Network Threat Hunter Training

Level 1

Thanks to our sponsors!

ACTIVE | COUNTERMEASURES

Other courses I'm teaching

- Advanced Network Threat Hunting
  - Shooting for Sept - Oct
  - $495

https://www.antisyphontraining.com/advanced-network-threat-hunting-w-chris-brenton/

- Getting Started with Packet Decoding
  - July 12 - 15
  - Pay what you can - $30+

https://www.antisyphontraining.com/event/getting-started-with-packet-decoding-w-chris-brenton/

Before we get started

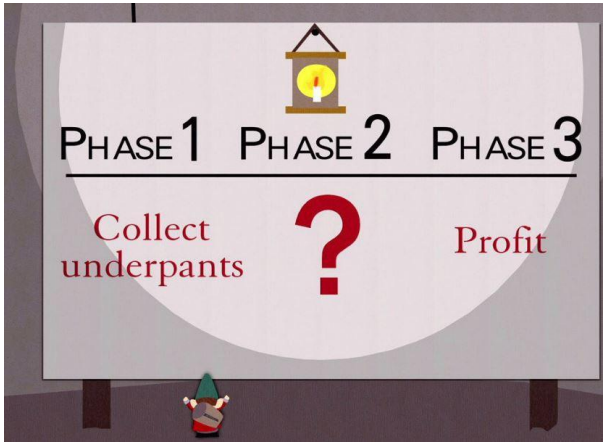
- You'll need the class VM to do the labs
- Or run the install script
- Or deploy on DigitalOcean
- Login info:
  - Name: thunt
  - Pass: aybab2u
- This should have been done before class :-)
- Slides are available on Discord

Logistics

- 10 minute break at top of each hour
- 20 minute break at 3 hour point
- Use the Discord channel for discussion
  - #acm-webcast-chat channel
- The team is monitoring for your questions

In this webcast

- I'm going to question some industry accepted standard practices
  - Because what we are doing is broken
  - And it's not getting any better
  - Will diverge from the norm
- Please keep an open mind
- Prime cognitive bias fodder



Modern attackers

- The vision of a lone hacker in the basement is dangerously outdated
- It's about profit, not mass infection
  - Attacks are now well funded
- Attacks are now targeted which means:
  - They do their homework on your environment
  - Malware is customized for your campaign
  - Attack infrastructure is customized as well
- ***Attackers innovate for each new target***

How we (try to) catch the bad guys

- Centralized log collection
- Write "signatures" to identify patterns that may indicate an attack
  - Patterns in the log messages
  - Matches against intel feeds
- Alert on signature matches
- Follow up on alerts

Limitations of system logging

- Syslog was not designed for security
  - Facility 13 is "security/log audit"
  - But rarely used in a general security context
  - More appropriate as a severity level
  - But there is no "security" severity level
- No standard for message context
  - Different platforms log events differently
  - Different applications log events differently
- Decoder ring not included

Limitations of deployment

- Every device and system?
- Are you sure?
- Are you REALLY sure?
  - I have yet to see an environment that can accurately make this claim
  - Even when you log, adversaries can disable this
- **"Fail open" system**
  - Can access Internet without logging and no alert
  - Can you detect disabled logging?

What are signatures?

- Basically RegEx for logs
- Match known bad patterns
- Because adversaries have stopped innovating and we now know all of the possible bad patterns they can use
- Oh wait…
- Sigs are also the 1990's anti-virus model

Lack of innovation

- Log RegEx matching is old
  - Older than IDS
  - Older than firewalls
- First SANS logging course early 2000's
- Not much has changed

OK to still wear parachute pants?

Are we getting better at detection?

- Interesting nuggets in Mandiant's M-Trends 2022 report
- Dwell time is down to less than 30 days
  - Skewed by Ransomware at 4 days
  - But drop shows no correlation to breach effect
- For threats Mandiant investigated:
  - 58% had been in place over 700 days
  - 20% had been in place over 90 days

https://www.mandiant.com/media/15671

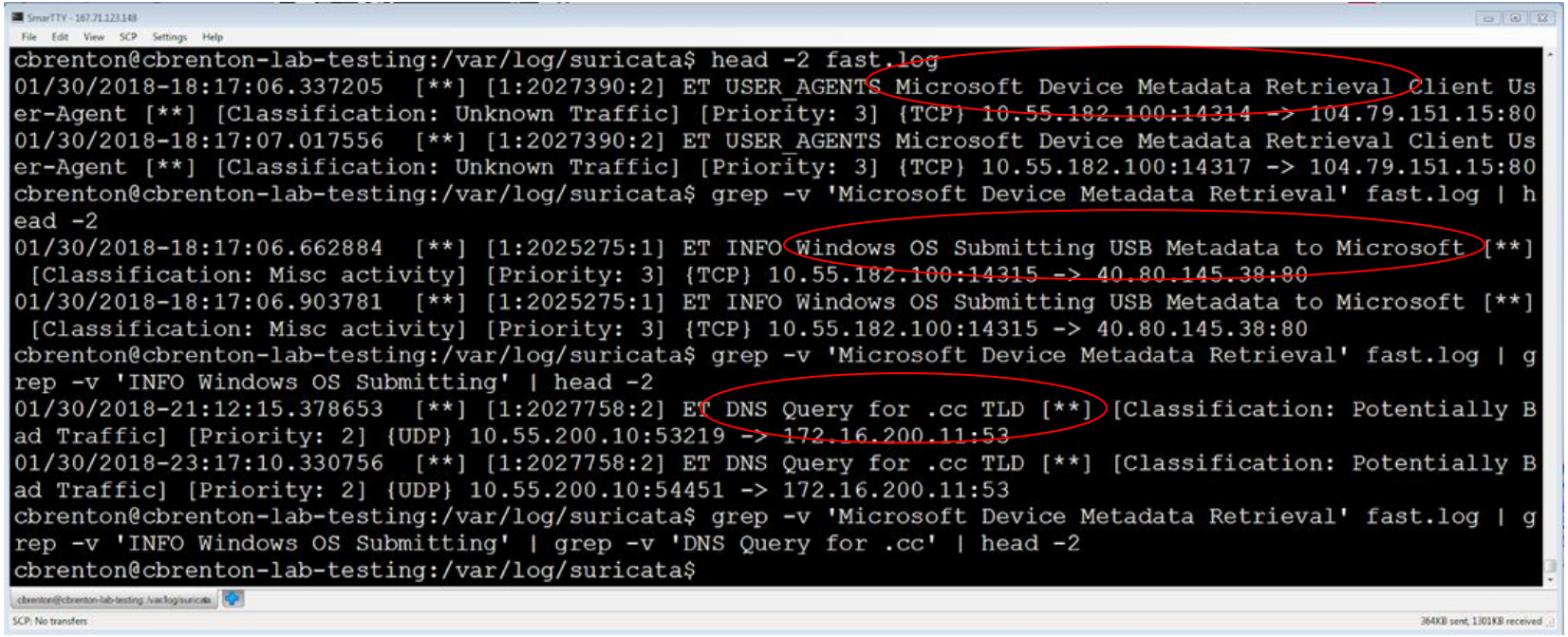
So is log review threat hunting?

- Just to review
  - Protocol can't describe security events
  - It's a fail open system
  - We try to pattern match on old attack patterns
  - False positive rates are extremely high
  - It's old technology
- The data says otherwise
- This process is clearly broken
- We need to assess new ideas and improve

I'm good, I use threat intel feeds

- Match on IP because someone said it's bad
- Also based on 1990's AV technology
- Is the data really actionable?
  - Adversaries frequently change IPs and DNS
  - Tend to use shared IP space
  - The accuracy is dependent on the reporter
- A threat intel match does not mean you've prevented an attack

Can I threat hunt with my NIDS?

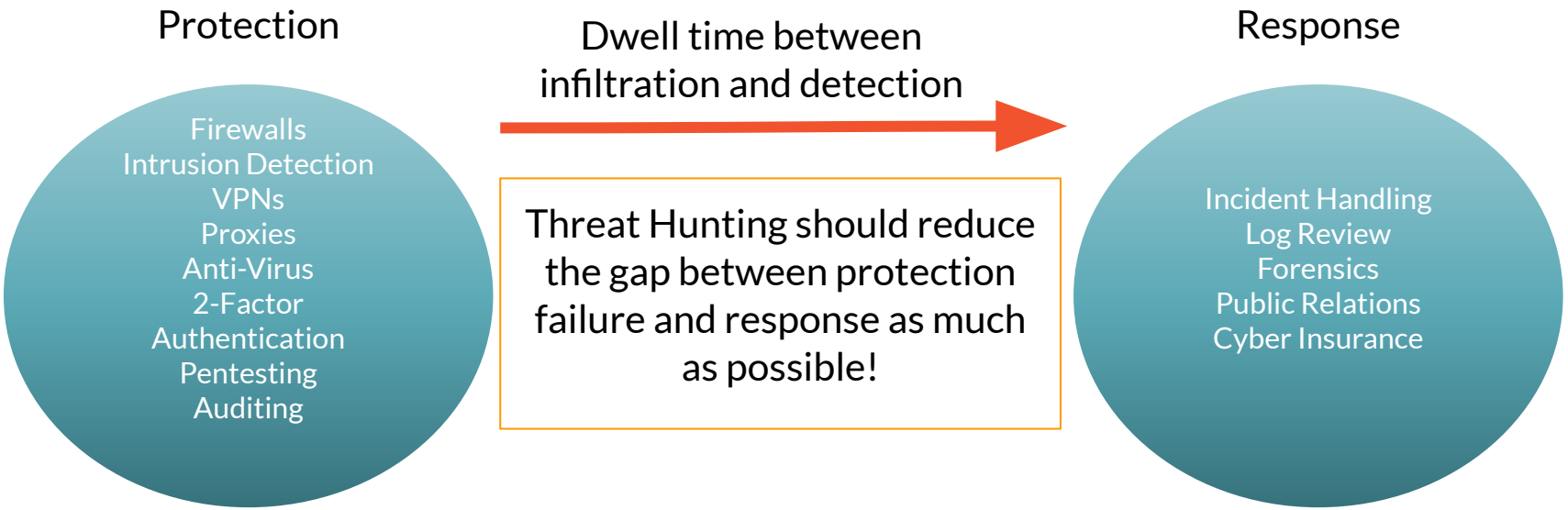


But empire and dnscat2 were missed

What Threat Hunting should be

- A proactive validation of all systems connected to the organization's network
- Needs to include all systems
  - Desktops, laptops, cellphones, tablets
  - Servers, network gear, printers
  - IoT, IIoT, any type of Internet "Thing"
- Execute without making assumptions
- Deliverable is a compromise assessment

The Purpose of Threat Hunting



What threat hunting is not

- Managing SOC alerts
- Check logs for suspect activity
- Check dashboards for unusual activity
- Monitor and respond to EDR alerts
- These are all reactive activities
- Threat hunting is a proactive process

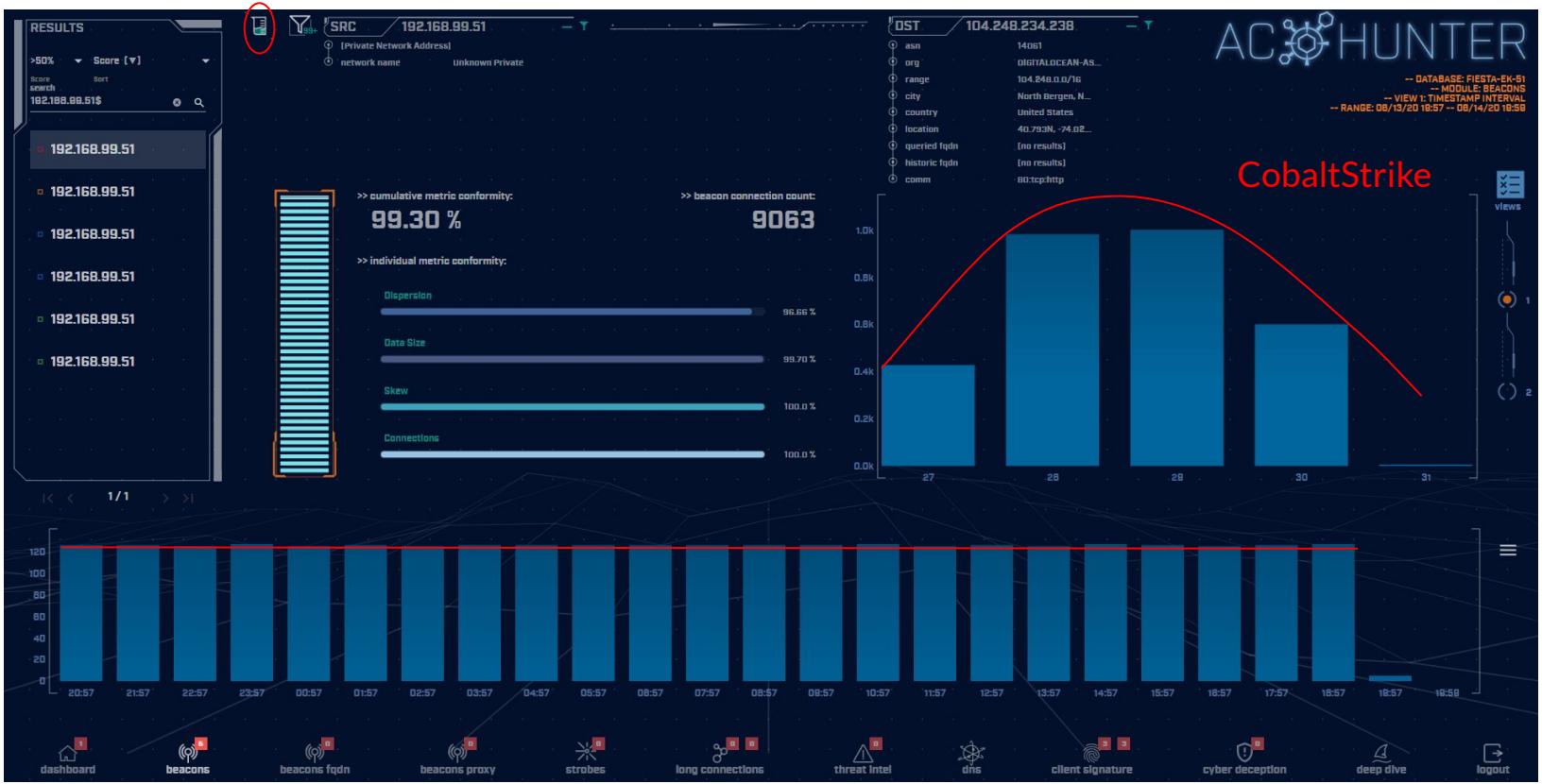
The process of threat hunting

- Review the integrity of every device
  - Desktops, servers, network gear, IoT, IIoT, etc.
- Generate one of 3 dispositions
  - I'm pretty certain the system is safe
  - I'm pretty certain the system is compromised
  - I'm unsure of state so will collect additional info to derive one of the above two results
- Leverage context for host log review

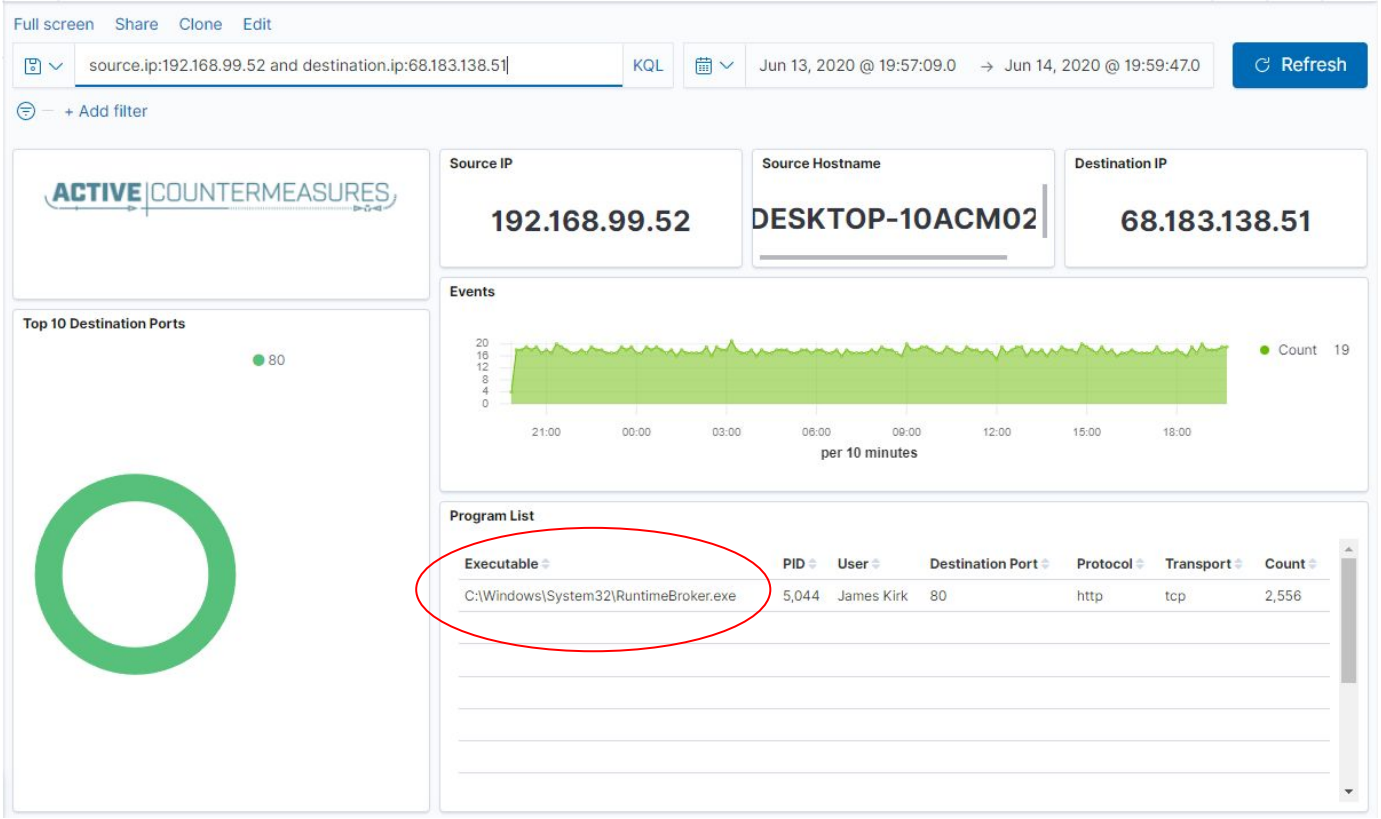
Proposal - Start with the network

- The network is the great equalizer
  - You see everything, regardless of platform
  - High level assessment of the terrain
- You can hide processes but not packets
- Malware is usually controlled
  - Which makes targeting C2 extremely effective
  - Identify compromise when C2 "calls home"
  - Must be frequent enough to be useful
- Wide view so you can target from there

Start on the network



THEN pivot to the system logs



Don't cross "the passive/active line"

- All threat hunting activity should be undetectable to an adversary
- Passive in nature
  - Review packets
  - Review SIEM logs
- If active techniques are required, we must trigger incident response first
  - Example: Isolating the suspect host
  - Example: Running commands on suspect host

C2 Detection Techniques

Where to Start

- Traffic to and from the Internet
  - Monitor internal interface of firewall
- Packet captures or Zeek data
- Analyze in large time blocks
  - More data = better fidelity
  - Minimum of 12 hours, 24 is ideal
- Analyze communications in pairs
  - Every outbound session passing the firewall
  - Ignore internal to internal (high false positive)

Threat score system - Updated

- Our job is to disposition IPs
- How do you know when to make a choice?
- A numeric system can help guide you
  - Score of 0 = system is safe
  - Score of 100 = system is compromised
- Score modifiers
  - Major - A clue that strongly indicates integrity state
  - Minor - A clue that peripherally indicates integrity state

Why not score each system?

