Threat Hunting via DNS

Eric Conrad (econrad@backshore.net)

https://ericconrad.com

Twitter: @eric_conrad

Welcome!

- Welcome to my talk!
- A copy of these slides are available on https://ericconrad.com

CIS 8.7: Malware Defenses

- *Enable Domain Name System (DNS) query logging to detect hostname lookups for known malicious domains.*[1]
- DNS logs are one of the most actionable threat hunting/SOC/SIEM data sources
- In addition to logging, viewing/dumping and inspecting the DNS cache is a good short-term investigative tool
- Note that DNS may be logged on the DNS server or endpoints, or sniffed on the network using tools like Zeek
  - Encrypted DNS is impacting both, as we will discuss shortly

Methods for Collecting DNS logs

- Sniff on the wire, analyze with Zeek
  - A great approach, now heavily impacted by DNS encryption (discussed next)
- Have clients resolve via local recursive DNS servers and log there
- All major DNS server software supports query logging (responses can be tricky):
  - Bind (syslog or local text file)
  - DNS Query Logging on Windows 2008/2012 (local text file)
  - DNS Analytical Logging on Windows 2012R2+ (logs in event log format to (**Logs\Microsoft\Windows\DNS-Server)**
- Sysmon supports Windows client logging

DNS Encryption

A big trend on the encryption front that is impacting a vital analytics source: DNS queries

DNS query encryption concerns itself primarily with increasing the privacy of users' communications

- This dovetails nicely with the push toward ubiquitous HTTPS from a traffic privacy perspective

Inscrutable DNS queries can pose secops challenges:

- Blindness to adversaries' intentional use of DNS
- Diminished user monitoring/analytic capabilities

Facing Reality

- This talk will not debate the merits of encrypted DNS vs. traditional DNS via UDP/TCP port 53 (sometimes called Do53)
  - Encrypted DNS provides privacy to the end user
  - Do53 provides easy centralized monitoring for companies, ISPs, etc.
    - And easy monetization for ISPs
- Years of network defense have taught me to be a realist, and not fight the incoming tide
- DNS over HTTPS (DoH) is coming on like a freight train
  - Network defenders need to prepare accordingly

DNS over HTTPS (DoH) and DNS over TLS (DoT)

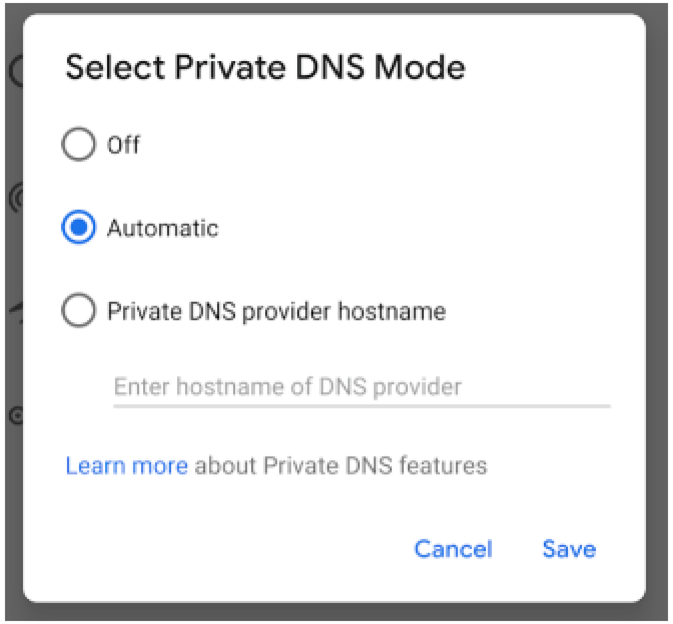
- DNS over HTTPS (DoH) and DNS over TLS (DoT) are impacting the ability to monitor DNS queries
  - This is true for Intrusion Detection Systems such as Zeek, as well as logging requests on the local DNS resolver/forwarder
- DNS over HTTPS uses TCP port 443 and looks like normal HTTPS traffic from a network perspective
- DNS over TLS uses TCP port 853, so network operators/defenders know that it’s (encrypted) DNS traffic
  - DoT can be easily blocked by a firewall, forcing resolution back to DNS
- In both cases: analyzing the content on the wire requires SSL/TLS interception/decryption

The Only Constant is Change

- This talk with track DoH in Firefox most closely
  - Firefox is the currently the most aggressive browser in regard to DNS encryption
- DoH/DoT adoption is evolving very rapidly
- I will track updates on https://ericconrad.com
- Jim Troutman's 2020 Shmoocon Firetalk is fantastic:
  - http://www.nepeeringforum.org/troutman/troutman-DoH-DoT-QuadX-Da-Faq.pdf

DoH and DoT

- The early trend: browsers tend to support DNS over HTTPS (for resolution within the browser), while **Linux** operating systems tend to support DNS over TLS for default operating system resolution
  - DNS over TLS is now used by default by Android (called "Private DNS Mode")
- Firefox and Chrome now support DNS over HTTPS
- Microsoft recently announced plans to support DoH in Windows 10
  - Windows 10 Insider Preview currently supports DoH (not enabled by default)
- In the short-term: DoH is "winning"



Select Private DNS Mode

○ Off

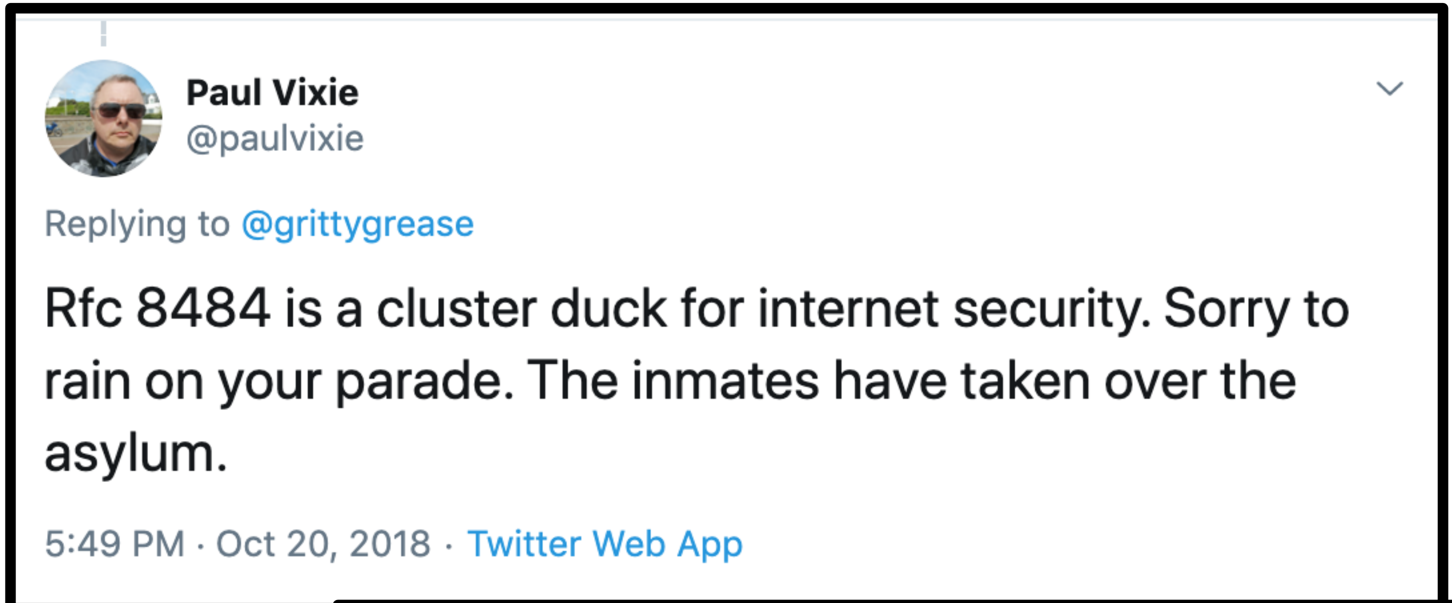
● Automatic

○ Private DNS provider hostname

Enter hostname of DNS provider

Learn more about Private DNS features

Paul Vixie on DoH

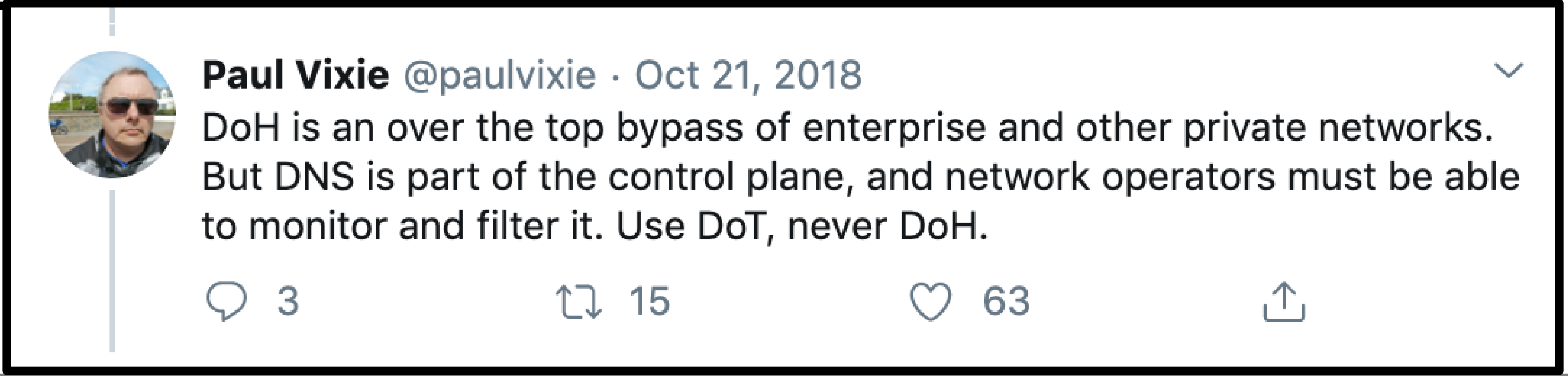


Paul Vixie
@paulvixie

Replying to @grittygrease

Rfc 8484 is a cluster duck for internet security. Sorry to rain on your parade. The inmates have taken over the asylum.

