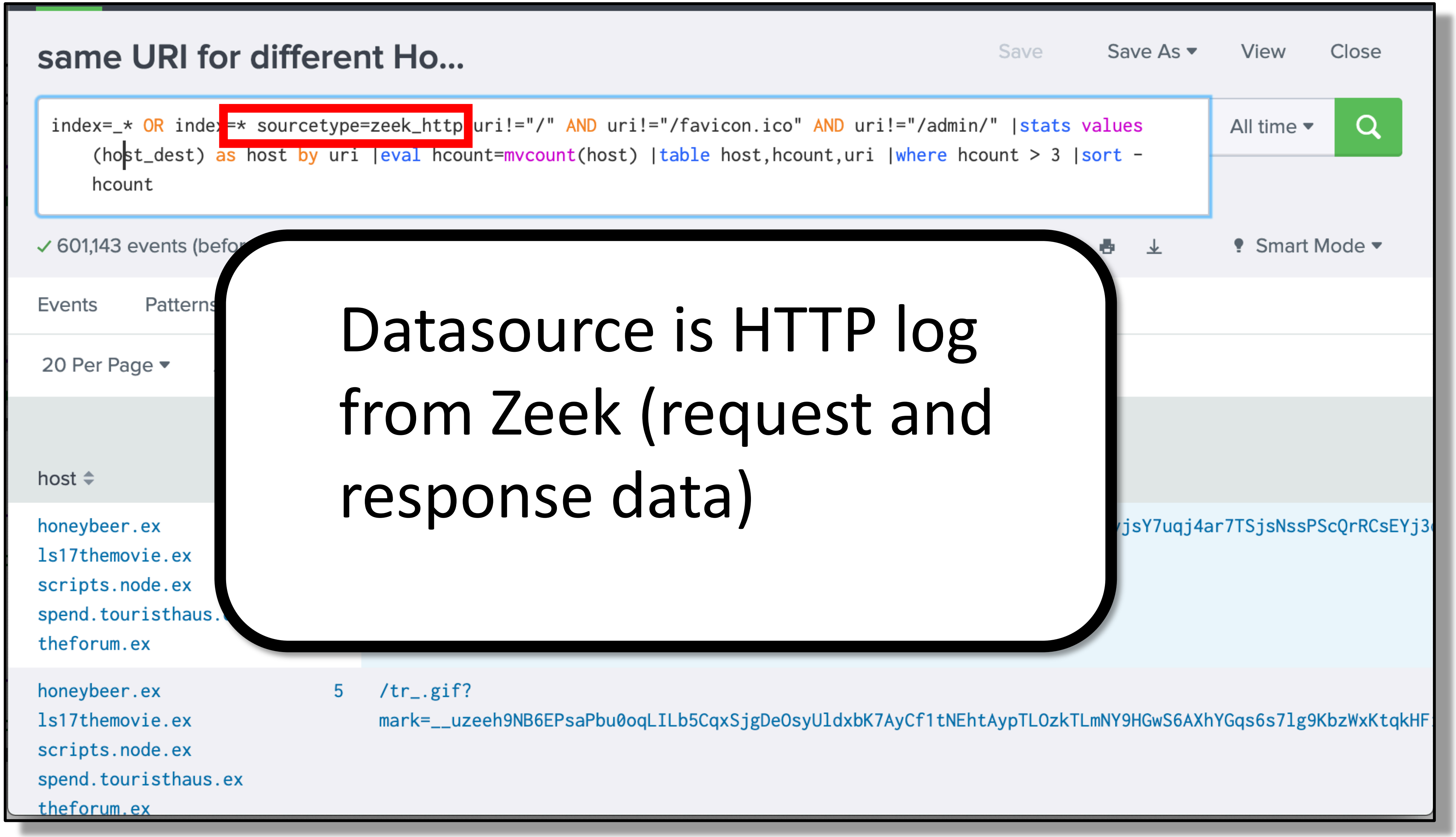


```
index=_* OR index=* sourcetype=zeek_http uri!="/" AND uri!="/favicon.ico" AND uri!="/admin/" |stats values
    (host_dest) as host by uri |eval hcount=mvcount(host) |table host,hcount,uri |where hcount > 3 |sort -
    hcount
```

✓ 601,143 events (before 01/05/2019 21:53:16.000)

host	hcount	uri
honeybeer.ex ls17themovie.ex scripts.node.ex spend.touristhaus.ex theforum.ex	5	/tr_.gif?mark=__uzeeaEEe7rVQQ_nQvEKeHR3Y1QQbo06oh3fmBxUl_ay21ONMZ3wAELRvjsY7uqj4ar7TSjsNssPScQrRCsEYj3 0WyRfDi7jelN77HnrnyoH2pWfIigTeEvhQQus4
honeybeer.ex ls17themovie.ex scripts.node.ex spend.touristhaus.ex theforum.ex	5	/tr_.gif? mark=__uzeeh9NB6EPsaPbu0oqLILb5CqxSjgDeOsyUldxbK7AyCf1tNEhtAypTLOzkTLmNY9HGwS6AXhYGqs6s7lg9KbzWxKtqkHF

Hipothesis #2: same URI for different Host names



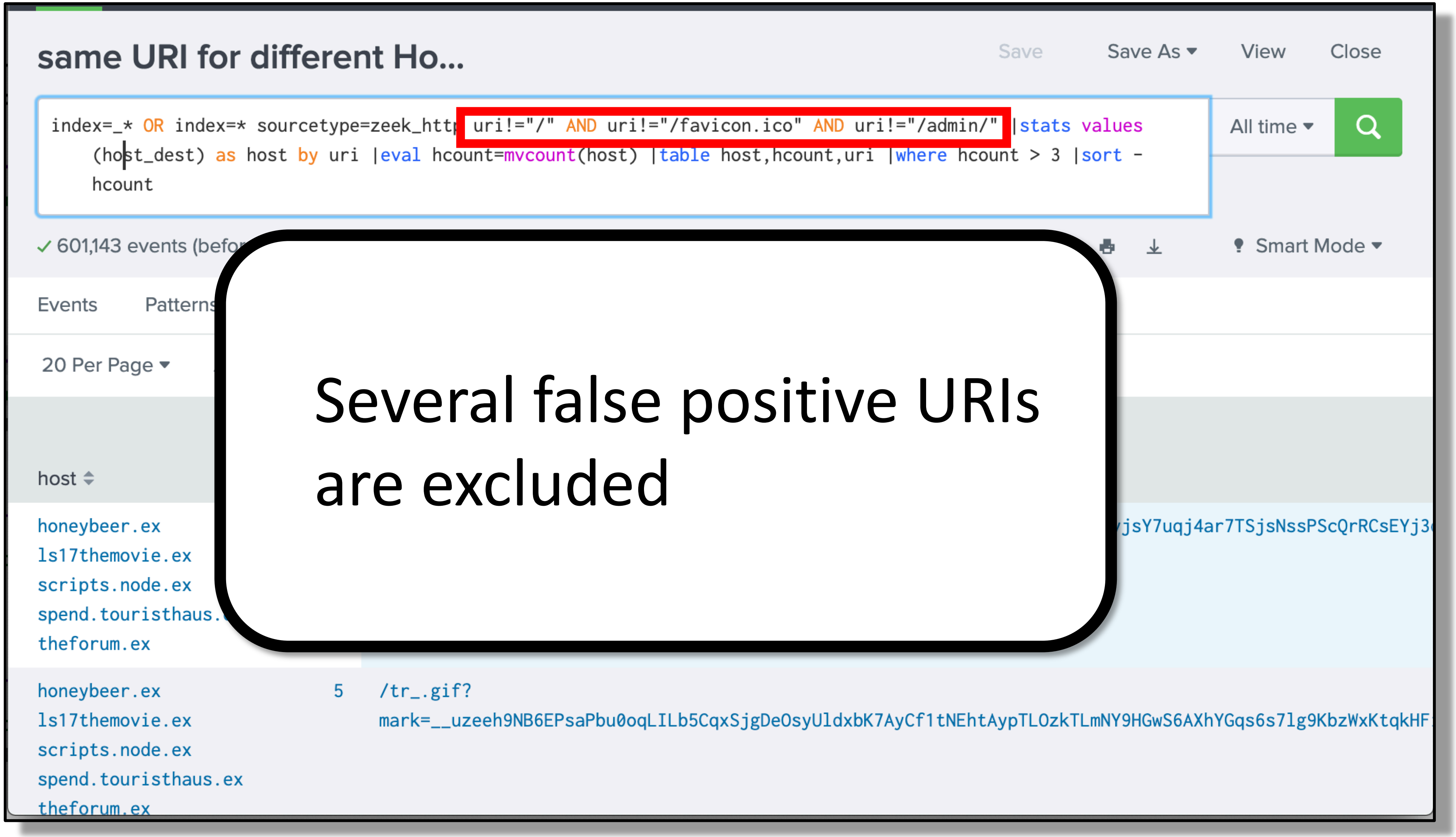
```
index=_* OR index=* sourcetype=zeek_http uri!="/" AND uri!="/favicon.ico" AND uri!="/admin/" |stats values
    (host_dest) as host by uri |eval hcount=mvcount(host) |table host,hcount,uri |where hcount > 3 |sort -
    hcount
```

```
host
honeybeer.ex                                                    jsY7uqj4ar7TSjsNssPScQrRCsEYj3
ls17themovie.ex
scripts.node.ex
spend.touristhaus.
theforum.ex

honeybeer.ex              5   /tr_.gif?
ls17themovie.ex               mark=__uzeeh9NB6EPsaPbu0oqLILb5CqxSjgDeOsyUldxbK7AyCf1tNEhtAypTLOzkTLmNY9HGwS6AXhYGqs6s7lg9KbzWxKtqkHF
scripts.node.ex
spend.touristhaus.ex
theforum.ex
```

Datasource is HTTP log from Zeek (request and response data)

Hipothesis #2: same URI for different Host names



```
index=_* OR index=* sourcetype=zeek_http uri!="/" AND uri!="/favicon.ico" AND uri!="/admin/" |stats values
    (host_dest) as host by uri |eval hcount=mvcount(host) |table host,hcount,uri |where hcount > 3 |sort -
    hcount
```

