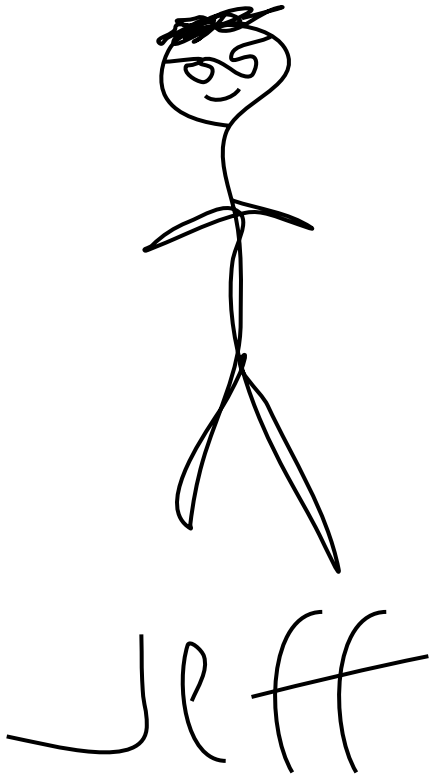
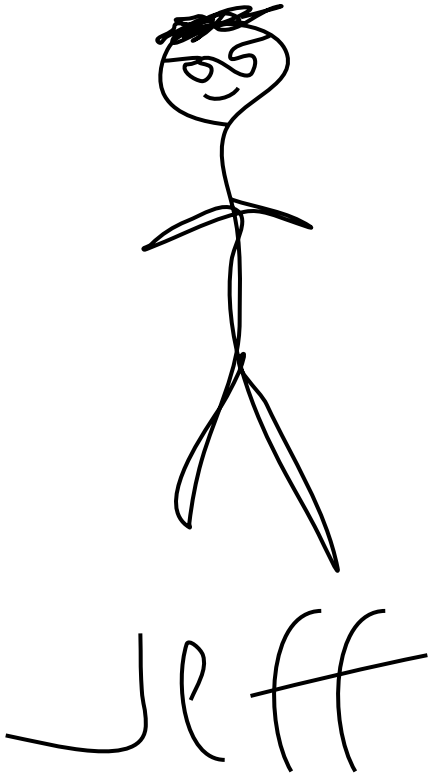
Enhance Encrypted Network Telemetry



`github.com/salesforce/ja3`

bro-pkg install ja3

$whoami



[redacted] security researcher [redacted]

[redacted]

$dayjob

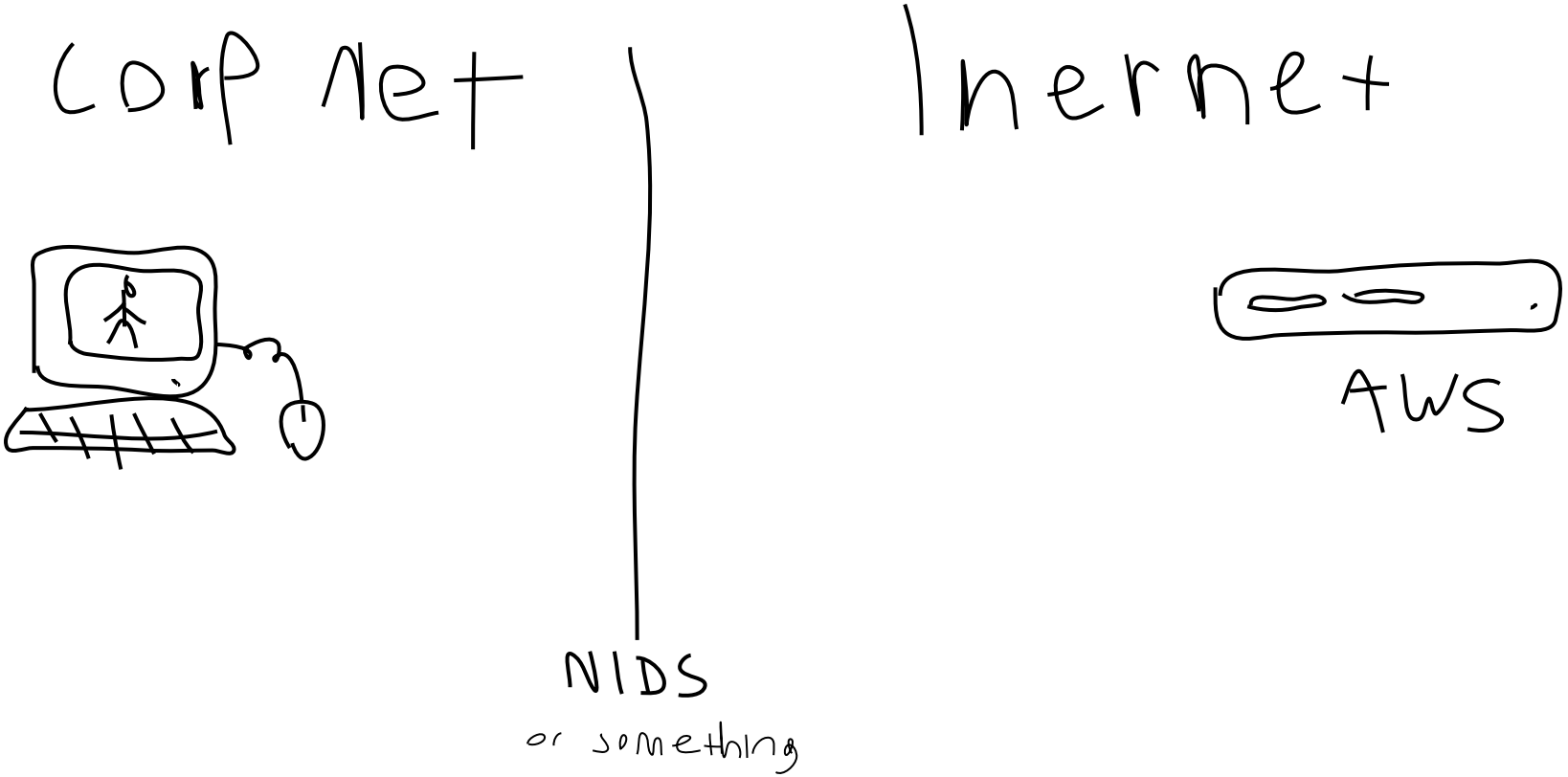
[redacted]

[redacted] his favorite tool, Bro. [redacted] creators of the JA3 fingerprint technique.