5:49 PM · Oct 20, 2018 · Twitter Web App

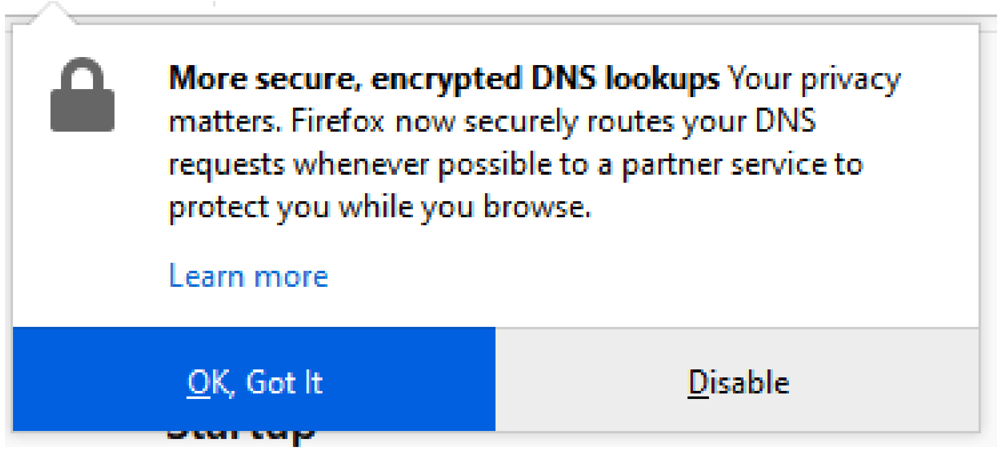


Paul Vixie @paulvixie · Oct 21, 2018

DoH is an over the top bypass of enterprise and other private networks. But DNS is part of the control plane, and network operators must be able to monitor and filter it. Use DoT, never DoH.

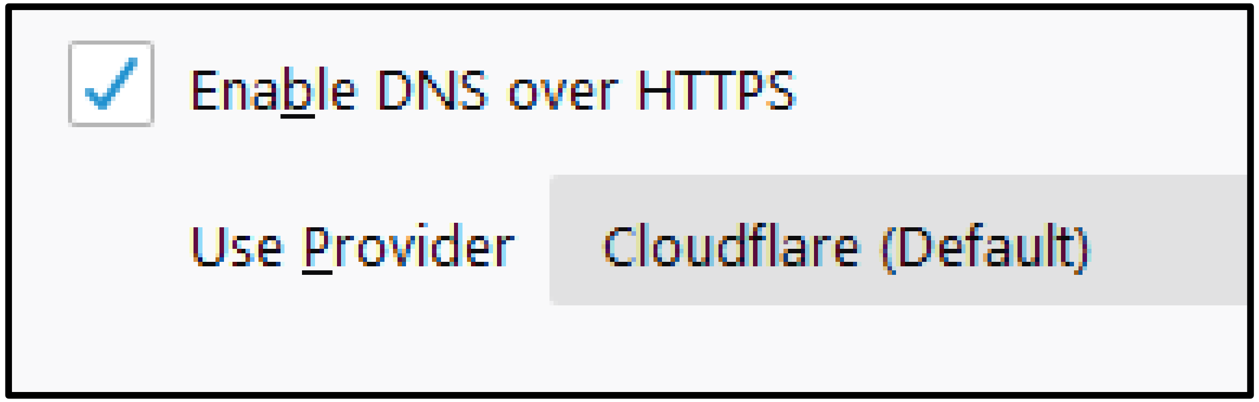
DoH Status update

- Chrome 83 (released May 19th 2020) enables DoH: "We've enabled an experiment in Chrome 83 for a fraction of our users with the following scope: platforms: Windows, Mac, Chrome OS."
- Firefox now enables DoH by default in the US (it prompts first)



More secure, encrypted DNS lookups Your privacy matters. Firefox now securely routes your DNS requests whenever possible to a partner service to protect you while you browse.

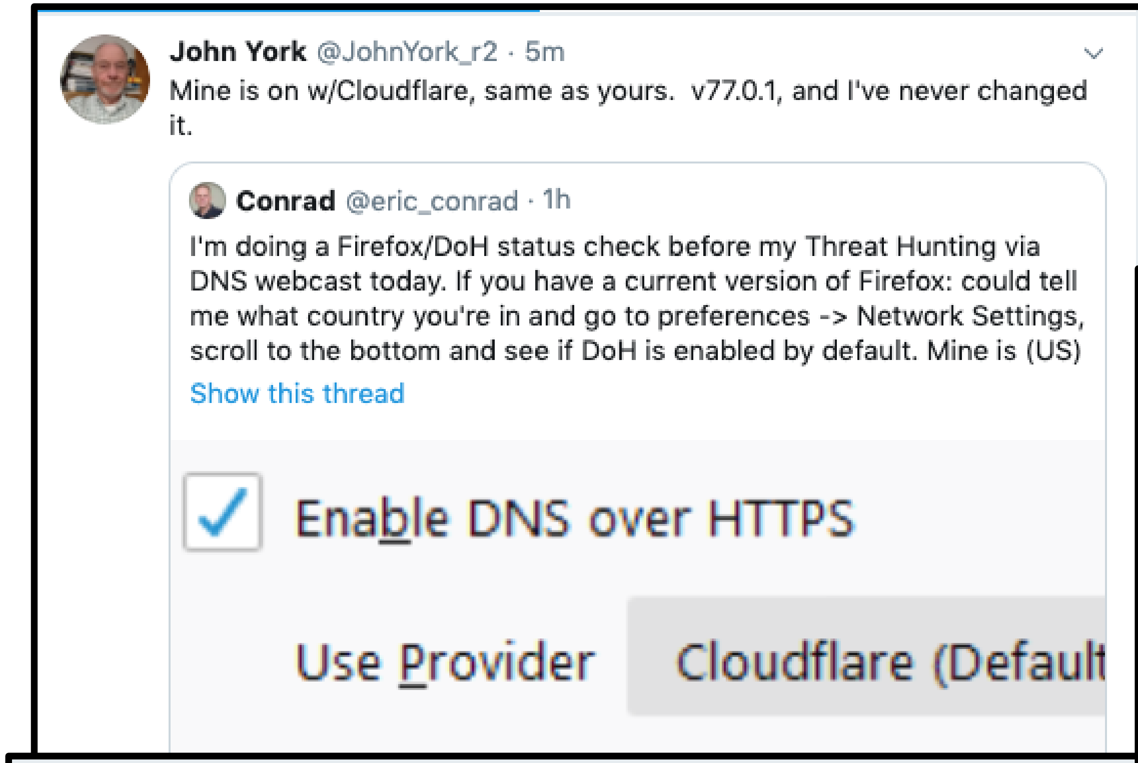
Learn more



☑ Enable DNS over HTTPS

Use Provider Cloudflare (Default)

Firefox/DoH Status Check (June 24th 2020)



John York @JohnYork_r2 · 5m

Mine is on w/Cloudflare, same as yours. v77.0.1, and I've never changed it.

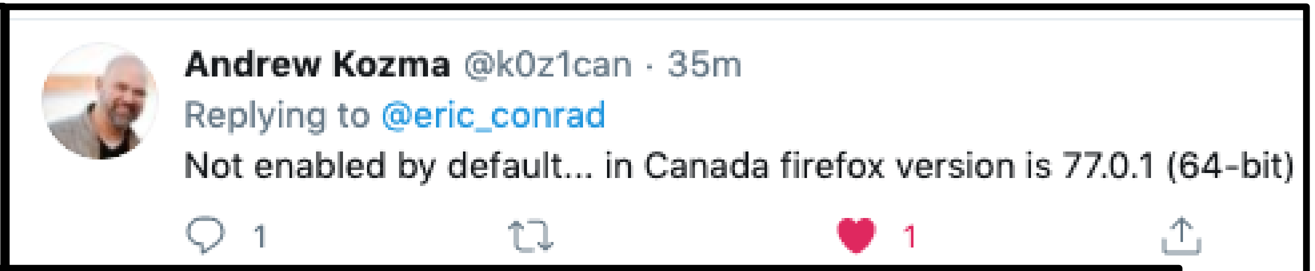
Conrad @eric_conrad · 1h

I'm doing a Firefox/DoH status check before my Threat Hunting via DNS webcast today. If you have a current version of Firefox: could tell me what country you're in and go to preferences -> Network Settings, scroll to the bottom and see if DoH is enabled by default. Mine is (US)

Show this thread

☑ Enable DNS over HTTPS

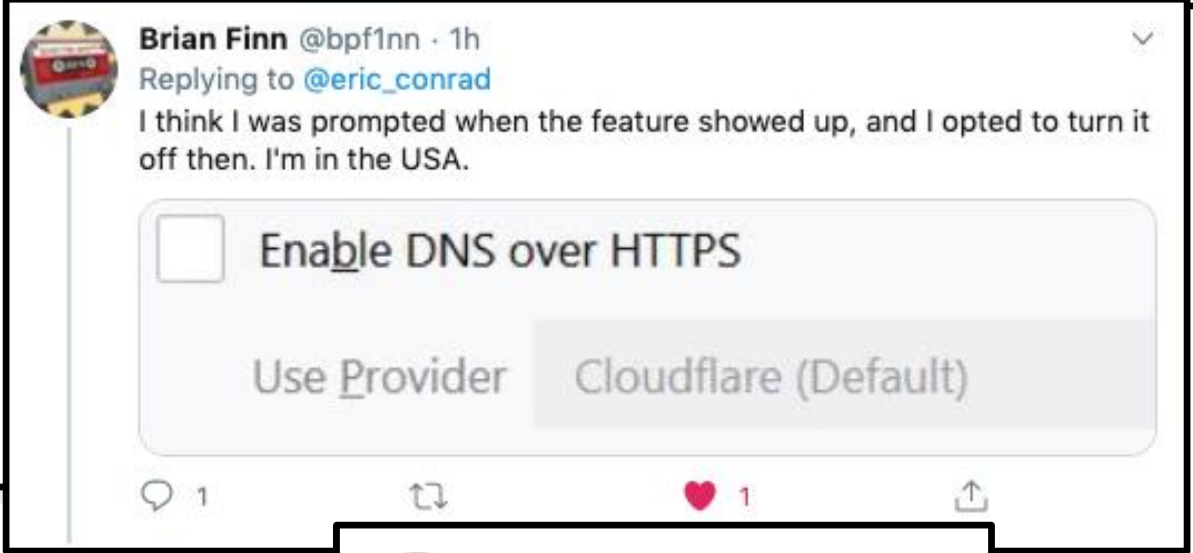
Use Provider Cloudflare (Default



Andrew Kozma @k0z1can · 35m

Replying to @eric_conrad

Not enabled by default... in Canada firefox version is 77.0.1 (64-bit)



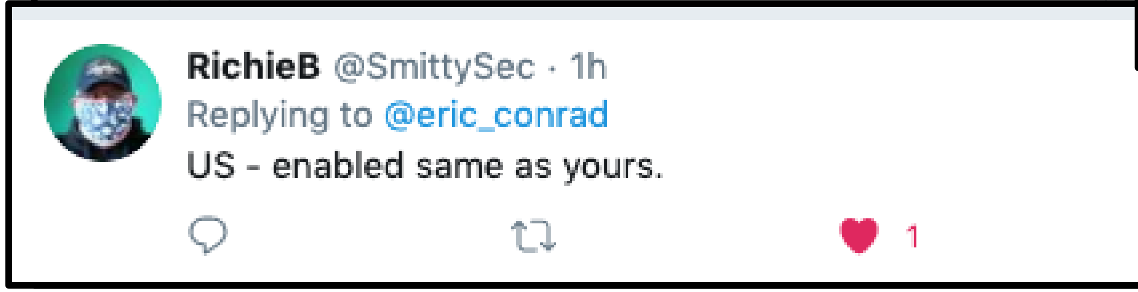
Brian Finn @bpf1nn · 1h

Replying to @eric_conrad

I think I was prompted when the feature showed up, and I opted to turn it off then. I'm in the USA.