✓ 601,143 events (befo

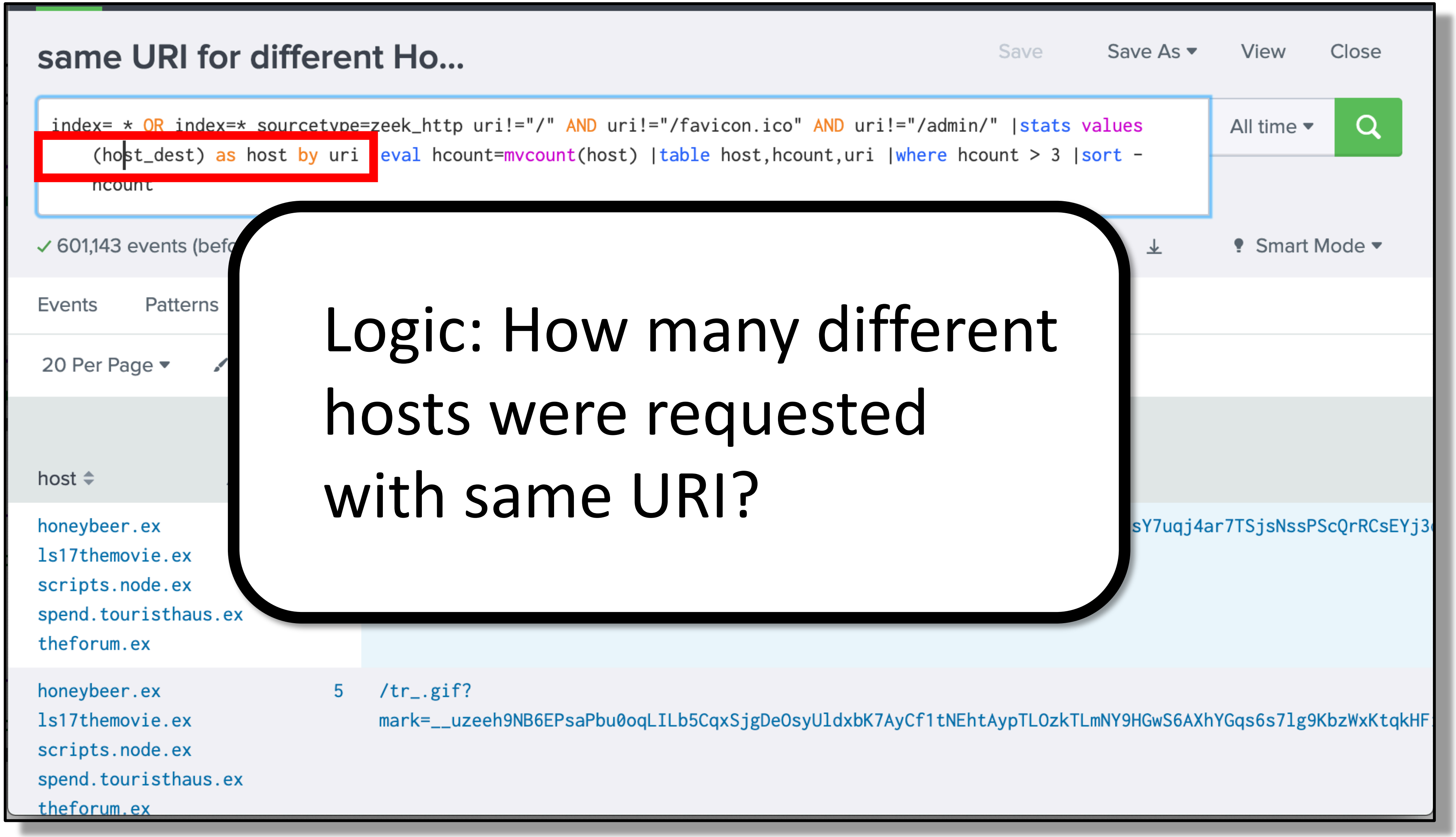
```
host ⇕

honeybeer.ex                                                                  jsY7uqj4ar7TSjsNssPScQrRCsEYj3
ls17themovie.ex
scripts.node.ex
spend.touristhaus.
theforum.ex

honeybeer.ex                 5   /tr_.gif?
ls17themovie.ex                  mark=__uzeeh9NB6EPsaPbu0oqLILb5CqxSjgDeOsyUldxbK7AyCf1tNEhtAypTLOzkTLmNY9HGwS6AXhYGqs6s7lg9KbzWxKtqkHF
scripts.node.ex
spend.touristhaus.ex
theforum.ex
```

Several false positive URIs are excluded

Hipothesis #2: same URI for different Host names



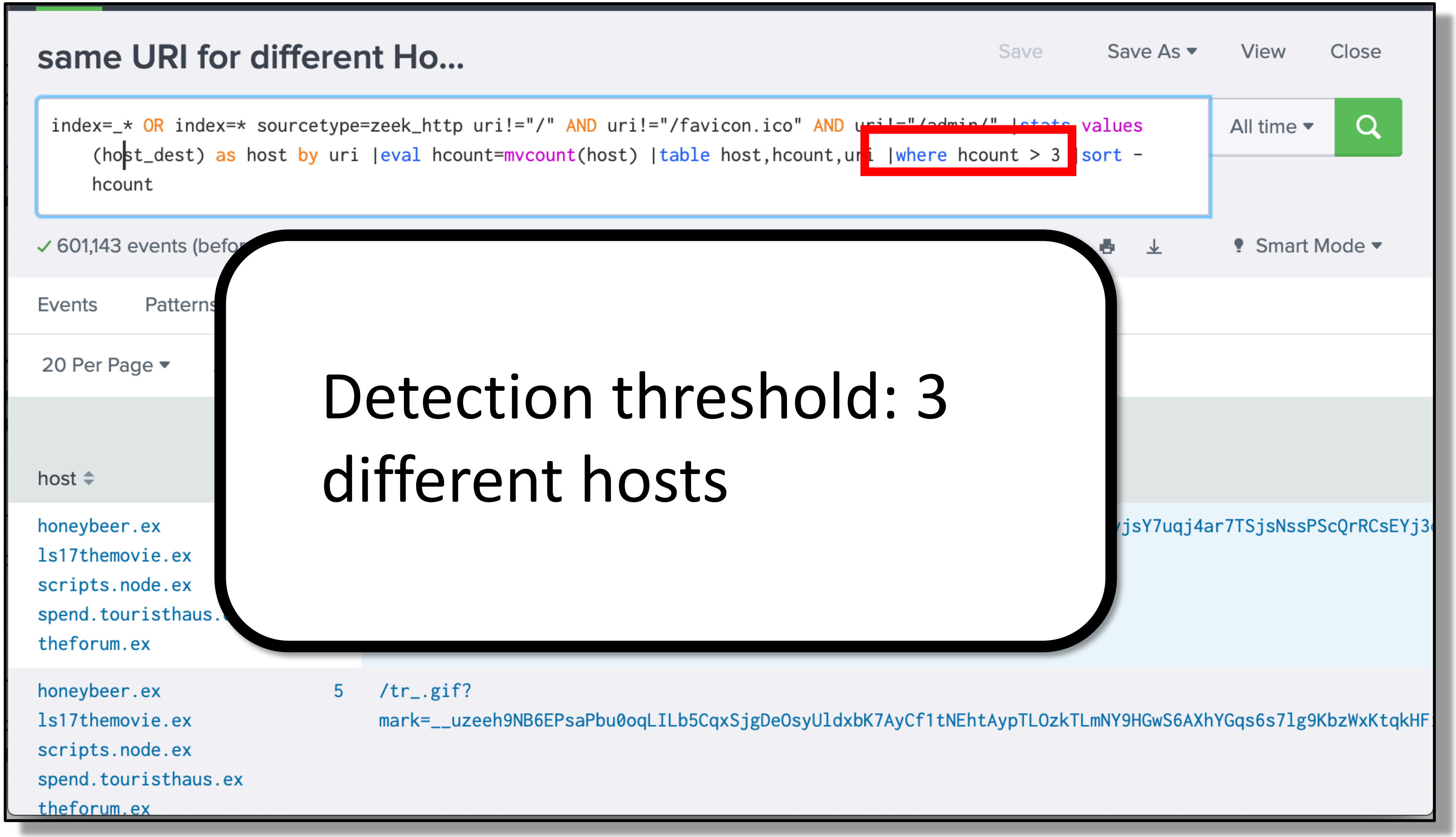
```
index= * OR index=* sourcetype=zeek_http uri!="/" AND uri!="/favicon.ico" AND uri!="/admin/" |stats values
    (host_dest) as host by uri  eval hcount=mvcount(host) |table host,hcount,uri |where hcount > 3 |sort -
    hcount
```

✓ 601,143 events (befo

host		
honeybeer.ex ls17themovie.ex scripts.node.ex spend.touristhaus.ex theforum.ex		sY7uqj4ar7TSjsNssPScQrRCsEYj3
honeybeer.ex ls17themovie.ex scripts.node.ex spend.touristhaus.ex theforum.ex	5	/tr_.gif? mark=__uzeeh9NB6EPsaPbu0oqLILb5CqxSjgDeOsyUldxbK7AyCf1tNEhtAypTLOzkTLmNY9HGwS6AXhYGqs6s7lg9KbzWxKtqkHF

Logic: How many different hosts were requested with same URI?

Hipothesis #2: same URI for different Host names



```
index=_* OR index=* sourcetype=zeek_http uri!="/" AND uri!="/favicon.ico" AND uri!="/admin/" |stats values
    (host_dest) as host by uri |eval hcount=mvcount(host) |table host,hcount,uri |where hcount > 3 |sort -
    hcount
```