- Data points are easily conflated
  - Points for persistence, JA3 hash & intel match
  - But are they all part of the same session?
- You are not hunting systems, you are hunting connection persistency
  - Your process & scoring should reflect this
- Fewer distractions
  - Focus on the persistence
  - Don't lose focus going after unrelated attributes

Threat hunting process order

- Connection persistency
- Business need for connection?
- Abnormal protocol behaviour
- Reputation check of external IP
- Investigation of internal IP
- Disposition
  - No threat detected = add to safelist
  - Compromised = Trigger incident handling

Does targeting C2 have blind spots?

- Attackers motivated by gain
  - Information
  - Control of resources
- Sometimes "gain" does not require C2
  - Just looking to destroy the target
  - Equivalent to dropping a cyber bomb
  - We are talking nation state at this level
- NotPetya
  - Worm with no C2 designed to seek and destroy

Techniques Vs Methodology

- We are going to deep dive on finding C2
- It's important to understand what needs to happen "under the hood"
- Some of these techniques don't scale
  - Manually breaking out connection pairs
  - But that's OK
- Will focus on tools in a later module
- For now, focus on just the techniques

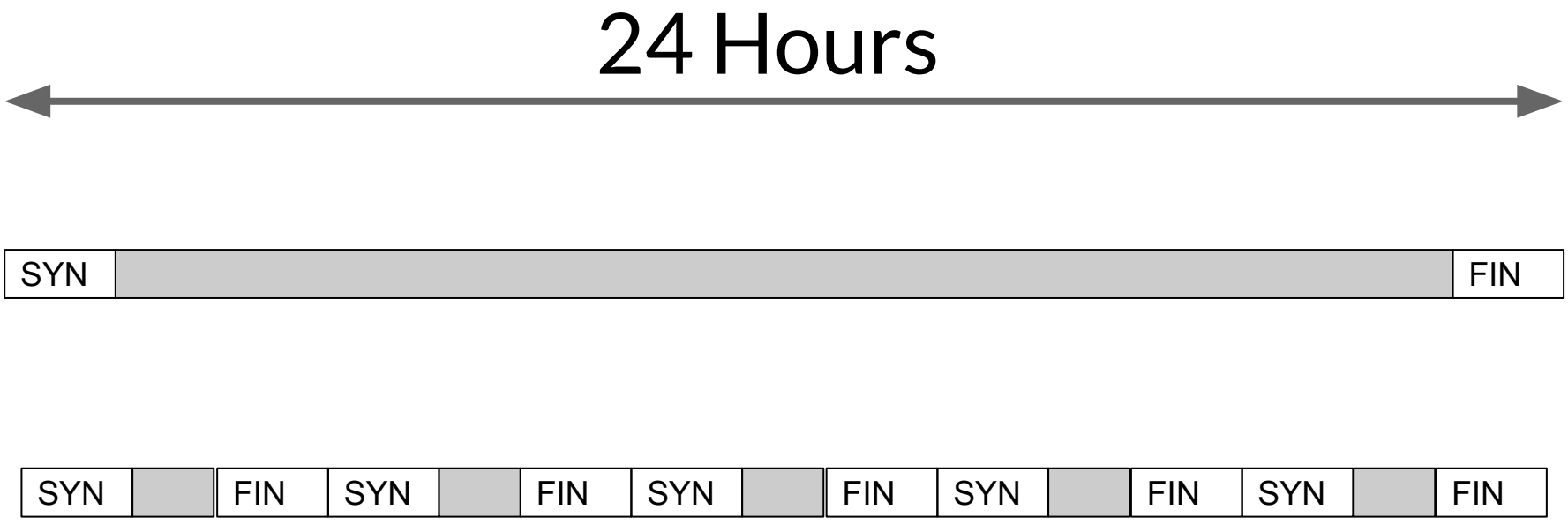
Bad guys Vs. Red Teams

- Bad guys = C2 is part of a business model
- Red team = C2 is why they get paid
- Much harder to detect red team C2 than the real bad guys
  - In the wild, most evil C2 beacons <= 1/minute
  - Red team on long term contract <= 1/week
- Focus will be on the bad guys

Long connections

- You are looking for:
- Total time for each connection
  - Which ones have gone on the longest?
- Cumulative time for all pair connections
  - Total amount of time the pair has been in contact
- Can be useful to ignore ports or protocols
  - C2 can change channels

Long connection examples



Connection timing from Zeek

```
cbrenton@zeek-3-3-rc2:/opt/bro/logs/2019-07-17$ zcat conn.00\:00\:00-01\:00\:00.log.gz | head -10
#separator \x09
#set_separator  ,
#empty_field    (empty)
#unset_field    -
#path   conn
#open   2019-07-17-00-00-00
#fields ts      uid     id.orig_h       id.orig_p       id.resp_h       id.resp_p       proto   ser
vice    duration        orig_bytes      resp_bytes      conn_state      local_orig      local_resp
missed_bytes    history orig_pkts       orig_ip_bytes   resp_pkts       resp_ip_bytes   tunnel_pare
nts
#types  time    string  addr    port    addr    port    enum    string  interval        count   cou
nt      string  bool    bool    count   string  count   count   count   count   set[string]
1563321592.266216       CRP5W73KxGUYtn2XQh      185.176.27.30   48086   104.248.191.205 20391   tcp
-       0.265051        0       0       REJ     F       F       0       SrR     2       80      1
40      (empty)
1563321592.266218       CjZ8aQ2AoHDrsheUAj      185.176.27.30   48086   104.248.191.205 20391   tcp
-       0.265051        0       0       REJ     F       F       0       SrR     2       80      1
40      (empty)
cbrenton@zeek-3-3-rc2:/opt/bro/logs/2019-07-17$
```

less -S conn.log

```
#separator \x09
#set_separator      ,
#empty_field        (empty)
#unset_field        -
#path               conn
#open               2021-10-13-15-47-50
#fields             ts                  uid                 id.orig_h           id.orig_p
#types              time                string              addr                port
1599652681.658987   Ci09jy2pQa8n4Nhpnk  192.168.125.105     43742               91.189.88.142
1599652681.909864   C7ebxg76JCvTenVC4   192.168.125.105     55418               91.189.91.38
1599652682.160692   Ciy54Bgp1AAP3g3Ai   192.168.125.105     56374               91.189.88.152
1599652682.411596   CIJ8Xh4WAfju0gEub6  192.168.125.105     36338               91.189.91.39
1599652681.643945   CfGhY0bXVYn9DET8    127.0.0.1           33915               127.0.0.53
1599652681.644119   CPCY5P1CD1nAxjVHG7  192.168.125.105     53240               8.8.8.8
1599652681.651291   CiKUI24evOEENjqzg5  127.0.0.1           58816               127.0.0.53
1599652681.651392   CEY8xNH9QzkxBCGvl   192.168.125.105     38521               8.8.8.8
1599652681.651543   CZs8CI12RnoQOgn0dg  192.168.125.105     55633               8.8.8.8
```

Longest duration with Zeek

```
thunt@thunt-labs:~/lab1$ cat conn.log | zeek-cut id.orig_h id.resp_h duration
 | sort -k 3 -rn | head
192.168.99.51   167.71.97.235   86389.659357
192.168.99.51   104.248.234.238 243.768999
192.168.99.51   104.118.9.117   166.139547
192.168.99.51   72.21.91.29     134.888177
192.168.99.51   52.184.216.246  129.075227
192.168.99.51   52.167.249.196  128.957107
192.168.99.51   52.184.216.246  128.481757
192.168.99.51   13.107.5.88     128.346889
192.168.99.51   52.179.219.14   128.116421
192.168.99.51   13.107.5.88     128.042647
thunt@thunt-labs:~/lab1$
```

Cumulative talk time with Zeek

```
thunt@thunt-labs:~/lab1$ cat conn.log | zeek-cut id.orig_h id.resp_h duration | sort
| grep -v -e '^$' | grep -v '-' | datamash -g 1,2 sum 3 | sort -k 3 -rn | head
192.168.99.51   167.71.97.235   86389.659357
192.168.99.51   52.179.219.14   4067.394413
192.168.99.51   52.184.217.56   2936.172839
192.168.99.51   52.184.216.246  2825.858
192.168.99.52   239.255.255.250 2507.626732
fe80::d048:42e0:8448:187c       ff02::c 2434.977049
192.168.99.51   239.255.255.250 2374.546469
fe80::2126:bcd7:16f4:8cdb       ff02::c 2368.234679
192.168.99.51   13.107.5.88     1317.047871
192.168.99.51   52.167.249.196  868.46966
thunt@thunt-labs:~/lab1$
```

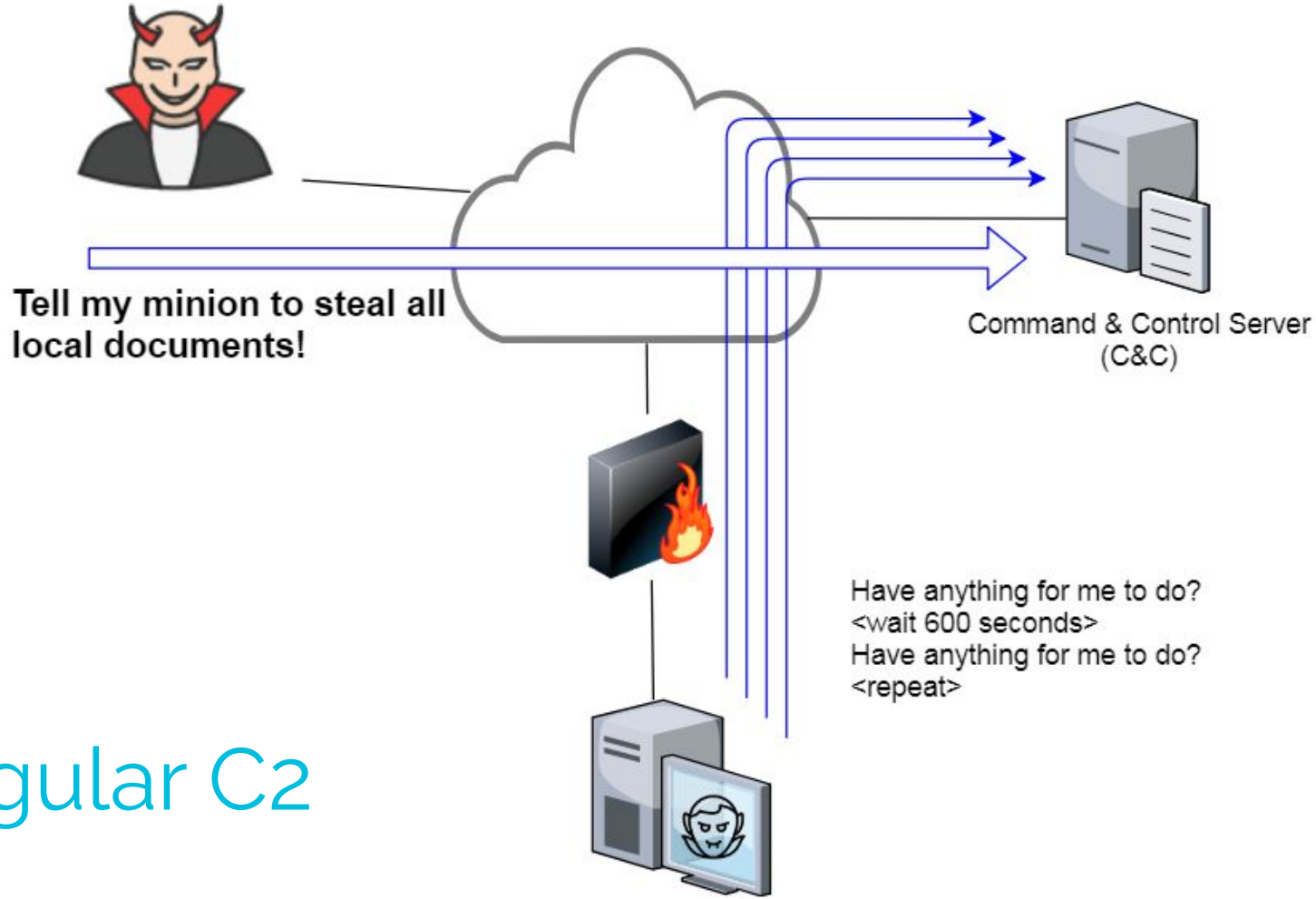
What about firewalls?

- Surprisingly hard to get this info
- "Timing" tends to be TTL, not duration
- BSD
  - pftop - output connection age in seconds
- Junos
  - show security flow session extensive node all
  - Duration in seconds

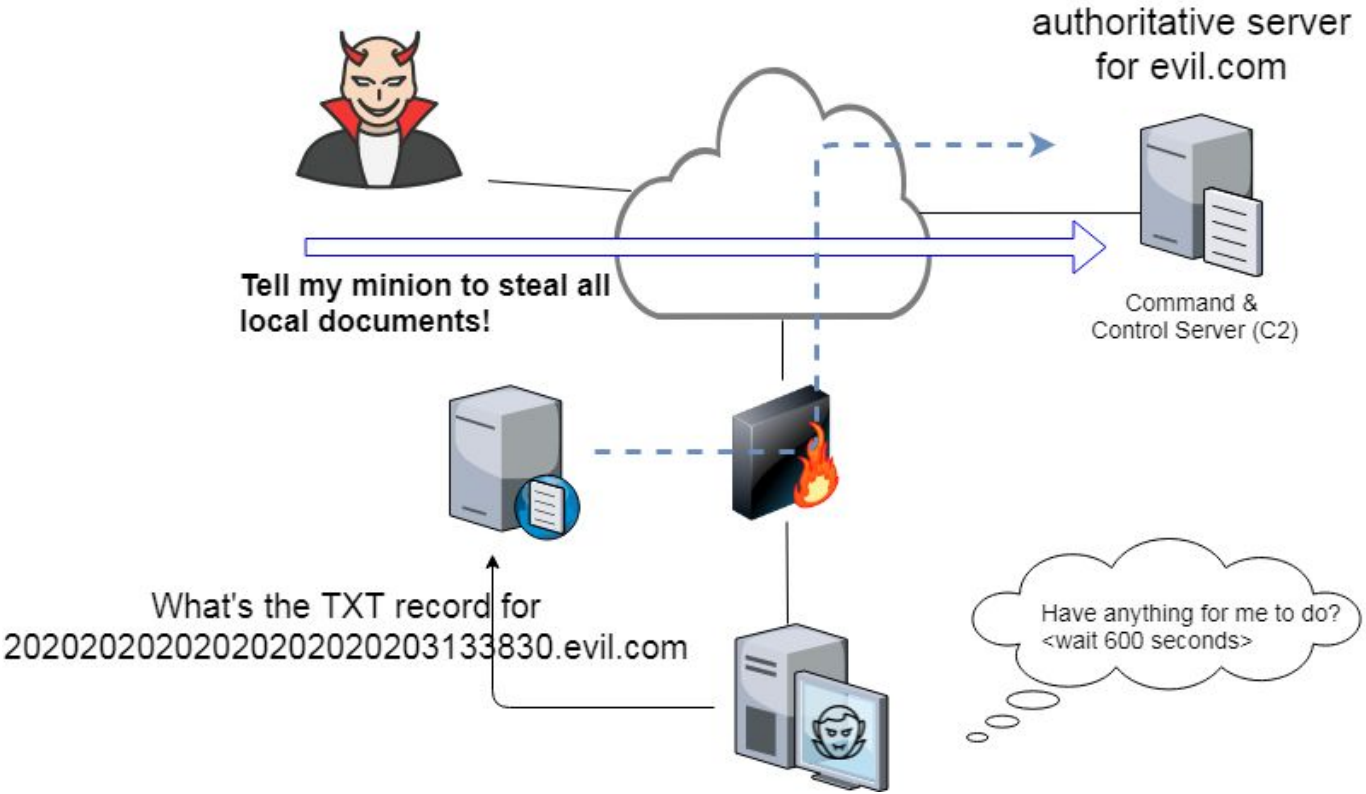
What is a beacon?