☐ Enable DNS over HTTPS

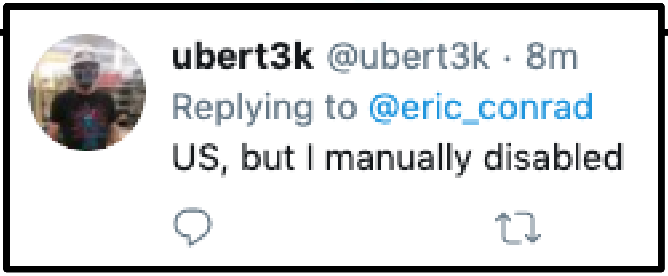
Use Provider Cloudflare (Default)



RichieB @SmittySec · 1h

Replying to @eric_conrad

US - enabled same as yours.



ubert3k @ubert3k · 8m

Replying to @eric_conrad

US, but I manually disabled

DoH in Firefox and Chrome

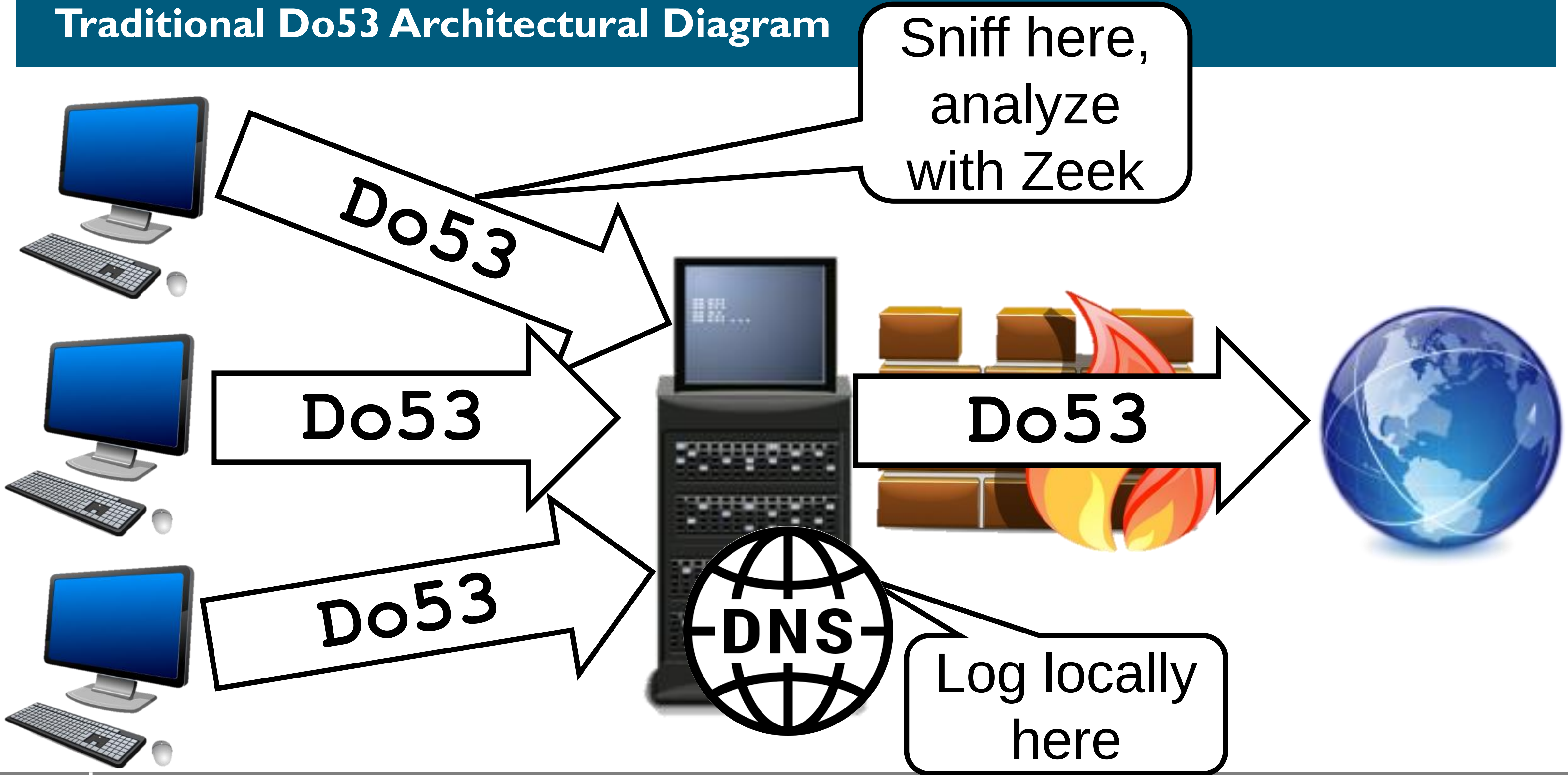
- Firefox bypasses the local system DNS settings when using DoH, and sets the DNS provider to Cloudflare by default
  - Other options include NextDNS and Custom
  - This bypass policy has proven to be controversial
- Chrome uses a different approach: If the system is using a provider on this list for DNS resolution, Chrome will “auto-upgrade” the DNS setting from DNS to DoH, and keep the same provider:
  - Cleanbrowsing, Cloudflare, Comcast, DNS.SB, Google, OpenDNS, Quad9
  - Otherwise: Chrome will continue using regular DNS, and the existing provider

What is your Organization's Encrypted DNS Policy?

Some options to consider:

- Embrace the privacy, and use it
  - Easy decision for organizations that don't currently log/analyze DNS
  - Great personal choice for home/travel/etc.
- Disable DoH and DoT (when possible), force resolution via Do53, and log via traditional methods
- Allow both DoH and DoT to local servers, and log there
- Worth noting: much like VPN traffic: most encrypted DNS will eventually resolve via Do53 upstream
  - One exception DoH/DoT traffic to an authoritative name server