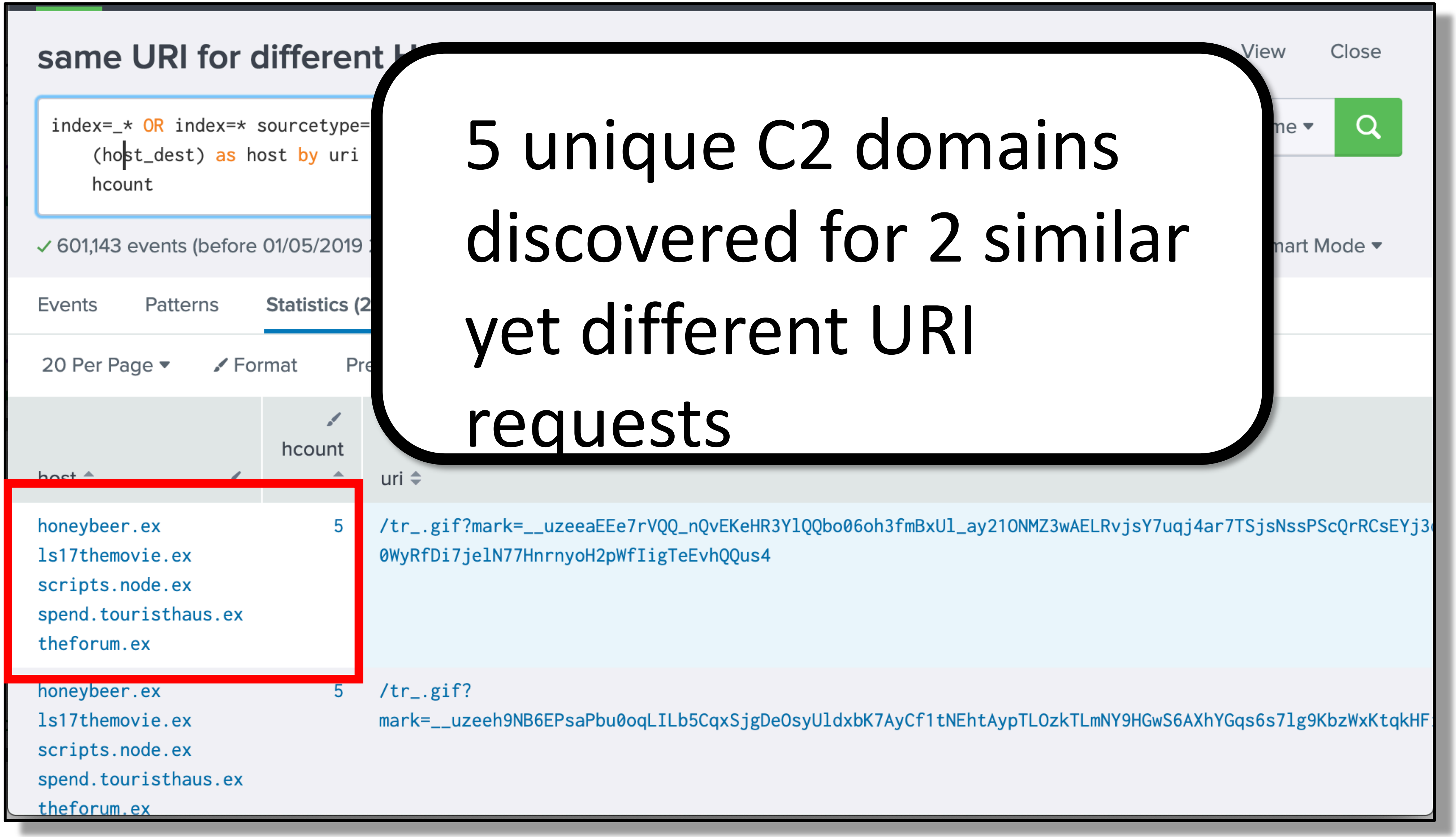
```
host ⇕

honeybeer.ex                                    jsY7uqj4ar7TSjsNssPScQrRCsEYj3
ls17themovie.ex
scripts.node.ex
spend.touristhaus.
theforum.ex

honeybeer.ex               5    /tr_.gif?
ls17themovie.ex                 mark=__uzeeh9NB6EPsaPbu0oqLILb5CqxSjgDeOsyUldxbK7AyCf1tNEhtAypTLOzkTLmNY9HGwS6AXhYGqs6s7lg9KbzWxKtqkHF
scripts.node.ex
spend.touristhaus.ex
theforum.ex
```

Detection threshold: 3 different hosts

Hipothesis #2: same URI for different Host names



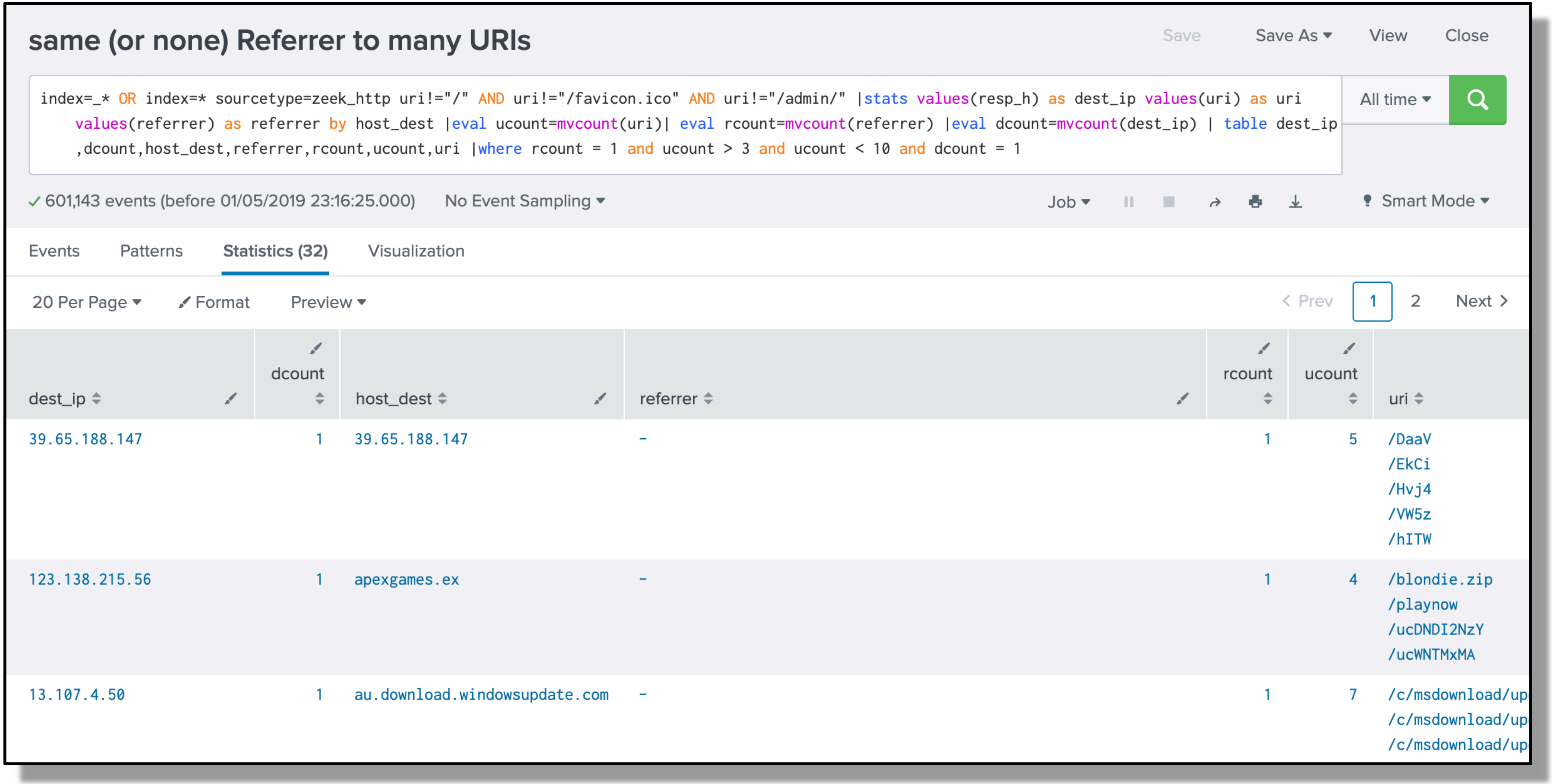
```
index=_* OR index=* sourcetype=
    (host_dest) as host by uri
    hcount
```

✓ 601,143 events (before 01/05/2019

host	hcount	uri
honeybeer.ex ls17themovie.ex scripts.node.ex spend.touristhaus.ex theforum.ex	5	/tr_.gif?mark=__uzeeaEEe7rVQQ_nQvEKeHR3Y1QQbo06oh3fmBxUl_ay21ONMZ3wAELRvjsY7uqj4ar7TSjsNssPScQrRCsEYj3 0WyRfDi7jelN77HnrnyoH2pWfIigTeEvhQQus4
honeybeer.ex ls17themovie.ex scripts.node.ex spend.touristhaus.ex theforum.ex	5	/tr_.gif? mark=__uzeeh9NB6EPsaPbu0oqLILb5CqxSjgDeOsyUldxbK7AyCf1tNEhtAypTLOzkTLmNY9HGwS6AXhYGqs6s7lg9KbzWxKtqkHF

5 unique C2 domains discovered for 2 similar yet different URI requests

Hipothesis #3: Same or none Referrer to many URIs



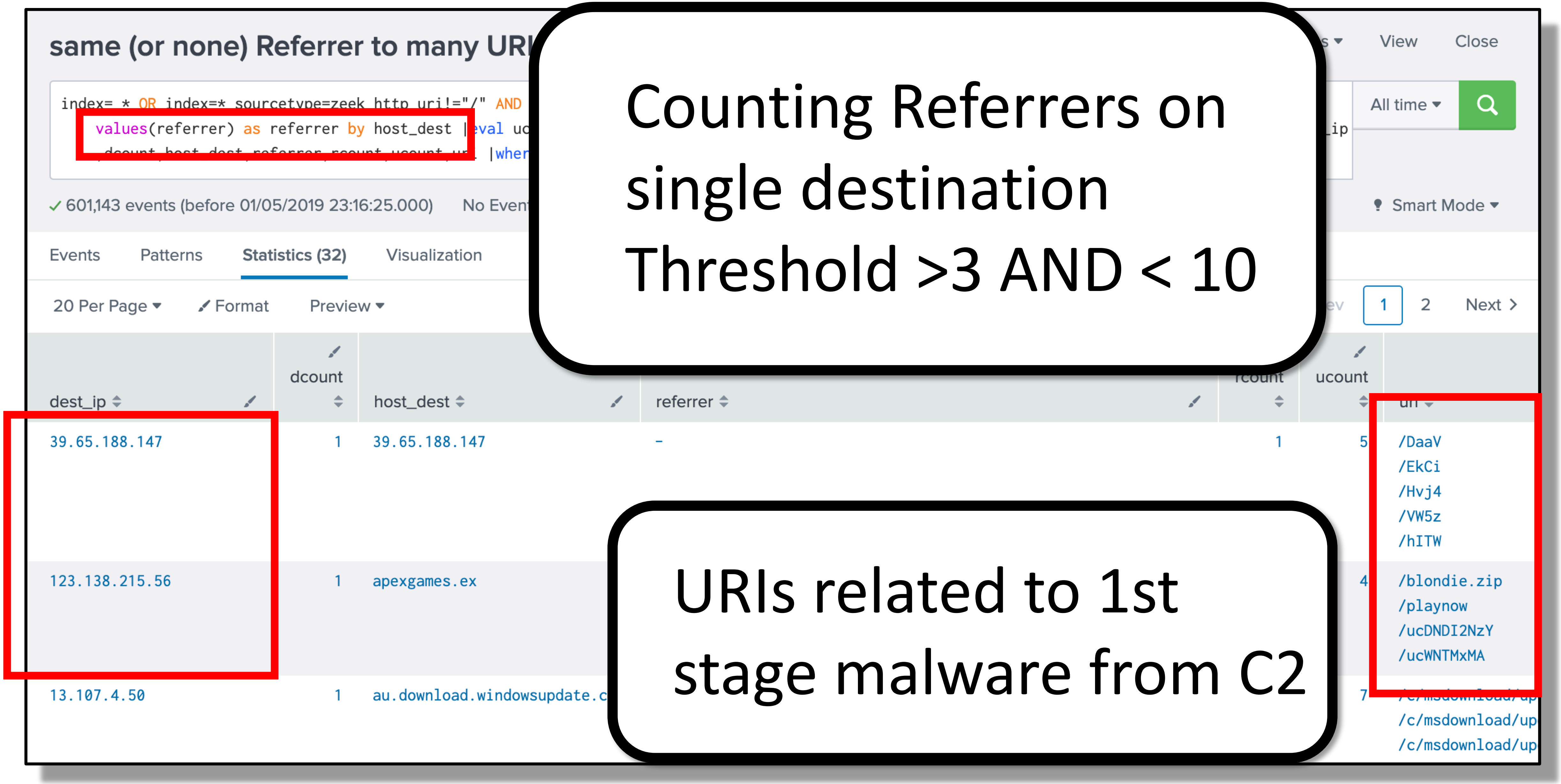
same (or none) Referrer to many URIs

```
index=_* OR index=* sourcetype=zeek_http uri!="/" AND uri!="/favicon.ico" AND uri!="/admin/" |stats values(resp_h) as dest_ip values(uri) as uri
    values(referrer) as referrer by host_dest |eval ucount=mvcount(uri)| eval rcount=mvcount(referrer) |eval dcount=mvcount(dest_ip) | table dest_ip
    ,dcount,host_dest,referrer,rcount,ucount,uri |where rcount = 1 and ucount > 3 and ucount < 10 and dcount = 1
```