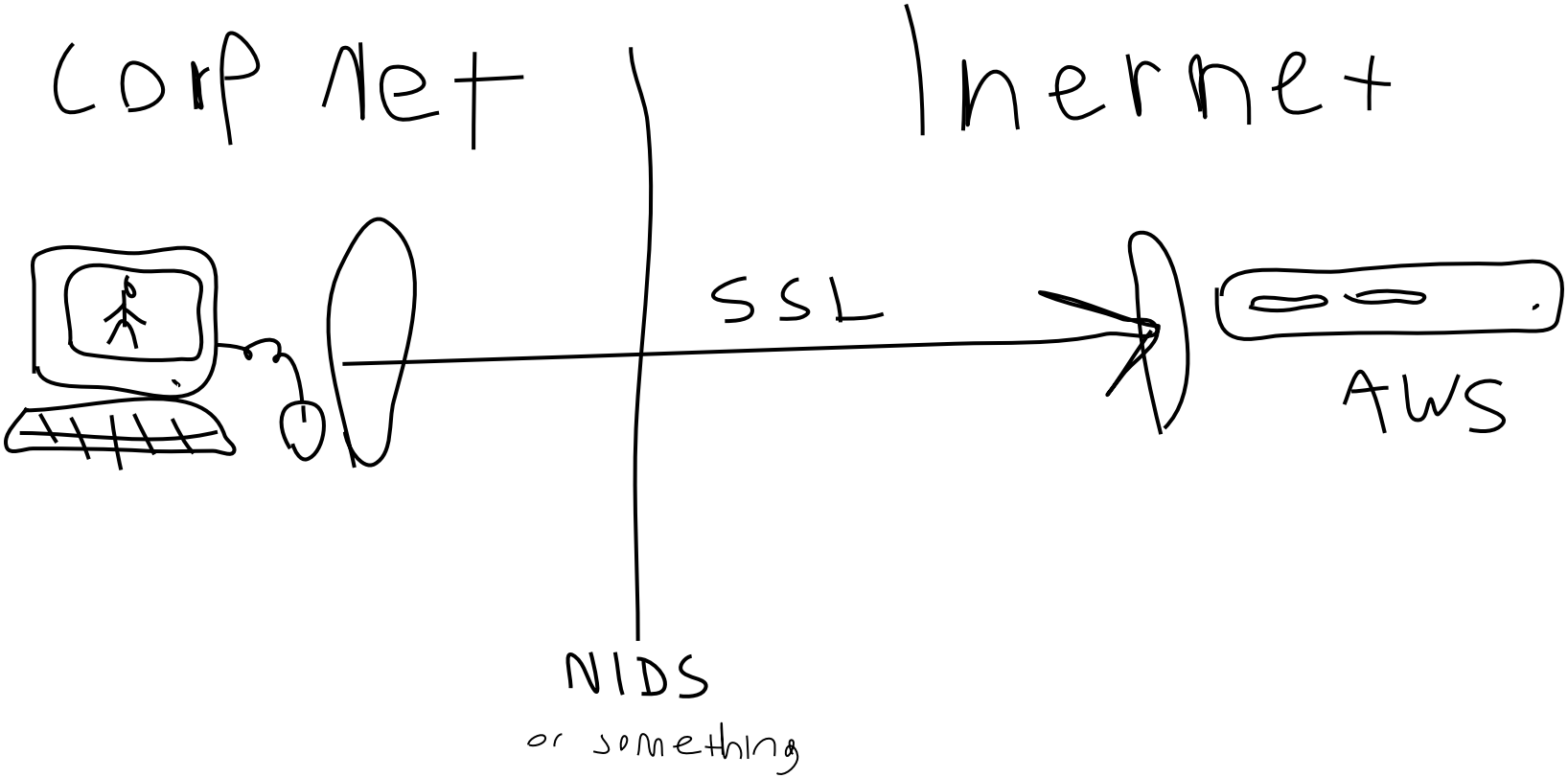
First Version I used:
Bro-0.7

How SSL works

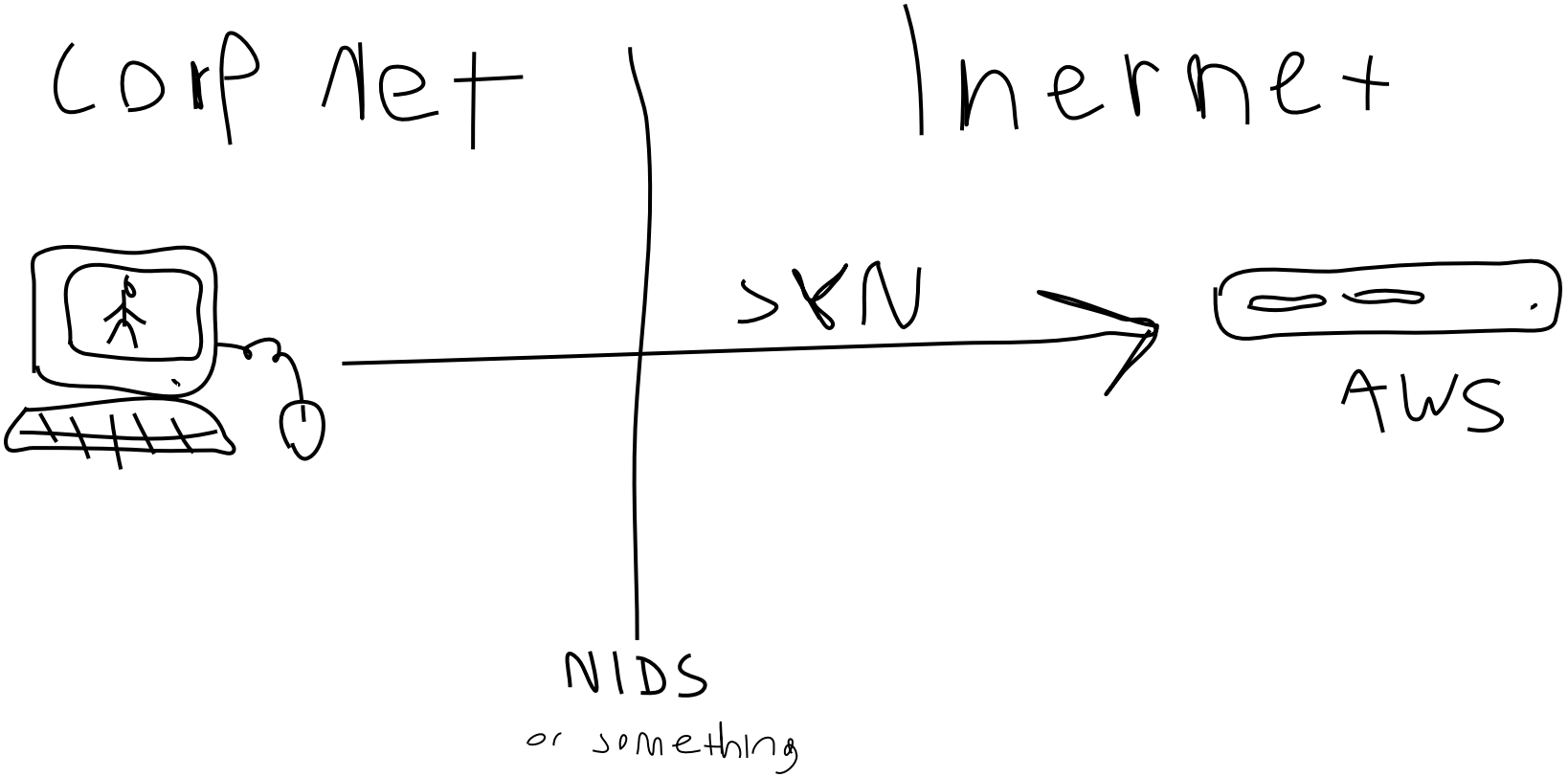
How SSL works



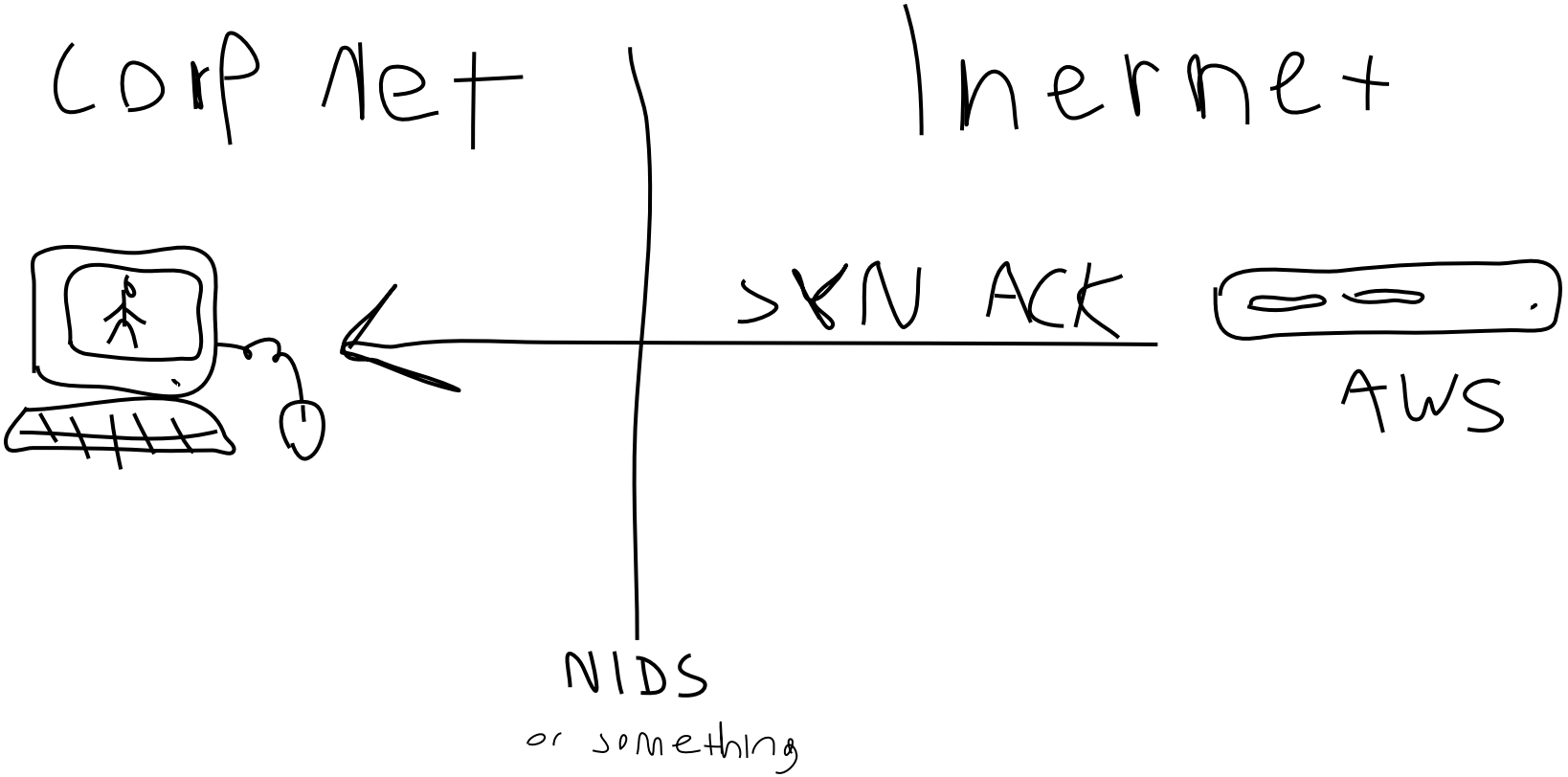
How SSL works



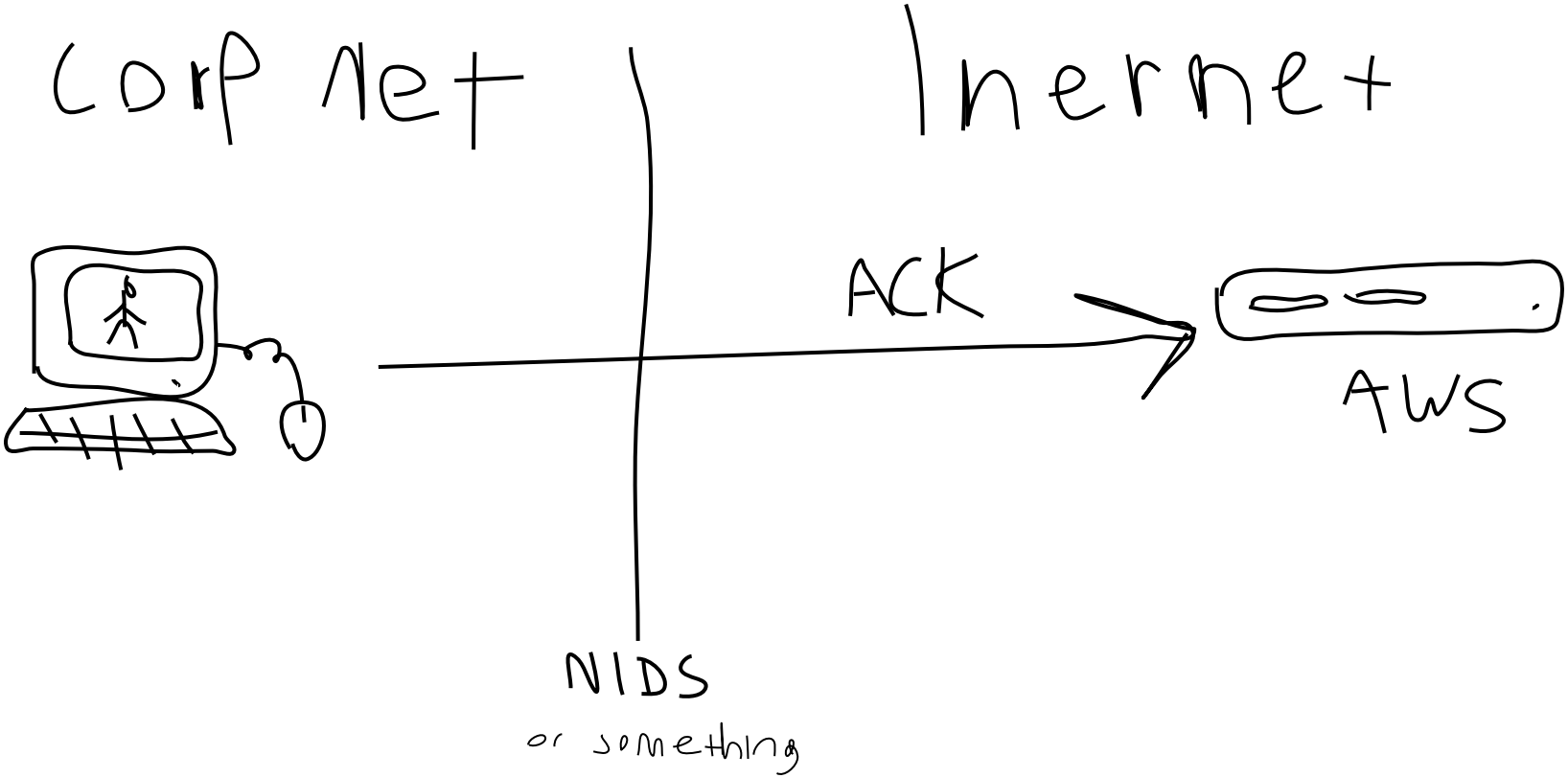
How SSL works



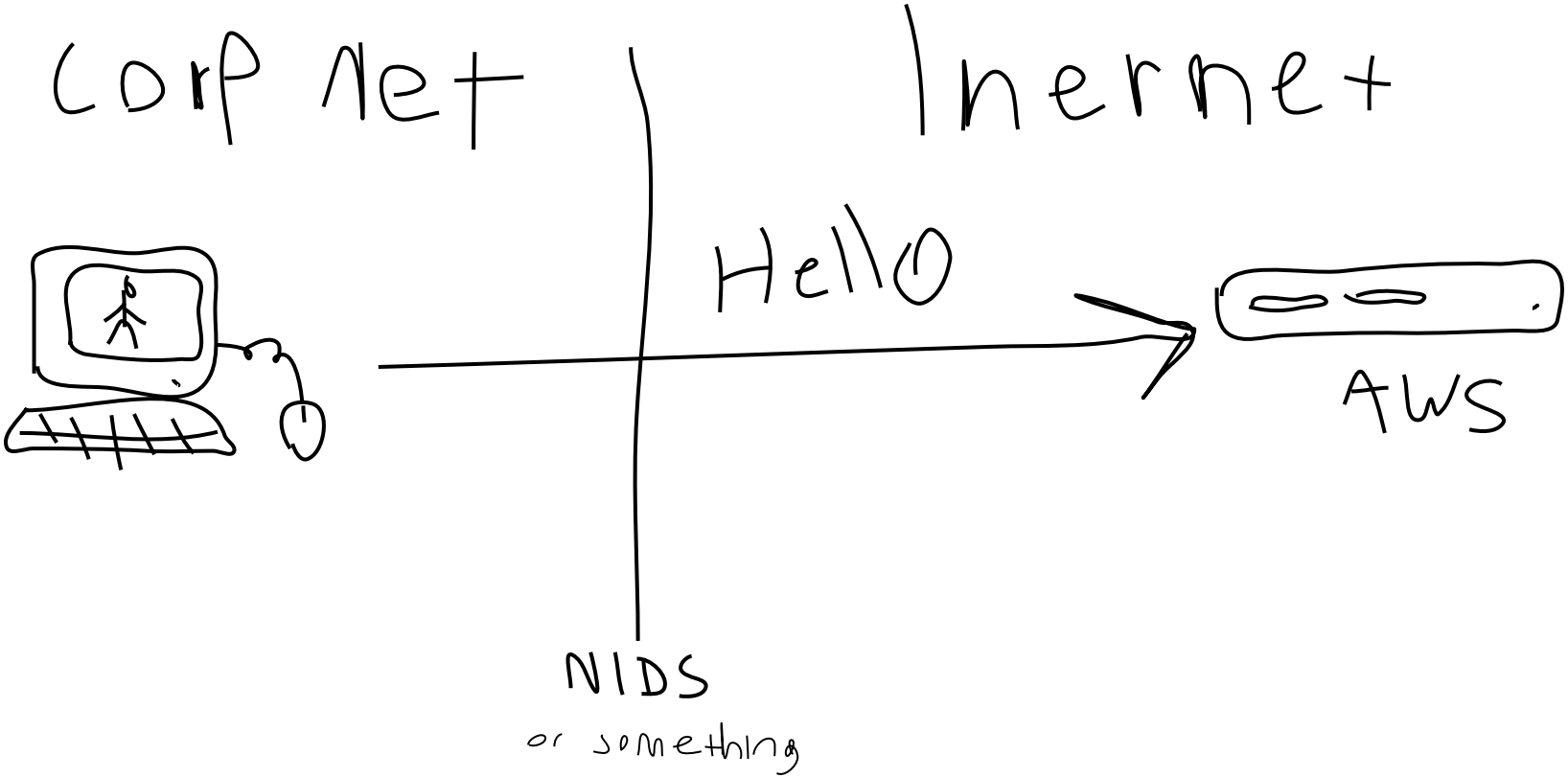
How SSL works



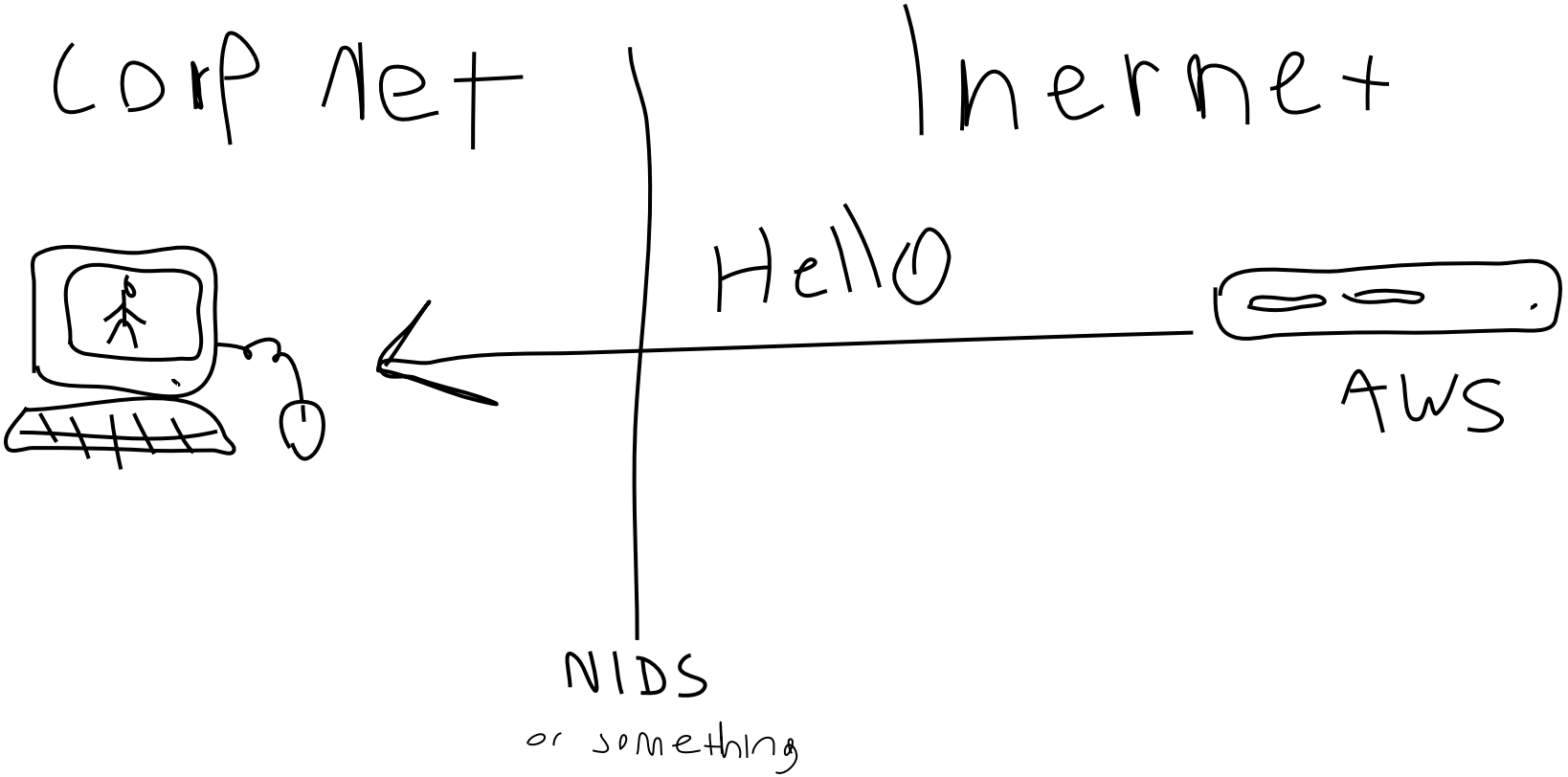
How SSL works



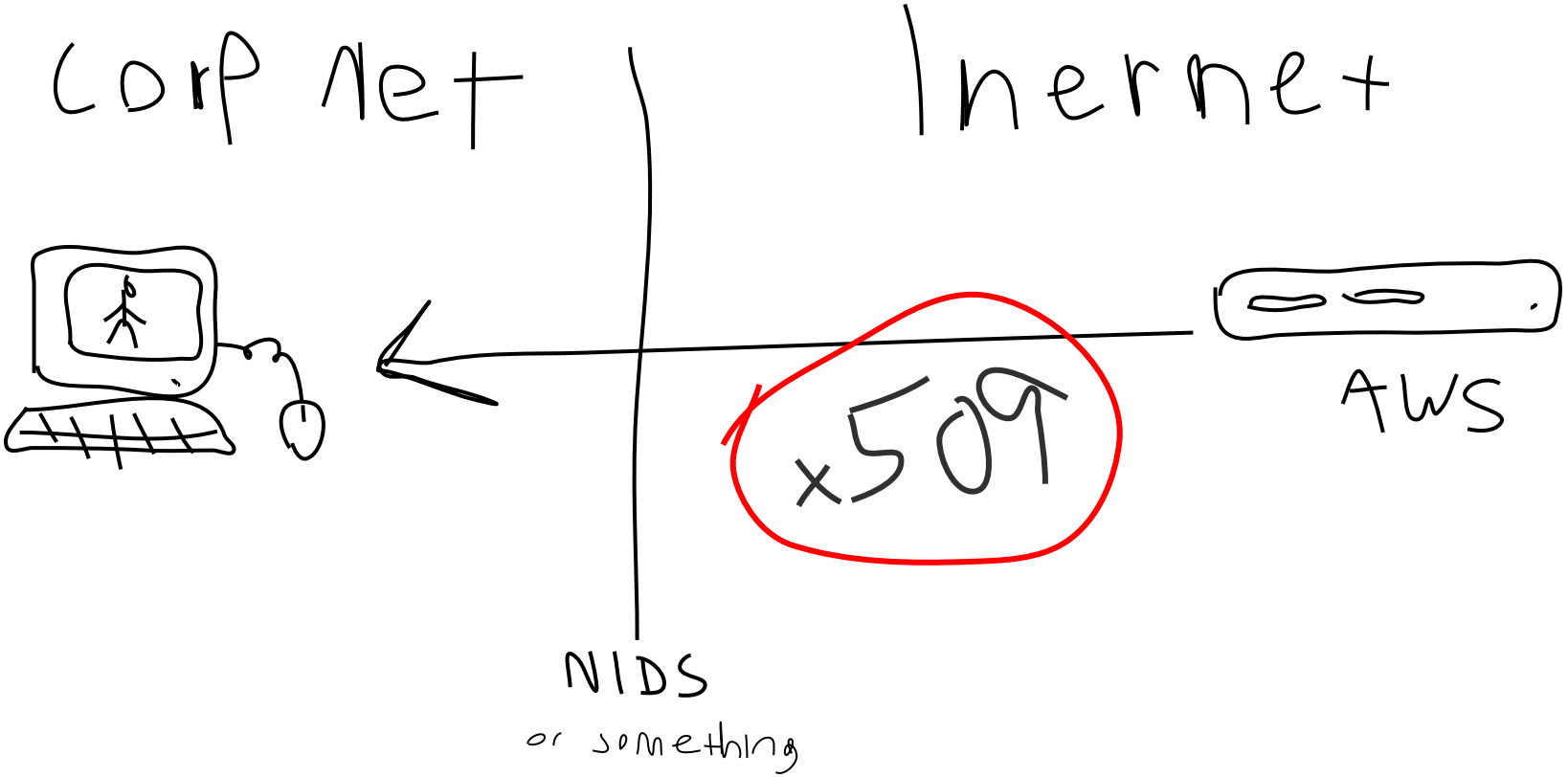
How SSL works

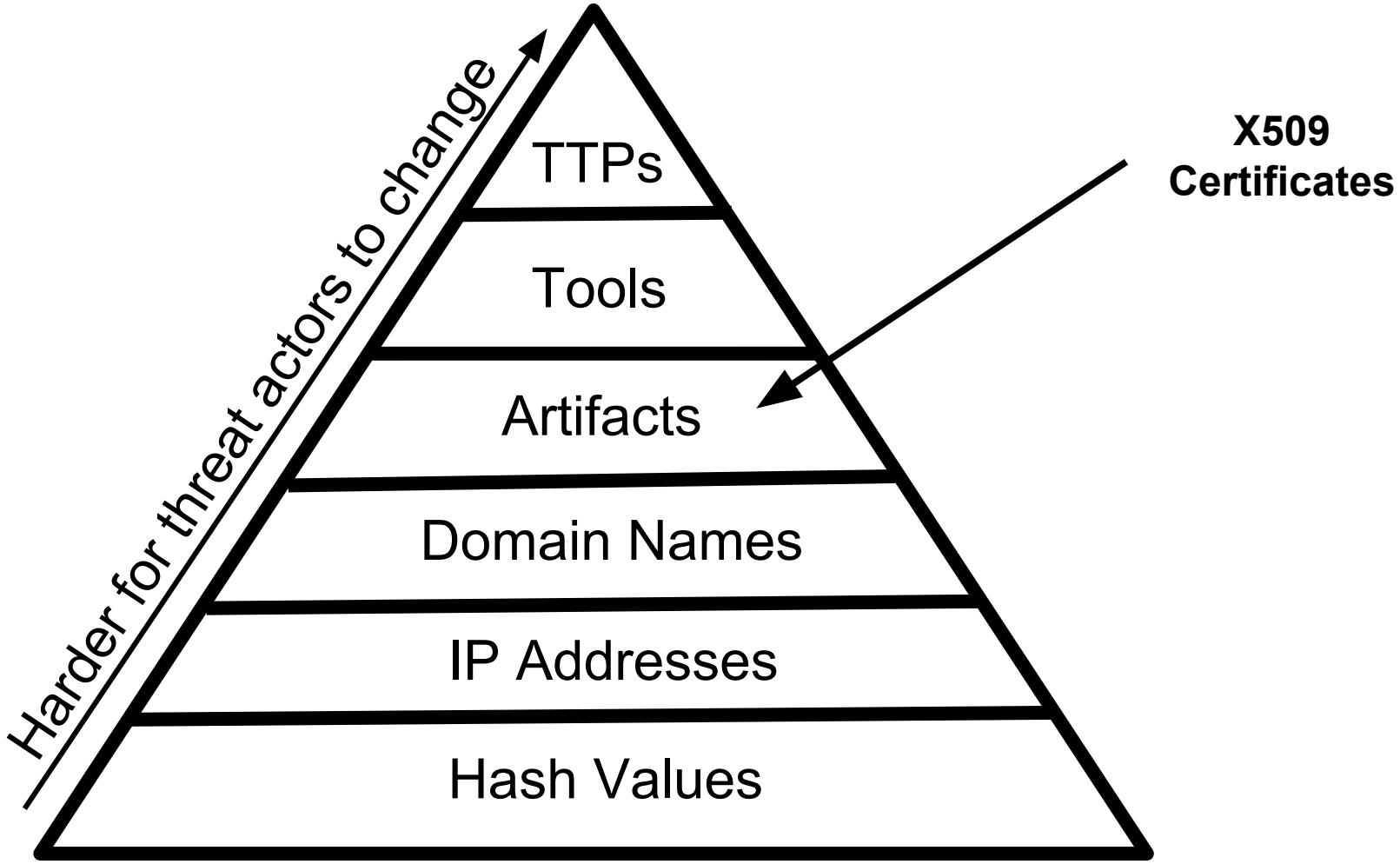


How SSL works



How SSL works

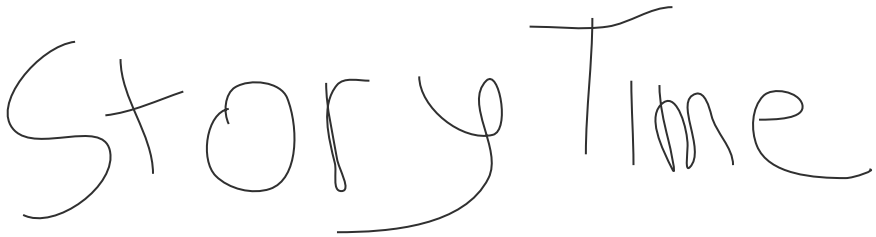



Harder for threat actors to change
TTPs
Tools
Artifacts
Domain Names
IP Addresses
Hash Values
X509
Certificates

event x509_certificate(f: fa_file, cert_ref: opaque of x509, cert: X509::Certificate)

```
type Certificate: record {
        version: count &log;    ##< Version number.
        serial: string &log;    ##< Serial number.
        subject: string &log;   ##< Subject.
        issuer: string &log;    ##< Issuer.
        cn: string &optional; ##< Last (most specific) common name.
        not_valid_before: time &log;    ##< Timestamp before when certificate is not valid.
        not_valid_after: time &log;     ##< Timestamp after when certificate is not valid.
        key_alg: string &log;   ##< Name of the key algorithm
        sig_alg: string &log;   ##< Name of the signature algorithm
        key_type: string &optional &log;        ##< Key type, if key parseable by openssl (either rsa, dsa or ec)
        key_length: count &optional &log;       ##< Key length in bits
        exponent: string &optional &log;        ##< Exponent, if RSA-certificate
        curve: string &optional &log;   ##< Curve, if EC-certificate
};
```

Metasploit SSL X509



Default Metasploit SSL Cert in Bro

x509.log

id	certificate.version	certificate.serial	certificate.subject	certificate.issuer	certificate.not_valid_before	certificate.not_valid_after	certificate.key_alg	certificat
FkBRWI3	3	2ABA7B7F	CN=vl3qykkr.com,O=UPdkxNE	CN=hrzvox.gov,O=bdlOFqMXlUf	1406814828	1409442828	rsaEncryption	sha1With

certificate.issuer:

CN=hrzvox.gov,O=bdlOFqMXlUfgoNQljMuRWgiJ,L=ZTIhjQVsJEuQIlS
gScdegcLSLJVRE,ST=WI,C=US