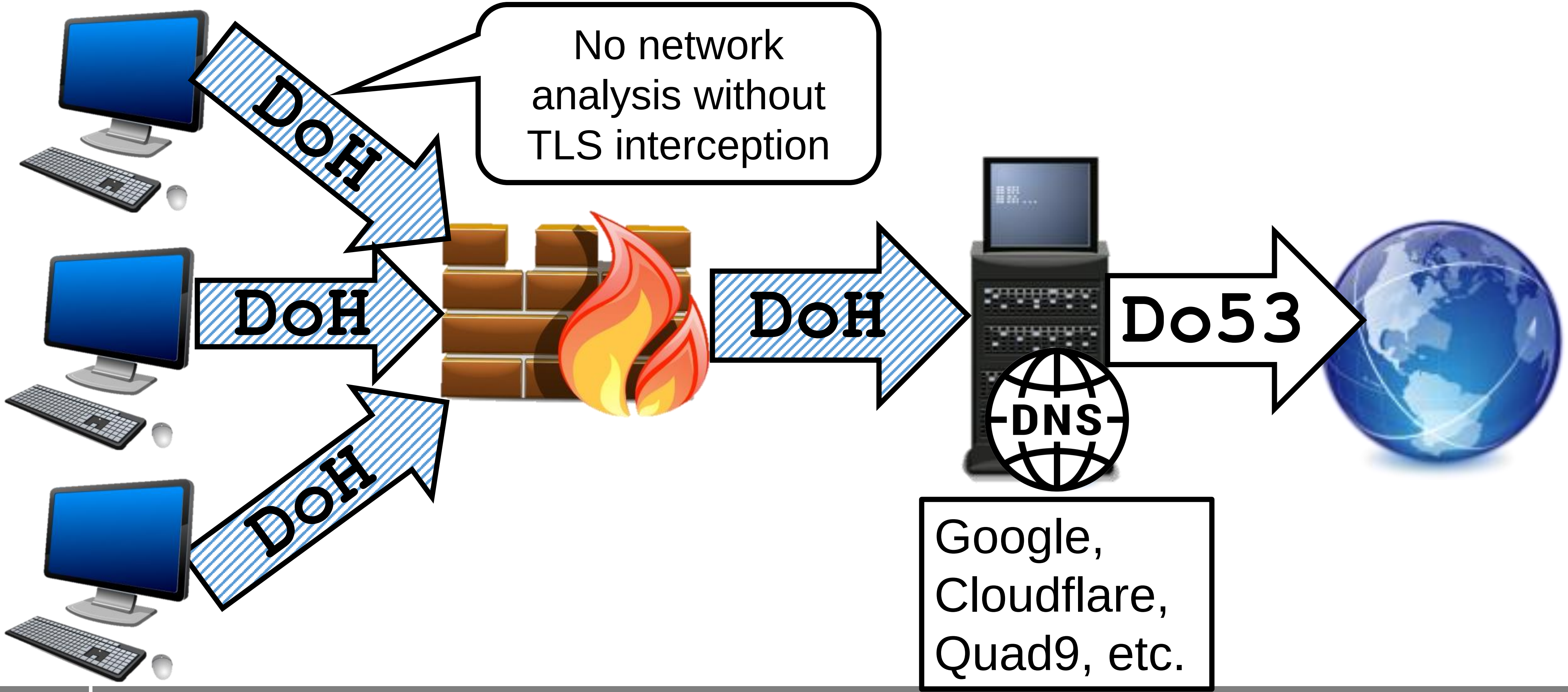
Traditional Do53 Architectural Diagram



Sniff here, analyze with Zeek

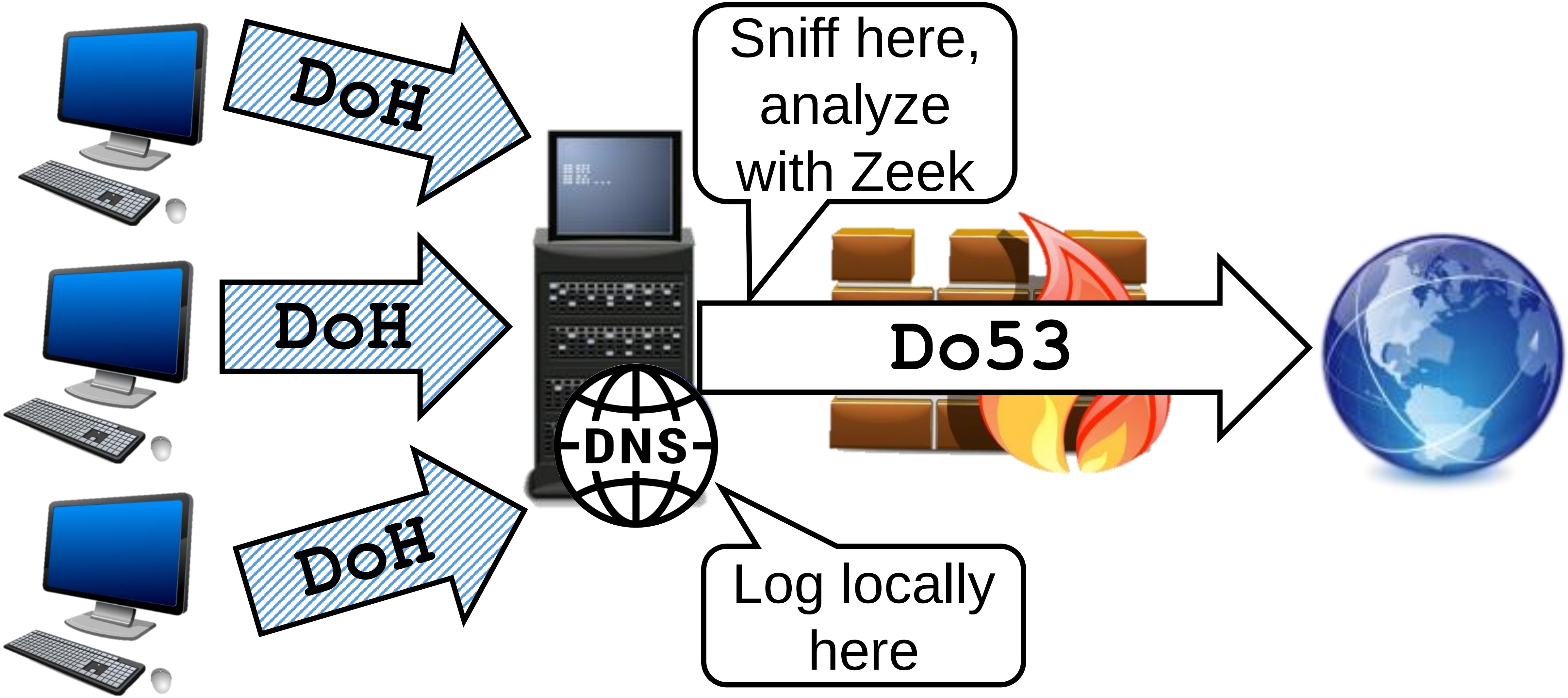
Log locally here

Third-Party DoH Architectural Diagram



Google, Cloudflare, Quad9, etc.

Custom DoH Architectural Diagram



Disabling DoH in Firefox and Chrome

Firefox:

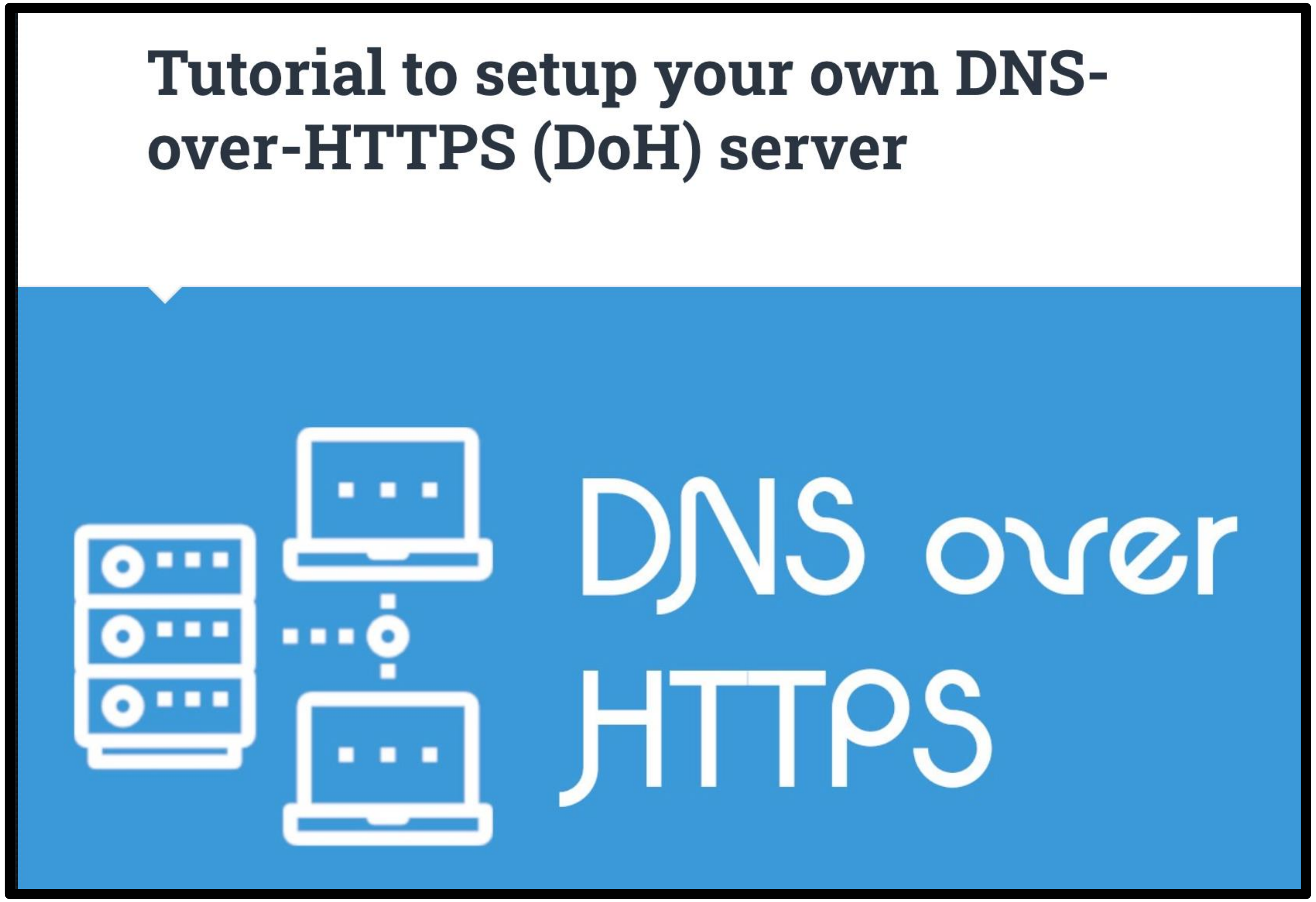
- To disable Firefox DoH for the enterprise: do not allow this canary domain to resolve: **use-application-dns.net**
- To disable DoH in a browser, go to Settings -> Network Settings -> Connection settings, and uncheck “Enable DNS over HTTPS”

Chrome:

- There is no canary domain support
- If using a supported DNS provider, Chrome will auto-upgrade any Do53 connection to DoH
- Workaround: if you don't use a supported DNS provider, Chrome will use Do53

Setting up your own DoH server

- This guide is fantastic
- Instructions for Ubuntu 18.04
  - Also has sections on setting up PiHole and DoT
- I was able to set up a DoH server in Digital Ocean’s cloud in <10 minutes

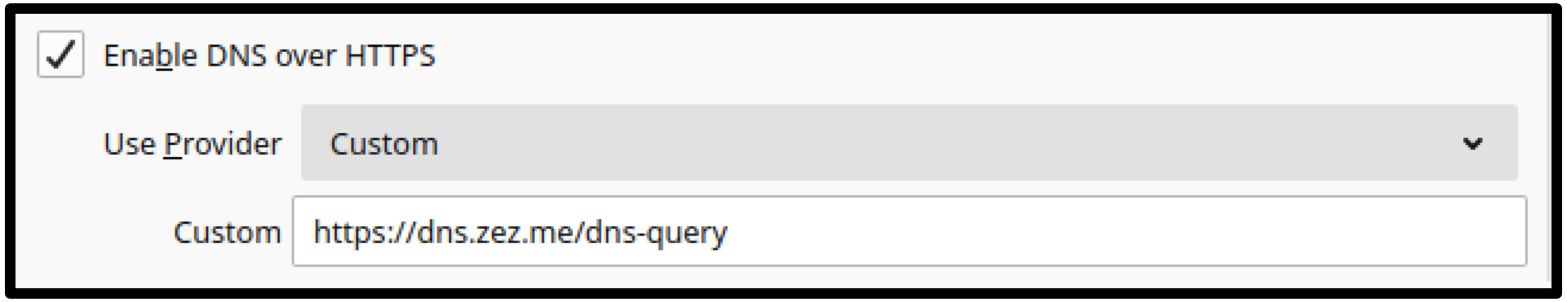


Tutorial to setup your own DNS-over-HTTPS (DoH) server

- https://www.aaflalo.me/2018/10/tutorial-setup-dns-over-https-server/

Logging on a local doh-server

- Configure Firefox to use a custom DoH server

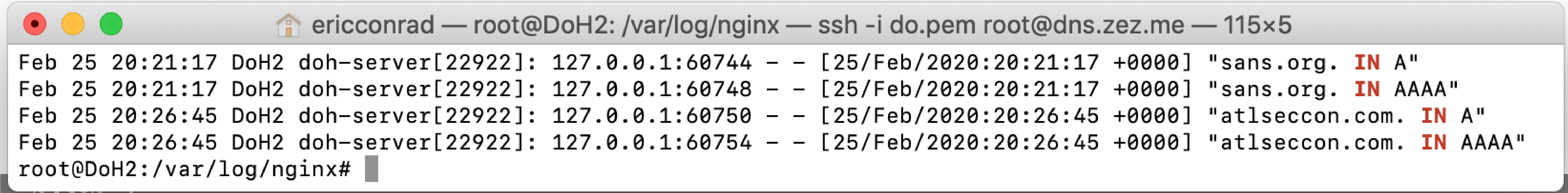


☑ Enable DNS over HTTPS

Use Provider: Custom

Custom: https://dns.zez.me/dns-query

- Set verbose to “true” in doh-server.conf
  - Logs queries only. Does not appear to have an option for logging responses, but it’s open source, and can be modified to do so

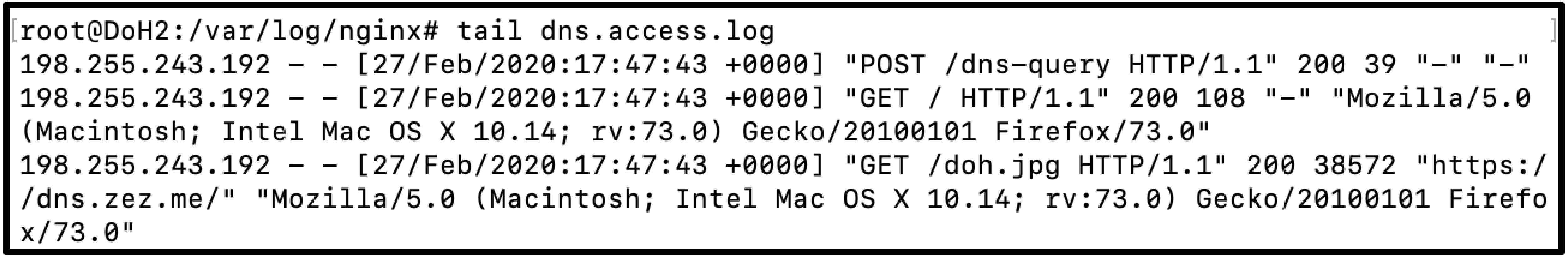


ericconrad — root@DoH2: /var/log/nginx — ssh -i do.pem root@dns.zez.me — 115×5

```
Feb 25 20:21:17 DoH2 doh-server[22922]: 127.0.0.1:60744 - - [25/Feb/2020:20:21:17 +0000] "sans.org. IN A"
Feb 25 20:21:17 DoH2 doh-server[22922]: 127.0.0.1:60748 - - [25/Feb/2020:20:21:17 +0000] "sans.org. IN AAAA"
Feb 25 20:26:45 DoH2 doh-server[22922]: 127.0.0.1:60750 - - [25/Feb/2020:20:26:45 +0000] "atlseccon.com. IN A"
Feb 25 20:26:45 DoH2 doh-server[22922]: 127.0.0.1:60754 - - [25/Feb/2020:20:26:45 +0000] "atlseccon.com. IN AAAA"
root@DoH2:/var/log/nginx# 
```