✓ 601,143 events (before 01/05/2019 23:16:25.000)

dest_ip	dcount	host_dest	referrer	rcount	ucount	uri
39.65.188.147	1	39.65.188.147	-	1	5	/DaaV /EkCi /Hvj4 /VW5z /hITW
123.138.215.56	1	apexgames.ex	-	1	4	/blondie.zip /playnow /ucDNDI2NzY /ucWNTMxMA
13.107.4.50	1	au.download.windowsupdate.com	-	1	7	/c/msdownload/up /c/msdownload/up /c/msdownload/up

Hipothesis #3: Same or none Referrer to many URIs



```
index= * OR index=* sourcetype=zeek_http uri!="/" AND
  values(referrer) as referrer by host_dest |eval uc
  ,dcount,host_dest,referrer,rcount,ucount,uri |wher
```

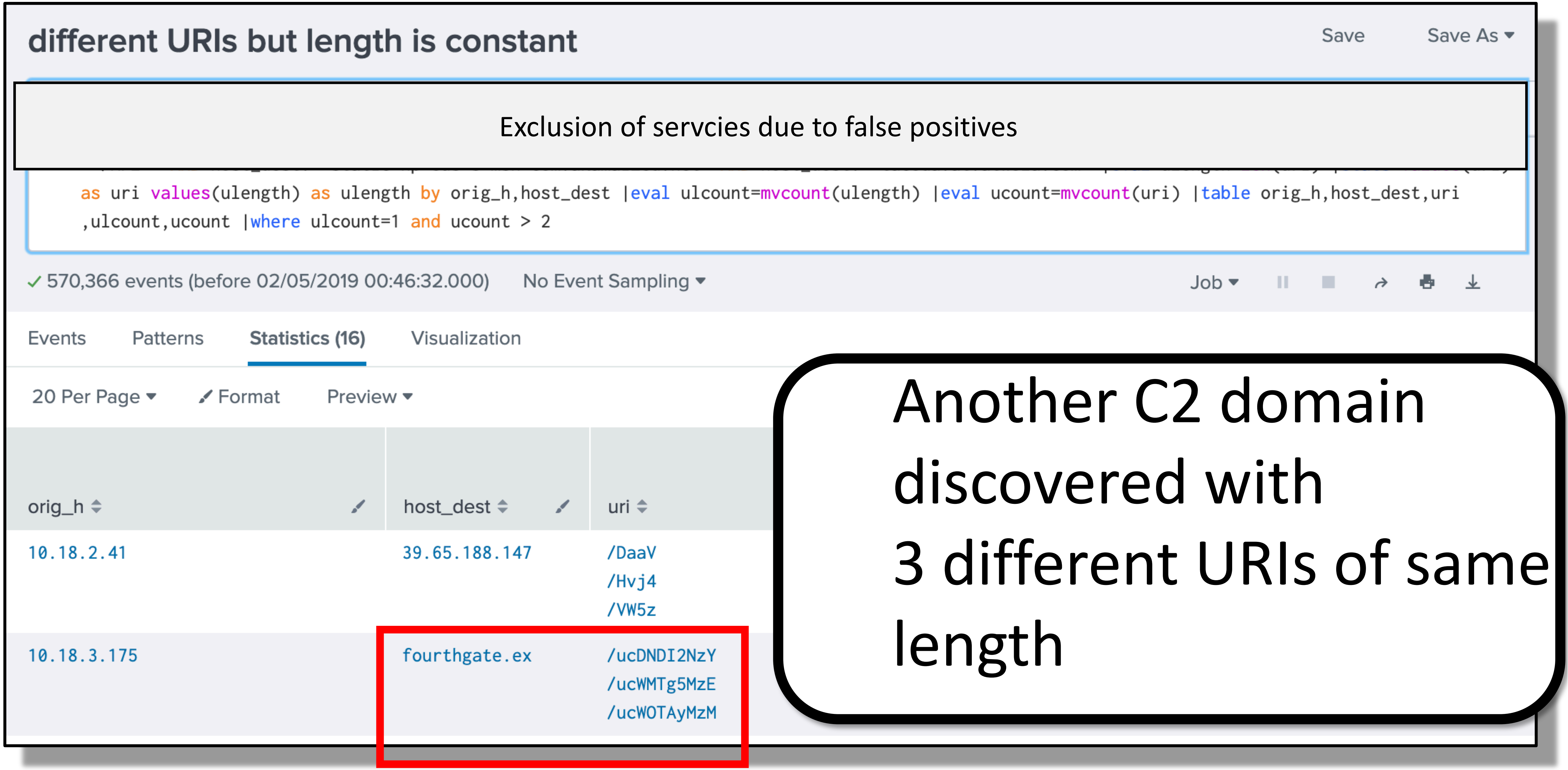
✓ 601,143 events (before 01/05/2019 23:16:25.000) No Event

dest_ip	dcount	host_dest	referrer	rcount	ucount	uri
39.65.188.147	1	39.65.188.147	-	1	5	/DaaV /EkCi /Hvj4 /VW5z /hITW
123.138.215.56	1	apexgames.ex			4	/blondie.zip /playnow /ucDNDI2NzY /ucWNTMxMA
13.107.4.50	1	au.download.windowsupdate.c			7	/c/msdownload/up /c/msdownload/up /c/msdownload/up

Counting Referrers on single destination Threshold >3 AND < 10

URIs related to 1st stage malware from C2

Hipothesis #4: different URIs but length is constant



different URIs but length is constant

Exclusion of servcies due to false positives

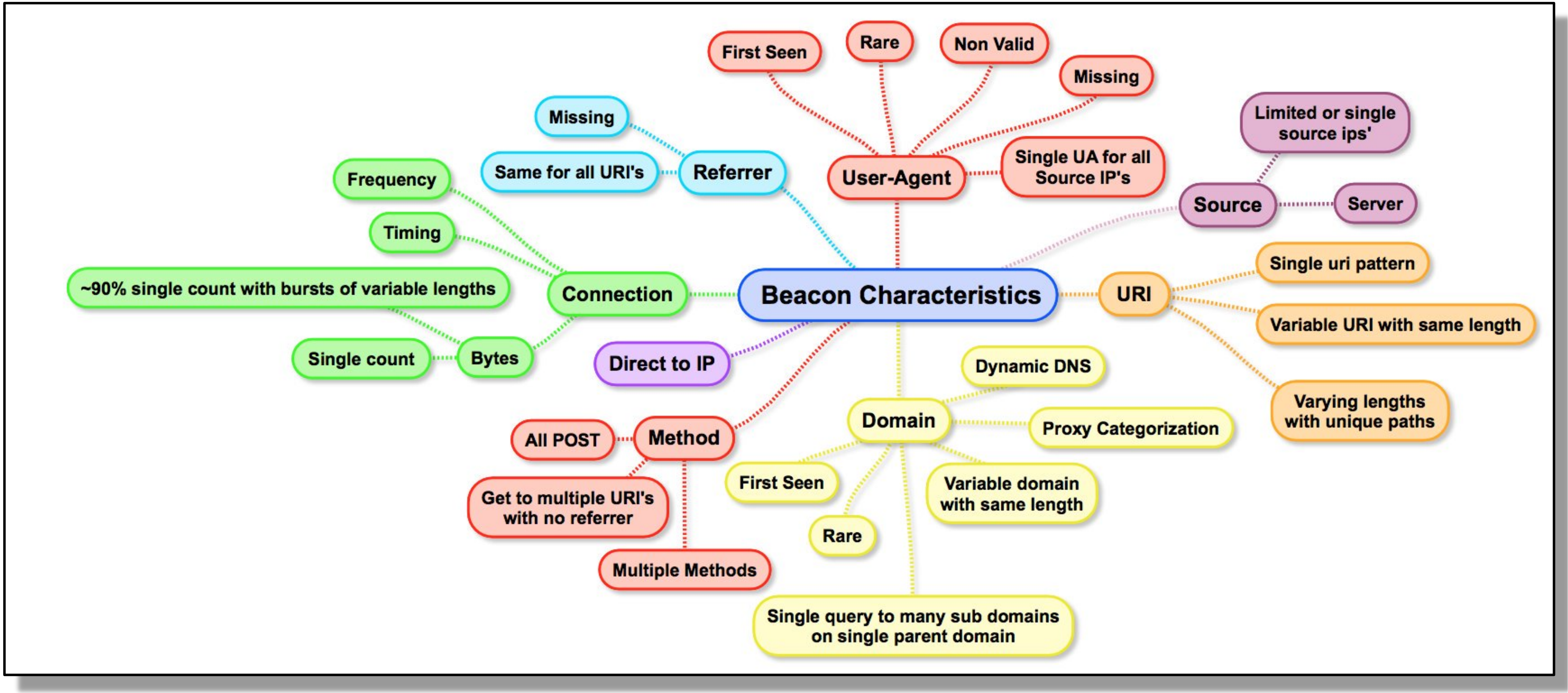
```
as uri values(ulength) as ulength by orig_h,host_dest |eval ulcount=mvcount(ulength) |eval ucount=mvcount(uri) |table orig_h,host_dest,uri
,ulcount,ucount |where ulcount=1 and ucount > 2
```

✓ 570,366 events (before 02/05/2019 00:46:32.000)

orig_h	host_dest	uri
10.18.2.41	39.65.188.147	/DaaV /Hvj4 /VW5z
10.18.3.175	fourthgate.ex	/ucDNDI2NzY /ucWMTg5MzE /ucWOTAyMzM