certificate.subject:

CN=vl3qykkr.com,O=UPdkxNEasODSAlkvuadEMm,L=SZewokfDFSkaAsf
KyeJMNtfleGT,ST=NV,C=US

/usr/share/metasploit-framework/lib/rex/socket/ssl_tcp_server.rb

```
def makessl(params)
  ssl_cert = params.ssl_cert
  if ssl_cert
    issuer = OpenSSL::X509::Name.new([
        ["C","US"],
        ['ST', Rex::Text.rand_state()],
        ["L", Rex::Text.rand_text_alpha(rand(20) + 10)],
        ["O", Rex::Text.rand_text_alpha(rand(20) + 10)],
        ["CN", Rex::Text.rand_hostname],
      ])
```

Default Metasploit SSL Cert in Bro

x509.log

id	certificate.version	certificate.serial	certificate.subject	certificate.issuer	certificate.not_valid_before	certificate.not_valid_after	certificate.key_alg	certificat
FkBRWI3	3	2ABA7B7F	CN=vl3qykkr.com,O=UPdlxNE	CN=hrzvox.gov,O=bdlOFqMXlUf	1406814828	1409442828	rsaEncryption	sha1With

certificate.issuer:

CN=hrzvox.gov,

O=bdlOFqMXlUfgoNQljMuRWgiJ,

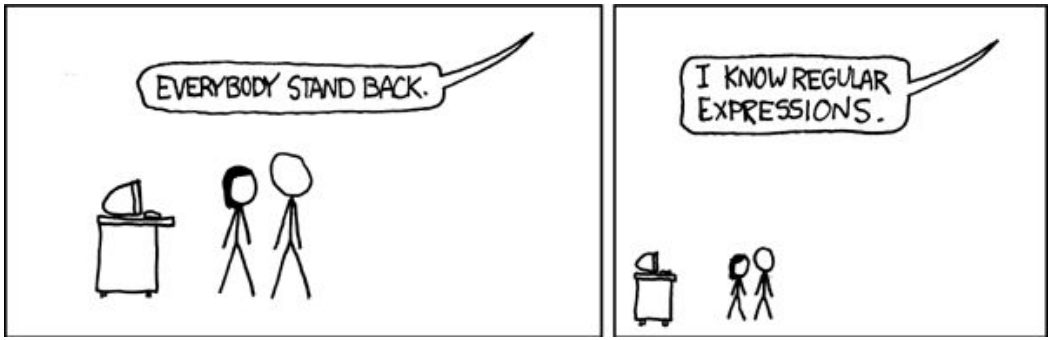
L=ZTIhjQVsJEuQIlSgScdegcLSLJVRE,

ST=WI,

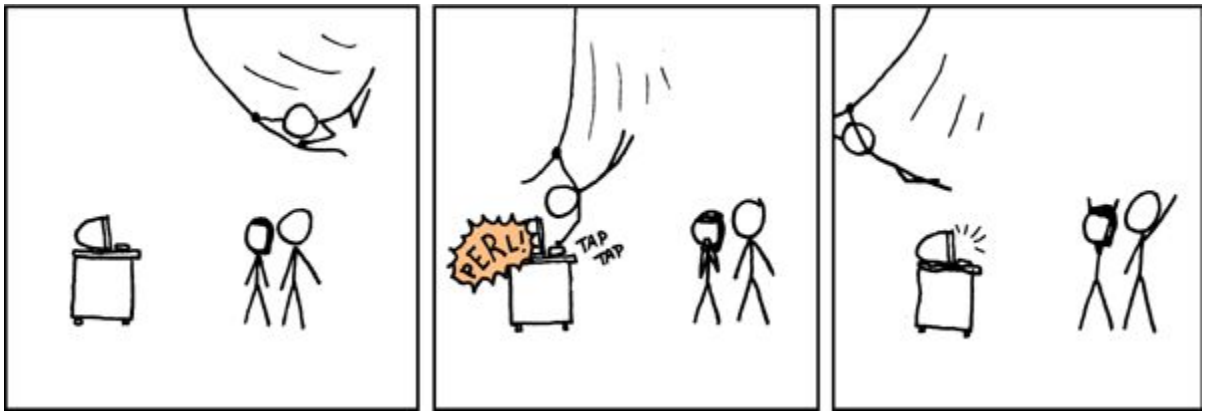
C=US

Regex match on rand mixed alpha?

```
bdlOFqMXlUfgoNQljMuRWgiJ
ZTIhjQVsJEuQIlSgScdegcLSLJVRE
alDSFlkasfQWAFlksSA
aAfkVCIQmdSDlEkfASgKJZEk
KfaNmtFxGPtqeK
jQVsJEuQIlSgoNQljMuR
CIQmddlOFqMXlUlDSFSgQljM
SgoNQljasfOFqMXl
KfIKwlMCZoetFFaLKXZ
```



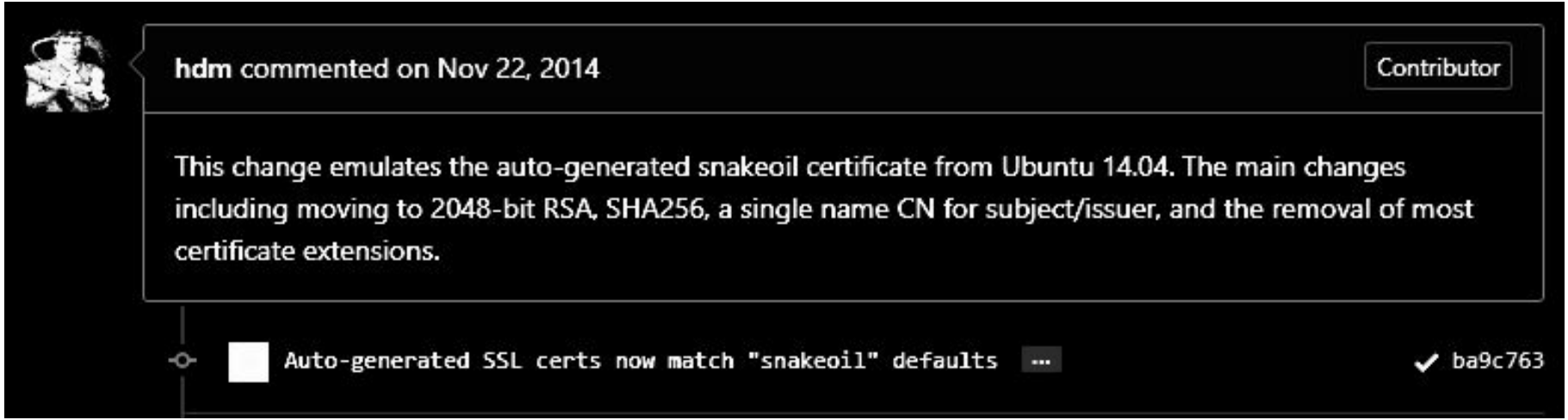
[a-z][A-Z]{2}



```
  if ( !(cert?$issuer) || (/C=US/ !in cert$issuer) )
        return;
local conn: connection;
for ( c in f$conns )
      conn=f$conns[c];
local metasploit = /[a-z][A-Z]{2}/;
  local x509_data: table[string] of string = table();
  local parts = split(cert$issuer, /,/);
  for ( part_index in parts )
          {
          local key_val = split1(parts[part_index], /=/);
          if ( 2 in key_val)
                  x509_data[key_val[1]] = key_val[2];
          }
  if ( "C" in x509_data && x509_data["C"] == "US" && "L" in x509_data && metasploit in x509_data["L"] )
          NOTICE([$note=Metasploit_SSL_Cert, $conn=conn,
                  $msg=fmt("Metasploit SSL, random issuer US city '%s'", x509_data["L"]),
                  $sub=cert$issuer,
                  $identifier=cert$issuer]);
```