Detection: DoH is HTTPS

- DoH **is** HTTPS
  - Uses web servers such as Nginx and Apache, leverages x.509 certs, etc.
- For example:
  - https://dns.zez.me – regular HTTPS site
  - https://dns.zez.me/dns-query - resolves DoH requests via a POST



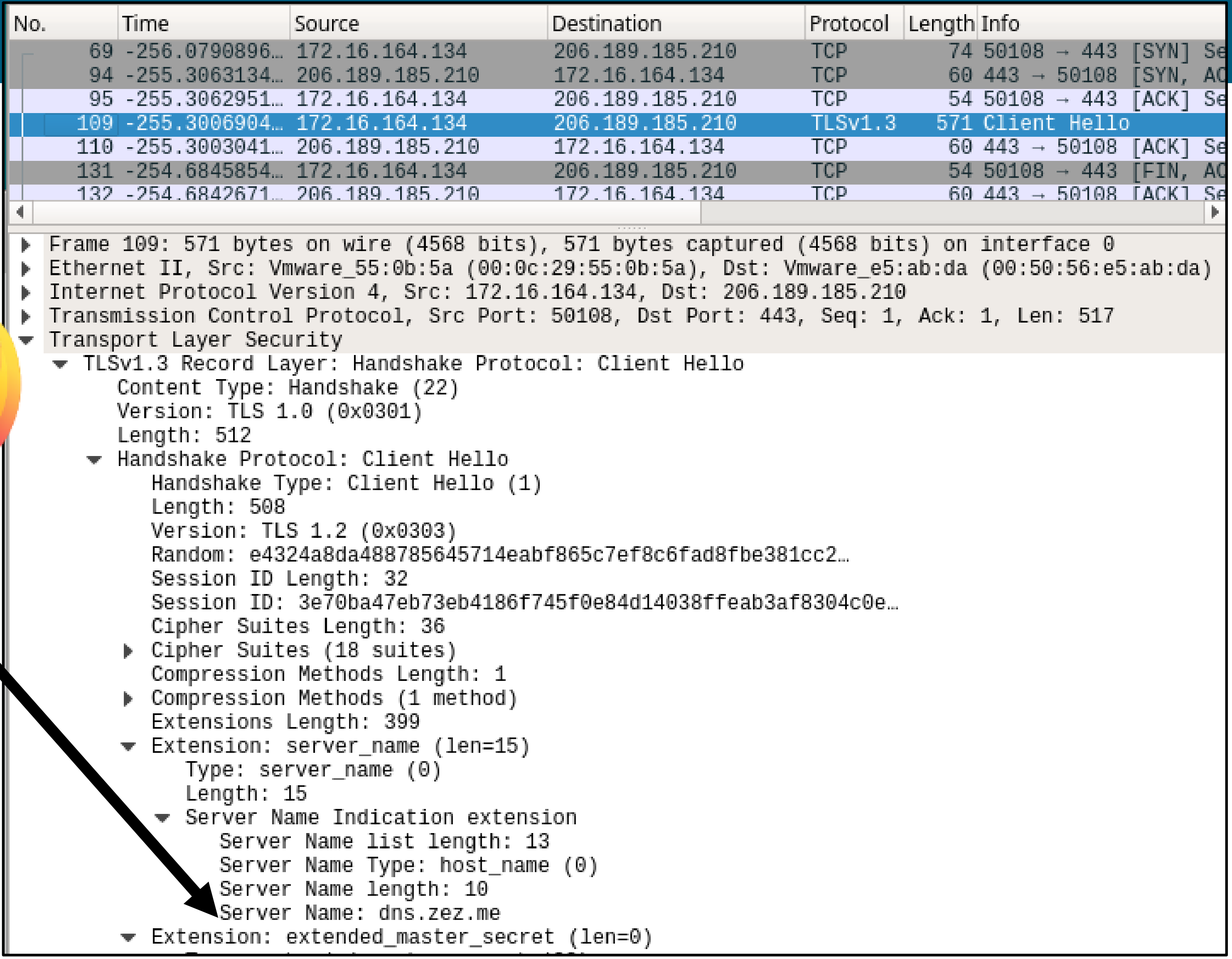
```
root@DoH2:/var/log/nginx# tail dns.access.log
198.255.243.192 - - [27/Feb/2020:17:47:43 +0000] "POST /dns-query HTTP/1.1" 200 39 "-" "-"
198.255.243.192 - - [27/Feb/2020:17:47:43 +0000] "GET / HTTP/1.1" 200 108 "-" "Mozilla/5.0
(Macintosh; Intel Mac OS X 10.14; rv:73.0) Gecko/20100101 Firefox/73.0"
198.255.243.192 - - [27/Feb/2020:17:47:43 +0000] "GET /doh.jpg HTTP/1.1" 200 38572 "https:/
/dns.zez.me/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:73.0) Gecko/20100101 Firefo
x/73.0"
```

DoH DNS Request

Firefox DoH request via **`dns.zez.me`**

The DoH virtual server name is shown in the Server Name Indication (SNI) field

The actual DNS query carried via DoH is encrypted



```
No.  Time             Source           Destination      Protocol  Length Info
 69 -256.0790896…  172.16.164.134   206.189.185.210  TCP           74 50108 → 443 [SYN] Se
 94 -255.3063134…  206.189.185.210  172.16.164.134   TCP           60 443 → 50108 [SYN, AC
 95 -255.3062951…  172.16.164.134   206.189.185.210  TCP           54 50108 → 443 [ACK] Se
109 -255.3006904…  172.16.164.134   206.189.185.210  TLSv1.3      571 Client Hello
110 -255.3003041…  206.189.185.210  172.16.164.134   TCP           60 443 → 50108 [ACK] Se
131 -254.6845854…  172.16.164.134   206.189.185.210  TCP           54 50108 → 443 [FIN, AC
132 -254.6842671…  206.189.185.210  172.16.164.134   TCP           60 443 → 50108 [ACK] Se
```

```
▶ Frame 109: 571 bytes on wire (4568 bits), 571 bytes captured (4568 bits) on interface 0
▶ Ethernet II, Src: Vmware_55:0b:5a (00:0c:29:55:0b:5a), Dst: Vmware_e5:ab:da (00:50:56:e5:ab:da)
▶ Internet Protocol Version 4, Src: 172.16.164.134, Dst: 206.189.185.210
▶ Transmission Control Protocol, Src Port: 50108, Dst Port: 443, Seq: 1, Ack: 1, Len: 517
▼ Transport Layer Security
  ▼ TLSv1.3 Record Layer: Handshake Protocol: Client Hello
      Content Type: Handshake (22)
      Version: TLS 1.0 (0x0301)
      Length: 512
    ▼ Handshake Protocol: Client Hello
        Handshake Type: Client Hello (1)
        Length: 508
        Version: TLS 1.2 (0x0303)
        Random: e4324a8da488785645714eabf865c7ef8c6fad8fbe381cc2…
        Session ID Length: 32
        Session ID: 3e70ba47eb73eb4186f745f0e84d14038ffeab3af8304c0e…
        Cipher Suites Length: 36
      ▶ Cipher Suites (18 suites)
        Compression Methods Length: 1
      ▶ Compression Methods (1 method)
        Extensions Length: 399
      ▼ Extension: server_name (len=15)
          Type: server_name (0)
          Length: 15
        ▼ Server Name Indication extension
            Server Name list length: 13
            Server Name Type: host_name (0)
            Server Name length: 10
            Server Name: dns.zez.me
      ▼ Extension: extended_master_secret (len=0)
```

Network-based DoH prevention

- If you can't configure each client or use canary domains to disable DoH: Network-based DoH prevention (such as firewalling) isn’t practically possible, short of SSL/TLS proxying and inspection
- HTTPS access to **known** DoH resolvers can be blocked
  - 1.1.1.1:443, 8.8.8.8:443, etc.
- HTTPS access to **unknown** DoH resolvers cannot be easily blocked
  - 206.189.185.210:443 (my custom DoH server)