Another C2 domain discovered with
3 different URIs of same length

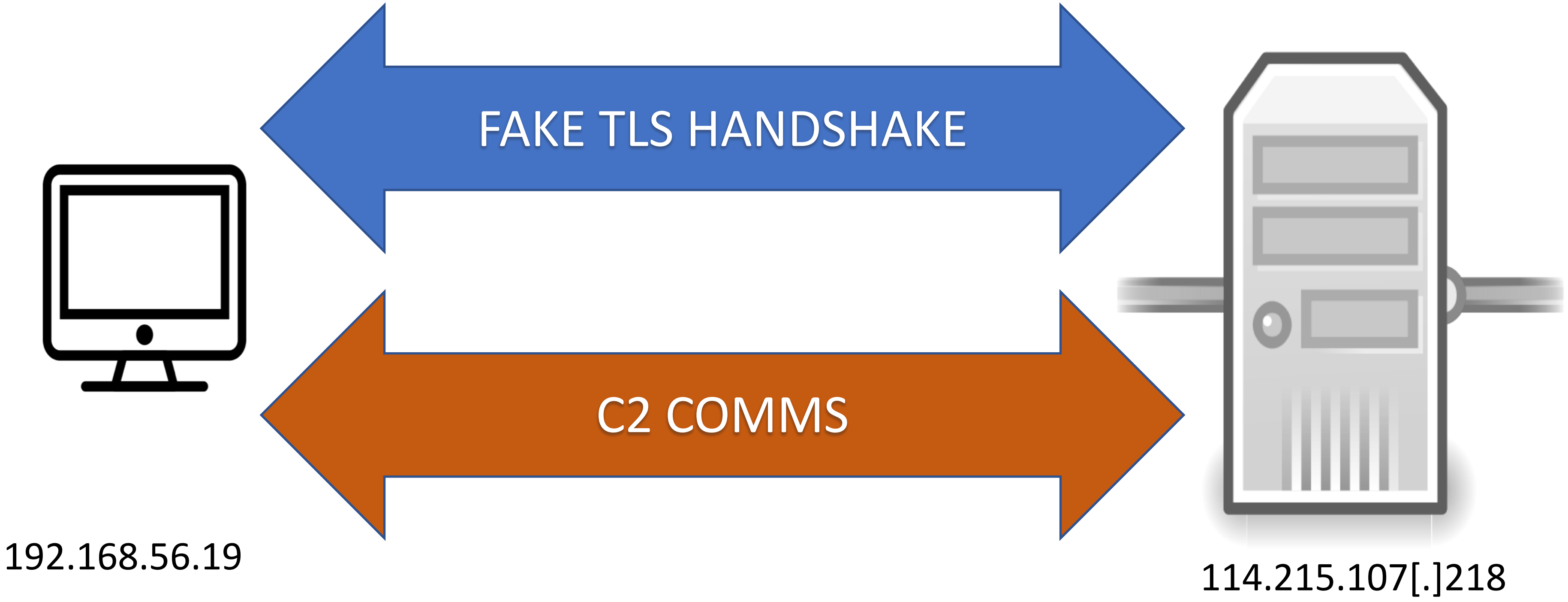
Jack Crook (still waiting for you, Jack, at x33fcon) has a great set for hipothesis inspirations:


Beacon Characteristics
User-Agent
First Seen
Rare
Non Valid
Missing
Single UA for all Source IP's
Referrer
Missing
Same for all URI's
Source
Limited or single source ips'
Server
URI
Single uri pattern
Variable URI with same length
Varying lengths with unique paths
Connection
Frequency
Timing
~90% single count with bursts of variable lengths
Bytes
Single count
Direct to IP
Method
All POST
Get to multiple URI's with no referrer
Multiple Methods
Domain
Dynamic DNS
Proxy Categorization
First Seen
Rare
Variable domain with same length
Single query to many sub domains on single parent domain

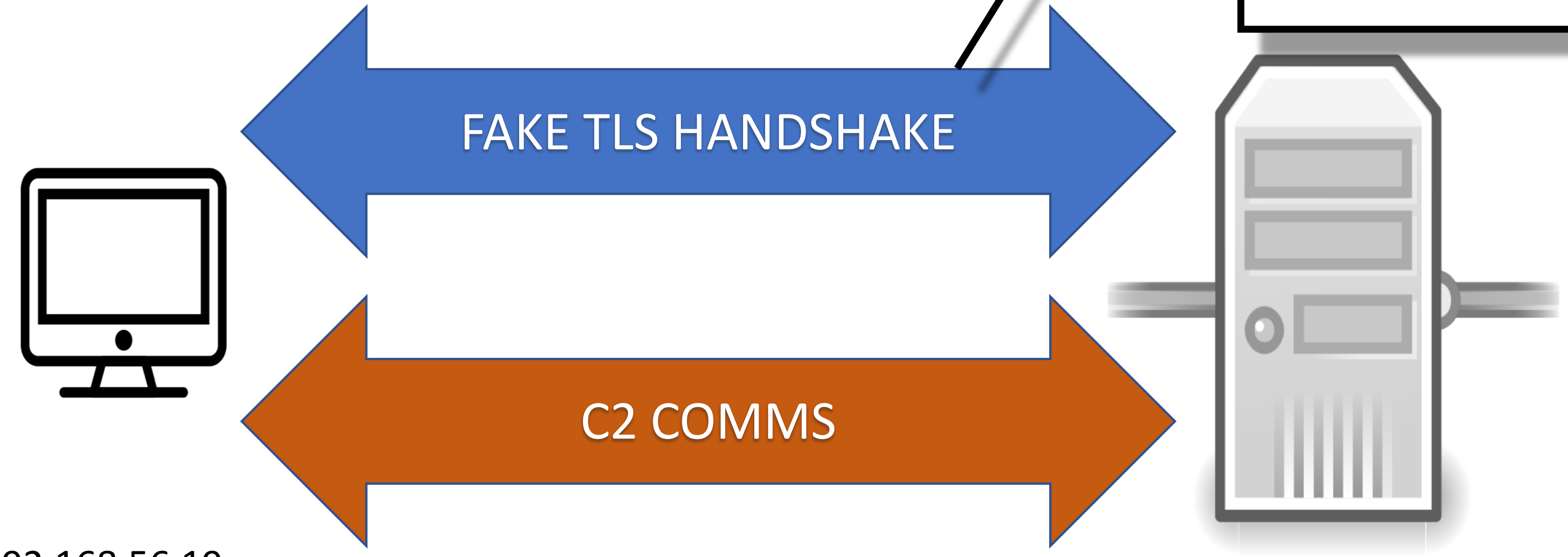
https://twitter.com/jackcr/status/1029457184164335617

PART II
Beaconing over **HTTPS**
{ FakeTLS example from LAZARUS APT }

FakeTLS – how does it work?



FakeTLS – how does it work?



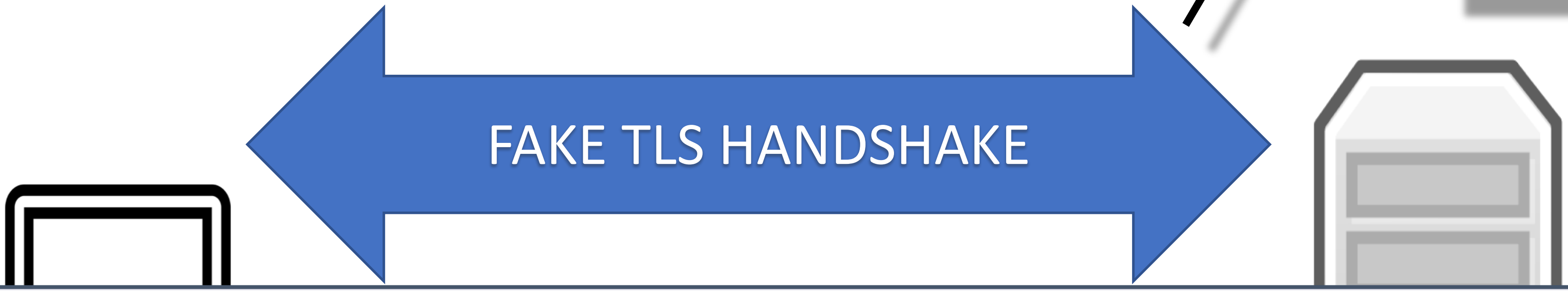
The Funny Part of mimicking TLS to popular sites e.g. wetransfer.com

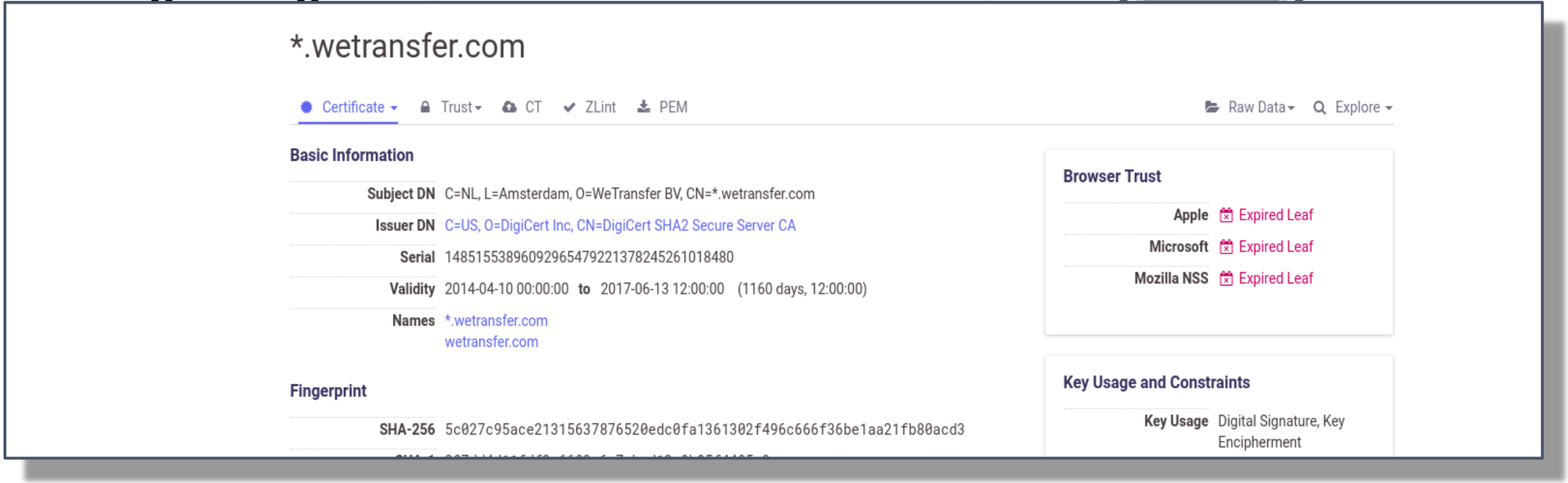
192.168.56.19

114.215.107[.]218

FakeTLS – how does it work?