Metasploit SSL Round 2

Metasploit SSL Cert Round 2



Metasploit SSL Cert Round 2

```
def self.ssl_generate_certificate
    yr   = 24*3600*365
    vf   = Time.at(Time.now.to_i - rand(yr * 3) - yr)
    vt   = Time.at(vf.to_i + (10 * yr))
    cn   = Rex::Text.rand_text_alpha_lower(rand(8)+2)
    key  = OpenSSL::PKey::RSA.new(2048){ }
    cert = OpenSSL::X509::Certificate.new
    cert.version    = 2
    cert.serial     = (rand(0xFFFFFFFF) << 32) + rand(0xFFFFFFFF)
    cert.subject    = OpenSSL::X509::Name.new([["CN", cn]])
    cert.issuer     = OpenSSL::X509::Name.new([["CN", cn]])
    cert.not_before = vf
    cert.not_after  = vt
    cert.public_key = key.public_key

    ef = OpenSSL::X509::ExtensionFactory.new(nil,cert)
    cert.extensions = [
      ef.create_extension("basicConstraints","CA:FALSE")
    ]
    ef.issuer_certificate = cert

    cert.sign(key, OpenSSL::Digest::SHA256.new)
```

Metasploit SSL Round 2

Bro logs

```
ip.orig_h: 10.1.2.3
ip.orig_P: 1984
ip.resp_h: 192.0.2.1
ip.resp_p: 443
subject: CN=qjpozixk
issuer:  CN=qjpozixk
version: TLSv12
certificate.sig_alg: sha256WithRSAEncryption
validation_status: self signed certificate
```

Snakeoil Cert

- Issuer contains CN only
- Issuer and Subject are the same
- 2048bit Key
- Version 3
- Valid for 10 years
    - Starting now
- Usually SHA1 (for now)
- CN = Hostname.Domain

Metasploit Cert

- Issuer contains CN only
- Issuer and Subject are the same
- 2048bit Key
- Version 3
- Valid for 10 years
    - Starting now - rand(yr * 3) - yr
- Always SHA256
- CN = rand_text_alpha_lower(rand(8)+2)

Snakeoil Cert

- Issuer contains CN only
- Issuer and Subject are the same
- 2048bit Key
- Version 3
- Valid for 10 years
  - Starting now
- Usually SHA1 (for now)
- CN = Hostname.Domain

Metasploit Cert

- Issuer contains CN only
- Issuer and Subject are the same
- 2048bit Key
- Version 3
- Valid for 10 years
  - Starting now - rand(yr * 3) - yr
- Always SHA256
- CN = rand_text_alpha_lower(rand(8)+2)

Bro Script

```
event x509_certificate(f: fa_file , cert_ref: opaque of x509 , cert:
X509::Certificate )
    {
    for ( cid in f$conns )
        { if ( cid$resp_h in 10.0.0.0/8 ) { return; } }
    if ( ! cert?$subject ) { return; }
    if ( ! cert?$issuer ) { return; }
    if ( cert$subject in falselist ) { return; }
    if ( cert$subject != cert$issuer ) { return; }
    if ( /^CN=[a-z]{2,10}$/ == cert$subject )
    if ( "sha256WithRSAEncryption" == cert$sig_alg )

    NOTICE([$note=Metasploit_SSL_Cert, $conn=f$conns[cid],
    $msg=fmt("Metasploit Randomly Generated SSL Cert, '%s'",
cert$subject),
    $sub=cert$issuer]);
    }
```

Bro Script

```
{ if ( cid$resp_h in 10.0.0.0/8 ) { return; } }
if ( ! cert?$subject ) { return; }
if ( ! cert?$issuer ) { return; }
if ( cert$subject in falselist ) { return; }
if ( cert$subject != cert$issuer ) { return; }
if ( /^CN=[a-z]{2,10}$/ == cert$subject )
if ( "sha256WithRSAEncryption" == cert$sig_alg )
```

NOTICE

Bro Script

```
if ( cert$subject != cert$issuer ) { return; }
if ( /^CN=[a-z]{2,10}$/ == cert$subject )
if ( "sha256WithRSAEncryption" == cert$sig_alg )
```

Metasploit SSL Round 3

rwhitcroft commented on Oct 5 • edited

Apparently auto-generated certs are getting snagged by some AV. The format of the generated certs is easy to regex against, and it doesn't help that the subject and issuer are identical.

Currently, the cert looks like this:

```
$ openssl s_client -connect 10.1.1.12:4444 2>/dev/null | egrep '^subject|^issuer'
subject=/CN=ucbigdz
issuer=/CN=ucbigdz
```

which will change slightly in length each time it's generated, but not much. Super easy to detect.

This PR makes them look a little better:

```
$ openssl s_client -connect 10.1.1.12:4444 2>/dev/null | egrep '^subject|^issuer'
subject=/C=US/ST=WI/L=Denise/O=Gary/CN=41tw6z.yl.biz
issuer=/C=US/O=Timothy/CN=Alan Jessica

$ openssl s_client -connect 10.1.1.12:4444 2>/dev/null | egrep '^subject|^issuer'
subject=/C=US/ST=WI/L=Steven/O=George/CN=p.h1qtuz.org
issuer=/C=US/O=Andrea/CN=Bobby Jeremy

$ openssl s_client -connect 10.1.1.12:4444 2>/dev/null | egrep '^subject|^issuer'
subject=/C=US/ST=VT/L=Antonio/O=Roy/CN=td.0swgljfedb.a.edu
issuer=/C=US/O=Nicholas/CN=Gregory Harry
```

 improve cert generation ✔ 3251adf

Certificate Style

subject=/C=US/ST=WI/L=Denise/O=Gary/CN=41tw6z.yl.biz
issuer=/C=US/O=Timothy/CN=Alan Jessica

subject=/C=US/ST=WI/L=Steven/O=George/CN=p.h1qtuz.org
issuer=/C=US/O=Andrea/CN=Bobby Jeremy

subject=/C=US/ST=VT/L=Antonio/O=Roy/CN=td.0swgljfedb.a.edu
issuer=/C=US/O=Nicholas/CN=Gregory Harry

Certificate Style

subject=/C=US/ST=WI/L=Denise/O=Gary/CN=41tw6z.yl.biz
issuer=/C=US/O=Timothy/CN=Alan Jessica

subject=/C=US/ST=WI/L=Steven/O=George/CN=p.h1qtuz.org
issuer=/C=US/O=Andrea/CN=Bobby Jeremy

subject=/C=US/ST=VT/L=Antonio/O=Roy/CN=td.0swgljfedb.a.edu
issuer=/C=US/O=Nicholas/CN=Gregory Harry

subject=^C=US\/ST=[A-Z]{2}\/L=[A-Z][a-z]+\/O=[A-Z][a-z]+\/CN=(\w|\.)+$
issuer=^C=US\/O=[A-Z][a-z]+\/CN=[A-Z][a-z]+\s[A-Z][a-z]+$

Extract and mixin cert ops from server module

Generic SSL routines can be in their own module, for import by consumers without having to drag the entire server infrastructure in with it.

This pulls the certificate methods into Rex::Socket::Ssl for use by consumers, and includes the module in Rex::Socket::SslTcpServer as the initial consumer.

---

master (#8)

RageLtMan committed 20 days ago

Showing 2 changed files with 150 additions and 136 deletions.

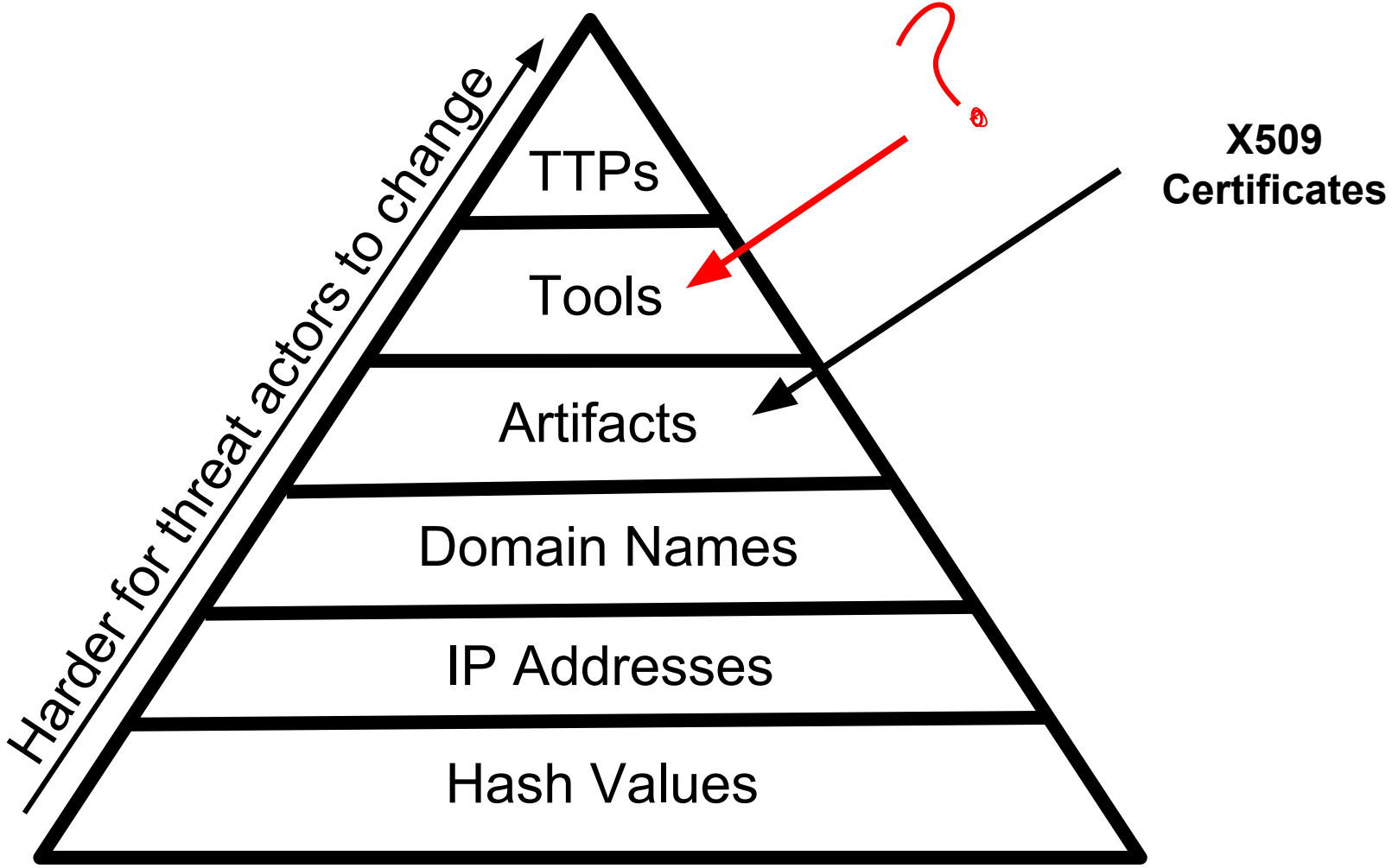
Certificate Style

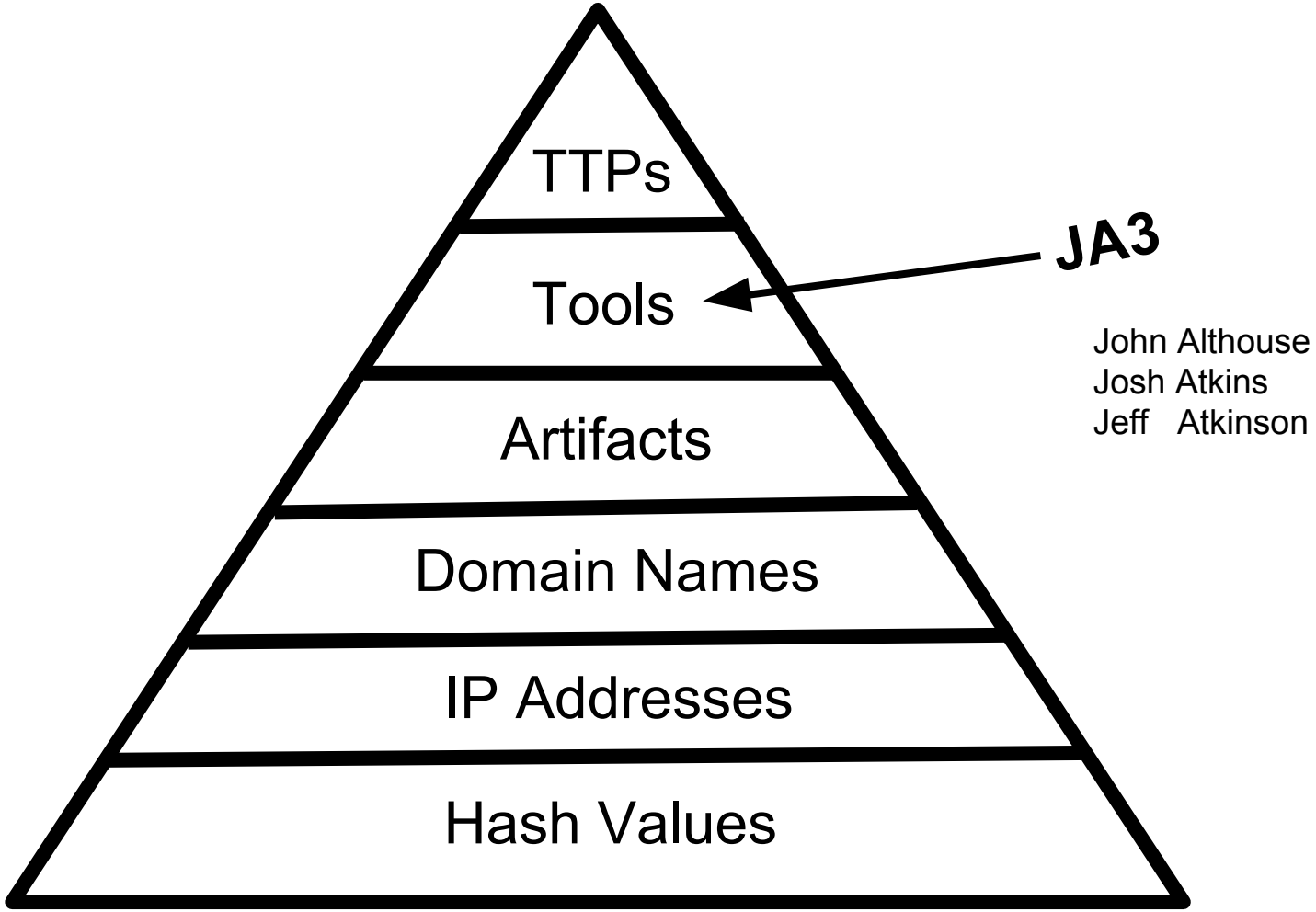
Subject: C=/C=US/ST=VA/O=Nienow LLC/OU=connect/CN=nienow.llc.org/emailAddress=connect@nienow.llc.org
Issuer: C=/C=US/ST=VA/O=Nienow LLC/OU=connect/CN=nienow.llc.org/emailAddress=connect@nienow.llc.org

Subject: C=/C=US/ST=TN/O=Toy-Rippin/OU=interface/CN=toy.rippin.org/emailAddress=interface@toy.rippin.org
Issuer: C=/C=US/ST=TN/O=Toy-Rippin/OU=interface/CN=toy.rippin.org/emailAddress=interface@toy.rippin.org

```
69  +   cert.serial     = (rand(0xFFFFFFFF) << 32) + rand(0xFFFFFFFF)
70  +   cert.subject    = OpenSSL::X509::Name.new([["C", subject]])
71  +   cert.issuer     = OpenSSL::X509::Name.new([["C", issuer]])
```

CertSubject: C=/C=US/ST=*
CertIssuer: C=/C=US/ST=*


Harder for threat actors to change
TTPs
Tools
Artifacts
Domain Names
IP Addresses
Hash Values
?
X509
Certificates


TTPs
Tools
Artifacts
Domain Names
IP Addresses
Hash Values
JA3
John Althouse
Josh Atkins
Jeff Atkinson

BLOG: IVAN RISTIĆ

« Improved handling of SSL warnings in Firefox 3.5 | Main | Examples of the information collected from SSL handshakes »

Analysis of Googlebot's frugal cipher suite list

July 02, 2009

Two weeks ago, I announced SSL Labs and my technique for passive SSL cipher suite analysis. It won't surprise you to learn that I've been carefully observing the cipher suites used in the requests that came to the web site since. (In fact, I announced the site slightly earlier than I had planned because I wanted to get my hands on some real-life data.) One client's SSL fingerprint immediately caught my attention, because it supported only 4 cipher suites. It was Googlebot.