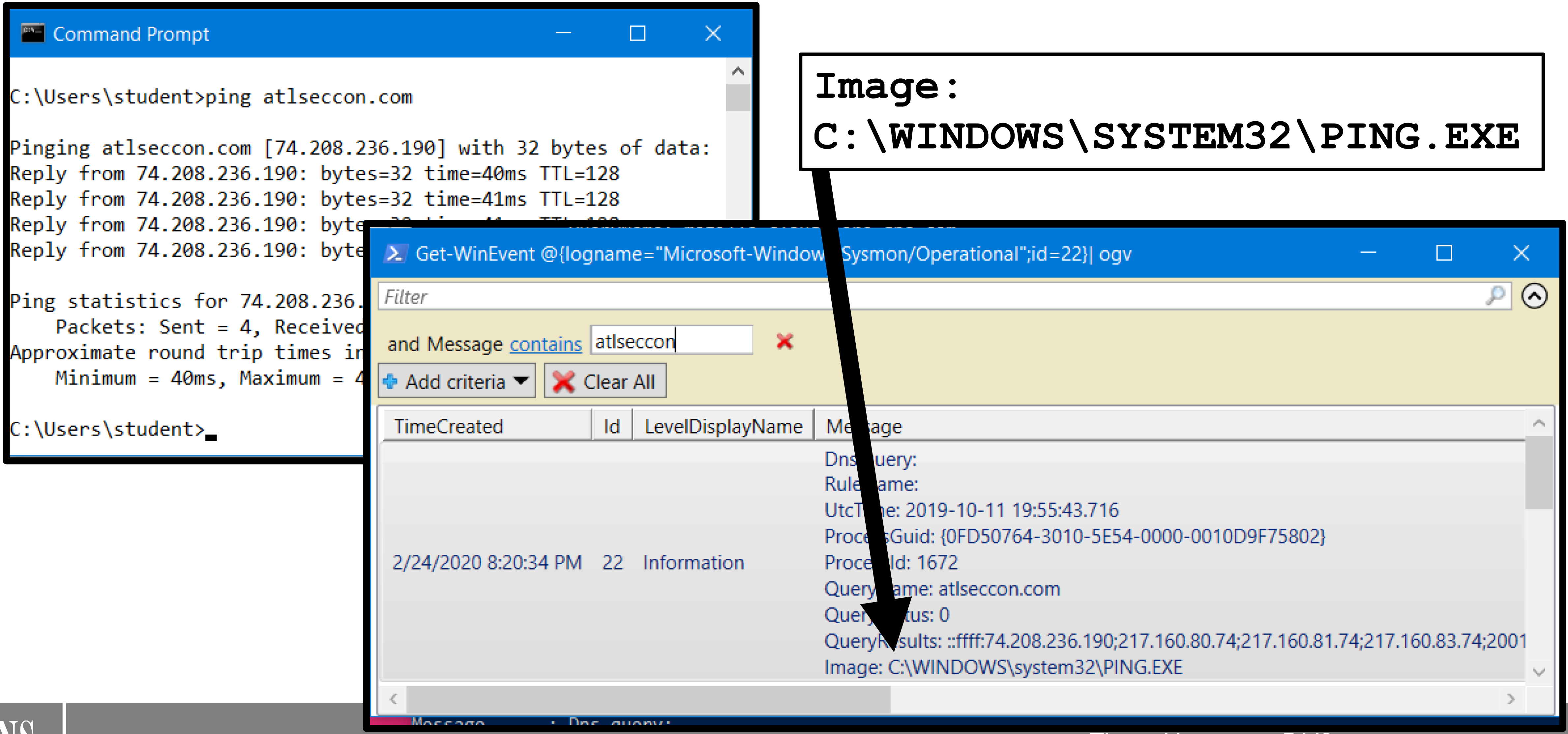
Network-based DoH Detection

- Known DoH resolvers can be detected via simple IP/port-based IDS rules (1.1.1.1:443, etc.)
- Beaconing detection can detect DoH to any site, including unknown resolvers
  - Browsers usually resolve via the same DoH server (HTTPS site) 1000+ times/day
- RITA is a great tool for detecting beaconing
  - https://www.blackhillsinfosec.com/projects/rita/
- Check out SANS STI student Drew Hjelm’s amazing paper: A New Needle and Haystack: Detecting DNS over HTTPS Usage
  - https://www.sans.org/reading-room/whitepapers/dns/paper/39160

DNS Logging via Sysmon

- Microsoft’s Sysmon can now log local DNS queries
- Plays nicely with centralized event collection via Windows Event Forwarding
- Killer threat hunting feature: it shows the client application that made the DNS request
- Note that Firefox’ DoH implementation bypasses local resolving entirely
- Sysmon **does not** log Firefox’s DoH DNS requests

Sysmon DNS Logging Example



```
C:\Users\student>ping atlseccon.com

Pinging atlseccon.com [74.208.236.190] with 32 bytes of data:
Reply from 74.208.236.190: bytes=32 time=40ms TTL=128
Reply from 74.208.236.190: bytes=32 time=41ms TTL=128
Reply from 74.208.236.190: byte
Reply from 74.208.236.190: byte

Ping statistics for 74.208.236.
    Packets: Sent = 4, Received
Approximate round trip times in
    Minimum = 40ms, Maximum = 4

C:\Users\student>_
```

Get-WinEvent @{logname="Microsoft-Window Sysmon/Operational";id=22}| ogv
Filter
Add criteria
Clear All

and Message contains atlseccon

TimeCreated	Id	LevelDisplayName	Message
2/24/2020 8:20:34 PM	22	Information	Dns uery: Rule ame: UtcT e: 2019-10-11 19:55:43.716 Proce sGuid: {0FD50764-3010-5E54-0000-0010D9F75802} Proce Id: 1672 Query ame: atlseccon.com Quer tus: 0 QueryR sults: ::ffff:74.208.236.190;217.160.80.74;217.160.81.74;217.160.83.74;2001 Image: C:\WINDOWS\system32\PING.EXE

```
Message        : Dns query:
```

Image:
C:\WINDOWS\SYSTEM32\PING.EXE

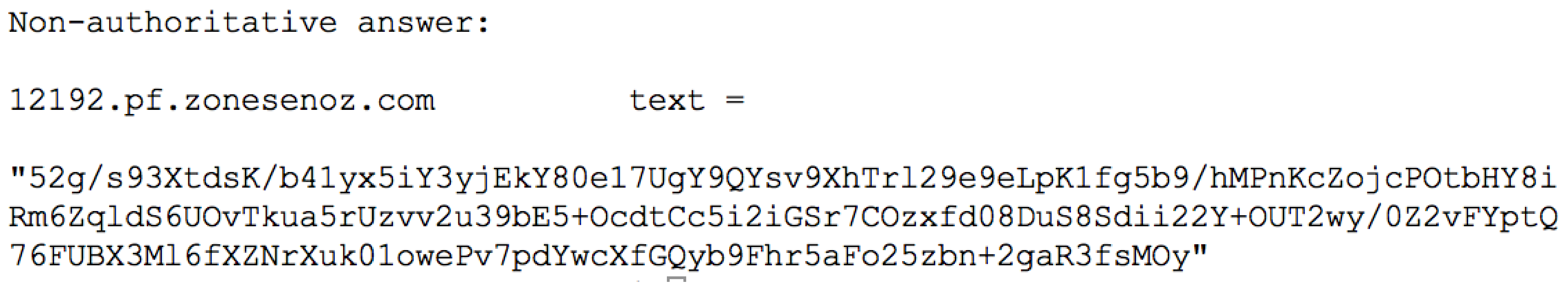
Now That We're Logging: Check Your DNS

- Malware, like most network software, uses DNS for resolving names to IP addresses (and so on)
- It also uses DNS for command and control (C2) traffic
    - It’s usually allowed outbound
    - It’s usually ignored

- The following should be monitored:
    - Requests to thousands of hosts or subdomains in one domain
    - Large DNS queries with high entropy
    - Large TXT record responses
    - Attempts to resolve NULL records
    - High volumes of DNS resolution failures
    - Requests to "baby" domains (registered very recently)

Zeus Botnet C2 via DNS

Note the large DNS TXT records used by the Zeus botnet for Command and Control (C2):



```
Non-authoritative answer:

12192.pf.zonesenoz.com          text =

"52g/s93XtdsK/b41yx5iY3yjEkY80e17UgY9QYsv9XhTrl29e9eLpK1fg5b9/hMPnKcZojcPOtbHY8i
Rm6ZqldS6UOvTkua5rUzvv2u39bE5+OcdtCc5i2iGSr7COzxfd08DuS8Sdii22Y+OUT2wy/0Z2vFYptQ
76FUBX3Ml6fXZNrXuk01owePv7pdYwcXfGQyb9Fhr5aFo25zbn+2gaR3fsMOy"
```

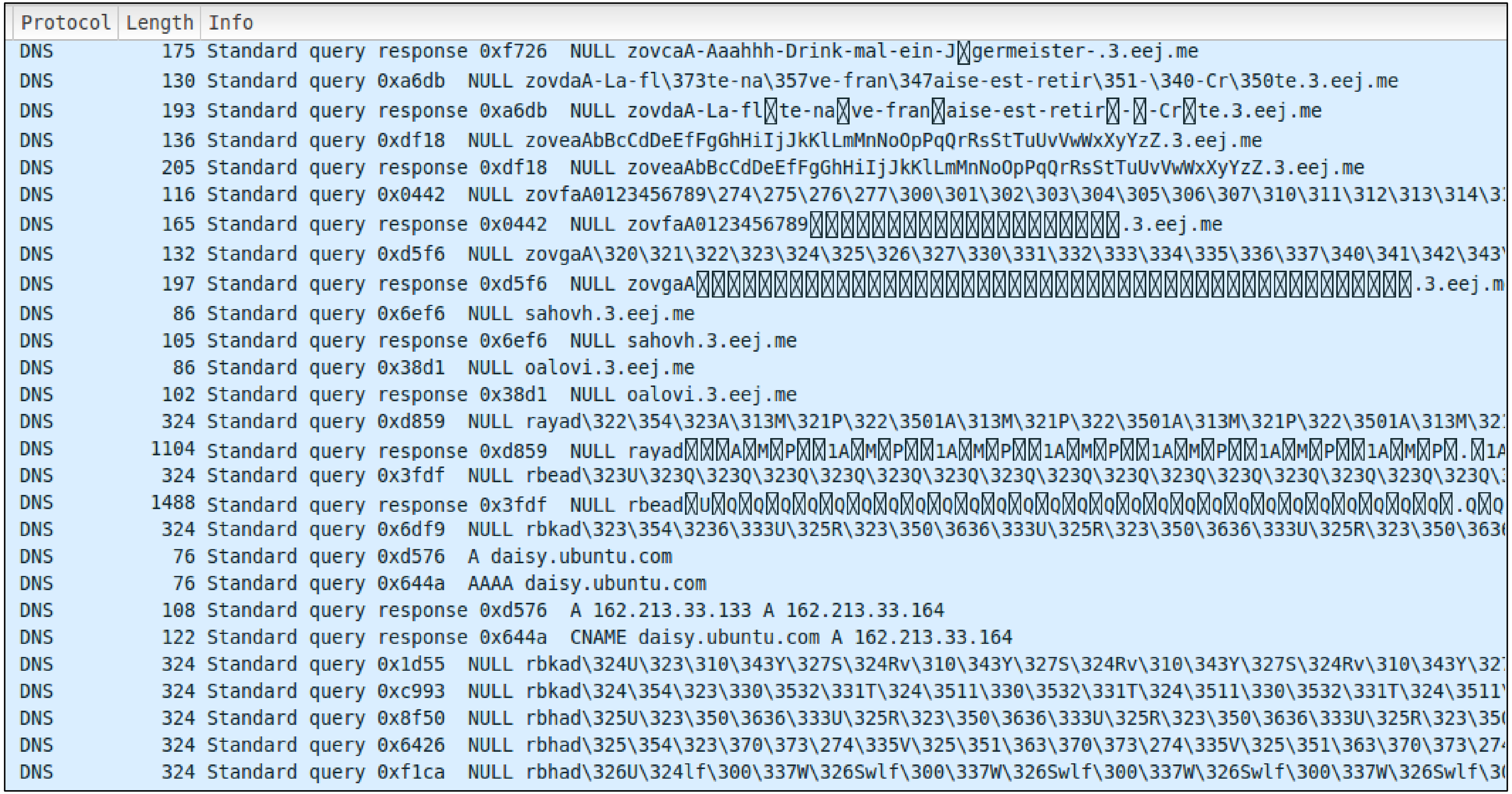
DNS: the Ideal C2 Channel

- DNS tunnels are the ideal C2 channel, IMO
  - DNS is usually allowed outbound
  - It’s usually ignored
  - Works via multiple forwarders (i.e. DNS proxies)
  - Locked down internal subnets with 'no internet access' often allow public DNS resolution
- An internal system has direct bidirectional internet access if it can resolve 'google.com' and receive the answer
- DNS tunnels are much more difficult to mitigate via preventive controls