- Repetitive connection establishment between two IP addresses
  - Easiest to detect
- Repetitive connection establishment between internal IP and FQDN
  - Beacon broken up over multiple IP's
    - Usually a CDN provider
  - Target IPs also destination for legitimate traffic
  - Far more difficult to detect

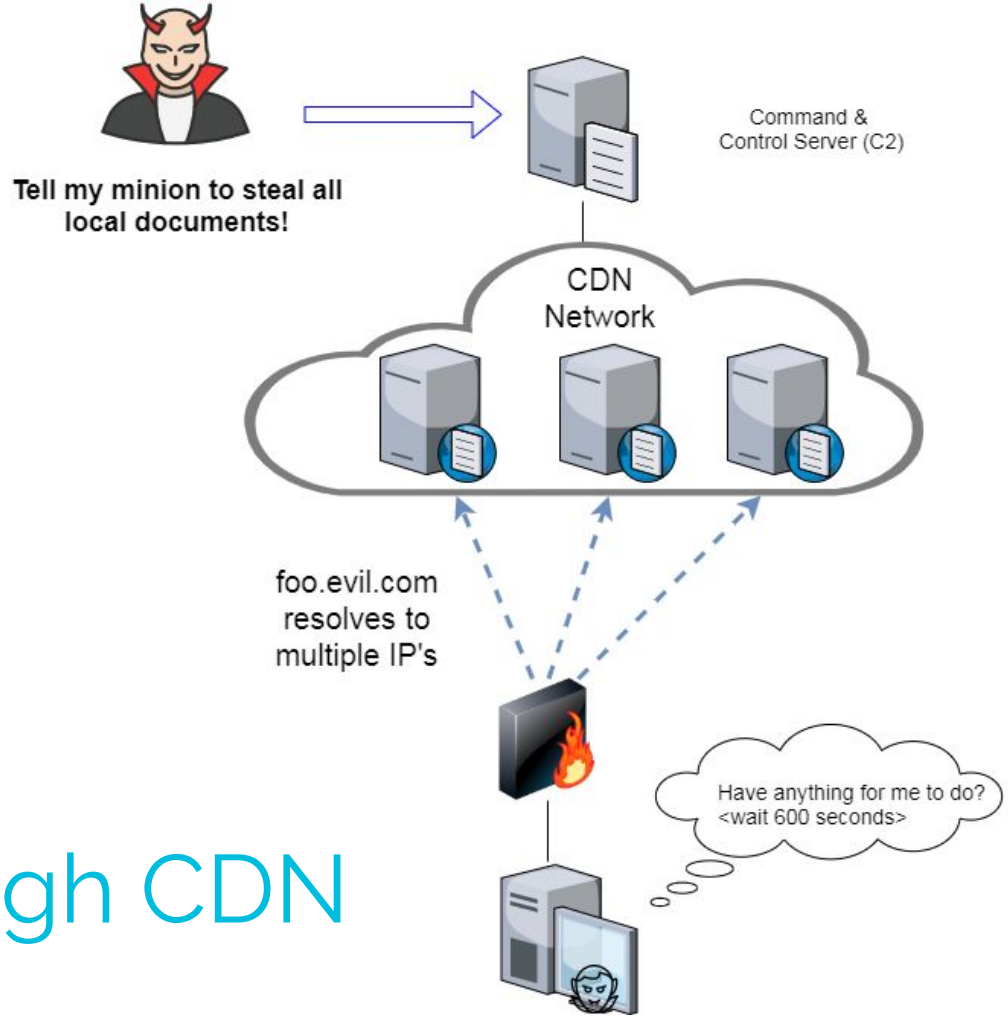
Regular C2


Tell my minion to steal all local documents!
Command & Control Server (C&C)
Have anything for me to do?
<wait 600 seconds>
Have anything for me to do?
<repeat>

C2 over DNS


authoritative server
for evil.com
Tell my minion to steal all
local documents!
Command &
Control Server (C2)
What's the TXT record for
20202020202020202020203133830.evil.com
Have anything for me to do?
<wait 600 seconds>

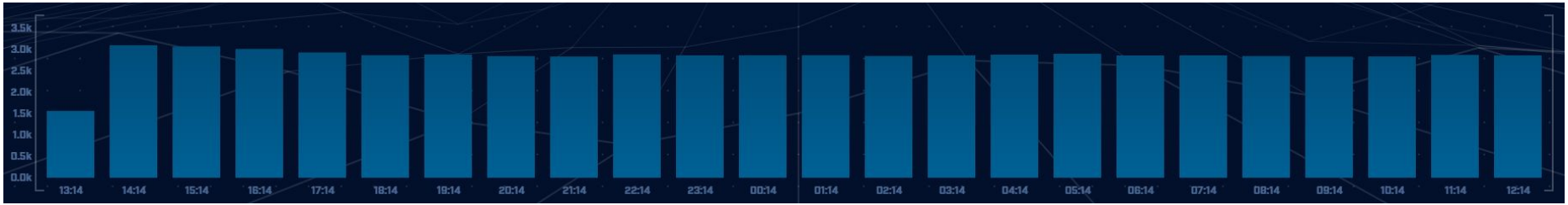
C2 through CDN



Beacon detection based on timing

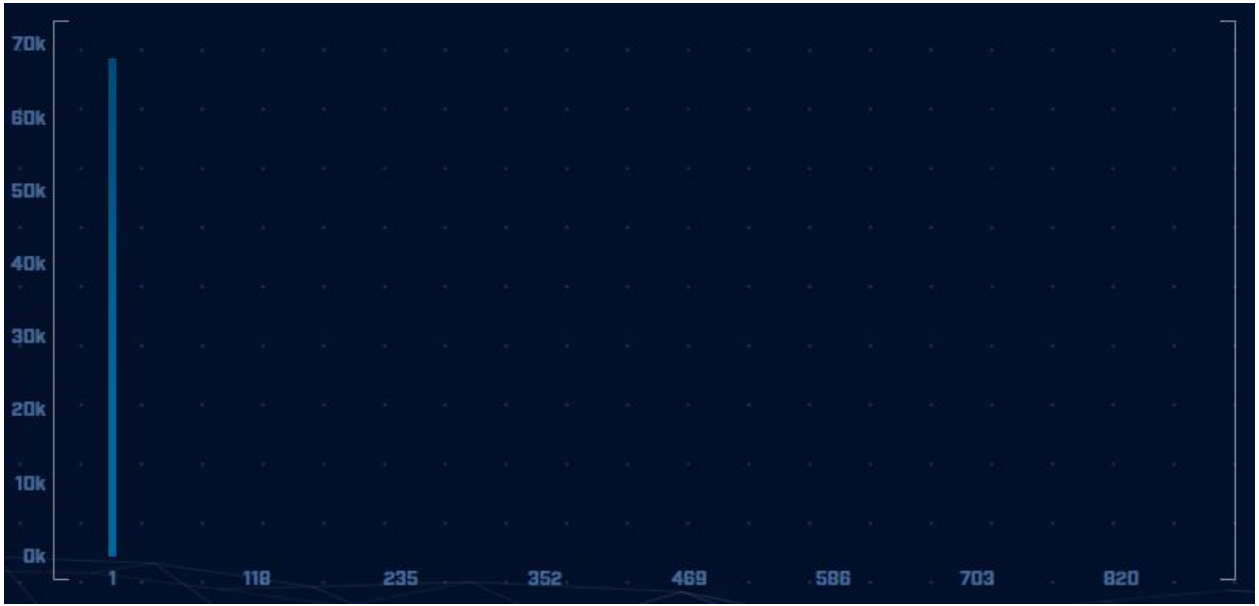
- May follow an exact time interval
  - Technique is less common today
  - Detectable by k-means
  - Potential false positives
- May introduce "jitter"
  - Vary connection sleep delta
  - Avoids k-means detection
  - False positives are extremely rare
- Short enough delta for terminal activities

Connection quantity VS time



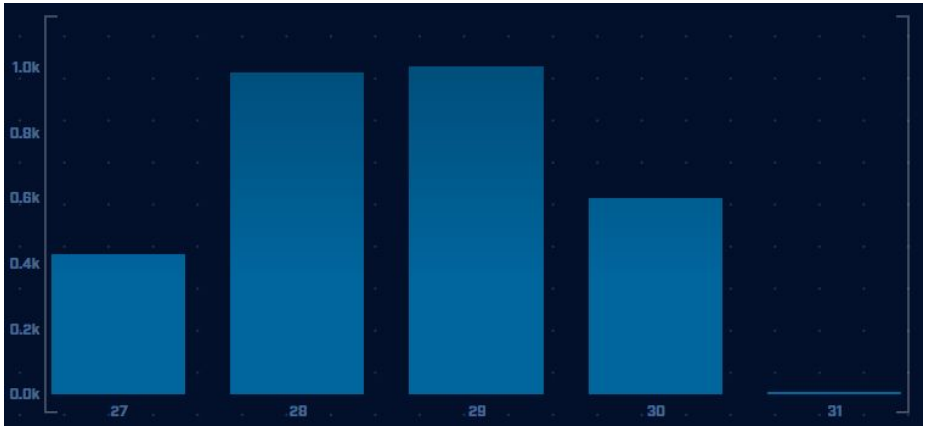
Each bar represents the number of times the source connected to the destination during that one hour time block

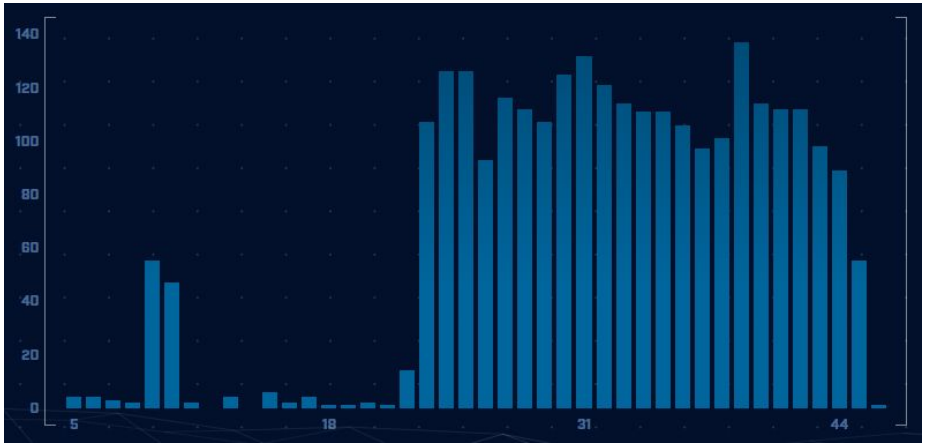
Connect time deltas with no jitter



How often a specific time delta was observed

Connection time deltas with jitter



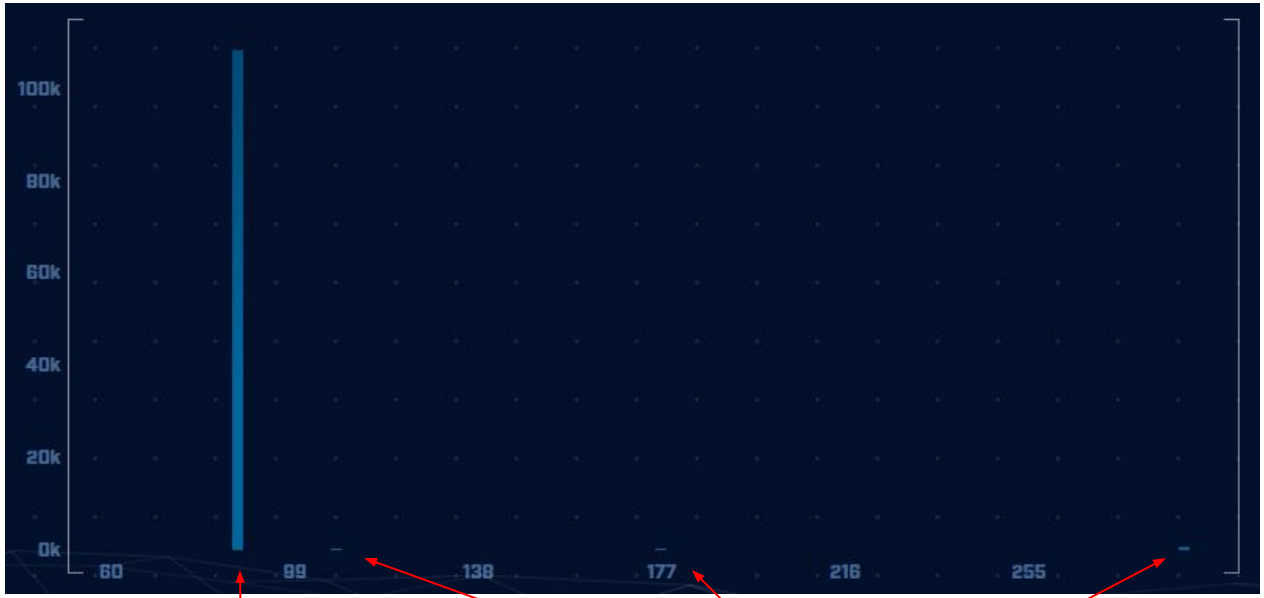


Cobalt Strike will typically produce a bell curve

Detection based on session size

- Focuses on detection of the heartbeat
- Variations from the heartbeat indicate activation of C2 channel
- Session size can help reveal info regarding commands being issued
- Possible to randomly pad but this is extremely rare

Session size analysis



Detecting beacons with jitter

- Easier to detect when normalized out over long periods of time
  - Average the time deltas for each hour
  - Plot over 24 hours
- Should make a beacon even more suspect
  - False positives don't obscure their beacon timing
  - High probability of being evil

Is there a business need?

Can I get false positives?

- Sort of…
- Checking for connection persistency
- Then checking for business need
- It's possible to have persistent connections with a legit business need
  - NTP
  - Windows Notification Services
  - Checking for patches

C2 Detection Techniques Part 2

What next?

- You've identified connection persistence
- You can't identify a business need
- Next steps
  - Protocol analysis
  - Reputation check of external target
  - Investigate internal IP address

Unexpected app or port usage

- There should be a business need for all outbound protocols
- Research non-standard or unknown ports
  - TCP/5222 (Chrome remote desktop)
  - TCP/5800 & 590X (VNC)
  - TCP/502 (Modbus)

Unknown app on standard port

- C2 wants to tunnel out of environment
  - Pick a port likely to be permitted outbound
  - Does not always worry about protocol compliance
- Check standard ports for unexpected apps
  - Indication of tunneling
- Different than app on non-standard port
  - This is sometimes done as "a feature"
  - Example: SSH listening on TCP/2222

Zeek decodes many apps

- Detect over 50 applications
  - HTTP, DNS, SIP, MYSQL, RDP, NTLM, etc. etc.
- Fairly easy to add new ones
  - Example: HL7 if you are in healthcare
- Checks all analyzers for each port
- Does not assume WKP = application

Zeek example

```
thunt@thunt-labs:~/lab1$ cat conn.log | zeek-cut id.orig_h id.resp_h id.resp_p
 proto service orig_ip_bytes resp_ip_bytes | column -t | head
192.168.99.51               104.248.234.238  80    tcp   http  689         403
192.168.99.51               23.223.200.136   80    tcp   -     80          40
192.168.99.51               104.248.234.238  80    tcp   http  729         443
192.168.99.52               224.0.0.251      5353  udp   dns   344         0
fe80::d048:42e0:8448:187c   ff02::fb         5353  udp   dns   424         0
fe80::d048:42e0:8448:187c   ff02::1:3        5355  udp   dns   81          0
192.168.99.52               224.0.0.252      5355  udp   dns   61          0
fe80::d048:42e0:8448:187c   ff02::1:3        5355  udp   dns   81          0
192.168.99.52               224.0.0.252      5355  udp   dns   61          0
192.168.99.51               104.248.234.238  80    tcp   http  689         403
thunt@thunt-labs:~/lab1$
```

Unexpected protocol use

- Attackers may bend but not break rules
- This can result in:
  - Full protocol compliance
  - Abnormal behaviour
- Need to understand "normal"
  - For the protocol
  - For your environment

Example: Too many FQDNs

- How many FQDNs do domains expose?
  - Most is < 10
  - Recognizable Internet based vendors 200 - 600
    - Microsoft
    - Akamai
    - Google
    - Amazon
- Greater than 1,000 is suspicious
- Could be an indication of C2 traffic

Detecting C2 over DNS

- Capture all DNS traffic
  - Capture tool of your choice
  - Longer the capture time, the better
- Filter so it's DNS traffic only
- Extract to text so we can sort and count
- Review total FQDNs per domain

Counting FQDNs per domain

```
cbrenton@cbrenton-lab-testing:~/lab-thunt$ tshark -r thunt-lab.pcapng -T fields -e dn
s.qry.name | sort | uniq | rev | cut -d '.' -f 1-2 | rev | sort | uniq -c | sort -rn
| head -10
  62468 r-1x.com
    154 akamaiedge.net
    125 akadns.net
    121 edgekey.net
    104 amazonaws.com
     67 microsoft.com
     51 dynect.net
     45 parsely.com
     44 akam.net
     43 cloudfront.net
cbrenton@cbrenton-lab-testing:~/lab-thunt$
```

Breaking it down

```
cbrenton@cbrenton-lab-testing:~/lab-thunt$ tshark -r thunt-lab.pcapng -T fields -e dn
s.qry.name | sort | uniq | head -4

0000011239458783cf.dnsc.r-1x.com
00000176d2f1ce66e2.dnsc.r-1x.com
0001011239458783cf.dnsc.r-1x.com
cbrenton@cbrenton-lab-testing:~/lab-thunt$ tshark -r thunt-lab.pcapng -T fields -e dn
s.qry.name | sort | uniq | rev | head -4

moc.x1-r.csnd.fc3878549321100000
moc.x1-r.csnd.2e66ec1f2d67100000
moc.x1-r.csnd.fc3878549321101000
cbrenton@cbrenton-lab-testing:~/lab-thunt$ tshark -r thunt-lab.pcapng -T fields -e dn
s.qry.name | sort | uniq | rev | cut -d '.' -f 1-2 | rev | head -4

r-1x.com
r-1x.com
r-1x.com
```