There were 115 visits from Googlebot in the two-week period, using 5 different user agent strings (although Googlebot will sometimes send a request without User-Agent set):

- `Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)`
- `SAMSUNG-SGH-E250/1.0 Profile/MIDP-2.0 Configuration/CLDC-1.1`

Ivan Ristić is an entrepreneur, software engineer, author, and application security researcher.

...then there was Lee Brotherston

SquareLemon Blog

TLS fingerprinting

Smarter Defending & Stealthier Attacking

Posted on September 25, 2015

Background

Transport Layer Security (TLS) provides security in the form of encryption to all manner of network connections from legitimate financial transactions, to private conversations, and malware calling home. The inability for an eavesdropper to analyze this encrypted traffic protects its users, whether they are legitimate or malicious. Those using TLS operate under the assumption that although an eavesdropper can easily observe the existence of their session, its source and destination IP addresses, that the content itself is secure and unreadable without access to cryptographic keying material at one or both ends of the connection. On the surface this holds true, barring any configuration flaws or exploitable vulnerabilities. However, using TLS Fingerprinting, it is easy to quickly and passively determine which client is being used, and then to apply this information from both the attacker and the defender perspectives.

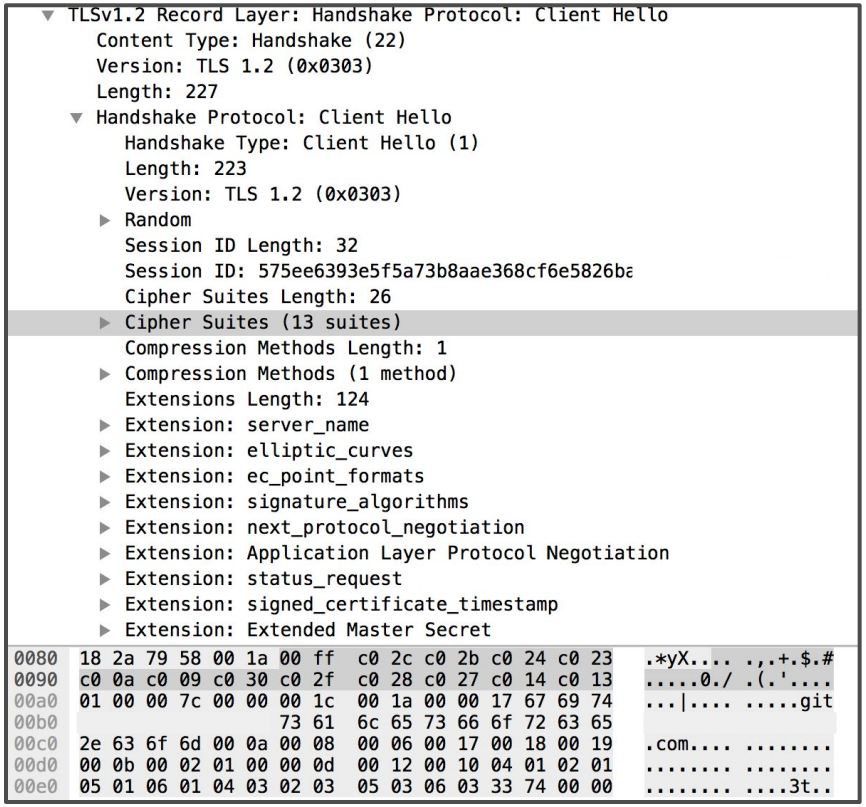
Previously, I have been able to demonstrate that certain clients could be differentiated from other network traffic. Specifically, that meant discriminating SuperFish, PrivDog, and GeniusBox from mainstream browsers when making HTTPS connections, and generating IDS signatures based on these findings to assist network administrators in being able to identify problematic hosts without requiring access to either endpoint. I have now expanded this technique to improve the accuracy of the fingerprints; provide tools to enable others to create fingerprints; and tools that will enable use by others in their own environments.

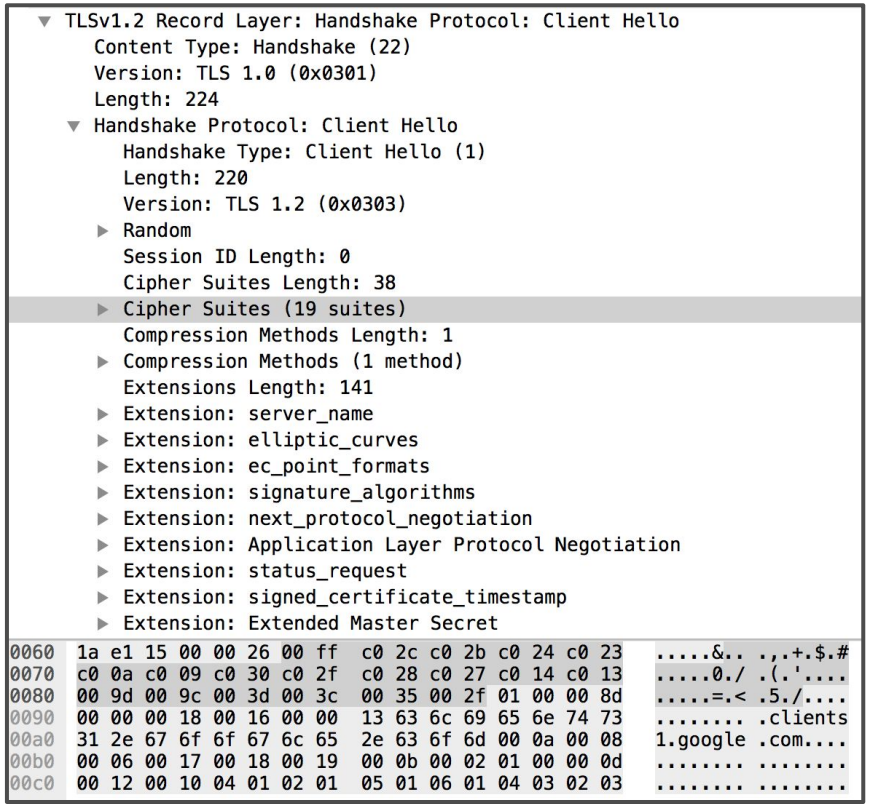
TLS

Our Requirements

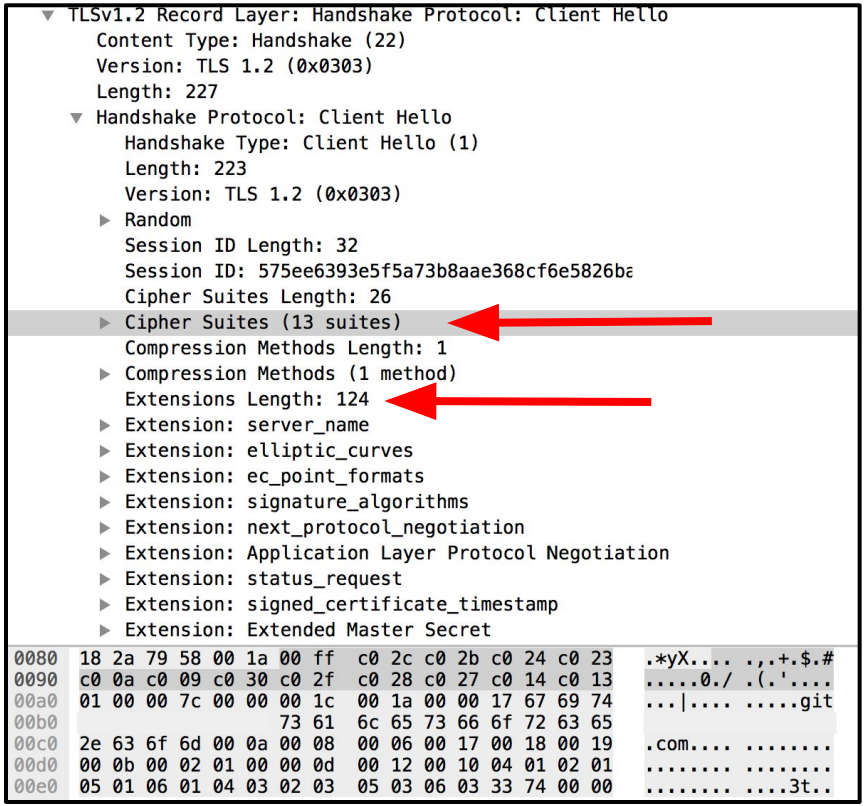
- Needs to work on existing tools
- Destination agnostic, focused on client
- Unique to client application
- Easy to create
- Easy to share
- Easy to consume by any tool

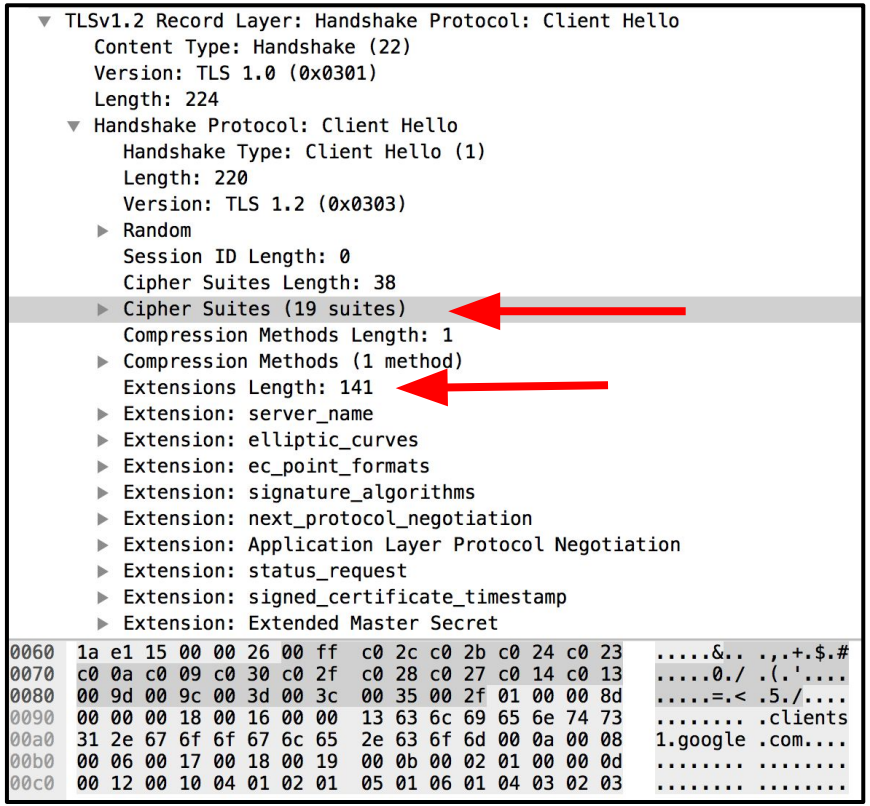
SSL Client Hello



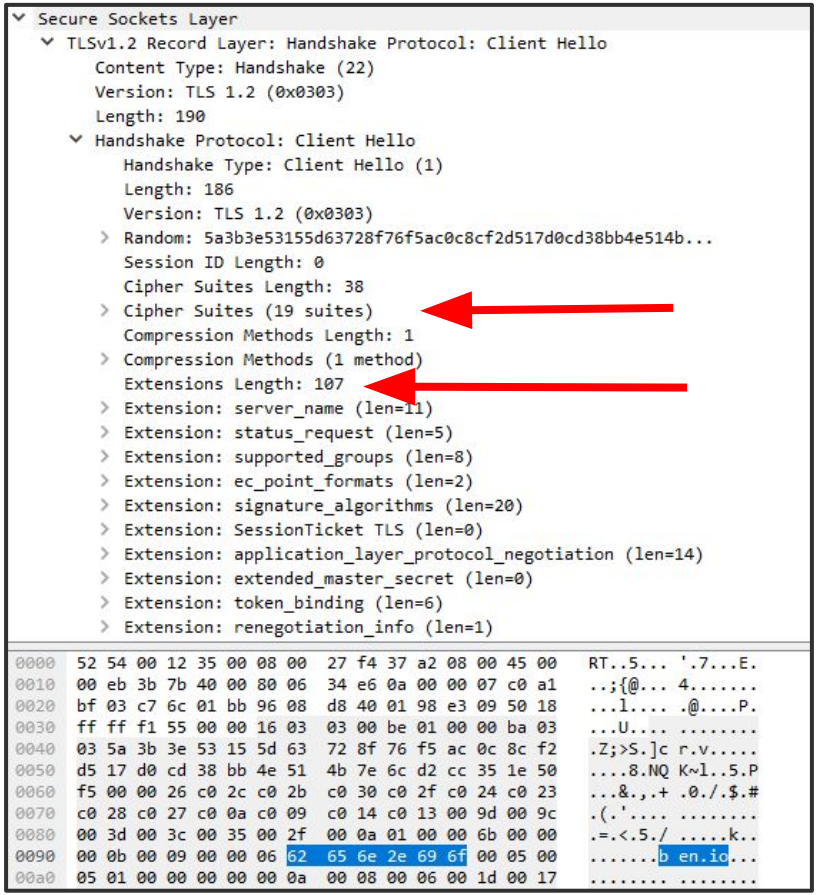


How SSL works



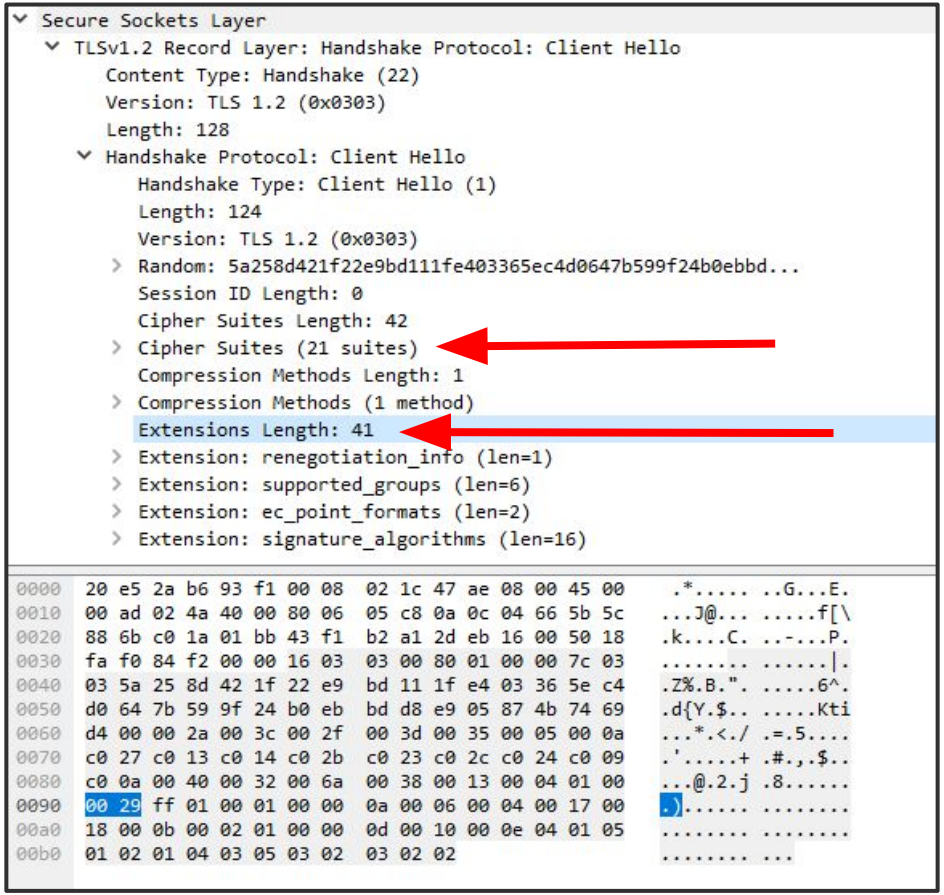


Microsoft Edge (Browser)