Iodine: Advanced DNS Tunneling

- Iodine offers a true routable tunnel via DNS
  - Can tunnel any IPv4 protocol
  - Quite easy to set up, and NIDS detection is poor
- Available at: http://code.kryo.se/iodine/
- Can forward via a local DNS server, or…
  - *it may also happen that _any_ traffic is allowed to the DNS port (53 UDP) of any computer. Iodine will detect this, and switch to raw UDP tunneling if possible.* [1]
  - [1] http://code.kryo.se/iodine/README.html

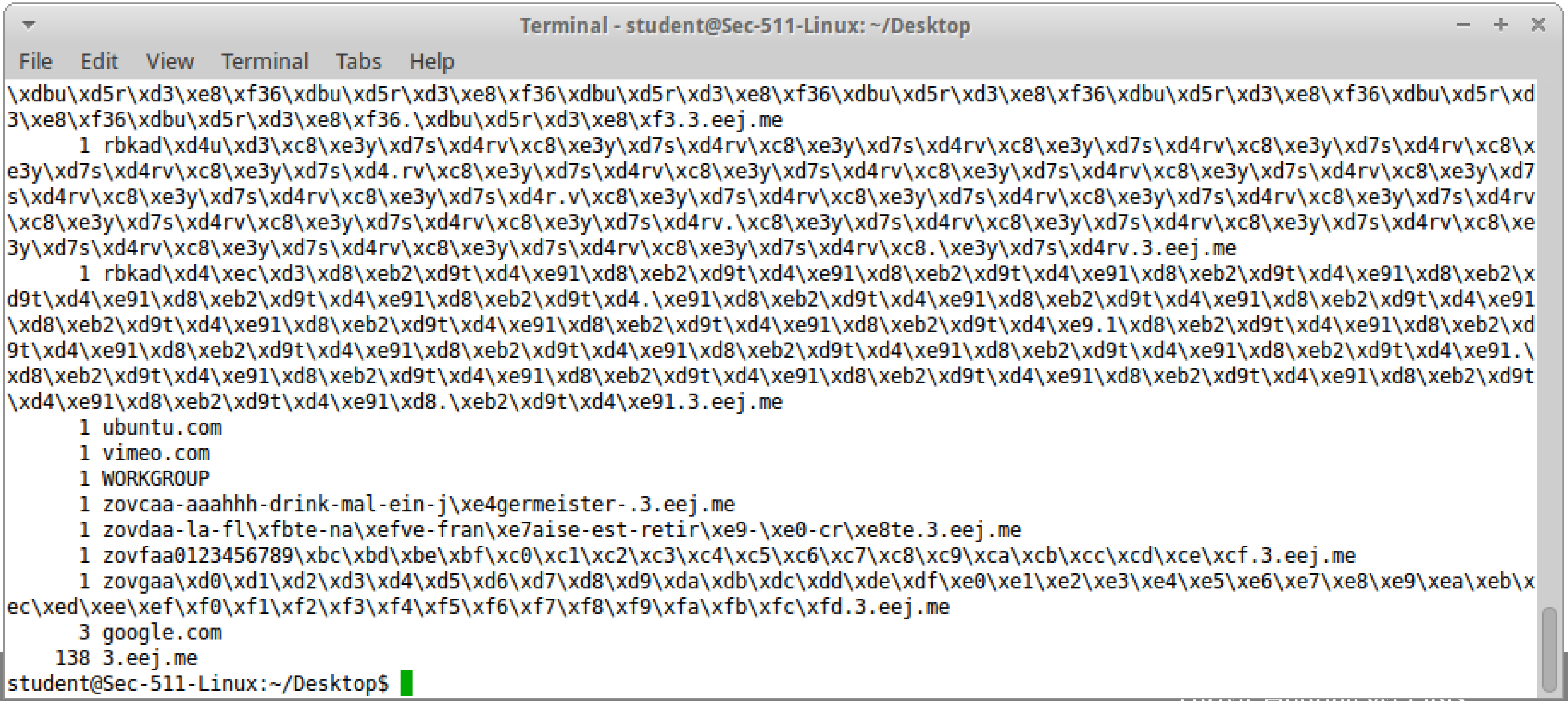
Iodine Wireshark View – DNS Tunnel (Forwarded via Local Resolver)



```
Protocol Length Info
DNS        175 Standard query response 0xf726  NULL zovcaA-Aaahhh-Drink-mal-ein-J⊠germeister-.3.eej.me
DNS        130 Standard query 0xa6db  NULL zovdaA-La-fl\373te-na\357ve-fran\347aise-est-retir\351-\340-Cr\350te.3.eej.me
DNS        193 Standard query response 0xa6db  NULL zovdaA-La-fl⊠te-na⊠ve-fran⊠aise-est-retir⊠-⊠-Cr⊠te.3.eej.me
DNS        136 Standard query 0xdf18  NULL zoveaAbBcCdDeEfFgGhHiIjJkKlLmMnNoOpPqQrRsStTuUvVwWxXyYzZ.3.eej.me
DNS        205 Standard query response 0xdf18  NULL zoveaAbBcCdDeEfFgGhHiIjJkKlLmMnNoOpPqQrRsStTuUvVwWxXyYzZ.3.eej.me
DNS        116 Standard query 0x0442  NULL zovfaA0123456789\274\275\276\277\300\301\302\303\304\305\306\307\310\311\312\313\314\3
DNS        165 Standard query response 0x0442  NULL zovfaA0123456789⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠.3.eej.me
DNS        132 Standard query 0xd5f6  NULL zovgaA\320\321\322\323\324\325\326\327\330\331\332\333\334\335\336\337\340\341\342\343
DNS        197 Standard query response 0xd5f6  NULL zovgaA⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠.3.eej.m
DNS         86 Standard query 0x6ef6  NULL sahovh.3.eej.me
DNS        105 Standard query response 0x6ef6  NULL sahovh.3.eej.me
DNS         86 Standard query 0x38d1  NULL oalovi.3.eej.me
DNS        102 Standard query response 0x38d1  NULL oalovi.3.eej.me
DNS        324 Standard query 0xd859  NULL rayad\322\354\323A\313M\321P\322\3501A\313M\321P\322\3501A\313M\321P\322\3501A\313M\32
DNS       1104 Standard query response 0xd859  NULL rayad⊠⊠⊠A⊠M⊠P⊠⊠1A⊠M⊠P⊠⊠1A⊠M⊠P⊠⊠1A⊠M⊠P⊠⊠1A⊠M⊠P⊠⊠1A⊠M⊠P⊠⊠1A⊠M⊠P⊠.⊠1A
DNS        324 Standard query 0x3fdf  NULL rbead\323U\323Q\323Q\323Q\323Q\323Q\323Q\323Q\323Q\323Q\323Q\323Q\323Q\323Q\323Q\323Q\3
DNS       1488 Standard query response 0x3fdf  NULL rbead⊠U⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠Q⊠.Q⊠Q
DNS        324 Standard query 0x6df9  NULL rbkad\323\354\3236\333U\325R\323\350\3636\333U\325R\323\350\3636\333U\325R\323\350\363
DNS         76 Standard query 0xd576  A daisy.ubuntu.com
DNS         76 Standard query 0x644a  AAAA daisy.ubuntu.com
DNS        108 Standard query response 0xd576  A 162.213.33.133 A 162.213.33.164
DNS        122 Standard query response 0x644a  CNAME daisy.ubuntu.com A 162.213.33.164
DNS        324 Standard query 0x1d55  NULL rbkad\324U\323\310\343Y\327S\324Rv\310\343Y\327S\324Rv\310\343Y\327S\324Rv\310\343Y\32
DNS        324 Standard query 0xc993  NULL rbkad\324\354\323\330\3532\331T\324\3511\330\3532\331T\324\3511\330\3532\331T\324\3511
DNS        324 Standard query 0x8f50  NULL rbhad\325U\323\350\3636\333U\325R\323\350\3636\333U\325R\323\350\3636\333U\325R\323\35
DNS        324 Standard query 0x6426  NULL rbhad\325\354\323\370\373\274\335V\325\351\363\370\373\274\335V\325\351\363\370\373\27
DNS        324 Standard query 0xf1ca  NULL rbhad\326U\324lf\300\337W\326Swlf\300\337W\326Swlf\300\337W\326Swlf\300\337W\326Swlf\3
```

Zeek View – sort, sed, etc…

```
$ cat dns.log |zeek-cut query | sort -u | sed
"s/^[a-zA-Z0-9-]*\.//g"| sort | uniq -c | sort -n
```


Terminal - student@Sec-511-Linux: ~/Desktop
File Edit View Terminal Tabs Help

```
\xdbu\xd5r\xd3\xe8\xf36\xdbu\xd5r\xd3\xe8\xf36\xdbu\xd5r\xd3\xe8\xf36\xdbu\xd5r\xd3\xe8\xf36\xdbu\xd5r\xd3\xe8\xf36\xdbu\xd5r\xd
3\xe8\xf36\xdbu\xd5r\xd3\xe8\xf36.\xdbu\xd5r\xd3\xe8\xf3.3.eej.me
      1 rbkad\xd4u\xd3\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\x
e3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4.rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7
s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4r.v\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv
\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv.\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe
3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8\xe3y\xd7s\xd4rv\xc8.\xe3y\xd7s\xd4rv.3.eej.me
      1 rbkad\xd4\xec\xd3\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\x
d9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4.\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91
\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe9.1\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd
9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91.\
xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8\xeb2\xd9t
\xd4\xe91\xd8\xeb2\xd9t\xd4\xe91\xd8.\xeb2\xd9t\xd4\xe91.3.eej.me
      1 ubuntu.com
      1 vimeo.com
      1 WORKGROUP
      1 zovcaa-aaahhh-drink-mal-ein-j\xe4germeister-.3.eej.me
      1 zovdaa-la-fl\xfbte-na\xefve-fran\xe7aise-est-retir\xe9-\xe0-cr\xe8te.3.eej.me
      1 zovfaa0123456789\xbc\xbd\xbe\xbf\xc0\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf.3.eej.me
      1 zovgaa\xd0\xd1\xd2\xd3\xd4\xd5\xd6\xd7\xd8\xd9\xda\xdb\xdc\xdd\xde\xdf\xe0\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\x
ec\xed\xee\xef\xf0\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\xfc\xfd.3.eej.me
      3 google.com
    138 3.eej.me
student@Sec-511-Linux:~/Desktop$
```