C2 sends back real (often expired) certificate





*.wetransfer.com

Basic Information

Subject DN	C=NL, L=Amsterdam, O=WeTransfer BV, CN=*.wetransfer.com
Issuer DN	C=US, O=DigiCert Inc, CN=DigiCert SHA2 Secure Server CA
Serial	14851553896092965479221378245261018480
Validity	2014-04-10 00:00:00 **to** 2017-06-13 12:00:00 (1160 days, 12:00:00)
Names	*.wetransfer.com wetransfer.com

Fingerprint

SHA-256	5c027c95ace21315637876520edc0fa1361302f496c666f36be1aa21fb80acd3

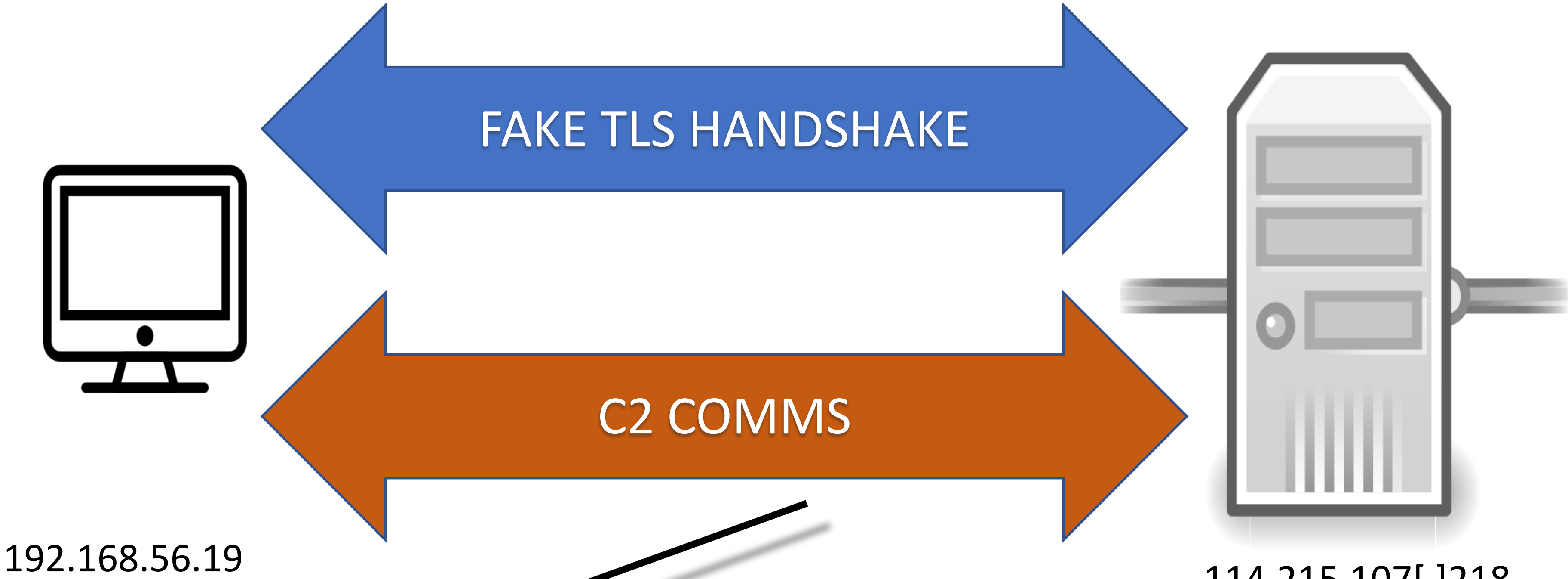
Browser Trust

Apple	Expired Leaf
Microsoft	Expired Leaf
Mozilla NSS	Expired Leaf

Key Usage and Constraints

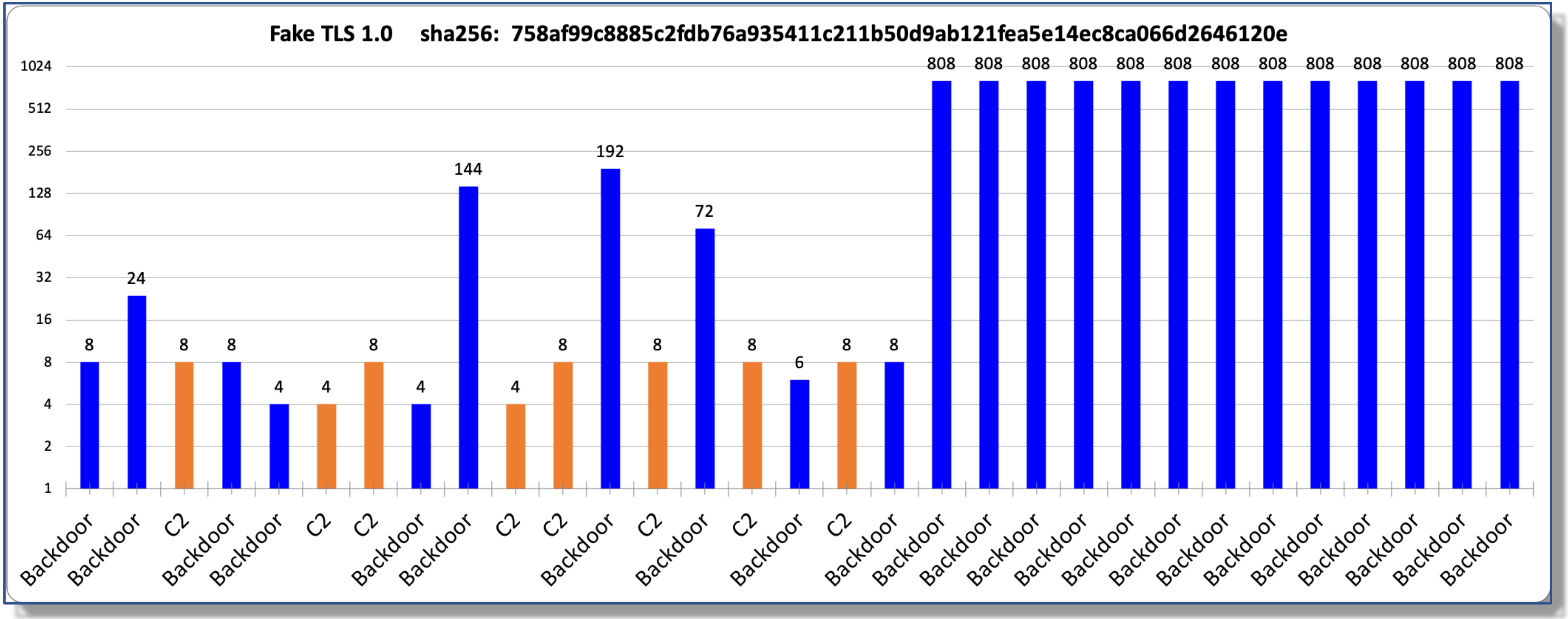
Key Usage	Digital Signature, Key Encipherment

FakeTLS – how does it work?



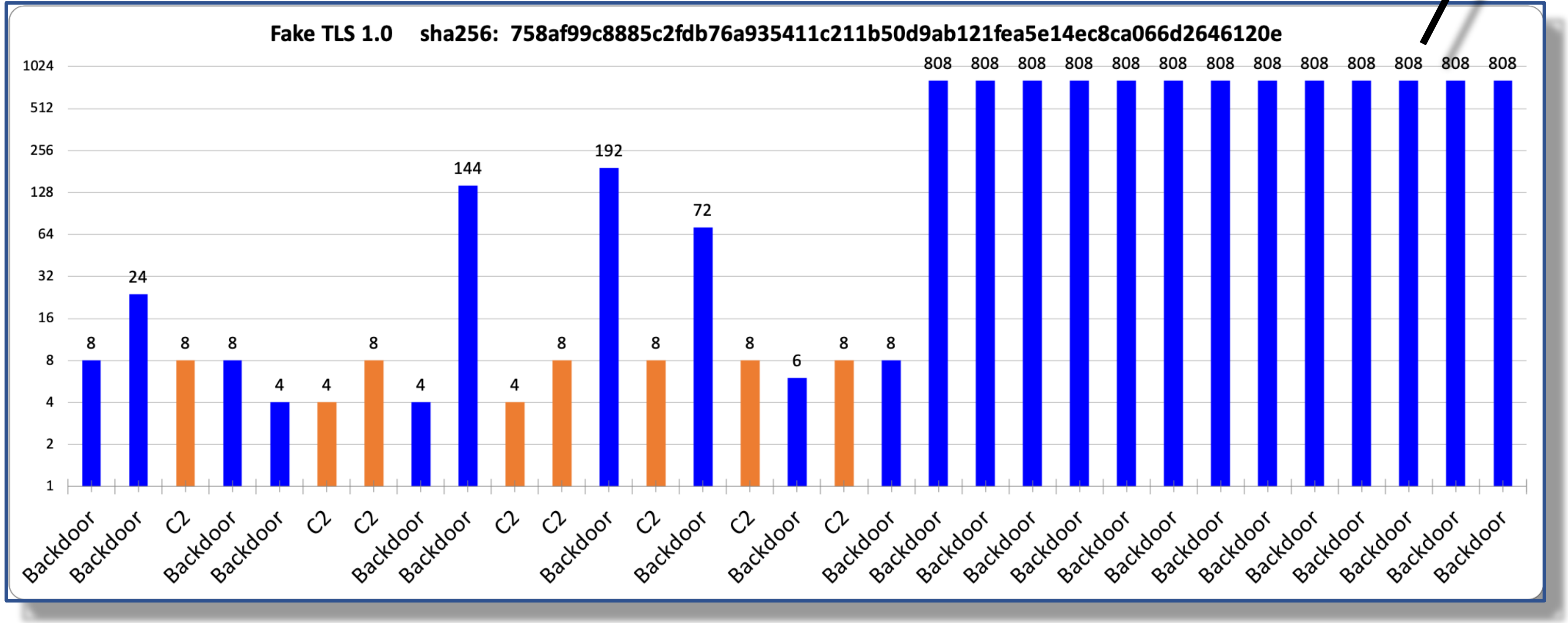
Non-TLS encryption with symmetric, shared RC4 key

FakeTLS – does it beacon?


Fake TLS 1.0 sha256: 758af99c8885c2fdb76a935411c211b50d9ab121fea5e14ec8ca066d2646120e

C2 COMMS (encrypted messages sizes in Bytes)

FakeTLS – does it beacon?