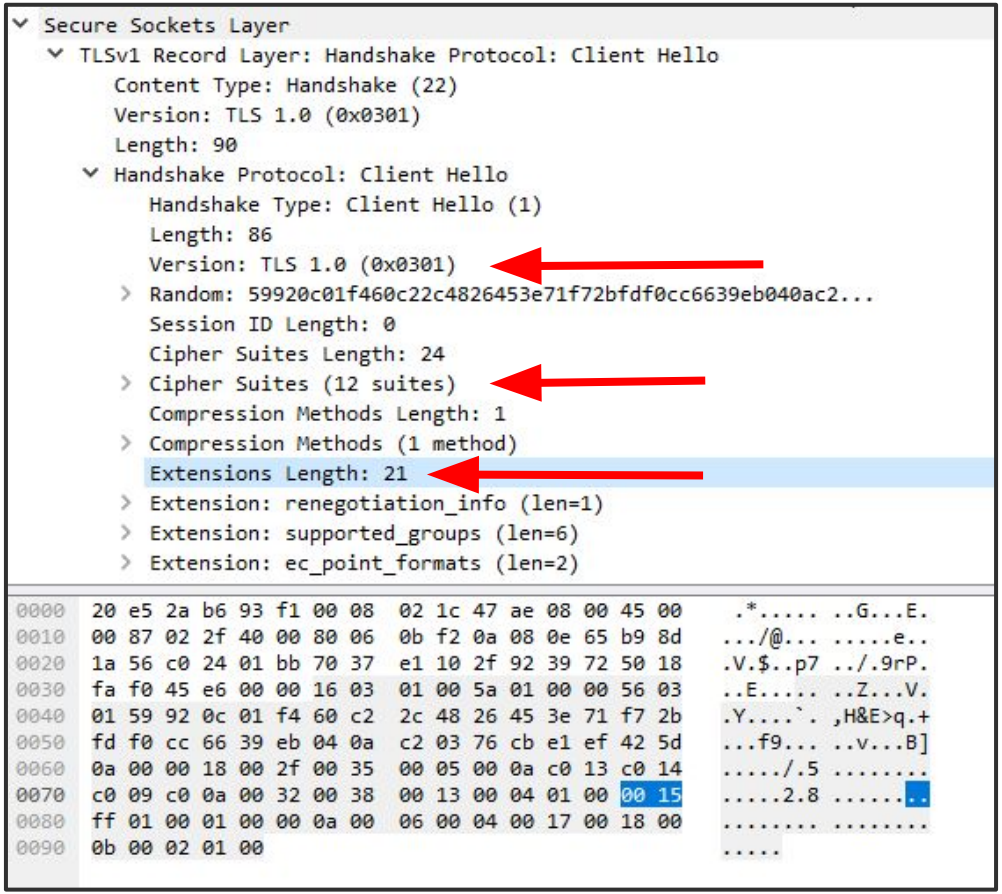
TLS Version: 1.2
Cipher Suites: 19
Extensions: 107

Dridex Malware (Banking Trojan)



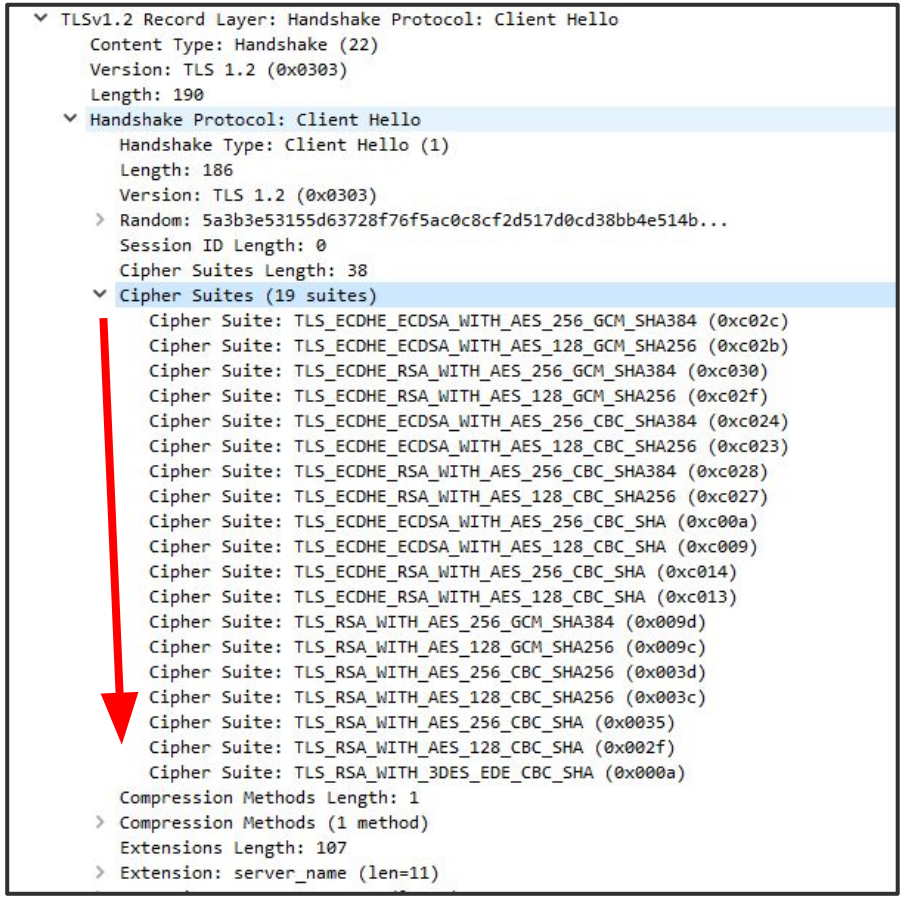
TLS Version 1.2
Cipher Suites: 21
Extensions: 41

Trickbot Malware (Banking Trojan)

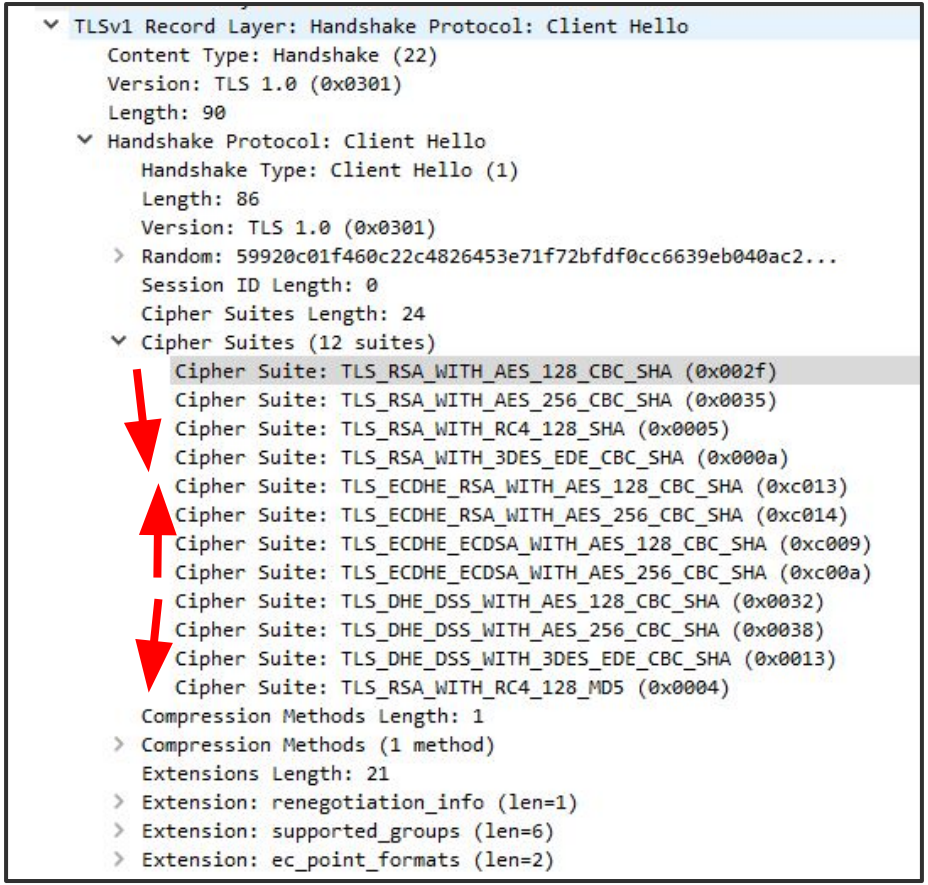


TLS Version: 1
Cipher Suites: 12
Extensions: 21

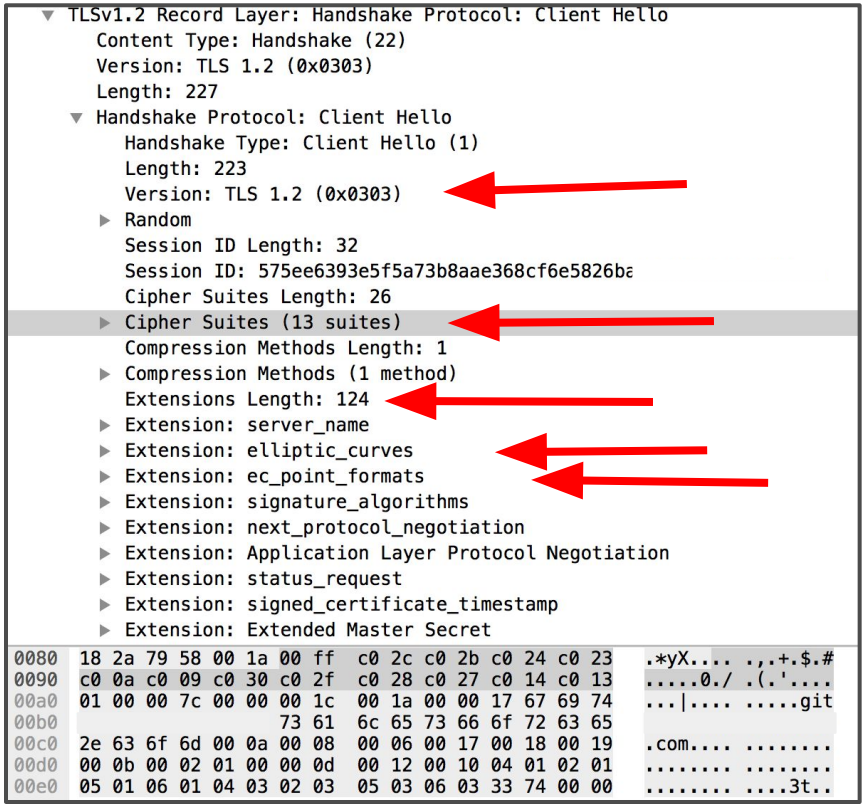
Microsoft Edge (Browser)

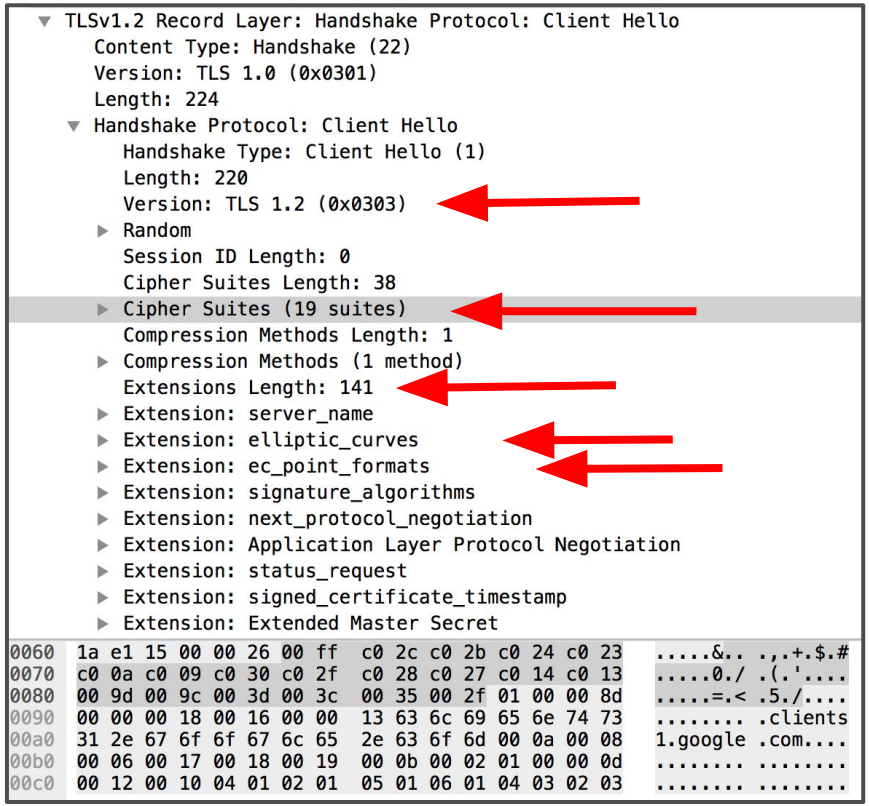


Trickbot Malware (Banking Trojan)



Fingerprinting TLS Clients





Fingerprinting TLS - The JA3 Method

Fingerprinting TLS - The JA3 Method

Version

771

Fingerprinting TLS - The JA3 Method

Version,Ciphers

771,49172-157-156-61-53-47-10

Fingerprinting TLS - The JA3 Method

Version,Ciphers,Extensions

771,49172-157-156-61-53-47-10,0-5-10-11-13

Fingerprinting TLS - The JA3 Method

Version,Ciphers,Extensions,EllipticCurves

771,49172-157-156-61-53-47-10,0-5-10-11-13,29-23-24

Fingerprinting TLS - The JA3 Method

Version,Ciphers,Extensions,EllipticCurves,ECPointFormats

771,49172-157-156-61-53-47-10,0-5-10-11-13,29-23-24,0

Fingerprinting TLS - The JA3 Method

Version,Ciphers,Extensions,EllipticCurves,ECPointFormats

771,49172-157-156-61-53-47-10,0-5-10-11-13,29-23-24,0

MD5 hash

Fingerprinting TLS - The JA3 Method

Version,Ciphers,Extensions,EllipticCurves,ECPointFormats

771,49172-157-156-61-53-47-10,0-5-10-11-13,29-23-24,0

MD5 hash

JA3 = `f4c4f050188e15839a6cd3af798b6c77`

Fingerprinting TLS - The JA3 Method

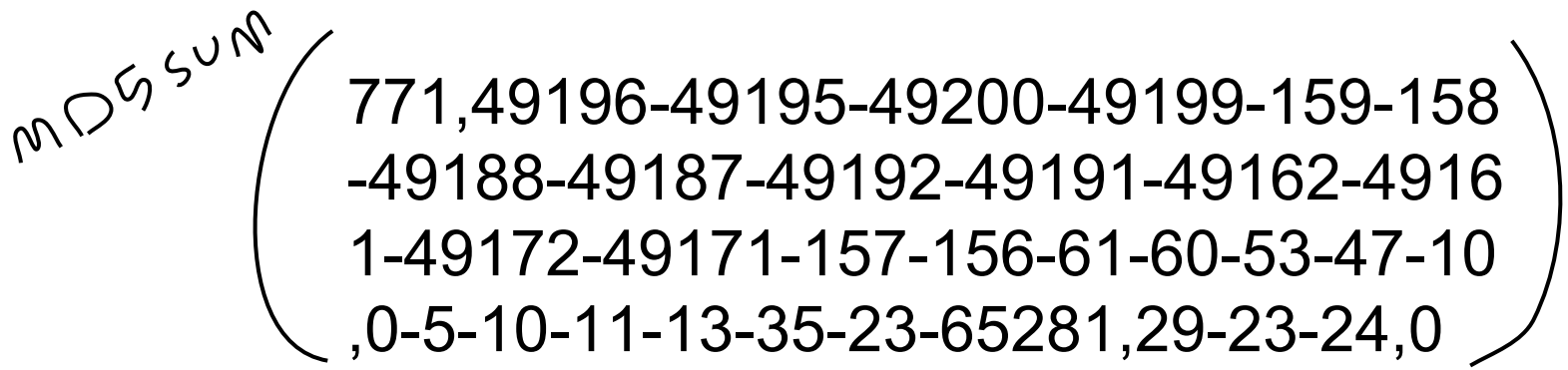
Version,Ciphers,Extensions,EllipticCurves,ECPointFormats