Show all instances of unique FQDNs queried

Reverse the characters on the line so we can "cut" first two fields

Cut out subdomains and reverse characters on the line. We can now count the number of unique FQDNs queried per domain

C2 over DNS analysis with Zeek

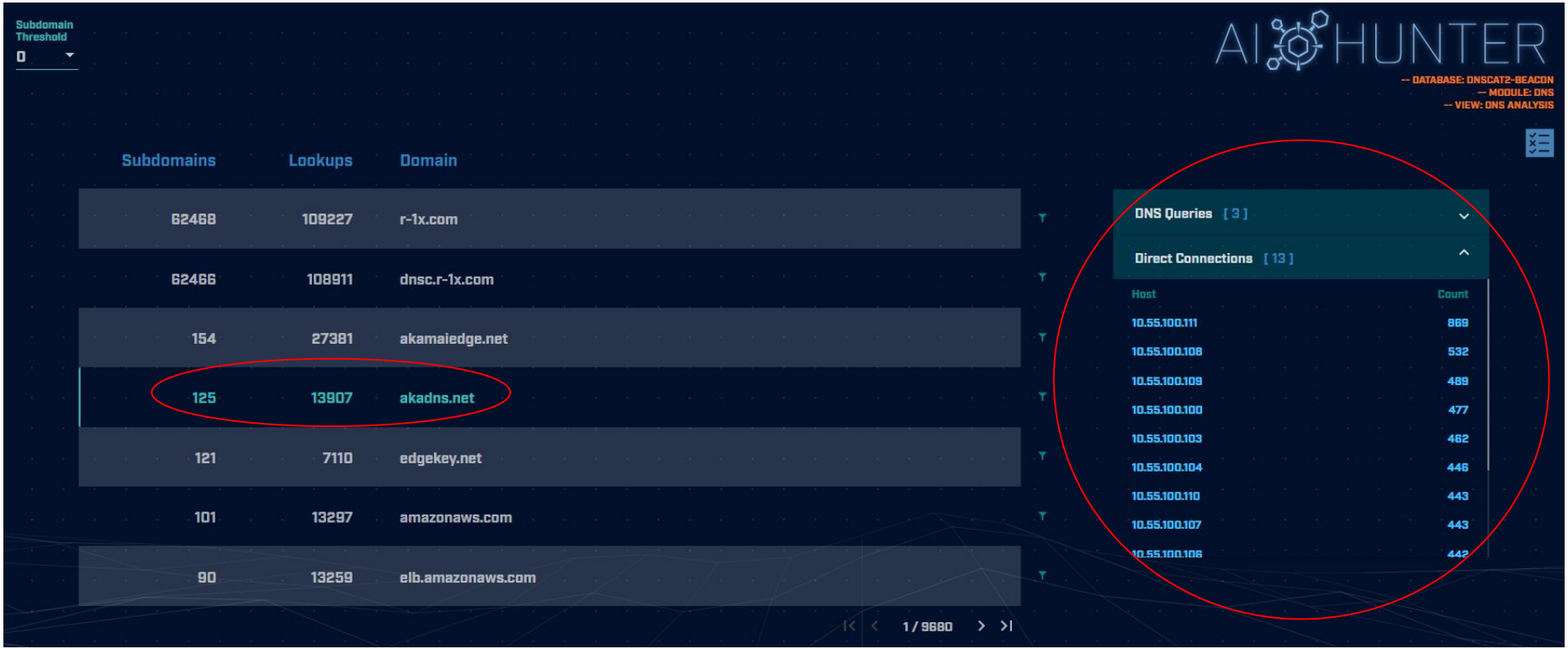
- Same process as last two slides
- Use zeek-cut to extract "query" field

```
cat dns.* | zeek-cut query | sort | uniq | rev | cut -d '.' -f
1-2 | rev | sort | uniq -c | sort -rn | head
```

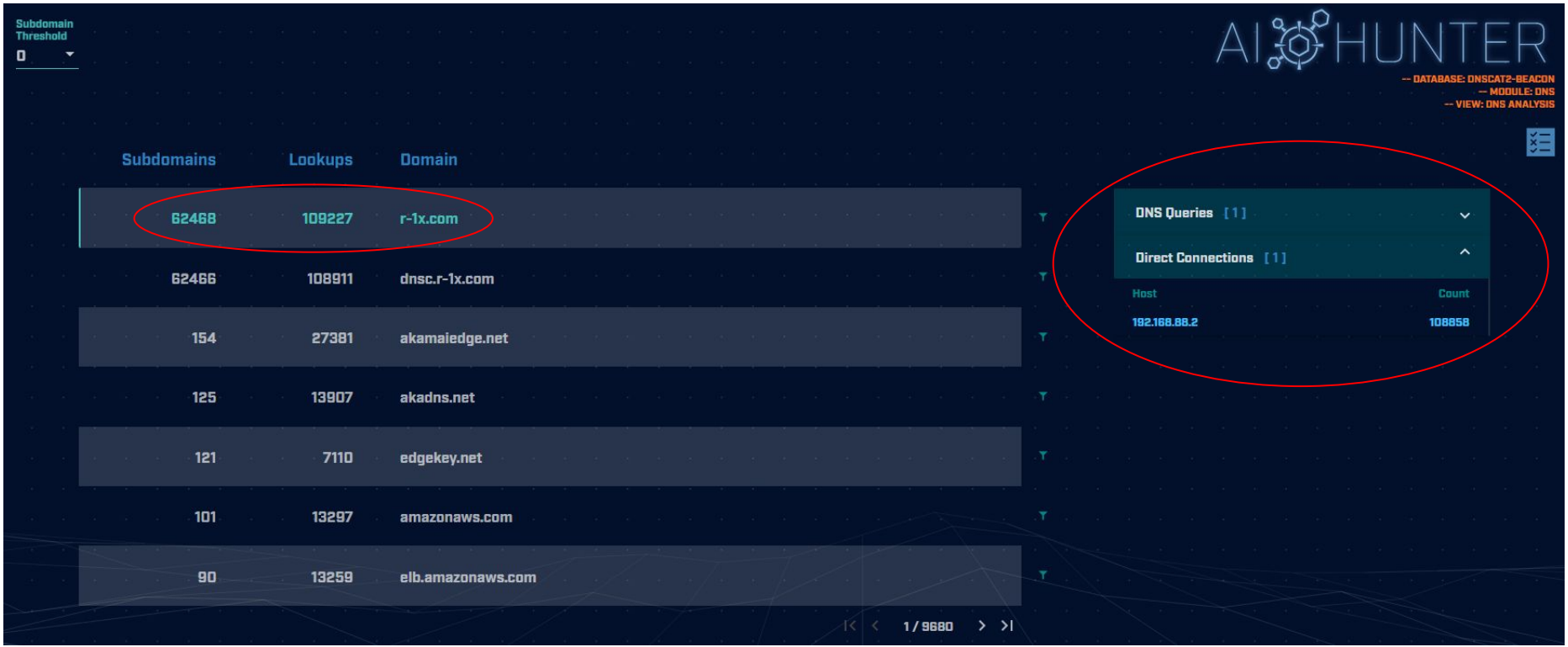
Bonus checks on DNS

- Check domains with a lot of FQDNs
- Get a list of the IPs returned
- Compare against traffic patterns
  - Are internal hosts visiting this domain?
  - Is it just your name servers?
- Unique trait of C2 over DNS
  - Lots or FQDN queries
  - But no one ever connects to these systems

Normal DNS query patten



Things that make you go "hummm"



Look for odd HTTP user agents

```
ritabeakerlab@ritabeakerlab:~/lab1$ cat http.log | zeek-cut id.orig_h id.resp_h user_agent
 | grep 10.0.2.15 | sort | uniq | cut -f 3 | sort | uniq -c | sort -rn
     15 Microsoft-CryptoAPI/10.0
     12 Microsoft-WNS/10.0
      1 Mozilla/5.0 (Windows; U; MSIE 7.0; Windows NT 5.2) Java/1.5.0_08
ritabeakerlab@ritabeakerlab:~/lab1$
```

10.0.2.15 identifies itself as:

Windows 10 when speaking to 27 IP's on the Internet
Windows XP when speaking to one IP on the Internet

Unique SSL Client Hello: Zeek + JA3

SSL/TLS Hash	Seen	Requests	Sources
5e573c9c9f8ba720ef9b18e9fce2e2f7	1	clientservices.googleapis.com	10.55.182.100
bc6c386f480ee97b9d9e52d472b772d8	2	clients4.google.com, 556-emw-319.mktoresp.com	10.55.182.100
f3405aa9ca597089a55cf8c62754de84	2	builds.cdn.getgo.com	10.55.182.100
28a2c9bd18a11de089ef85a160da29e4	2	mediaredirect.microsoft.com	10.55.100.105, 10.55.182.100
08bf94d7f3200a537b5e3b76b06e02a2	4	files01.netgate.com	192.168.88.2

Check destination IP address

- Start simple
  - Who manages ASN?
  - Geolocation info?
  - IP delegation
  - PTR records
- Do you recognize the target organization?
  - Business partner or field office
  - Current vendor (active status)
- Other internal IP's connecting?

Check threat intel on target IP

- Need to understand:
  - When was the record first created?
  - Why was the record created?

https://www.abuseipdb.com/check/<ip address>
https://dnslytics.com/ip/<IP address>
https://transparencyreport.google.com/safe-browsing/search?url=<IP, FQDN or URL>

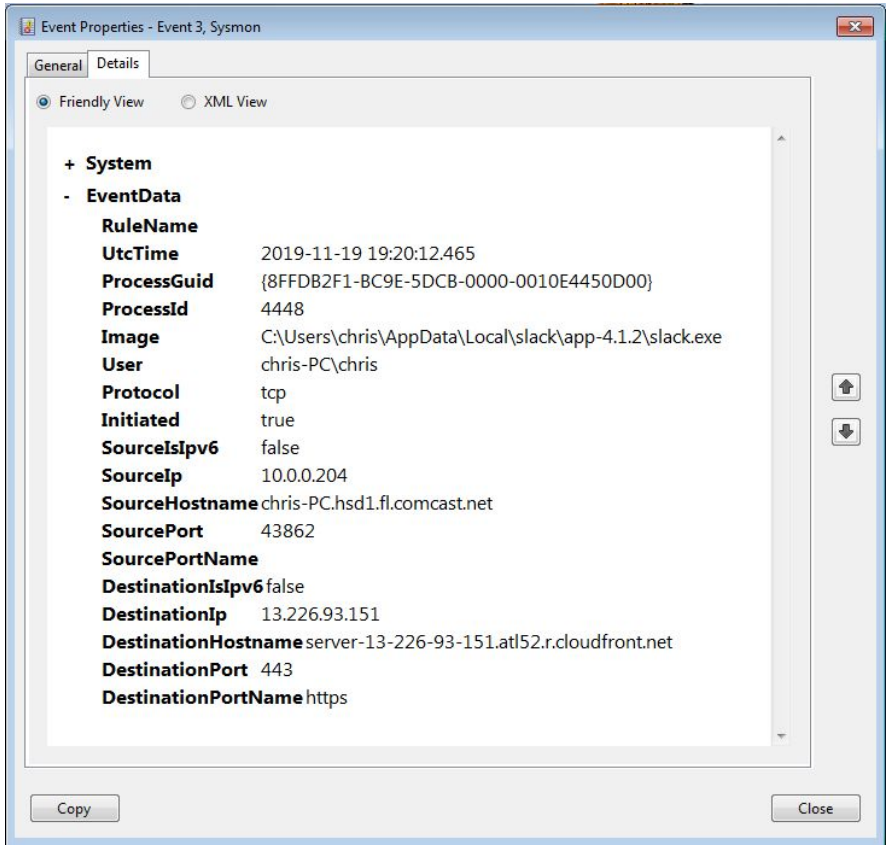
Internal system

- Info available varies greatly between orgs
- Inventory management systems
- Security tools like Carbon Black
- OS projects like BeaKer
- Internal security scans
- DHCP logs
- Login events
- Passive fingerprinting

Leverage internal host logging

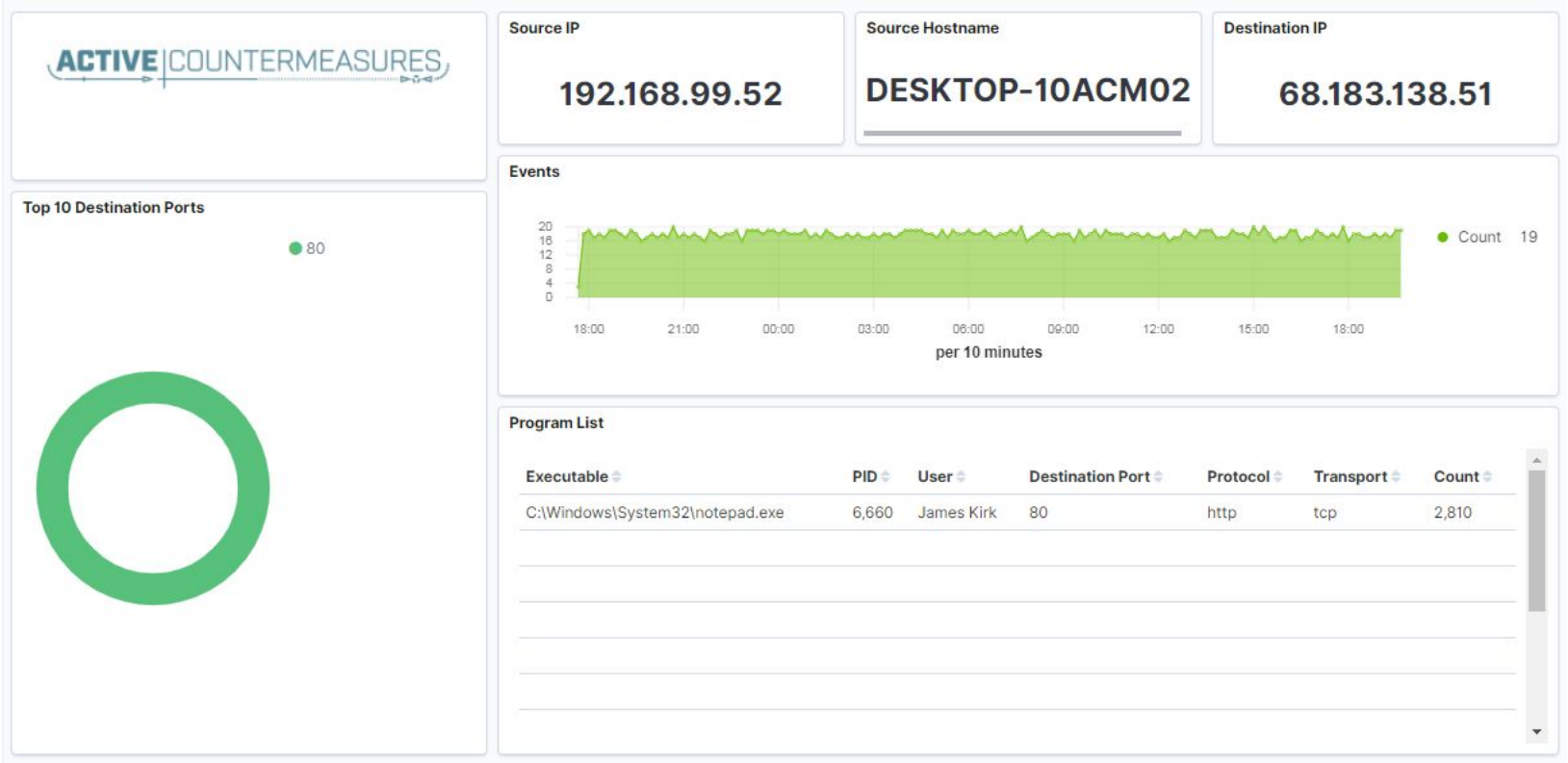
- Network shows suspicious traffic patterns
- Use this data to pivot to host logs
- Filter your logs based on:
  - Suspect internal host
  - Timeframe being analyzed
- Anything stand out as unique or odd?

Sysmon Event ID Type 3's



Map outbound connections to the applications that created them.

Sysmon Type 3 + BeaKer



But I have no system logs!

- Might be a good time to start collecting them
- Full packet captures from system
- Apply additional network tools to collect more data

What next?