Programmatic Entropy Analysis

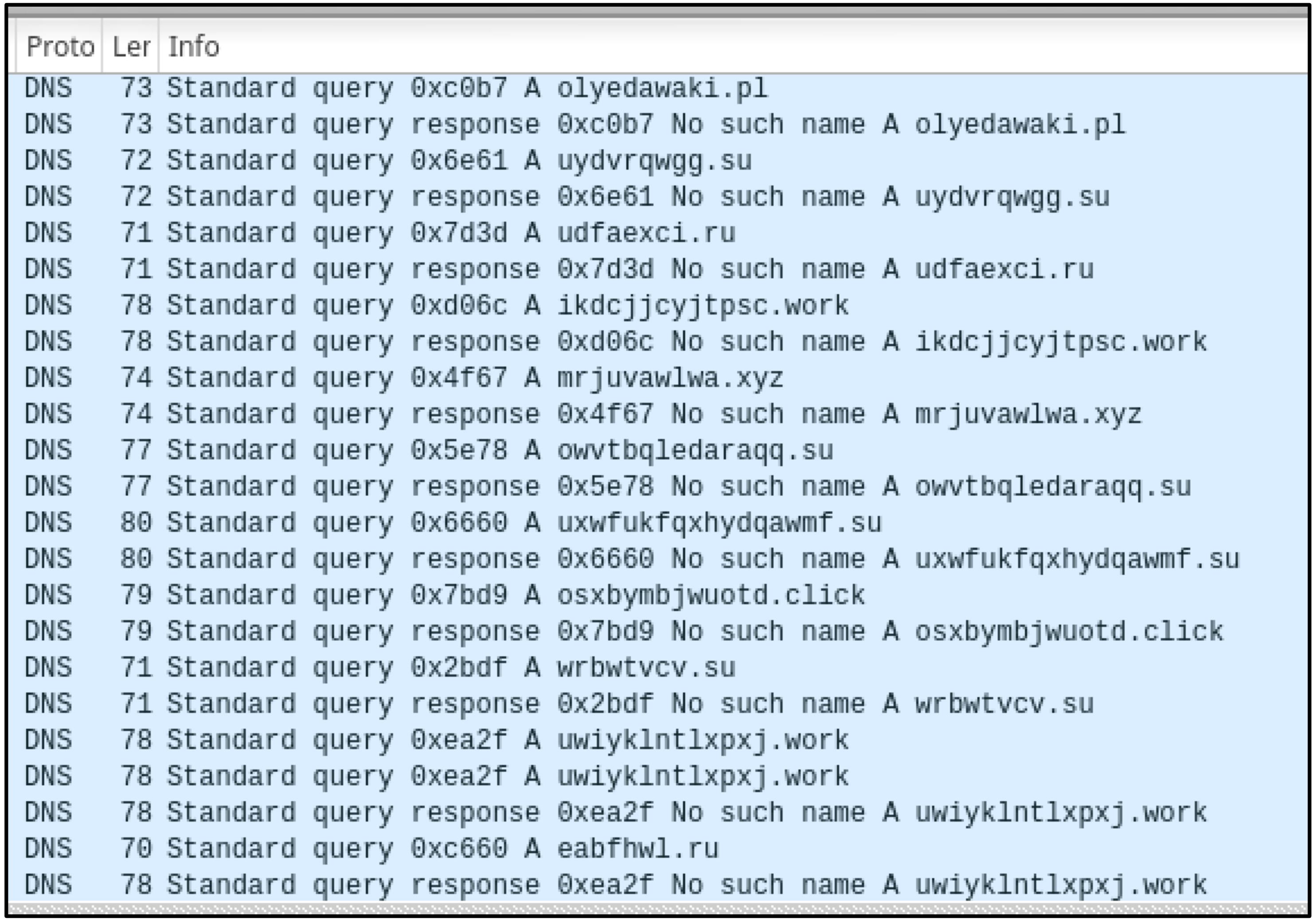
- Without trying, the human brain often can detect something as potentially random generated
  - Programmatically achieving this proves more difficult than expected
- Many tools exist for calculating entropy, the often built-in Linux tool, **`ent`** being a simple example
- Classic entropy analysis using tools like **`ent`** can be leveraged to determine the degree of randomness of provided input…
  - …but ASCII has 256 characters
  - A DNS name containing letters (26 characters) and numbers (10 characters) uses a maximum of 36 of 256 total ASCII values (14%)
  - Any cryptologist will tell you: that equals low entropy

Bring Out the Baggett

- Solving problems like detecting random (before morning break) is why you always have @**MarkBaggett** (GSE #15) take your classes
  - freq.py tool is a huge boon to finding random generated strings where they perhaps shouldn't be
  - https://github.com/sans-blue-team/freq.py
- The approach looks at the likelihood of character occurrence based on frequency analysis
  - Simple example: in English text, "**`q`**" is pretty much followed by a "**`u,`**" so seeing a "q" followed by sc would be rather unlikely to occur

Domain Generation Algorithms DGAs

- One of the most obvious, and incredibly useful, ways to employ **`freq.py`** is looking at DNS names for signs of randomness
- You will necessarily need to do whitelisting
  - Public CDNs (Content Delivery Networks)
  - Major cloud services (Microsoft, Amazon, Google) often have their own CDN



```
Proto  Ler  Info
DNS     73 Standard query 0xc0b7 A olyedawaki.pl
DNS     73 Standard query response 0xc0b7 No such name A olyedawaki.pl
DNS     72 Standard query 0x6e61 A uydvrqwgg.su
DNS     72 Standard query response 0x6e61 No such name A uydvrqwgg.su
DNS     71 Standard query 0x7d3d A udfaexci.ru
DNS     71 Standard query response 0x7d3d No such name A udfaexci.ru
DNS     78 Standard query 0xd06c A ikdcjjcyjtpsc.work
DNS     78 Standard query response 0xd06c No such name A ikdcjjcyjtpsc.work
DNS     74 Standard query 0x4f67 A mrjuvawlwa.xyz
DNS     74 Standard query response 0x4f67 No such name A mrjuvawlwa.xyz
DNS     77 Standard query 0x5e78 A owvtbqledaraqq.su
DNS     77 Standard query response 0x5e78 No such name A owvtbqledaraqq.su
DNS     80 Standard query 0x6660 A uxwfukfqxhydqawmf.su
DNS     80 Standard query response 0x6660 No such name A uxwfukfqxhydqawmf.su
DNS     79 Standard query 0x7bd9 A osxbymbjwuotd.click
DNS     79 Standard query response 0x7bd9 No such name A osxbymbjwuotd.click
DNS     71 Standard query 0x2bdf A wrbwtvcv.su
DNS     71 Standard query response 0x2bdf No such name A wrbwtvcv.su
DNS     78 Standard query 0xea2f A uwiyklntlxpxj.work
DNS     78 Standard query 0xea2f A uwiyklntlxpxj.work
DNS     78 Standard query response 0xea2f No such name A uwiyklntlxpxj.work
DNS     70 Standard query 0xc660 A eabfhwl.ru
DNS     78 Standard query response 0xea2f No such name A uwiyklntlxpxj.work
```

DGA++ - Beyond Domain Generation Algorithms

Though DGA detection can be very effective, think more broadly about places where adversaries might programmatically generate large volumes

Detecting randomness can be a tremendous indicator of otherwise unknown malice

- Thread/Process names
- File names (binaries, scripts, etc.)
- Workstation names
- Service names
- Subdomains (Domain Shadowing[1])
- Certificate subject names and issuers
- Usernames
- Many additional possibilities

`freq_server.py` - freq-ing At Scale

As additional use cases are discovered, you will soon feel the need to wield **`freq.py`** at scale

Although the initial script is, without question, a work of art, it was not intended to have a system perform 100,000+ **`freq.py/sec`**

Have no fear, **`@MarkBaggett`** worked with SANS SIEM course author and 511 instructor **Justin Henderson** (**`@SecurityMapper, GSE #108, SANS SIEM Author`**) and developed a new feature/deployment model

- **`freq_server.py – https://github.org/sans-blue-team/freq.py/`**
- **`freq_server.py`** designed to allow for remote calls from tools such as LogStash
- Implementation and analysis techniques discussed in SANS SIEM class

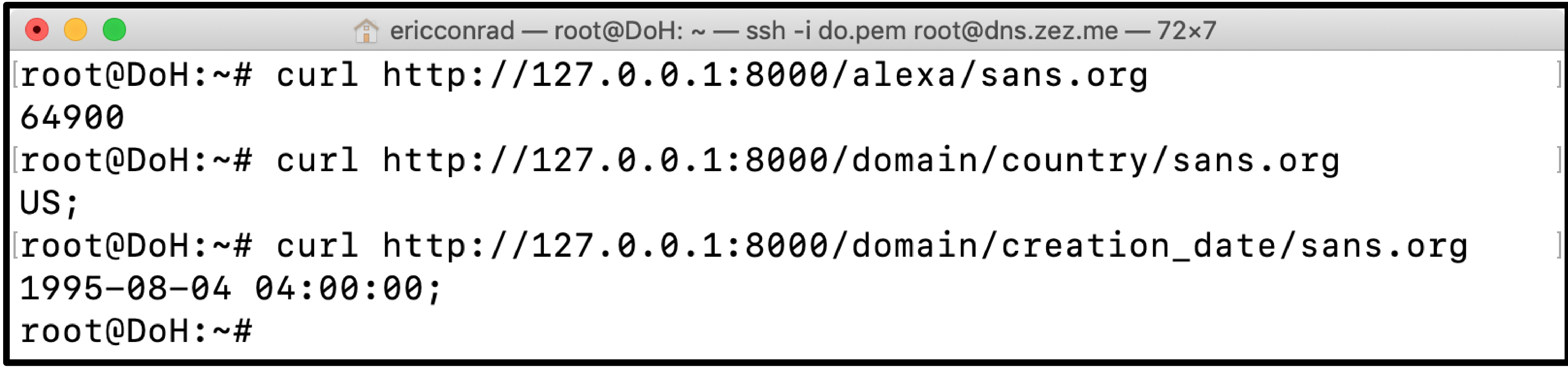
dnstwist

- Use **dnstwist** to protect against cousin domains (`sec530.com` vs. `sec53O.com`) and Internationalized Domain Name (IDN) homoglyph attacks
  - pɑypɑl.com vs. paypal.com
  - Block with firewall/proxy, or detect via DNS and other sources
  - dnstwist calculates permutations against a given domain
  - Also checks to see if any domains have been registered
  - And provides additional information about the domain
- Use dnstwist with scripting to handle evil cousins and homographs

Baby Domain Detection: domain_stats

- Domain_stats is another great tool by Mark Baggett
  - https://github.com/MarkBaggett/domain_stats
- Can query the Alexa or Cisco Umbrella top million
- Can also query RDAP data to discover domain creation time (to discover newly-registered "baby domains")
  - And much more
- RDAP (Registration Data Access Protocol) is the (eventual) replacement for WHOIS
  - WHOIS: blobs of inconsistent and poorly-formatted data
  - RDAP: can output in **JSON**

domain_stats in action


ericconrad — root@DoH: ~ — ssh -i do.pem root@dns.zez.me — 72×7

```
root@DoH:~# curl http://127.0.0.1:8000/alexa/sans.org
64900
root@DoH:~# curl http://127.0.0.1:8000/domain/country/sans.org
US;
root@DoH:~# curl http://127.0.0.1:8000/domain/creation_date/sans.org
1995-08-04 04:00:00;
root@DoH:~#
```

Thank you! – econrad@backshore.net

- Thank you for attending my talk!
- A copy of these slides are available on https://ericconrad.com