771,49172-157-156-61-53-47-10,,,

MD5 hash

JA3 = `4dd4fca5534245b13b641d54a7035851`

Fingerprinting TLS - The JA3 Method



Google Chrome

JA3 = 94c485bca29d5392be53f2b8cf7f4304

Microsoft Edge

JA3 = 10ee8d30a5d01c042afd7b2b205facc4

Tor Client

JA3 = e7d705a3286e19ea42f587b344ee6865

Dridex Malware

JA3 = 74927e242d6c3febf8cb9cab10a7f889

Trickbot Malware

JA3 = 6734f37431670b3ab4292b8f60f29984

Mapping JA3 to Client Application

Branch: master ▾ trisul-scripts / lua / frontend_scripts / reassembly / ja3 / prints / **ja3fingerprint.json** Find file Copy path

keith4 removing extra commas from Apache hashes d8b2493 9 days ago

3 contributors

624 lines (615 sloc) 154 KB Raw Blame History


```
{"desc":"Adium 1.5.10 (a)","ja3_hash":"93948924e733e9df15a3bb44404cd909","ja3_str":"769,255-49188-49187-49162-49161-49160-49192-49191-4917
{"desc":"Adium 1.5.10 (b)","ja3_hash":"e4adf57bf4a7a2dc08e9495f1b05c0ea","ja3_str":"769,255-49188-49187-49162-49161-49160-49192-49191-4917
{"desc":"AirCanada Android App","ja3_hash":"d5169d6e19447685bf6f1af8c055d94d","ja3_str":"769,52393-52392-52244-52243-49195-49199-49196-492
{"desc":"AirCanada Android App","ja3_hash":"0bb402a703d08a608bf82763b1b63313","ja3_str":"769,52393-52392-52244-52243-49195-49199-49196-492
{"desc":"Android App","ja3_hash":"662fdc668dd6af994a0f903dbcf25d66","ja3_str":"769,49195-49199-158-49162-49161-49171-49172-51-57-156-47-53
{"desc":"Android Google API Access","ja3_hash":"515601c4141e718865697050a7a1765f","ja3_str":"769,49195-49196-49199-49200-158-159-49161-491
{"desc":"Android Webkit Thing","ja3_hash":"855953256ecc8e2b6d2360aff8e5d337","ja3_str":"769,49195-49199-49162-49161-49171-49172-51-50-57-4
{"desc":"Android Webkit Thing","ja3_hash":"99d8afeec9a4422120336ad720a5d692","ja3_str":"769,49195-49199-49162-49161-49171-49172-51-50-57-4
{"desc":"Android Webkit Thing","ja3_hash":"85bb8aa8e5ba373906348831bdbed41a","ja3_str":"769,49195-49196-49199-49200-158-159-49161-49162-49
{"desc":"Android Webkit Thing","ja3_hash":"1aab4c2c84b6979c707ed052f724734b","ja3_str":"769,49195-49196-49199-49200-158-159-49161-49162-49
{"desc":"Android Webkit Thing","ja3_hash":"5331a12866e19199b363f6e903381498","ja3_str":"769,49195-49196-49199-49200-158-159-49161-49162-49
{"desc":"Android Webkit Thing","ja3_hash":"25b72c88f837567856118febcca761e0","ja3_str":"769,49200-49196-49192-49188-49172-49162-163-159-10
{"desc":"Apple Push Notification System, apple.WebKit.Networking,CalendarAgent,Go for Gmail","ja3_hash":"d4693422c5ce1565377aca25940ad80c"
```

github.com/salesforce/ja3/tree/master/lists

```
"Copyright (c) 2017 salesforce.com inc.
All rights reserved.
Licensed under the BSD 3-Clause license.
For full license text see LICENSE.txt file in the repo root or https://opensource.org/licenses/BSD-3-Clause",
61d50e7771aee7f2f4b89a7200b4d45e,"AcroCEF"
49a6cf42956937669a01438f26e7c609,"AIM"
561145462cfc7de1d6a97e93d3264786,"Airmail 3"
f6fd83a21f9f3c5f9ff7b5c63bbc179d,"Alation Compose"
6003b52942a2e1e1ea72d802d153ec08,"Amazon Music"
eb149984fc9c44d85ed7f12c90d818be,"Amazon Music,Dreamweaver,Spotify"
8e3f1bf87bc652a20de63bfd4952b16a,"AnypointStudio"
5507277945374659a5b4572e1b6d9b9f,"apple.geod"
f753495f2eab5155c61b760c838018f8,"apple.geod"
ba40fea2b2638908a3b3b482ac78d729,"apple.geod,parsecd,apple.photomoments"
474e73aea21d1e0910f25c3e6c178535,"apple.WebKit.Networking"
eeeb5e7485f5e10cbc39db4cfb69b264,"apple.WebKit.Networking"
d4693422c5ce1565377aca25940ad80c,"apple.WebKit.Networking,CalendarAgent,Go for Gmail"
63de2b6188d5694e79b678f585b13264,"apple.WebKit.Networking,Chatter,FieldServiceApp,socialstudio"
3e4e87dda5a3162306609b7e330441d2,"apple.WebKit.Networking,itunesstored"
7b343af1092863fdd822d6f10645abfb,"apple.WebKit.Networking,itunesstored"
a312f9162a08eeedf7feb7a13cd7e9bb,"apple.WebKit.Networking,Spotify,WhatsApp,Skype,iTunes"
c5c11e6105c56fd29cc72c3ac7a2b78b,"AT&T Connect"
fa030dbcb2e3c7141d3c2803780ee8db,"Battle.net,Dropbox"
0ef9ca1c10d3f186f5786e1ef3461a46,"bitgo,ShapeShift"
cdec81515ccc75a5aa41eb3db22226e6,"BlueJeans,CEPHtmlEngine"
83e04bc58d402f9633983cbf22724b02,"Charles,Google Play Music Desktop Player,Postman,Slack,and other desktop programs"
424008725394c634a4616b8b1f2828a5,"Charles,java,eclipse"
be9f1360cf52dc1f61ae025252f192a3,"Chromium"
def8761e4bcaaf91d99801a22ac6f6d4,"Chromium"
fc5cb0985a5f5e295163cc8ffff8a6e1,"Chromium"
e7d46c98b078477c4324031e0d3b22f5,"Cisco AnyConnect Secure Mobility Client"
ed36017db541879619c399c95e22067d,"Cisco AnyConnect Secure Mobility Client"
5ee1a653fb824db7182714897fd3b5df,"Citrix Viewer"
```

Mapping JA3 to Client Application

Events | Patterns | Statistics (15) | Visualization

100 Per Page | Format | Preview

JA3	ClientApplication	count	percent
f58966d34ff9488a83797b55c804724d	Google Chrome	451	40.962761
94c485bca29d5392be53f2b8cf7f4304	Google Chrome	339	30.790191
bc6c386f480ee97b9d9e52d472b772d8	Google Chrome	172	15.622162
be1a7de97ea176604a3c70622189d78d		44	3.996367
c07cb55f88702033a8f52c046d23e0b2	Used by many programs on OSX,apple.WebKit.Networking	27	2.452316
83e04bc58d402f9633983cbf22724b02	Charles,Google Play Music Desktop Player,Postman,Slack,and other desktop programs	19	1.725704
a312f9162a08eeedf7feb7a13cd7e9bb	apple.WebKit.Networking,Spotify,WhatsApp,Skype,iTunes	13	1.180745
6cd1b944f5885e2cfbe98a840b75eeb8	Google Chrome	13	1.180745
da949afd9bd6df820730f8f171584a71	Google Chrome	6	0.544959
0b61c673ee71fe9ee725bd687c455809	Google Chrome	5	0.454133
baaac9b6bf25ad098115c71c59d29e51	Google Chrome	4	0.363306
62448833d8230241227c03b7d441e31b	parsecd,apple.geod,apple.photomoments,photoanalysisd,FreedomProxy	4	0.363306
b4f4e6164f938870486578536fc1ffce	Google Chrome	2	0.181653
f28d34ce9e732f644de2350027d74c3f	Used by many programs,Quip,Aura,Spotify,Chatty	1	0.090827
f1c5cf087b959cec31bd6285407f689a	Used by many programs on OSX,apple.WebKit.Networking	1	0.090827

Large Network Example

JA3	ClientApplication	count	percent
94c485bca29d5392be53f2b8cf7f4304	Google Chrome	25091171	39.108785
f58966d34ff9488a83797b55c804724d	Google Chrome	9917687	15.458373
3e4e87dda5a3162306609b7e330441d2	apple.WebKit.Networking,itunesstored	4099667	6.390016
bc6c386f480ee97b9d9e52d472b772d8	Google Chrome	2960252	4.614048
1885aa9927f99ed538ed895d9335995c		2862309	4.461387
0ffee3ba8e615ad22535e7f771690a28	firefox	1836254	2.862109
c07cb55f88702033a8f52c046d23e0b2	Used by many programs on OSX,apple.WebKit.Networking	1707345	2.661183
187dfde7edc8ceddccd3deeccc21daeb	eclipse,java,studio,STS	1494424	2.329310
6cd1b944f5885e2cfbe98a840b75eeb8	Google Chrome	1193719	1.860611
37f691b063c10372135db21579643bf1		843801	1.315205
1fbe5382f9d8430fe921df747c46d95f	FieldServiceApp,socialstudio	783763	1.221626
e4d448cdfe06dc1243c1eb026c74ac9a	Used by many programs on OSX,apple.WebKit.Networking	717555	1.118429
c05de18b01a054f2f6900ffe96b3da7a	Used by many programs on OSX,apple.WebKit.Networking	715799	1.115692
7b343af1092863fdd822d6f10645abfb	apple.WebKit.Networking,itunesstored	429911	0.670088

Domain=m.google.com

Events | Patterns | Statistics (37) | Visualization

100 Per Page | Format | Preview

JA3	ClientApplication	count	percent
3e4e87dda5a3162306609b7e330441d2	apple.WebKit.Networking,itunesstored	1354514	96.446367
c07cb55f88702033a8f52c046d23e0b2	Used by many programs on OSX,apple.WebKit.Networking	18605	1.324744
bc6c386f480ee97b9d9e52d472b772d8	Google Chrome	14959	1.065136
ab2382d4d8acbfb77e96935ba1e287b7		4358	0.310306
6cd1b944f5885e2cfbe98a840b75eeb8	Google Chrome	3837	0.273208
d5e8fe02b8ad14691619ceb0eba94fc6		2449	0.174378
1ad848b33ae442ed4c23356f04a7dc8e		2369	0.168681
f1c5cf087b959cec31bd6285407f689a	Used by many programs on OSX,apple.WebKit.Networking	1895	0.134931
c56cc8486d51dd06dce2bdf0c8ae9dbc		435	0.030974
b9ffee599a89a2a54c859098a580fe15		314	0.022358