- Disposition session
  - "I think it's safe" = add to safelist
  - "I think we've detected a compromise" = Incident response mode
- Remember to leave no footprints
  - All actions should be undetectable to potential adversaries
  - Passive activities only
- Incident response may include active tasks

C2 Detection Tools

tcpdump

- What's it good for?
  - Lightweight packet capturing tool
  - Cross platform support (windump on Windows)
- When to use it
  - Audit trail of all traffic
  - Can also filter to see only specific traffic
  - Can be fully automated
- Where to get it

https://www.tcpdump.org/

Tcpdump example

- Debian/Ubuntu
  - Place the following in /etc/rc.local
- Red Hat/CentOS, Fedora
  - Place the following in /etc/rc.d/rc.local
- Grabs all traffic and rotates every 60 min
  - Date/time stamped and compressed

```
#Place _above_ any "exit" line
mkdir -p /opt/pcaps
screen -S capture -t capture -d -m bash -c "tcpdump -ieth0 -G
3600 -w '/opt/pcaps/`hostname -s`.%Y%m%d%H%M%S.pcap' -z bzip2"
```

capinfos

- Print summary info regarding pcaps
- For a decent hunt you want 12+ hours
- 86,400 seconds = 24 hours

```
cbrenton@guess:~/c2$ capinfos -aeu evilosx_24hr.pcap
File name:           evilosx_24hr.pcap
Capture duration:    86291.558021 seconds
First packet time:   2021-02-17 03:40:26.100491
Last packet time:    2021-02-18 03:38:37.658512
cbrenton@guess:~/c2$
```

tshark

- What's it good for?
  - Extracting interesting fields from packet captures
  - Multiple passes to focus on different attributes
  - Combine with text manipulation tools
  - Can be automated
- When to use it
  - Both major and minor attributes
- Where to get it

https://www.wireshark.org/

Tshark example - DNS queries

```
$ tshark -r thunt-lab.pcapng -T fields -e dns.qry.name
udp.port==53 | head -10

6dde0175375169c68f.dnsc.r-1x.com
6dde0175375169c68f.dnsc.r-1x.com
0b320175375169c68f.dnsc.r-1x.com
0b320175375169c68f.dnsc.r-1x.com
344b0175375169c68f.dnsc.r-1x.com
344b0175375169c68f.dnsc.r-1x.com
0f370175375169c68f.dnsc.r-1x.com
0f370175375169c68f.dnsc.r-1x.com
251e0175375169c68f.dnsc.r-1x.com
251e0175375169c68f.dnsc.r-1x.com
```

Tshark example - user agents

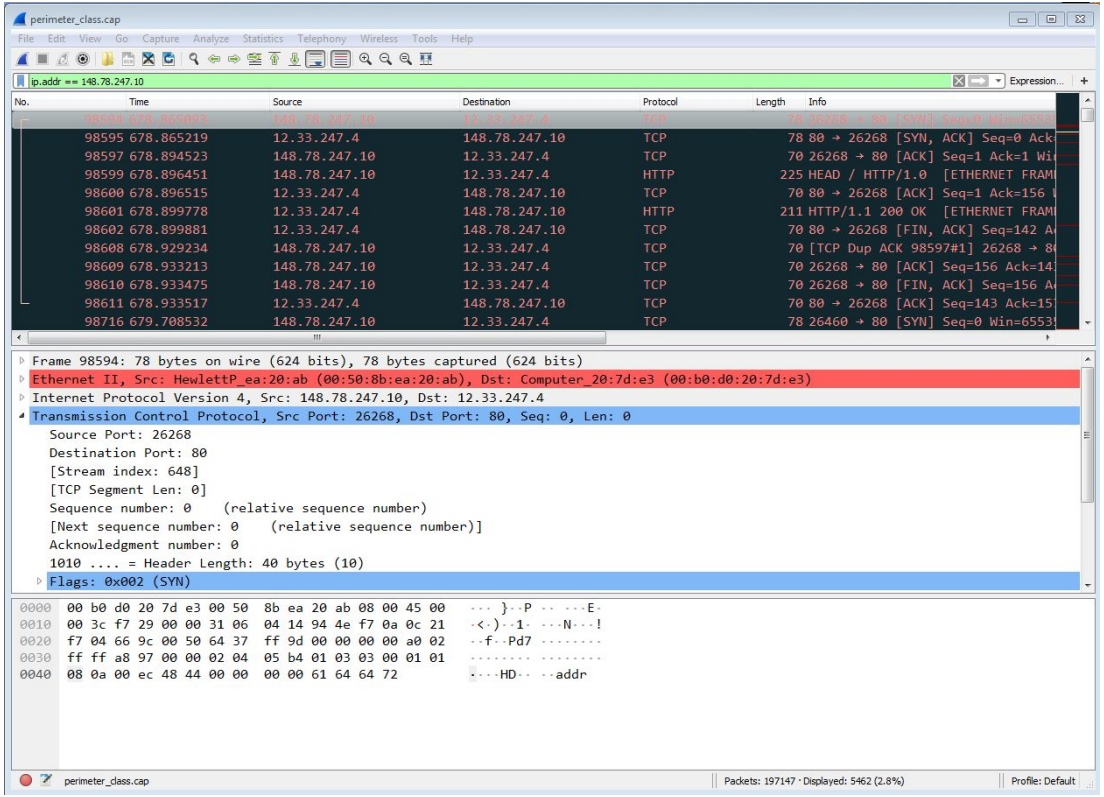
```
$ tshark -r sample.pcap -T fields -e http.user_agent tcp.
dstport==80 | sort | uniq -c | sort -n | head -10
      2 Microsoft Office/16.0
      2 Valve/Steam HTTP Client 1.0 (client;windows;10;1551832902)
      3 Valve/Steam HTTP Client 1.0
     11 Microsoft BITS/7.5
     11 Windows-Update-Agent
     12 Microsoft-CryptoAPI/6.1
    104 PCU
```

Wireshark

- What's it good for?
  - Packet analysis with guardrails
  - Stream level summaries
- When to use it
  - As part of a manual analysis
  - When steps cannot be automated
- Where to get it

https://www.wireshark.org/

Useful when I have a target



Bro/Zeek

- Old name = Bro New name = Zeek
- What's it good for?
  - Near real time analysis
  - More storage friendly than pcaps
- When to use it
  - When you need to scale
  - When you know what attributes to review
- Where to get it

https://www.zeek.org/
sudo apt -y install zeek

Zeek example - cert check

```
$ cat ssl* | zeek-cut id.orig_h id.resp_h id.resp_p
validation_status | grep 'self signed' | sort | uniq
122.228.10.51   192.168.88.2    9943    self signed certificate in
certificate chain
24.111.1.134    192.168.88.2    9943    self signed certificate in
certificate chain
71.6.167.142    192.168.88.2    9943    self signed certificate in
certificate chain
```

-d for human readable times

- Zeek-cut prints epoch time by default
- "-d" converts to human readable

```
cbrenton@cbrenton-beacon-src-test:~/foo$ cat conn.01\:00\:00-02\
:00\:00.log | zeek-cut ts id.orig_h | head -8
1645578000.318671       167.172.154.151
1645578000.318784       167.172.154.151
1645578000.318841       167.172.154.151
1645578000.334906       167.172.154.151
1645578000.334948       167.172.154.151
1645578000.334977       167.172.154.151
1645578001.228742       167.172.154.151
1645578001.360749       167.172.154.151
cbrenton@cbrenton-beacon-src-test:~/foo$ cat conn.01\:00\:00-02\
:00\:00.log | zeek-cut -d ts id.orig_h | head -8
2022-02-23T01:00:00+0000        167.172.154.151
2022-02-23T01:00:00+0000        167.172.154.151
2022-02-23T01:00:00+0000        167.172.154.151
2022-02-23T01:00:00+0000        167.172.154.151
2022-02-23T01:00:00+0000        167.172.154.151
2022-02-23T01:00:00+0000        167.172.154.151
2022-02-23T01:00:01+0000        167.172.154.151
2022-02-23T01:00:01+0000        167.172.154.151
cbrenton@cbrenton-beacon-src-test:~/foo$
```

ngrep

- Pattern match on passing packets
- Like "grep" for network traffic
- Useful for quick checks
  - NIDS with signature better choice for long term
- Useful switches
  - "-q" = Don't print "#" for non-matches
  - "-I" = Read a pcap file

https://github.com/jpr5/ngrep
sudo apt install ngrep

Ngrep example

```
cbrenton@cbrenton-lab-testing:~/pcaps$ ngrep -q -I odd.pcap Admin | head -15
input: odd.pcap
match: Admin

T 148.78.247.10:26922 -> 12.33.247.4:80 [AP]
  GET /cfide/Administrator/startstop.html HTTP/1.0..Host: 12.33.247.4..User-Agent: Mozilla/5.0 [en] (Win
  95; U)..Referer: http://12.33.247.4/..X-Forwarded-For: 148.64.147.168..Cache-Control: max-stale=0..Pra
  gma: no-cache......Cv

T 12.33.247.4:80 -> 148.78.247.10:26922 [AP]
  HTTP/1.1 404 Not Found..Date: Tue, 25 Jun 2002 00:34:58 GMT..Server: Apache..Connection: close..Conten
  t-Type: text/html; charset=iso-8859-1....<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">.<HTML><HEA
  D>.<TITLE>404 Not Found</TITLE>.</HEAD><BODY>.<H1>Not Found</H1>.The requested URL /cfide/Administrato
  r/startstop.html was not found on this server.<P>.</BODY></HTML>.....

T 12.33.247.4:80 -> 148.78.247.10:26922 [AFP]
cbrenton@cbrenton-lab-testing:~/pcaps$ _
```

Datamash

- What's it good for?
  - Similar to the R-base tools, but more extensive
  - Performing simple calculation on data
- When to use it
  - Performing calculations on multiple lines
  - Statistical analysis
- Where to get it

https://www.gnu.org/software/datamash/
sudo apt install datamash

Datamash example

```
cbrenton@cbrenton-lab-testing:~/lab3$ cat conn.log | zeek-cut
id.orig_h id.resp_h duration | sort -k3 -rn | head -5
192.168.1.105   143.166.11.10   328.754946
192.168.1.104   63.245.221.11   41.884228
192.168.1.104   63.245.221.11   31.428539
192.168.1.105   143.166.11.10   27.606923
192.168.1.102   192.168.1.1     4.190865

cbrenton@cbrenton-lab-testing:~/lab3$ cat conn.log | zeek-cut
id.orig_h id.resp_h duration | grep -v -e '^$' | grep -v '-' | sort |
datamash -g 1,2 sum 3| sort -k3 -rn | head -5
192.168.1.105   143.166.11.10   356.361869
192.168.1.104   63.245.221.11   73.312767
192.168.1.102   192.168.1.1     5.464553
192.168.1.103   192.168.1.1     4.956918
192.168.1.105   192.168.1.1     1.99374
```

Duplicate IPs

RITA

- What's it good for?
  - Beacon & long conn at scale
  - Some secondary attributes
- When to use it
  - Can better organize Zeek data
  - Good when you are comfortable scripting
  - Will scale but can be time consuming
- Where to get it

https://github.com/activecm/rita

RITA example - beacons

```
cbrenton@cbrenton-lab-testing:~$ rita show-beacons thunt-lab | head
Score,Source IP,Destination IP,Connections,Avg. Bytes,Intvl Range,Size Range,Top Intv
l,Top Size,Top Intvl Count,Top Size Count,Intvl Skew,Size Skew,Intvl Dispersion,Size
Dispersion
1,10.55.100.111,165.227.216.194,20054,92,29,52,1,52,7774,20053,0,0,0,0
1,192.168.88.2,165.227.88.15,108858,199,860,230,1,89,53341,108319,0,0,0,0
0.838,10.55.200.10,205.251.194.64,210,308,29398,4,300,70,109,205,0,0,0,0
0.835,10.55.200.11,205.251.197.77,69,308,1197,4,300,70,38,68,0,0,0,0
0.834,10.55.100.111,34.239.169.214,34,1259,5,14388,1,156,15,30,0,0,0,0
0.834,192.168.88.2,13.107.5.2,27,198,2,33,12601,73,4,15,0,0,0,0
0.833,10.55.100.107,23.52.162.184,24,2397,43356,52,1800,467,18,18,0,0,0,0
0.833,10.55.100.107,23.52.161.212,24,5404,43235,52,1800,505,19,21,0,0,0,0
0.833,10.55.100.111,23.52.161.212,27,5379,37752,92,1800,505,17,20,0,0,0,0
cbrenton@cbrenton-lab-testing:~$ _
```

Scale is 0 - 1 with 1.0 being a perfect beacon score

RITA example - C2 over DNS

```
thunt@thunt-one-day:~$ rita show-exploded-dns test | head -10
Domain,Unique Subdomains,Times Looked Up
cymru.com,227,502
hash.cymru.com,224,485
malware.hash.cymru.com,222,341
akadns.net,134,19282
edgekey.net,116,6342
akamaiedge.net,116,19680
microsoft.com,91,3116
amazonaws.com,89,6369
com.edgekey.net,83,5401
thunt@thunt-one-day:~$ _
```

Passer

```
TC,172.1.199.23,TCP_43,open,
TC,172.16.199.23,TCP_55443,open,
UC,172.16.199.23,UDP_626,open,serialnumberd/clientscanner likely nmap
scan Warnings:scan
UC,172.16.199.23,UDP_1194,open,openvpn/client Warnings:tunnel
UC,172.16.199.23,UDP_3386,open,udp3386/client
UC,172.16.199.23,UDP_5632,open,pcanywherestat/clientscanner
Warnings:scan
UC,172.16.199.23,UDP_64738,open,shodan_host/clientscanner abcdefgh
Unlisted host Warnings:scan
DN,2001:db8:1001:0000:0000:0000:0000:0015,AAAA,ns3.markmonitor.com.,
DN,fe80:0000:0000:0000:189f:545b:7d4c:eeb8,PTR,Apple
TV._device-info._tcp.local.,model=J105aA
```

Beacon/Threat Simulator

- Permits you to test your C2 detection setup
- Target any TCP or UDP port
- Can jitter timing
- Can jitter payload size
- Not designed to exfiltrate data!

```
beacon-simulator.sh <target IP> 80 300 10 tcp 5000
```

Connect to TCP/80 on target IP every 300 seconds, +/-10 seconds, vary payload between 0-5,000 bytes

https://github.com/activecm/threat-tools

C2 Labs

What We Will Cover

- This section is mostly hands on labs
- Implement what you have learned
- Lab format:
  - Given a problem
    - Use earlier content to help solve
  - Given hints
    - If you don't know where to start, try the hints
  - Given the exact commands
  - Solution
    - Complete walk through of the solution

Reminder

- All lab files are on the VM
  - No network access needed
  - Unless you want to do third party research
  - Can also be done from your host system browser
- Login info
  - Name = thunt
  - Password = aybab2u
- Labs are in /home/thunt/lab*

Find long connections

- Files located in /home/thunt/lab1
- Provided with pcap and Zeek log files
- Identify
  - Top 10 longest connections between private and legal IP addresses (internal to external)
  - Top 10 cumulative communication time between private and legal IP addresses (internal to external)

Find long conns - Hints

- Long connections is a relative term. You need to know the length of time being audited.
- Pcaps don't store connection duration
- Zeek stores duration in conn.log
- Zeek-cut extracts fields from Zeek logs
- Datamash is useful for adding values

Useful commands to try

```
capinfos -aeu <pcap file>

cat conn.log | zeek-cut id.orig_h
id.resp_h duration | sort -k 3 -rn | head

cat conn.log | zeek-cut id.orig_h
id.resp_h duration | sort | grep -v -e
'^$' | grep -v '-' | datamash -g 1,2 sum 3
| sort -k 3 -rn | head
```

Long conns - Answers

- Need to ID how long the pcap captured
- Use Zeek conn.log to easily get duration
- Need to extract:
  - Source IP (id.orig_h)
  - Destination IP (id.resp_h)
  - Duration of each connection (duration)
- Need to be able to:
  - Add up connection time between IP's
  - Present longest results first

less -S conn.log

```
#separator \x09
#set_separator      ,
#empty_field        (empty)
#unset_field        -
#path               conn
#open               2021-10-13-15-47-50
#fields             ts                  uid                 id.orig_h       id.orig_p       id.resp_h
#types              time                string              addr            port            addr
1599652681.658987   Ci09jy2pQa8n4Nhpnk  192.168.125.105     43742           91.189.88.142   80
1599652681.909864   C7ebxg76JCvTenVC4   192.168.125.105     55418           91.189.91.38    80
1599652682.160692   Ciy54Bgp1AAP3g3Ai   192.168.125.105     56374           91.189.88.152   80
1599652682.411596   CIJ8Xh4WAfju0gEub6  192.168.125.105     36338           91.189.91.39    80
```

"duration" = duration

Identify time window being audited

```
thunt@thunt:~/lab1$ capinfos -aeu trace1.pcap
File name:           trace1.pcap
Capture duration:    86398.498096 seconds
First packet time:   2020-06-04 16:59:02.292525
Last packet time:    2020-06-05 16:59:00.790621
thunt@thunt:~/lab1$ _
```

24 hours = 86,400 seconds

Plan B for files too large for capinfos:

```
tcpdump -tttt -n -r <filename> | awk 'NR==1; END{print}'
```

Longest unique connections

```
thunt@thunt:~/lab1$ cat conn.log | zeek-cut id.orig_h id.resp_h duration | sort -k 3 -rn | head
192.168.99.51   167.71.97.235   86389.659357
192.168.99.51   104.248.234.238 243.768999
192.168.99.51   104.118.9.117   166.139547
192.168.99.51   72.21.91.29     134.888177
192.168.99.51   52.184.216.246  129.075227
192.168.99.51   52.167.249.196  128.957107
192.168.99.51   52.184.216.246  128.481757
192.168.99.51   13.107.5.88     128.346889
192.168.99.51   52.179.219.14   128.116421
192.168.99.51   13.107.5.88     128.042647
thunt@thunt:~/lab1$
```

Duration is just short of the full 86,398 second capture time

Longest cumulative time

```
thunt@thunt:~/lab1$ cat conn.log | zeek-cut id.orig_h id.resp_h duration | sort |
grep -v '-' | datamash -g 1,2 sum 3 | sort -k 3 -rn | head
192.168.99.51   167.71.97.235   86389.659357
192.168.99.51   52.179.219.14   4067.394413
192.168.99.51   52.184.217.56   2936.172839
192.168.99.51   52.184.216.246  2825.858
192.168.99.52   239.255.255.250 2507.626732
fe80::d048:42e0:8448:187c       ff02::c 2434.977049
192.168.99.51   239.255.255.250 2374.546469
fe80::2126:bcd7:16f4:8cdb       ff02::c 2368.234679
192.168.99.51   13.107.5.88     1317.047871
192.168.99.51   52.167.249.196  868.46966
thunt@thunt:~/lab1$ _
```

Note the first entry is still the same, but all others are new.
IPv6 addresses have shifted info to the right.