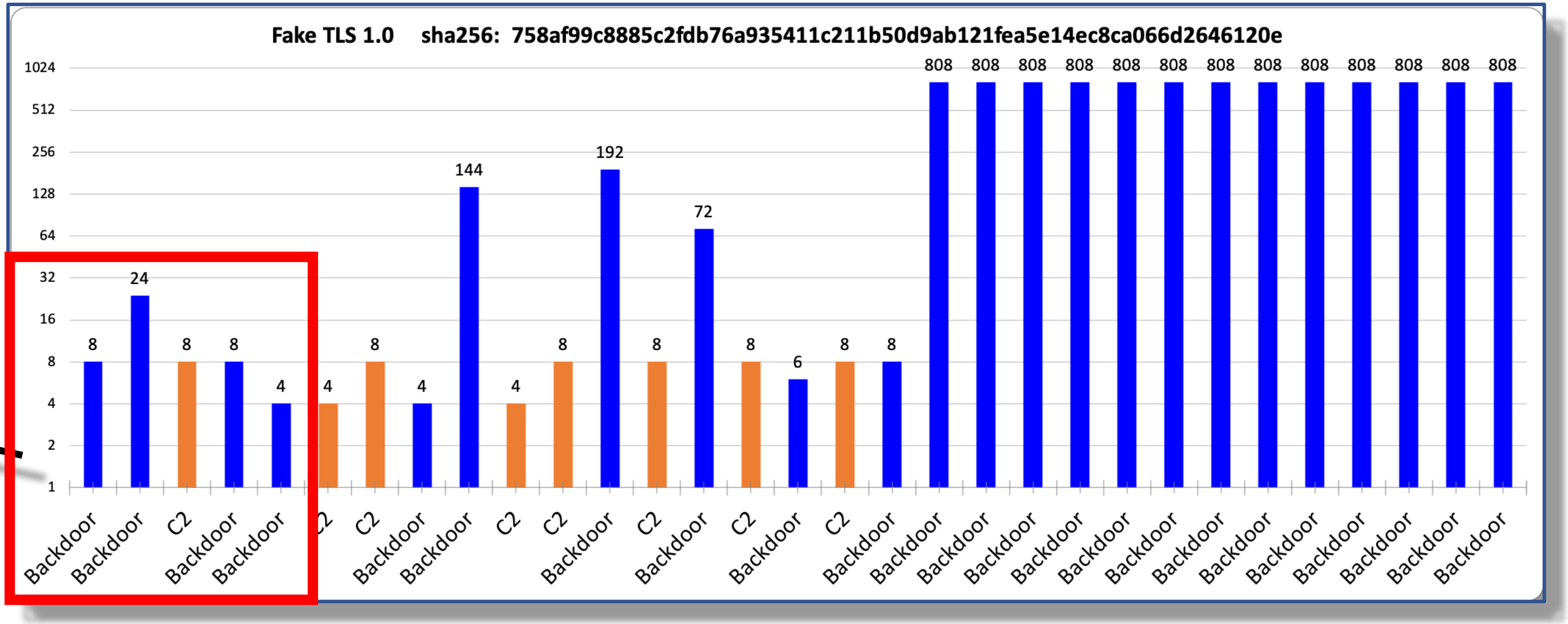


Maximum message size of 808 Bytes

C2 COMMS

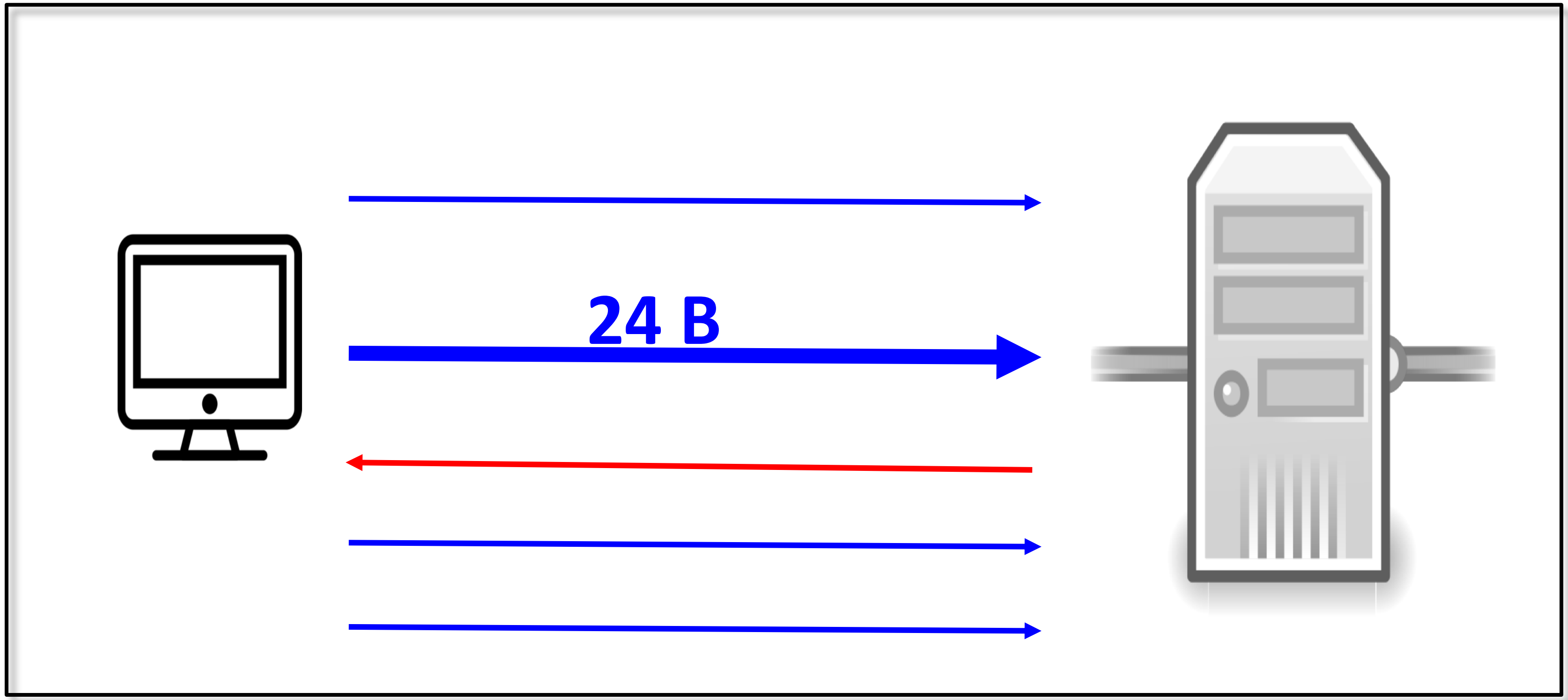
FakeTLS – interesting part shortly after handshake

The beginning of **REAL** comms has fixed size messages



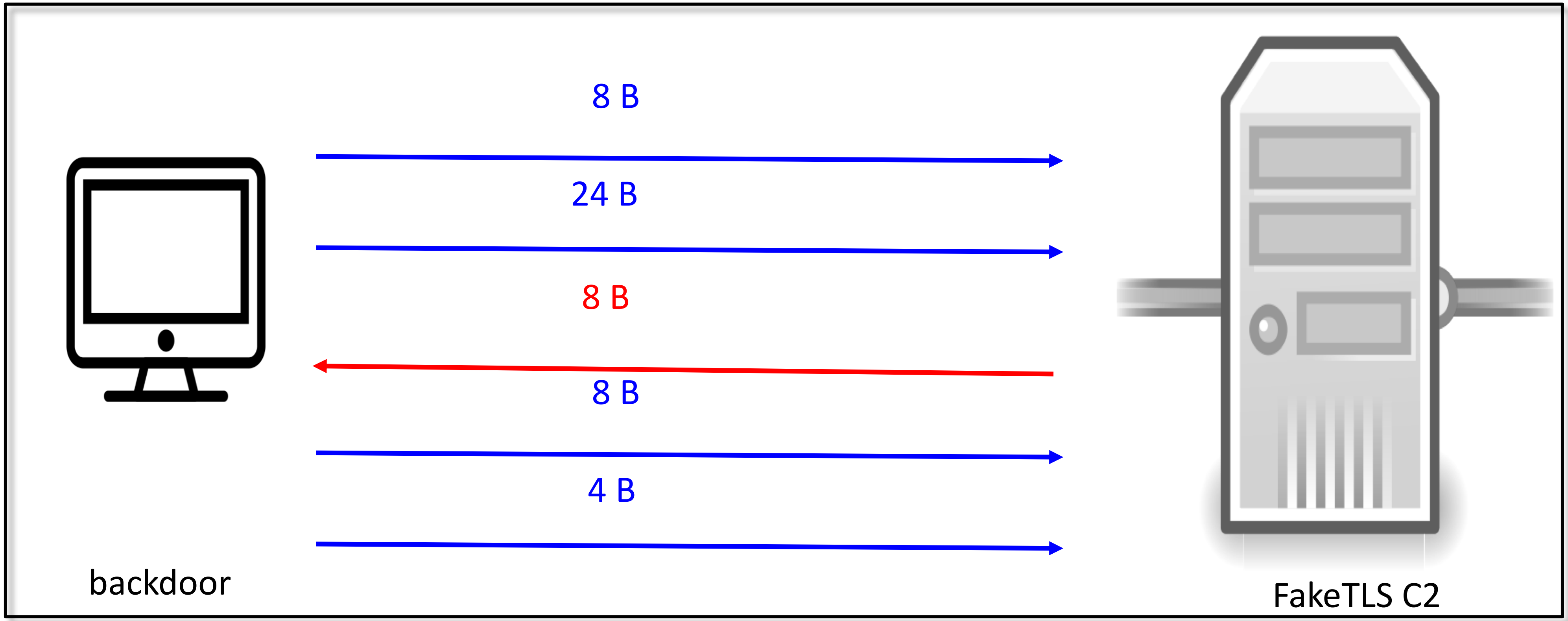
C2 COMMS

FakeTLS – is it really hardcoded?



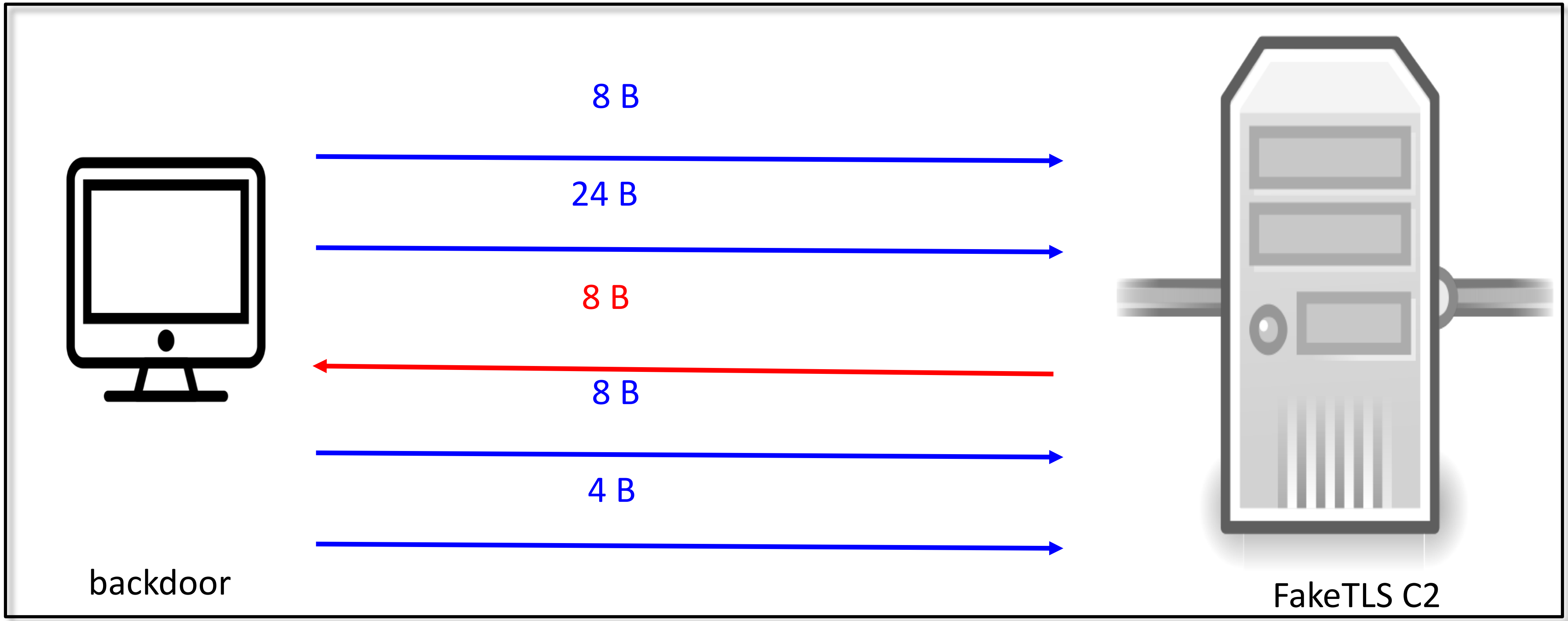
```
# Message 2 construction in code
push    0x17 # Encrypted Data Header in SSL message
push    1      #  TLS 1.0
lea edx, [esp + 0x34]
push 0x18    # 24 bytes - Encrypted Message Length
```

FakeTLS detection using SSL profiling



Analysing the **sizes of first 5 messages** of Encrypted Application Data (after TLS handshake) can help you detect traffic to **unknown C2** infrastructure that uses FakeTLS

FakeTLS – what’s wrong with those msg sizes?