Investigate the longest talkers

- Let's investigate the external IP of the two longest session
  - 167.71.97.235
  - 52.179.219.14
- We'll use three common research methods
  - Check the dns.log file
  - AbuseIPDB
    - https://www.abuseipdb.com/
  - ThreatCrowd
    - https://www.threatcrowd.org/

Investigate - hints

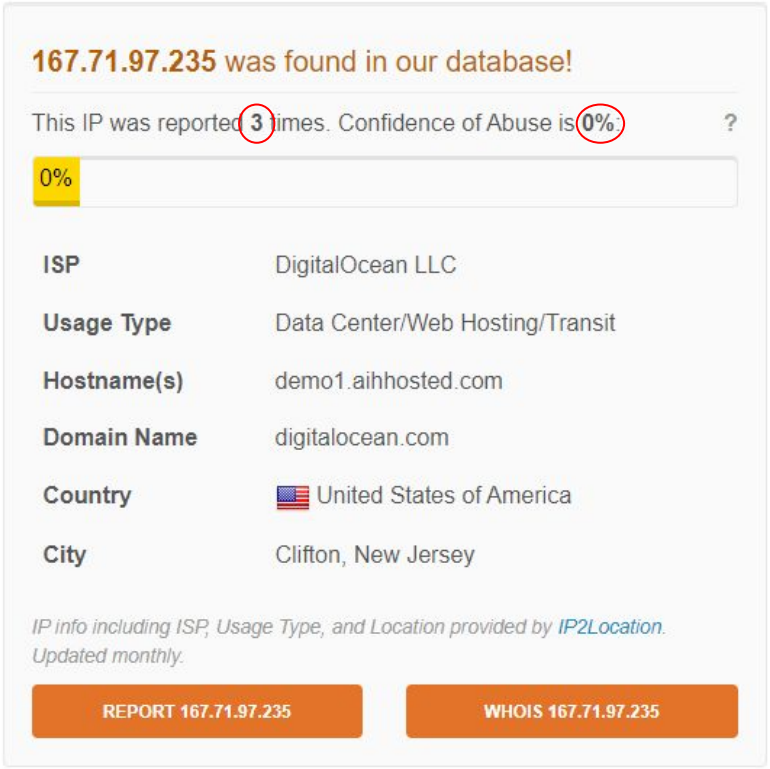
- You were given the two IP addresses to research
- The dns.log file shows all DNS queries and answers that were returned
- Use a browser to connect to the two research Websites and enter each IP

One out of two is not bad

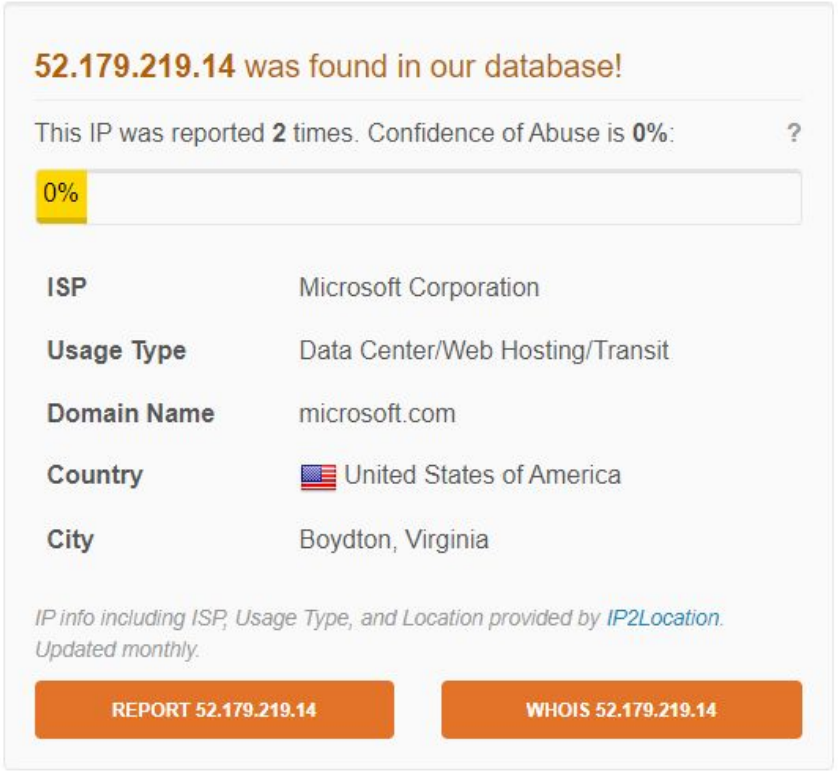
```
thunt@thunt-labs:~/lab1$ cat dns.log | zeek-cut query answers | sort | uniq |
 grep 167.71.97.235
thunt@thunt-labs:~/lab1$ cat dns.log | zeek-cut query answers | sort | uniq |
 grep 52.179.219.14
array503.prod.do.dsp.mp.microsoft.com   52.179.219.14
thunt@thunt-labs:~/lab1$
```

Second IP was contacted because system was trying to reach a microsoft.com host.

AbuseIPDB on first IP

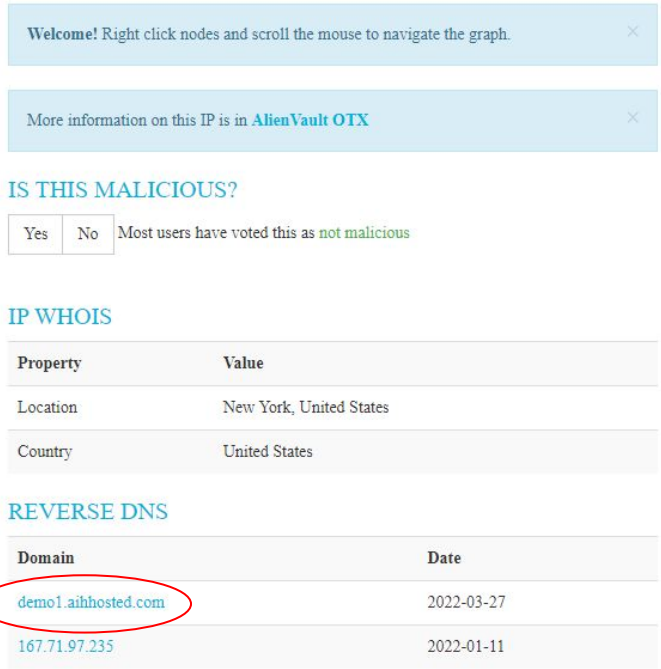


AbuseIPDB data on 2nd IP



ThreatCrowd data on first IP

IP > 167.71.97.235



Is aihhosted.com a business partner? When in doubt, check with purchasing

We could try to verify this entry with a host/dig/nslookup query but that will actively send them data

ThreatCrowd data on 2nd IP

IP WHOIS

Property	Value
Location	Wilmington, United States
Country	United States

REVERSE DNS

Domain	Date
array503.prod.do.dsp.mp.microsoft.com	2022-04-11
orgeover-prod.do.dsp.mp.microsoft.com	2022-01-03
52.179.219.14	2021-10-12
array503-prod.do.dsp.mp.microsoft.com	2021-08-28
geo-prod.do.dsp.mp.microsoft.com	2021-03-16
dbk2gusuwcoxoqjmsyhz2m3zfacameap.qw776iawpez267udlbikkahi5yivxqkc.1.0.od6u6m3cwr	2021-02-05
etclpbjw6fdjpulrfarjm4vxsjkpxr46.kzfhjer7xzp7voaycava.1.0.ozlnabtsgij2f5y455ywbx	2020-12-18
geo.prod.do.dsp.mp.microsoft.com	2020-12-18
geo-prod.dodsp.mp.microsoft.com.nsatc.net	2020-09-23
sbzurncdc4clwz5.eastus2.atlas.cloudapp.azure.com	2020-05-29
runnercitus-eastus2-a149423e-0.postgres.database.azure.com	2020-05-16

Next lab

- We verified aihhosted is a business partner
- The 2nd IP (52.179.219.14) looks like Microsoft, but we want to verify
- Is there anything else in the data that can help with verification?
- Start with conn.log
  - Move to any applicable application logs

Hints

- We want to further verify 52.179.219.14
- Search conn.log for this IP address
- Is the "service" recognized?
- Check for a log file with that service name
- Search that log for the above IP
- Any helpful digital verification?

Looks like it's SSL/TLS traffic

```
thunt@thunt-labs:~/lab1$ grep 52.179.219.14 conn.log | head -5
1591290650.463848       Ce8vuV9pdZN1TTE21       192.168.99.51   52863   52.179.219.14
443     tcp     ssl     65.389372       1270    3035    SF      -       -       0
ShADadFf        14      1842    12      3527    -
1591292050.459124       CbnymM8GhENDKN6ol       192.168.99.51   52938   52.179.219.14
443     tcp     ssl     95.406423       1270    3036    SF      -       -       0
ShADdaFf        14      1842    12      3528    -
1591293617.574816       CdOPg52V3t5AGGkjyf      192.168.99.51   52999   52.179.219.14
443     tcp     ssl     68.280122       1270    3036    SF      -       -       0
ShADdaFf        14      1842    11      3488    -
1591295064.955993       CKXOFb4bJ1gZgX3sW2      192.168.99.51   53150   52.179.219.14
443     tcp     ssl     128.116421      1269    3036    RSTR    -       -       0
ShADdar 12      1761    11      3488    -
1591295092.098734       CjkEjk4m4GL1LSMJMd      192.168.99.51   53153   52.179.219.14
443     tcp     ssl     113.248030      1246    3036    SF      -       -       0
ShADdaFf        14      1818    11      3488    -
thunt@thunt-labs:~/lab1$ _
```

Entry in ssl.log

```
thunt@thunt-labs:~/lab1$ grep 52.179.219.14 ssl.log | head -2
1591290650.502177       Ce8vuV9pdZN1TTE21       192.168.99.51   52863   52.179.219.14
443     TLSv12  TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 x25519  array503.prod.do.dsp.
mp.microsoft.com        F       -       h2      T       Fd3zBI3qZR5omLoAi7,FWU71E32do
A3zMCOH (empty) CN=*.prod.do.dsp.mp.microsoft.com,OU=DSP,O=Microsoft,L=Redmond,ST=WA,
C=US    CN=Microsoft ECC Content Distribution Secure Server CA 2.1,O=Microsoft Corpor
ation,L=Redmond,ST=Washington,C=US       -       -
1591292050.498723       CbnymM8GhENDKN6ol       192.168.99.51   52938   52.179.219.14
443     TLSv12  TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 x25519  array503.prod.do.dsp.
mp.microsoft.com        F       -       h2      T       Fle59121NCCmoWeZnf,FMXEBQ1sio
EzUSPeza        (empty) CN=*.prod.do.dsp.mp.microsoft.com,OU=DSP,O=Microsoft,L=Redmon
d,ST=WA,C=US    CN=Microsoft ECC Content Distribution Secure Server CA 2.1,O=Microsof
t Corporation,L=Redmond,ST=Washington,C=US       -       -
thunt@thunt-labs:~/lab1$
```

What have we learned?

- Connection was SSL/TLS
- Server had a digital certificate
- The server_name matches the DNS query
- Only thing left would be to check that the cert is valid
  - Zeek can do this automatically
  - We have that feature turned off for these labs
  - Assume the cert is valid

Answers

- Longest connection appears to be business partner related
- Second longest is is used in keeping Windows 10 updated
- Neither appear to be malware related

Let's look for beacons

- Beacons are hard to detect!
- Neither pcaps or Zeek logs record dwell time between connections
- Using connect quantity misses low & slow
- Using session size also problematic
- RITA to the rescue!
- We've already imported data into RITA

RITA online help

```
thunt@thunt-labs:~$ rita | less
NAME:
  rita - Look for evil needles in big haystacks.

USAGE:
  rita [global options] command [command options] [arguments...]

VERSION:
  v4.3.1

COMMANDS:
    delete, delete-database  Delete imported database(s)
    import                   Import zeek logs into a target database
    html-report              Create an html report for an analyzed database
    show-beacons-fqdn        Print hosts which show signs of C2 software (FQDN Analysis)
    show-beacons-proxy       Print hosts which show signs of C2 software (internal -> Proxy)
    show-beacons             Print hosts which show signs of C2 software
```

List imported data

```
thunt@thunt-labs:~$ rita list
lab1
lab2
lab3
thunt@thunt-labs:~$ _
```

Syntax = rita <command> <dbase>

```
thunt@thunt-labs:~$ rita show-long-connections lab1 | head
Source IP,Destination IP,Port:Protocol:Service,Duration,State
192.168.99.51,167.71.97.235,9200:tcp:-,86389.7,closed
192.168.99.51,104.248.234.238,80:tcp:http,243.769,closed
192.168.99.51,104.118.9.117,443:tcp:ssl,166.14,closed
192.168.99.51,72.21.91.29,80:tcp:- 80:tcp:http,134.888,closed
192.168.99.51,52.184.216.246,443:tcp:ssl,129.075,closed
192.168.99.51,52.167.249.196,443:tcp:ssl,128.957,closed
192.168.99.51,13.107.5.88,443:tcp:ssl,128.347,closed
192.168.99.51,52.179.219.14,443:tcp:ssl,128.117,closed
192.168.99.51,52.184.217.56,443:tcp:ssl,126.653,closed
thunt@thunt-labs:~$ _
```

Lab time!

- Using RITA, identify potential beacons
- We are still working with "lab1"
- Consider any session scoring .8 or higher worthy of deeper analysis

Hints

- RITA is the best tool for beacon detection
- Remember the syntax:
  - rita <command> <database>
- Finding RITA's beacon commands

```
thunt@thunt-labs:~$ rita | grep beacons
     show-beacons-fqdn        Print hosts which show signs of C2 software (FQDN Analysis)
     show-beacons-proxy       Print hosts which show signs of C2 software (internal -> Proxy)
     show-beacons             Print hosts which show signs of C2 software
thunt@thunt-labs:~$
```

Commands

```
rita show-beacons-fqdn lab1
rita show-beacons-proxy lab1
rita show-beacons-fqdn lab1
```

Answers - FQDN beacons

rita show-beacons-fqdn lab1 -H | less -S

SCORE	SOURCE IP	FQDN	CONNECTIONS	AVG BYTES
0.624	192.168.99.51	tile-service.weather.microsoft.com	48	5436
0.585	192.168.99.51	array509.prod.do.dsp.mp.microsoft.com	30	5258
0.548	192.168.99.51	kv501.prod.do.dsp.mp.microsoft.com	44	7560
0.535	192.168.99.51	geover.prod.do.dsp.mp.microsoft.com	40	7857
0.51	192.168.99.51	disc501.prod.do.dsp.mp.microsoft.com	52	8088
0.487	192.168.99.51	array506.prod.do.dsp.mp.microsoft.com	25	5244
0.438	192.168.99.51	cp501.prod.do.dsp.mp.microsoft.com	50	8058
0.391	192.168.99.51	ctldl.windowsupdate.com	28	834412
0.373	192.168.99.51	array503.prod.do.dsp.mp.microsoft.com	38	5279
0.337	192.168.99.51	settings-win.data.microsoft.com	50	5959

All score < .8 so no further investigation needed

Answers - no proxy traffic found

```
thunt@thunt-labs:~$ rita show-beacons-proxy lab1 -H
No results were found for lab1
```

Answers - IP beacons

rita show-beacons lab1 -H | less -S

SCORE	SOURCE IP	DESTINATION IP	CONNECTIONS	AVG BYTES	INTVL RANGE	SIZE RANGE
0.885	192.168.99.51	104.248.234.238	3011	1101	246	621
0.835	192.168.99.51	52.179.224.121	72	396	11	2
0.586	192.168.99.51	208.67.220.220	60	245	7741	30
0.585	192.168.99.51	52.184.217.56	30	5258	2687	122
0.535	192.168.99.51	23.197.120.174	40	7857	16	1329
0.526	192.168.99.51	208.67.222.222	297	231	2005	39
0.487	192.168.99.51	52.184.216.246	25	5244	13031	122
0.373	192.168.99.51	52.179.219.14	38	5279	3562	162
0.319	192.168.99.51	52.167.249.196	47	5976	14399	1331

Two entries scoring .8 or higher

Is there a way to visualize beacons?

```
thunt@thunt-labs:~/lab1$ beacon-data 192.168.99.51 104.248.234.238
00 126
01 125
02 126
03 126
04 126
05 126
06 126
07 126
08 126
09 125
10 127
11 126
12 125
13 126
14 125
15 126
16 126
17 126
18 126
19 118
20 126
21 125
22 126
23 125
```

We cover these types of techniques in the Advanced Threat Hunting class

Answers - Final

- Two IPs are worth investigating
  - 104.248.234.238
  - 52.179.224.121

Payload analysis with ngrep

- We found a suspicious IP pair
  - 192.168.99.51 to 104.248.234.238
- Let's analyze the payloads in these sessions
- Multiple tools can help here
  - But ngrep easily focuses on payload
- Use "host" parameter to focus in on the above IPs

Payload analysis - hints

- Ngrep is normally used to search for patterns within the payload of all packets
- You can use BP filters to:
  - Focus on specific IP addresses
  - Focus on specific ports
  - "host" focuses on specific IP addresses
- Helpful switches
  - "-q" = Don't print "#" for packets that don't match
  - "-I" (capital letter i) = Read from pcap file

Useful commands to try

```
ngrep -q -I trace1.pcap host 192.168.99.51 and host
104.248.234.238 | less
```

Things that make you go "humm"

```
thunt@thunt:~/lab1$ ngrep -q -I trace1.pcap host 192.168.99.51 and host 104.248.23
4.238 | head -20
input: trace1.pcap
filter: ( host 192.168.99.51 and host 104.248.234.238 ) and ((ip || ip6) || (vlan
&& (ip || ip6)))

T 192.168.99.51:52833 -> 104.248.234.238:80 [AP] #4
  GET /rmvk30g/eghmbblnphlaefbmmnoenohhoncmcepapefjjekpleokhjfjmnmijghedkienpli
  dbbcmgdjldbegpeemiboacnfcpnbnnhlmjbpcejfpecdioiddklfegefcjbcnagjclnoijpajlpkk
  egakmpdddojnlphegeehaacmofggdfkagpbighfkndllaamndepdanhnogedkaodhgakiigohemin
  oolnaobdiiokpebghapnghbebkepiffoo(jden;1;4;1 HTTP/1.1..Accept: text/html, ima
  ge/gif, image/jpeg, *; q=.2, */*; q=.2..Connection: keep-alive..User-Agent: M
  ozilla/4.0 (Windows 7 6.1) Java/1.7.0_11..Host: 104.248.234.238..Cache-Contro
  l: no-cache....

T 104.248.234.238:80 -> 192.168.99.51:52833 [A] #5
  ......

T 104.248.234.238:80 -> 192.168.99.51:52833 [AP] #6
  HTTP/1.1 200 OK..Date: Thu, 4 Jun 2020 16:59:22 GMT..Server: Apache/2.2.15 (C
  entOS)..X-Powered-By: PHP/5.3.27..Content-Type: application/octet-stream..Con
  nection: close..Content-Length: 0....
```

User agent string analysis

- Is it normal for the source IP to ID as a Windows 7 system?
- Let's find out together
- Run this command:

```
cat http.log | zeek-cut id.orig_h id.resp_h user_agent | grep
192.168.99.51 | sort | uniq | cut -f 3 | sort | uniq -c | sort -rn
```

What you should see

```
thunt@thunt-labs:~/lab1$ cat http.log | zeek-cut id.orig_h id.resp_h user_agent | grep 192.168.99.
51 | sort | uniq | cut -f 3 | sort | uniq -c | sort -rn
     29 Microsoft-WNS/10.0
     16 Microsoft-Delivery-Optimization/10.0
      8 Microsoft-CryptoAPI/10.0
      1 WicaAgent
      1 Mozilla/4.0 (Windows 7 6.1) Java/1.7.0_11
```

Source IP normally identifies as Windows 10
Beacon session is the only time it claims to be Windows 7

What data are we sending?

- Is the URI in the ngrep output sent consistently?
- We could eyeball it, but…
- Zeek stores this type of data
  - It's in the http.log file
- Let's use this log to identify all of the URI's requested from this external host

URI request - hints

- Zeek-cut is your friend
- We should extract
  - Source IP
  - Destination IP
  - The "uri" string
- Grep can focus on the traffic we care about
- Remember the threat hunter's mantra
  - sort | uniq | sort

Useful commands to try

```
cat http.log | zeek-cut id.orig_h id.resp_h uri |
grep 104.248.234.238 | sort | uniq -c | sort -rn
```

Single minded request

```
thunt@thunt:~/lab1$ cat http.log | zeek-cut id.orig_h id.resp_h uri | grep 104.248
.234.238 | sort | uniq -c | sort -rn
   3011 192.168.99.51   104.248.234.238 /rmvk30g/eghmbblnphlaefbmmnoenohhoncmcepap
efjjekpleokhjfjmnmijghedkienplidbbcmgdjldbegpeemiboacnfcpnbnnhlmjbpcejfpecdioiddkl
fegefcjbcnagjclnoijpajlpkkegakmpdddojnlphegeehaacmofggdfkagpbighfkndllaamndepdanhn
ogedkaodhgakiigoheminoolnaobdiiokpebghapnghbebkepiffooljden;1;4;1
thunt@thunt:~/lab1$
```

Answers

- 3,011 connections to external host
- Always sending the same odd "GET" request
- HTTP header data looks forged
- This really looks like a C2 channel
- Google search for "rmvk30g"
  - Looks like Fiesta EK malware

https://www.malware-traffic-analysis.net/2014/04/05/index.html

Lab - Look for C2 over DNS

- Check to see if C2 over DNS is in play
- Consider any domain with more than 1,000 FQDNs in it suspect
  - Not interested in total quantity of queries
  - Interest in quantities of unique FQDNs

Hints

- Type "rita" to show a list of commands
- Look for any that seem "dns" related
- RITA labels "unique queries" as "Unique Subdomains"

Commands

```
rita show-exploded-dns lab1 -H | head -20
```

Answers

```
thunt@thunt-labs:~/lab1$ rita show-exploded-dns lab1 -H | head -20
+-----------------------------------------------------+-------------------+------------------+
|                       DOMAIN                        | UNIQUE SUBDOMAINS | TIMES LOOKED UP  |
+-----------------------------------------------------+-------------------+------------------+
| microsoft.com                                       |                24 |              226 |
+-----------------------------------------------------+-------------------+------------------+
| mp.microsoft.com                                    |                14 |              117 |
+-----------------------------------------------------+-------------------+------------------+
| dsp.mp.microsoft.com                                |                 9 |              109 |
+-----------------------------------------------------+-------------------+------------------+
| do.dsp.mp.microsoft.com                             |                 8 |              107 |
+-----------------------------------------------------+-------------------+------------------+
| prod.do.dsp.mp.microsoft.com                        |                 8 |              107 |
+-----------------------------------------------------+-------------------+------------------+
| delivery.mp.microsoft.com                           |                 4 |                6 |
+-----------------------------------------------------+-------------------+------------------+
| dl.delivery.mp.microsoft.com                        |                 3 |                3 |
+-----------------------------------------------------+-------------------+------------------+
| live.com                                            |                 2 |               10 |
+-----------------------------------------------------+-------------------+------------------+
| update.microsoft.com                                |                 2 |                9 |
thunt@thunt-labs:~/lab1$ _
```

Noting of note
Unique queries are well under 1,000

Let's move to lab2

- Let's check the data in the lab2 directory
- Ww will also use "lab2" database in RITA

```
thunt@thunt-labs:~/lab1$ cd ../lab2
thunt@thunt-labs:~/lab2$ ls
conn.log  dns.log  packet_filter.log  weird.log
thunt@thunt-labs:~/lab2$ _
```

Next lab

- Working with data in the lab2 directory
- Let's repeat our check for C2 over DNS
- Effectively the same commands as we used in the last lab
- Pipe through "less -S" instead of "head" if lines of data are really long

Commands

```
rita show-exploded-dns lab2 -H | less -S
```

Answers - You should see

DOMAIN	UNIQUE SUBDOM
honestimnotevil.com	
5da0b7f90908be408ac43eb80a.honestimnotevil.com	
8806d9a9068226a33b26e65071a0d496c751246292ec22b36bb5761c2762.5da0b7f90908be408ac43eb80a.honestimnotevil.com	
60a5291b4324545e080e62a0ea.honestimnotevil.com	
6a22df8dcd8e5032f95c2406362b70ddc5843efe182166d82ecf895312d7.60a5291b4324545e080e62a0ea.honestimnotevil.com	
8810f36b0b8e785c93544806d213e9c249d806a1b09b25b0bbdba6a4d016.a62e1536e8f6f362509c462faa.honestimnotevil.com	
71b3a90c8ae03782a44b552c8162238aed61cea42db89d05185f96cb2cc0.c3d37e9c6fc2384d2379ff9f16.honestimnotevil.com	
c3d37e9c6fc2384d2379ff9f16.honestimnotevil.com	
a62e1536e8f6f362509c462faa.honestimnotevil.com	

Navigate up/down/left/right using arrow keys

Answers - data output

```
thunt@thunt-labs:~/lab2$ rita show-exploded-dns lab2 | head
Domain,Unique Subdomains,Times Looked Up
honestimnotevil.com,2074,2074
5da0b7f90908be408ac43eb80a.honestimnotevil.com,21,21
8806d9a9068226a33b26e65071a0d496c751246292ec22b36bb5761c2762.5da0b7f90908be408ac43eb80a.honestimno
tevil.com,21,21
60a5291b4324545e080e62a0ea.honestimnotevil.com,7,7
6a22df8dcd8e5032f95c2406362b70ddc5843efe182166d82ecf895312d7.60a5291b4324545e080e62a0ea.honestimno
tevil.com,7,7
8810f36b0b8e785c93544806d213e9c249d806a1b09b25b0bbdba6a4d016.a62e1536e8f6f362509c462faa.honestimno
tevil.com,4,4
71b3a90c8ae03782a44b552c8162238aed61cea42db89d05185f96cb2cc0.c3d37e9c6fc2384d2379ff9f16.honestimno
tevil.com,4,4
c3d37e9c6fc2384d2379ff9f16.honestimnotevil.com,4,4
a62e1536e8f6f362509c462faa.honestimnotevil.com,4,4
```

Greater than 1,000 unique queries!

Answers

- We looked up 2,074 FQDNs within honestimnoteveil.com
- This extremely high for a domain we do not recognize
- Could very well indicate C2 over DNS

C2 over DNS only TXT queries?

```
thunt@thunt:~/lab2$ cat dns.log | zeek-cut qtype_name query | grep honestimnotevil
 | cut -f 1 | sort | uniq -c | sort -rn
    707 MX
    692 TXT
    675 CNAME
thunt@thunt:~/lab2$
```

707 + 692 + 675 = 2,074 (same as number of FQDNs found in first lab)

What's with the odd FQDNs?

```
thunt@thunt-labs:~/lab2$ cat dns.log | zeek-cut query | head
79f50108263fa92265480800043dbf9bba0.honestimnotevil.com
58cc010826f99c2b2f7167004499f9c8af.honestimnotevil.com
3d06010826a90a57036d2100456f759c3a.honestimnotevil.com
3657010826070191 8be7af0046fee50649.honestimnotevil.com
5c73010826f935d832b7620047712fe0a4.honestimnotevil.com
c4b30108267ad7b7c8931e00482fb1ae06.honestimnotevil.com
c244010826dc5cff732c1000495c204bd8.honestimnotevil.com
c94f010826e6597c4bfd7e004b46fbe42d.honestimnotevil.com
082a0108260d28f9002dea004c12ca08a3.honestimnotevil.com
5f120108261bca94ef3860004ad631a265.honestimnotevil.com
thunt@thunt-labs:~/lab2$
```

We cover decoding this C2 channel in the Packet Decoding class

Next lab!

- Working with the lab2 data, check for:
  - Beacons
  - Long connections
- Anything of note?

Hints

- Each of these was covered when investigating the lab1 data
- Refer back and repeat the commands as needed to investigate each

Commands

```
rita show-beacons-fqdn lab2
rita show-beacons-proxy lab2
rita show-beacons-fqdn lab2

rita show-long-connections lab2

cat conn.log | zeek-cut id.orig_h id.resp_h duration | sort | grep -v
-e '^$' | grep -v '-' | datamash -g 1,2 sum 3 | sort -k 3 -rn | head
```

Answers - No beacons found

```
thunt@thunt-labs:~/lab2$ rita show-beacons-fqdn lab2
No results were found for lab2
thunt@thunt-labs:~/lab2$ rita show-beacons-proxy lab2
No results were found for lab2
thunt@thunt-labs:~/lab2$ rita show-beacons lab2
No results were found for lab2
thunt@thunt-labs:~/lab2$
```

Answers - No long conns of note

```
thunt@thunt-labs:~/lab2$ rita show-long-connections lab2
No results were found for lab2
thunt@thunt-labs:~/lab2$
thunt@thunt-labs:~/lab2$ cat conn.log | zeek-cut id.orig_h id.resp_h duration | sort | grep -v -e
'^$' | grep -v '-' | datamash -g 1,2 sum 3 | sort -k 3 -rn | head
172.31.26.157   172.31.0.2      1134.198964
thunt@thunt-labs:~/lab2$ _
```

Answers - Final

- Lab1 data has a C2 beacon
- Lab2 data has C2 over DNS
- All other data looks clear

What have we learned?

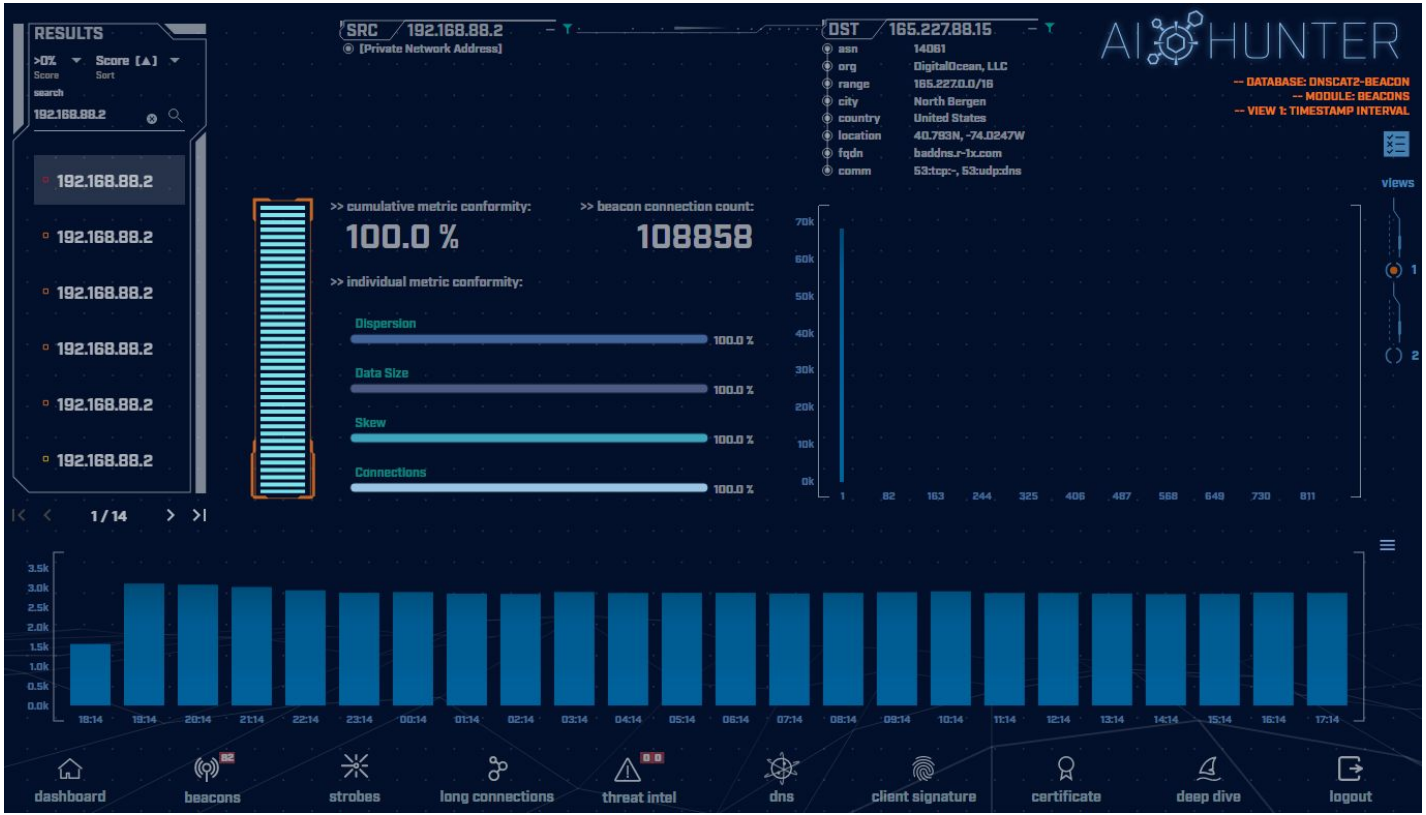
- RITA provides a consistent interface for identifying C2
- Screens pull in additional helpful info
- Even very slow beacons can be detected
- Investigation can be scripted
- Open source, so anyone can use it for free

Quick demo

- Similar data, seen through AI-Hunter
- Inexpensive commercial solution
- Automates much of the hunting process

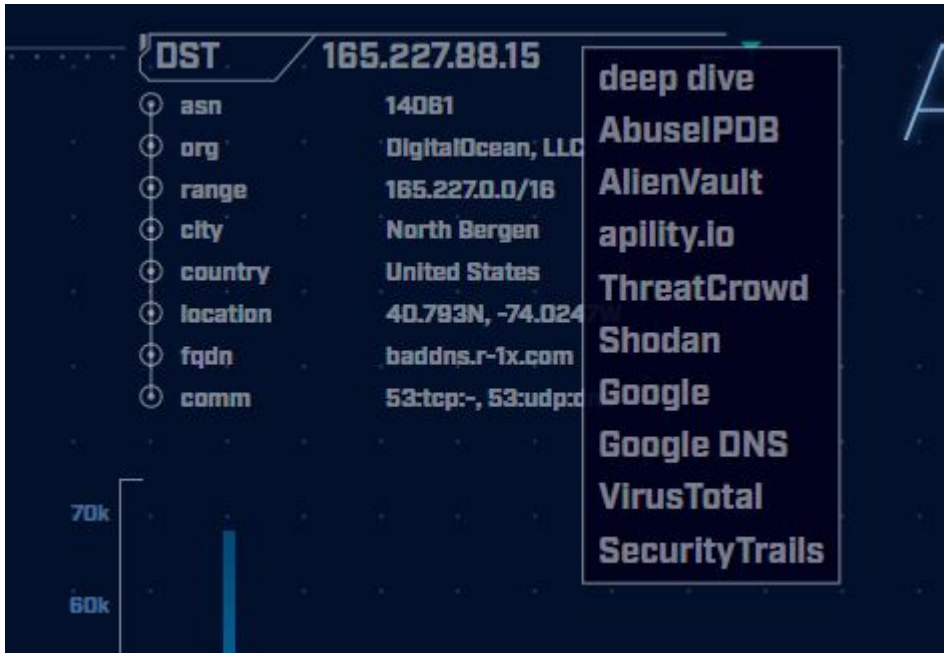
24 active hunts of 24-hours of data every single day
Top results scored, alerts sent to SIEM