In TLS algorithms every message is hashed (e.g. md5) for integrity check

length(md5(msg)) = 16B

8B < 16B ;)

FakeTLS – where to hunt unknown C2 infrastructure?

Reactive:

- own network traffic detection
- Can your network traffic analyser process TLS data after the handshake?

Proactive:

- pcaps from sandboxes e.g. Hybrid-Analysis

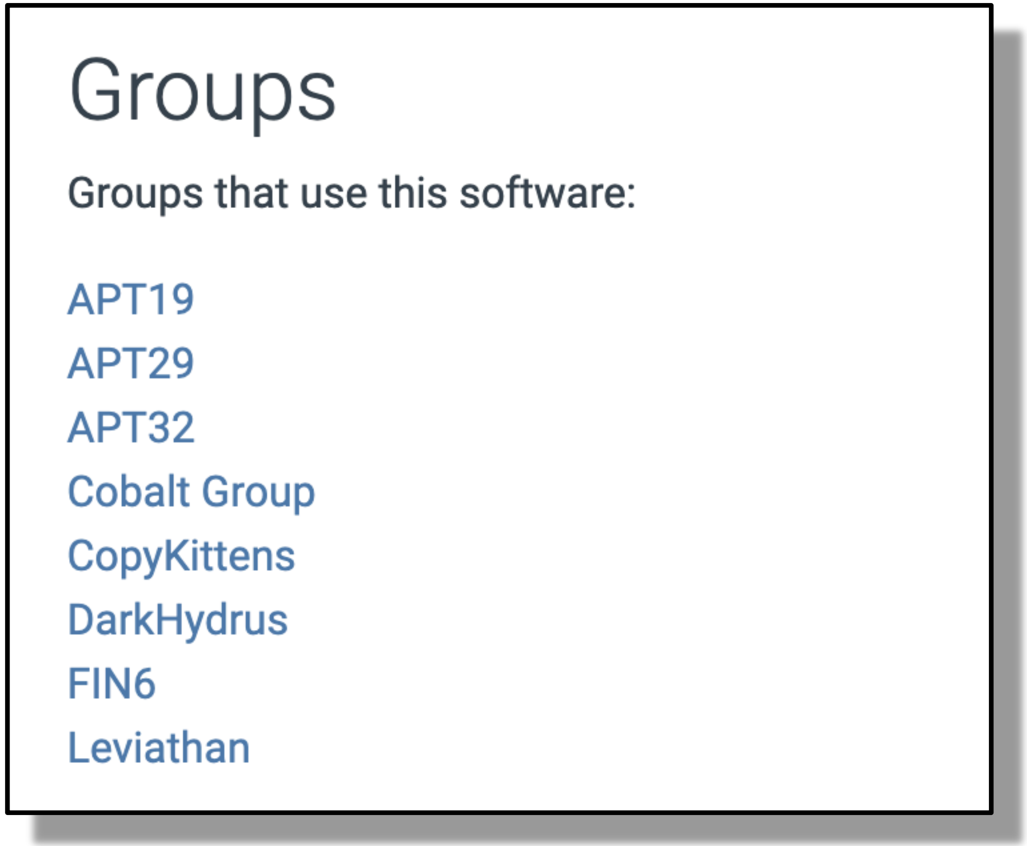
PART III
Let’s hunt them **early** – C2 scanning

NBA in 1990s – „Offense starts with defense”

http://b-rise.com

Quick intro to wide topic

COBALT STRIKE
ADVANCED THREAT TACTICS FOR PENETRATION TESTERS

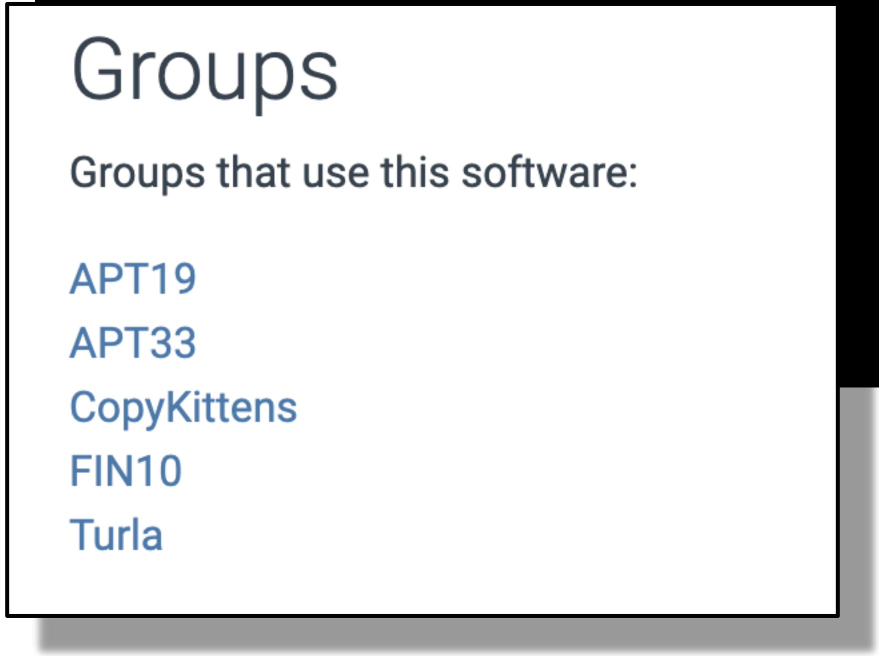


Groups

Groups that use this software:

APT19
APT29
APT32
Cobalt Group
CopyKittens
DarkHydrus
FIN6
Leviathan

```
================================================================================
Empire: PowerShell post-exploitation agent | [Version]: 0.5.1-beta
================================================================================
[Web]: https://www.PowerShellEmpire.com/ | [Twitter]: @harmj0y, @sixdub
================================================================================
```



Groups

Groups that use this software:

APT19
APT33
CopyKittens
FIN10
Turla

https://attack.mitre.org/

Finding defaults: #1 Cobalt Strike console port

Management console port for Teamserver is by default: **50050/tcp**

Finding defaults: #2 Cobalt Strike **idle** DNS answer

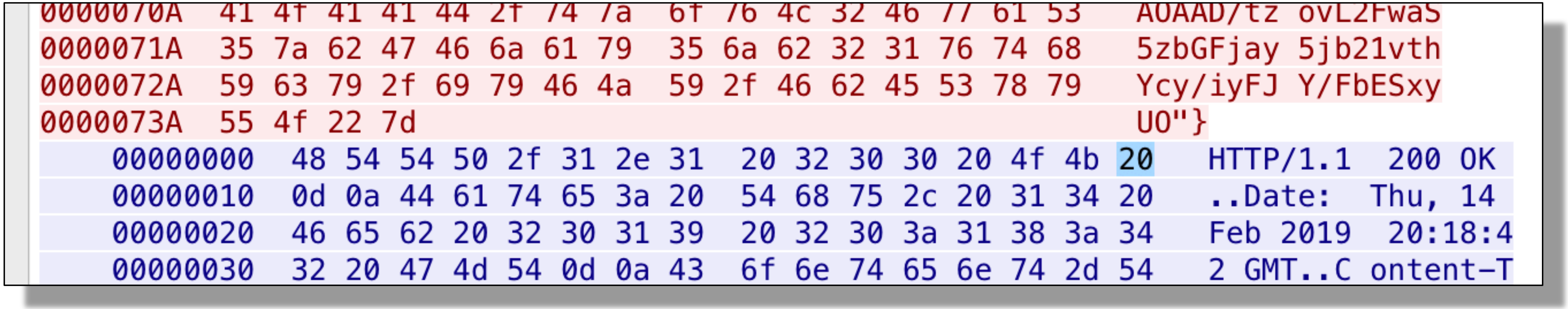
DNS answer for ANY request is: **0.0.0.0**

Finding defaults: #3 Cobalt Strike 404 answer

CS (NanoHTTPD) answers with:

```
HTTP/1.1 404 Not Found
Content-Type: text/plain
Date: Mon, 30 Feb 2019 13:37:00 GMT
Content-Length: 0
```

Finding defaults: #4 Cobalt Strike „space”



```
0000070A  41 4f 41 41 44 2f 74 7a  6f 76 4c 32 46 77 61 53   AOAAD/tz ovL2FwaS
0000071A  35 7a 62 47 46 6a 61 79  35 6a 62 32 31 76 74 68   5zbGFjay 5jb21vth
0000072A  59 63 79 2f 69 79 46 4a  59 2f 46 62 45 53 78 79   Ycy/iyFJ Y/FbESxy
0000073A  55 4f 22 7d                                        UO"}
    00000000  48 54 54 50 2f 31 2e 31  20 32 30 30 20 4f 4b 20   HTTP/1.1  200 OK
    00000010  0d 0a 44 61 74 65 3a 20  54 68 75 2c 20 31 34 20   ..Date:  Thu, 14
    00000020  46 65 62 20 32 30 31 39  20 32 30 3a 31 38 3a 34   Feb 2019  20:18:4
    00000030  32 20 47 4d 54 0d 0a 43  6f 6e 74 65 6e 74 2d 54   2 GMT..C ontent-T
```

CS responds with additional space after **200 OK**

Hunting for NanoHTTPD servers.

Corrected in Cobalt Strike v. 3.13

Conclusion

- Adversary tools and procedures very often have **patterns**
- Threat analyst job is to **uncover** human traces and adversaries weaknesses
- Burn the **defaults**, burn what is **known** (opensource, commercial C2)