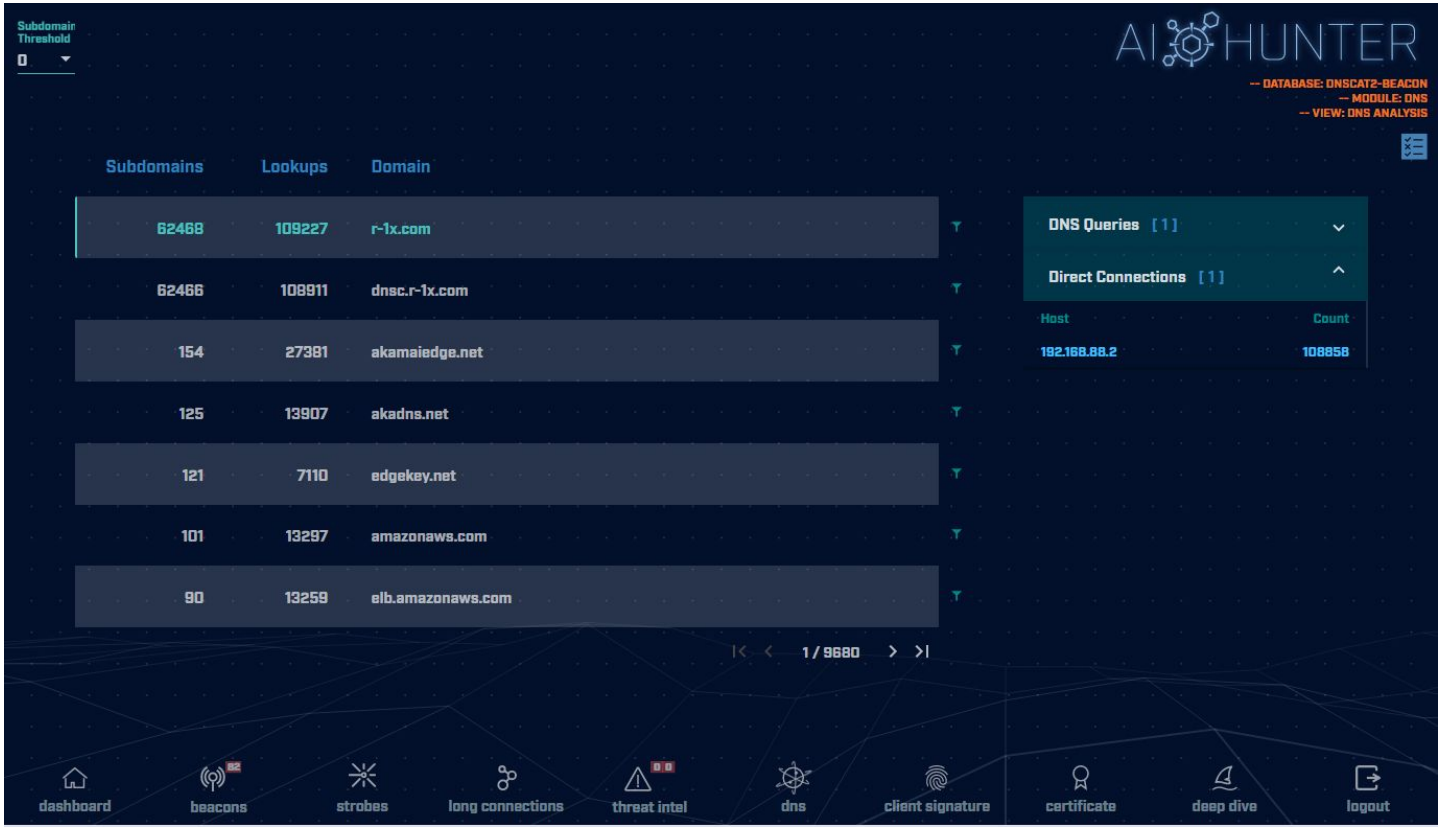
Long connections with lots of intel
View both individual and cumulative



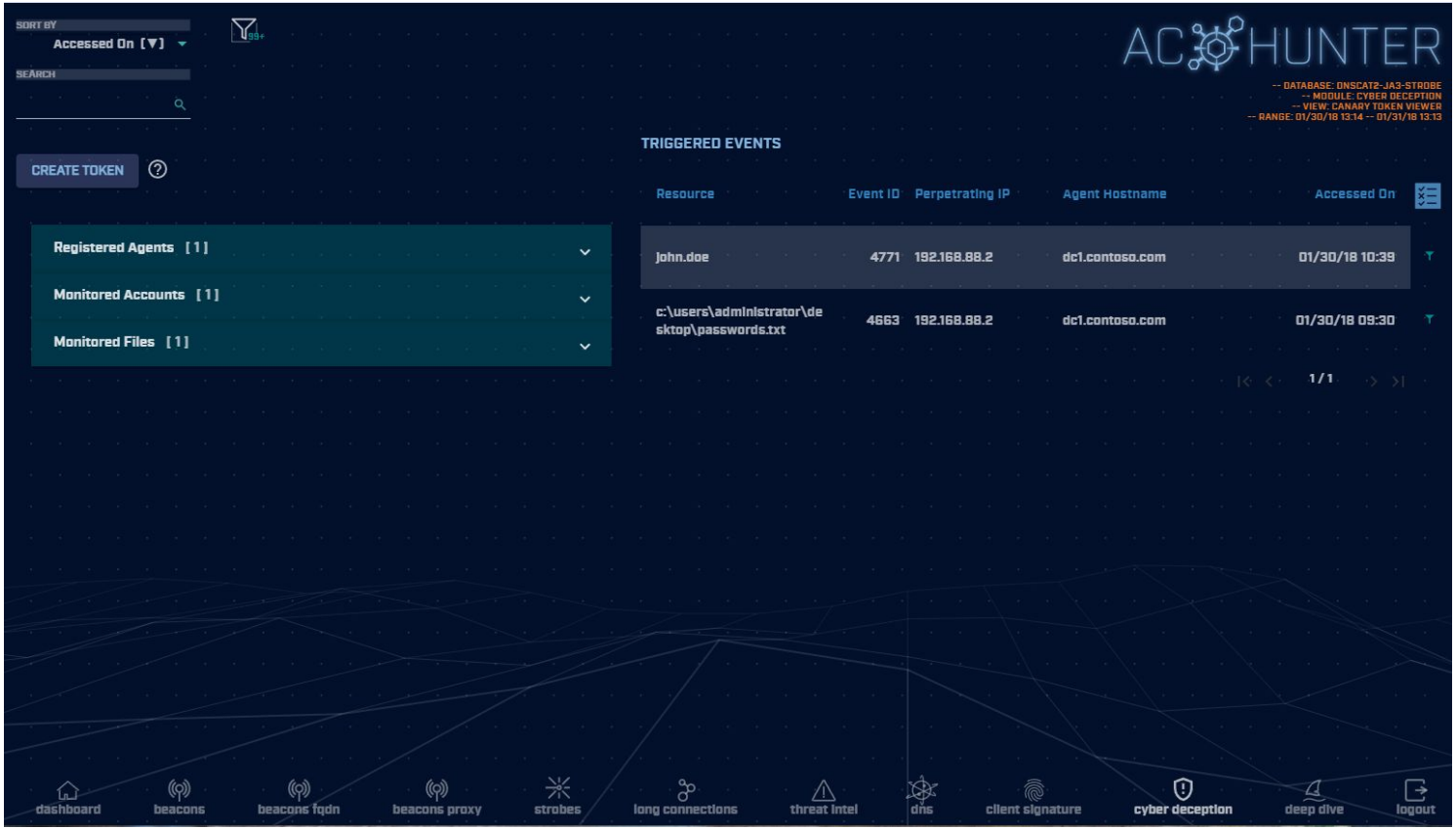
Clear beacon analysis
By both timing and session size

Resources to dig deeper

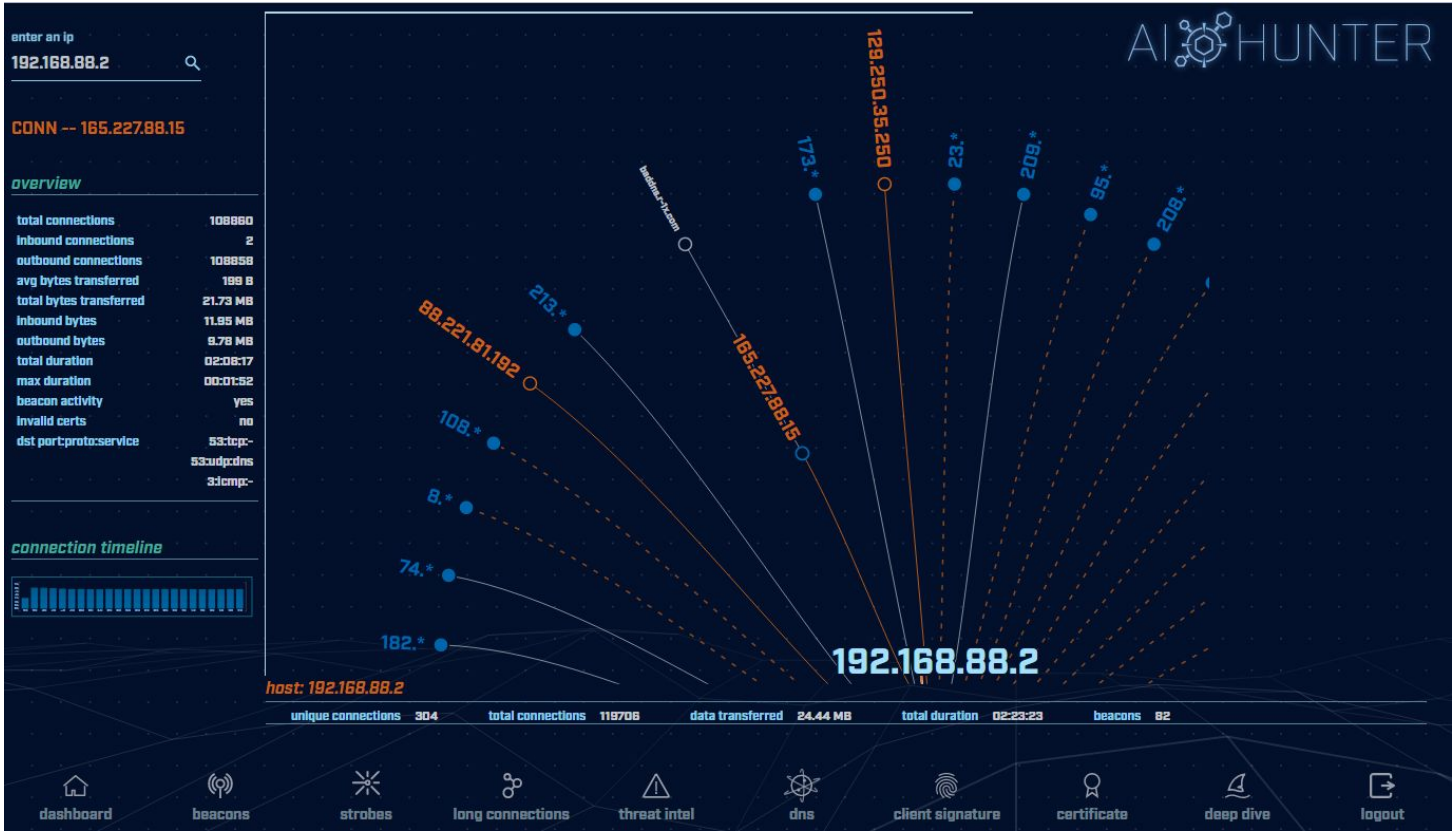




C2 over DNS analysis



Cyber Deception/Honey Tokens
Lateral movement detection with very low false positive rate



Deep dive analysis

Take home lab

- This is a bonus lab to do on your own
  - Wait at least a week
  - Will help identify what training "stuck"
- Move to the "lab3" directory
- Check for C2/DNS, long conns & beacons
- Investigate any suspect external IP's
- Do you see anything of concern?
- Hints and answers after "Wrap Up" slide

Keep honing your skills

- Check out our blog
- "Malware of the day"
  - Skip to the bottom
  - Grab the pcap
  - Find the C2 channel
  - Go back and read the blog to check your work
- Subscribe to get notifications

https://www.activecountermeasures.com/subscribe/

Other courses I'm teaching

- Advanced Network Threat Hunting
  - Shooting for Sept - Oct
  - $495

https://www.antisyphontraining.com/advanced-network-threat-hunting-w-chris-brenton/

- Getting Started with Packet Decoding
  - July 12 - 15
  - Pay what you can - $30+

https://www.antisyphontraining.com/event/getting-started-with-packet-decoding-w-chris-brenton/

Wrap Up

- Thanks for attending!
- Very special thank you to the folks behind the scenes
  - They give up their free time to help us all out
- Content feedback?
  - Please email: chris@activecountermeasures.com

Take home lab

- Move to the "lab3" directory
- Check for:
  - Beacons (all types)
  - Long connections
  - C2 over DNS
- Investigate any suspect external IP's
- Do you see anything of concern?

Hints for the take home lab

- Repeat what we did with lab1 & lab2
- Us "up arrow" key to scroll through command buffer to see commands you ran previously
- You've got this! :-)

Useful commands to try (1 of 2)

```
cat conn.log | zeek-cut id.orig_h id.resp_h
duration | sort -k 3 -rn | head

cat conn.log | zeek-cut id.orig_h id.resp_h
duration | sort | grep -v -e '^$' | grep -v '-'
| datamash -g 1,2 sum 3 | sort -k 3 -rn | head

cat conn.log | zeek-cut id.orig_h id.resp_h |
sort | uniq -c | sort -rn | head

host <IP address to investigate>
```

Useful commands to try (2/2)

```
rita show-databases
rita show-long-connections lab3 | head

rita show-long-connections lab3 | cut -d ,
-f 1,2,4 | sort | datamash -H -t , -g 1,2
sum 3 | sort -t , -k 3 -rn | head

rita show-beacons lab1 | head

rita show-exploded-dns lab1 | head
```

Answers - Long connections

```
thunt@thunt:~/lab3$ cat conn.log | zeek-cut id.orig_h id.resp_h duration | sort -k
 3 -rn | head
192.168.99.52   167.71.97.235   86387.734233
192.168.99.52   162.250.5.77    86347.153666
192.168.99.52   52.117.209.74   9868.617938
192.168.99.52   162.250.2.168   6735.118200
192.168.99.52   52.184.217.56   129.924272
192.168.99.52   52.184.212.181  129.754188
192.168.99.52   52.184.213.21   129.130822
192.168.99.52   52.184.212.181  129.123714
192.168.99.52   52.167.17.97    129.057349
192.168.99.52   52.167.17.97    128.896376
thunt@thunt:~/lab3$
```

Answers - Cumulative comm time

```
thunt@thunt:~/lab3$ cat conn.log | zeek-cut id.orig_h id.resp_h duration | sort |
grep -v -e '^$' | grep -v '-' | datamash -g 1,2 sum 3 | sort -k 3 -rn | head
192.168.99.52   167.71.97.235   86387.734233
192.168.99.52   162.250.5.77    86347.153666
192.168.99.52   52.117.209.74   9868.617938
192.168.99.52   52.184.217.56   7065.516309
192.168.99.52   52.184.213.21   7056.53546
192.168.99.52   162.250.2.168   6735.1182
192.168.99.52   52.184.212.181  6646.856637
192.168.99.52   239.255.255.250 2294.038962
fe80::d048:42e0:8448:187c       ff02::c 2281.05815
fe80::2126:bcd7:16f4:8cdb       ff02::c 2242.310744
thunt@thunt:~/lab3$
```

Same two top IPs

Answers - Beacons

```
thunt@thunt:~/lab3$ cat conn.log | zeek-cut id.orig_h id.resp_h | sort | uniq -c |
 sort -rn | head
    339 192.168.99.52   224.0.0.251
    319 192.168.99.52   208.67.222.222
    288 fe80::fd16:6e8:118e:81cd        ff02::fb
    288 fe80::fd16:6e8:118e:81cd        ff02::16
    288 fe80::d048:42e0:8448:187c       ff02::fb
    288 fe80::d048:42e0:8448:187c       ff02::16
    288 fe80::b8d7:3773:ab6e:7fc9       ff02::fb
    288 fe80::b8d7:3773:ab6e:7fc9       ff02::16
    288 fe80::5d7e:4fb3:8fbc:d59        ff02::fb
    288 fe80::5d7e:4fb3:8fbc:d59        ff02::16
thunt@thunt:~/lab3$
```

Nothing of note

Answers - RITA

```
thunt@thunt:~/lab1$ rita show-long-connections lab3 | head -5
Source IP,Destination IP,Port:Protocol:Service,Duration
192.168.99.52,167.71.97.235,9200:tcp:-,86387.7
192.168.99.52,162.250.5.77,5938:tcp:-,86347.2
192.168.99.52,52.117.209.74,5938:tcp:-,9868.62
192.168.99.52,162.250.2.168,5938:tcp:-,6735.12
thunt@thunt:~/lab1$ rita show-beacons lab3 | head -5
Score,Source IP,Destination IP,Connections,Avg. Bytes,Intvl Range,Size Range,Top I
ntvl,Top Size,Top Intvl Count,Top Size Count,Intvl Skew,Size Skew,Intvl Dispersion
,Size Dispersion
0.835,192.168.99.52,52.230.222.68,59,546,31350,2696,840,181,46,48,0,0,0,0
0.834,192.168.99.52,52.242.211.89,21,826,1651,2696,1680,181,14,11,0,0,0,0
0.833,192.168.99.52,104.71.255.238,24,5429,21721,40,1800,505,16,22,0,0,0,0
0.658,192.168.99.52,52.184.213.21,65,5392,2199,120,900,1883,28,33,0.99757,0,1,0
thunt@thunt:~/lab1$ rita show-exploded-dns lab3 | head -5
Domain,Unique Subdomains,Times Looked Up
microsoft.com,10,237
teamviewer.com,6,36
mp.microsoft.com,5,111
8.e.f.ip6.arpa,4,20
thunt@thunt:~/lab1$
```

Answers - Investigate IPs

```
thunt@thunt:~/lab3$ host 167.71.97.235
235.97.71.167.in-addr.arpa domain name pointer demo1.aihhosted.com.
thunt@thunt:~/lab3$ host 162.250.5.77
77.5.250.162.in-addr.arpa domain name pointer US-NJC-ANX-R010.teamviewer.com.
thunt@thunt:~/lab3$ _
```

Business need?

Answers - Final

- Two long connections found
- Unlikely (but not impossible) we have any beacons
- For the two long connections
  - First was discussed earlier (business partner)
  - The second is TeamViewer
- Is there a business need to run TeamViewer on this system?