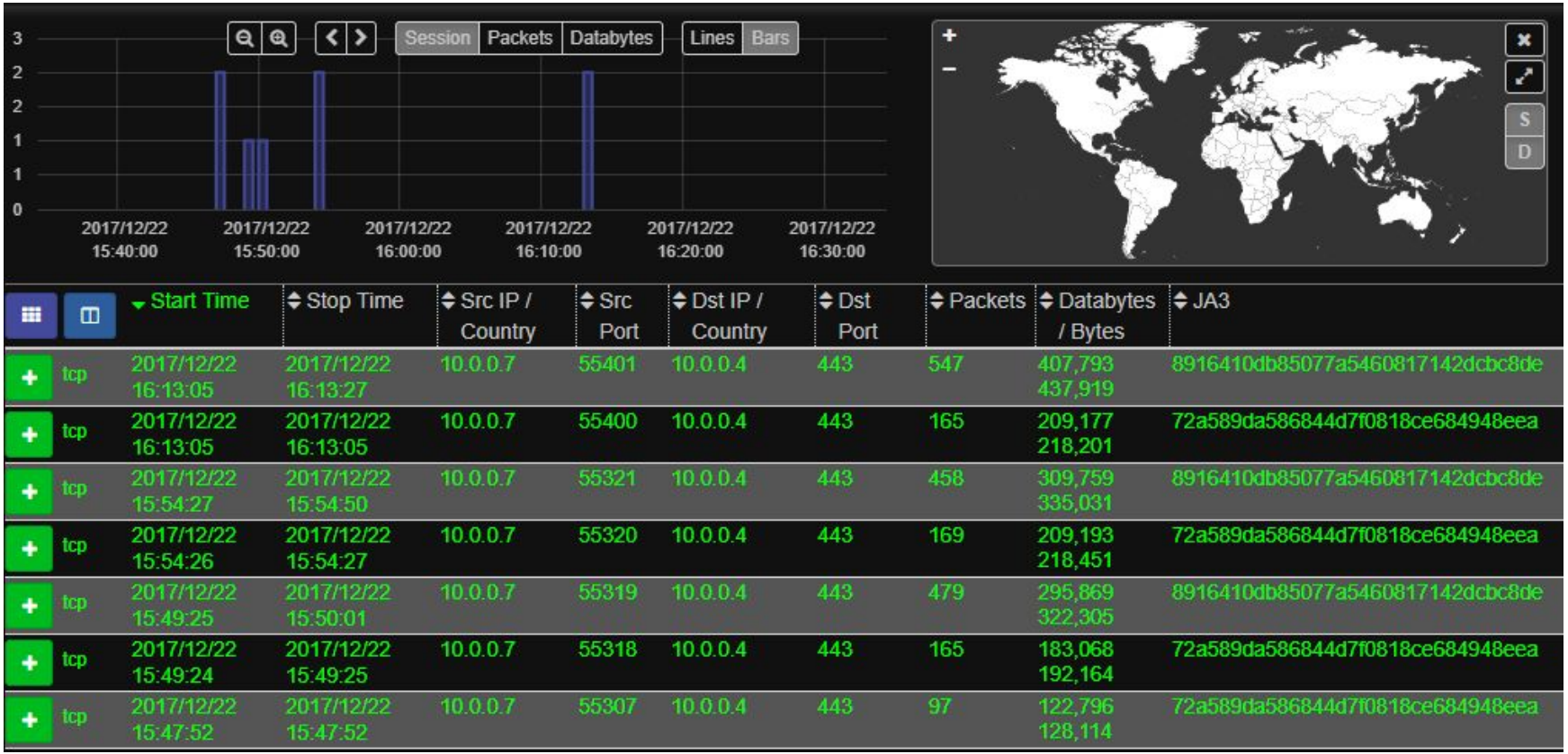
Malware and Sandboxes

Baseline your sandbox

https://github.com/gbarford/testssl

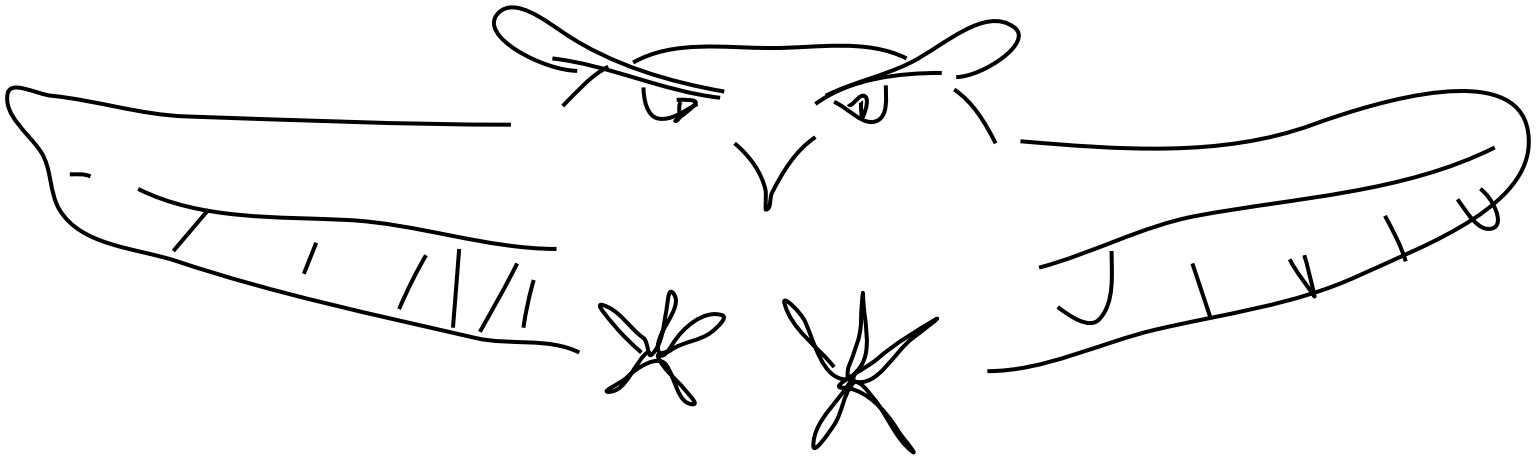
```
Win10-socket:               c12f54a3f91dc7bafd92cb59fe009a35
Win10-socket-SNI:           3b5074b1b5d032e5620f69f9f700ff0e
Win10-powershell:           fc54e0d16d9764783542f0146a98b300
Win10-powershell-SNI:       54328bd36c14bd82ddaa0c04b25ed9ad
Win10-iexplore:             be6155e945a3e59a1dd0841b86f6c945
Win10-iexplore-SNI:         10ee8d30a5d01c042afd7b2b205facc4

Win2016-socket:             043c543b63b895881d9abfbc320cb863
Win2016-socket-SNI:         7c410ce832e848a3321432c9a82e972b
Win2016-powershell:         17b69de9188f4c205a00fe5ae9c1151f
Win2016-powershell-SNI:     235a856727c14dba889ddee0a38dd2f2
Win2016-iexplore:           4f2e9c50db9bd107439136bd24740c0d
Win2016-iexplore-SNI:       f88610704d61a237aa9e5e0849573998
```

Metasploit SSL Round 3



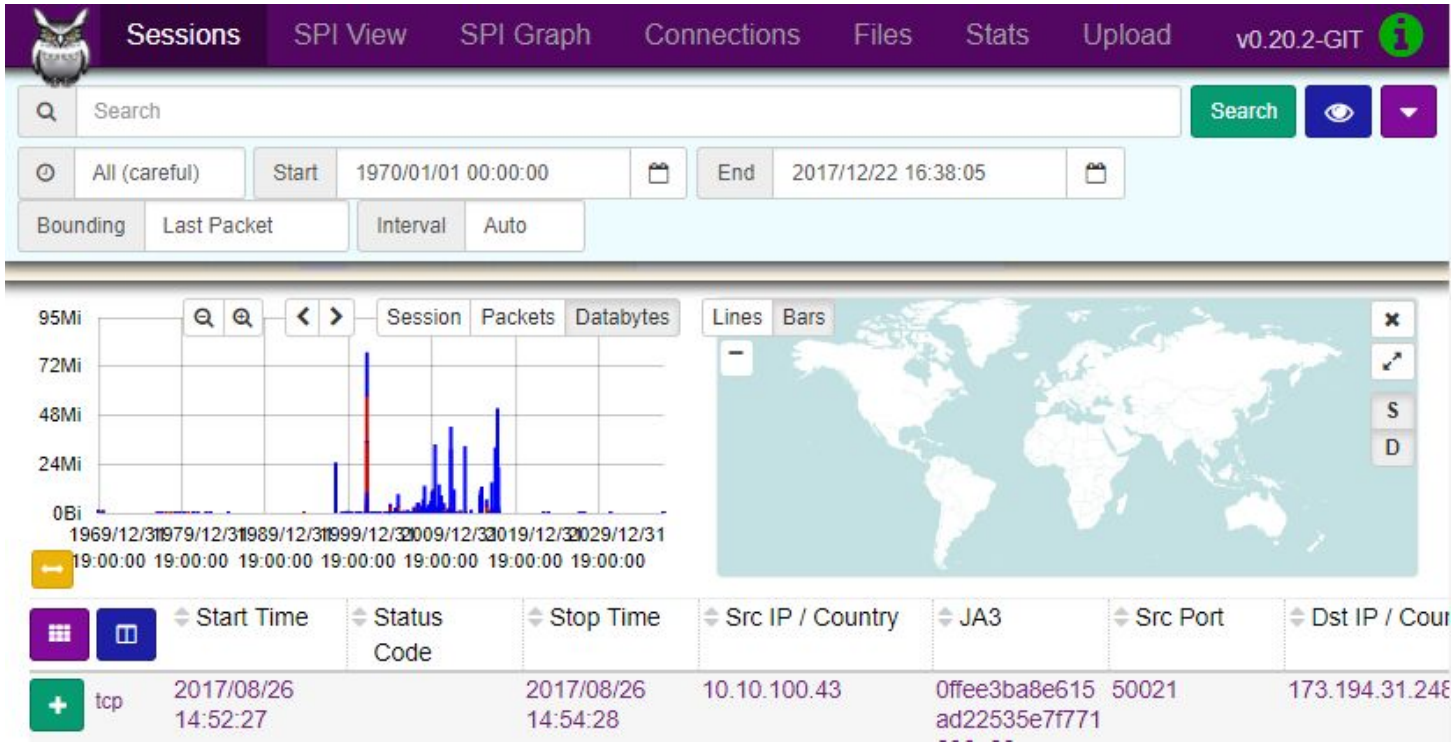
Start Time	Stop Time	Src IP / Country	Src Port	Dst IP / Country	Dst Port	Packets	Databytes / Bytes	JA3
tcp 2017/12/22 16:13:05	2017/12/22 16:13:27	10.0.0.7	55401	10.0.0.4	443	547	407,793 437,919	8916410db85077a5460817142dcbc8de
tcp 2017/12/22 16:13:05	2017/12/22 16:13:05	10.0.0.7	55400	10.0.0.4	443	165	209,177 218,201	72a589da586844d7f0818ce684948eea
tcp 2017/12/22 15:54:27	2017/12/22 15:54:50	10.0.0.7	55321	10.0.0.4	443	458	309,759 335,031	8916410db85077a5460817142dcbc8de
tcp 2017/12/22 15:54:26	2017/12/22 15:54:27	10.0.0.7	55320	10.0.0.4	443	169	209,193 218,451	72a589da586844d7f0818ce684948eea
tcp 2017/12/22 15:49:25	2017/12/22 15:50:01	10.0.0.7	55319	10.0.0.4	443	479	295,869 322,305	8916410db85077a5460817142dcbc8de
tcp 2017/12/22 15:49:24	2017/12/22 15:49:25	10.0.0.7	55318	10.0.0.4	443	165	183,068 192,164	72a589da586844d7f0818ce684948eea
tcp 2017/12/22 15:47:52	2017/12/22 15:47:52	10.0.0.7	55307	10.0.0.4	443	97	122,796 128,114	72a589da586844d7f0818ce684948eea

Moloch

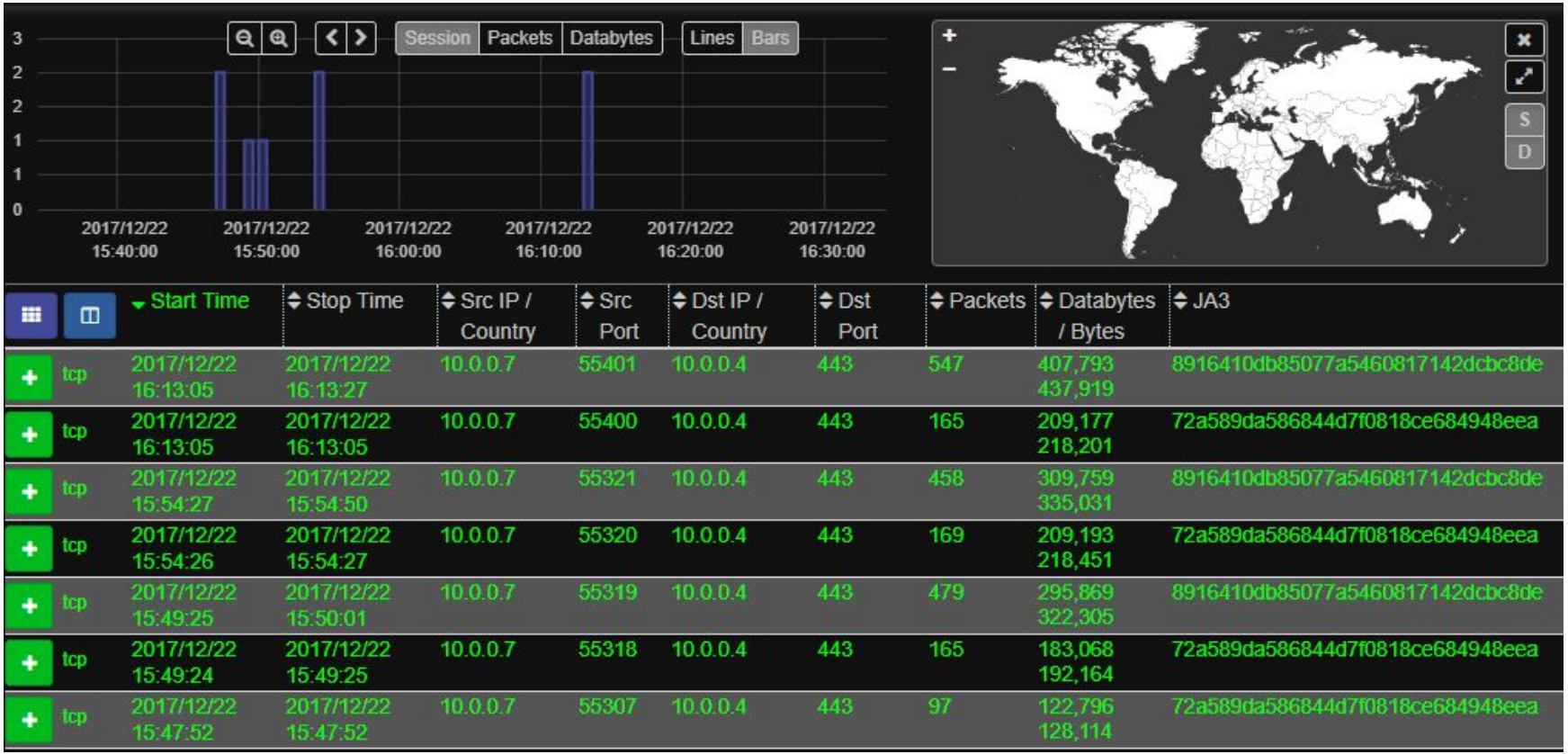


Moloch

molo.ch



Meterpreter on Windows 10



Start Time	Stop Time	Src IP / Country	Src Port	Dst IP / Country	Dst Port	Packets	Databytes / Bytes	JA3
tcp 2017/12/22 16:13:05	2017/12/22 16:13:27	10.0.0.7	55401	10.0.0.4	443	547	407,793 437,919	8916410db85077a5460817142dcbc8de
tcp 2017/12/22 16:13:05	2017/12/22 16:13:05	10.0.0.7	55400	10.0.0.4	443	165	209,177 218,201	72a589da586844d7f0818ce684948eea
tcp 2017/12/22 15:54:27	2017/12/22 15:54:50	10.0.0.7	55321	10.0.0.4	443	458	309,759 335,031	8916410db85077a5460817142dcbc8de
tcp 2017/12/22 15:54:26	2017/12/22 15:54:27	10.0.0.7	55320	10.0.0.4	443	169	209,193 218,451	72a589da586844d7f0818ce684948eea
tcp 2017/12/22 15:49:25	2017/12/22 15:50:01	10.0.0.7	55319	10.0.0.4	443	479	295,869 322,305	8916410db85077a5460817142dcbc8de
tcp 2017/12/22 15:49:24	2017/12/22 15:49:25	10.0.0.7	55318	10.0.0.4	443	165	183,068 192,164	72a589da586844d7f0818ce684948eea
tcp 2017/12/22 15:47:52	2017/12/22 15:47:52	10.0.0.7	55307	10.0.0.4	443	97	122,796 128,114	72a589da586844d7f0818ce684948eea

/meterpreter/reverse_https

Start Time	Stop Time	Src IP / Country	Src Port	Dst IP / Country	Dst Port	Packets	Databytes / Bytes	JA3
2017/12/22 16:13:05	2017/12/22 16:13:27	10.0.0.7	55401	10.0.0.4	443	547	407,793 437,919	8916410db85077a5460817142dcbc8de
2017/12/22 16:13:05	2017/12/22 16:13:05	10.0.0.7	55400	10.0.0.4	443	165	209,177 218,201	72a589da586844d7f0818ce684948eea
2017/12/22 15:54:27	2017/12/22 15:54:50	10.0.0.7	55321	10.0.0.4	443	458	309,759 335,031	8916410db85077a5460817142dcbc8de
2017/12/22 15:54:26	2017/12/22 15:54:27	10.0.0.7	55320	10.0.0.4	443	169	209,193 218,451	72a589da586844d7f0818ce684948eea
2017/12/22 15:49:25	2017/12/22 15:50:01	10.0.0.7	55319	10.0.0.4	443	479	295,869 322,305	8916410db85077a5460817142dcbc8de
2017/12/22 15:49:24	2017/12/22 15:49:25	10.0.0.7	55318	10.0.0.4	443	165	183,068 192,164	72a589da586844d7f0818ce684948eea

/meterpreter/reverse_https

Start Time	Stop Time	Src IP / Country	Src Port	Dst IP / Country	Dst Port	Packets	Databytes / Bytes	JA3
2017/12/22 16:13:05	2017/12/22 16:13:27	10.0.0.7	55401	10.0.0.4	443	547	407,793 437,919	8916410db85077a5460817142dcbc8de
2017/12/22 16:13:05	2017/12/22 16:13:05	10.0.0.7	55400	10.0.0.4	443	165	209,177 218,201	72a589da586844d7f0818ce684948eea
2017/12/22 15:54:27	2017/12/22 15:54:50	10.0.0.7	55321	10.0.0.4	443	458	309,759 335,031	8916410db85077a5460817142dcbc8de
2017/12/22 15:54:26	2017/12/22 15:54:27	10.0.0.7	55320	10.0.0.4	443	169	209,193 218,451	72a589da586844d7f0818ce684948eea
2017/12/22 15:49:25	2017/12/22 15:50:01	10.0.0.7	55319	10.0.0.4	443	479	295,869 322,305	8916410db85077a5460817142dcbc8de
2017/12/22 15:49:24	2017/12/22 15:49:25	10.0.0.7	55318	10.0.0.4	443	165	183,068 192,164	72a589da586844d7f0818ce684948eea

/meterpreter/reverse_https

Dst Port	Packets	Databytes / Bytes	JA3
443	547	407,793 437,919	8916410db85077a5460817142dcbc8de
443	165	209,177 218,201	72a589da586844d7f0818ce684948eea

/meterpreter/reverse_winhttps

	Start Time	Stop Time	Src IP / Country	Src Port	Dst IP / Country	Dst Port	Packets	Databytes / Bytes	JA3
tcp	2017/12/22 17:20:38	2017/12/22 17:21:01	10.0.0.7	49710	10.0.0.4	443	468	312,137 337,883	8916410db85077a5460817142dcbc8de
tcp	2017/12/22 17:20:37	2017/12/22 17:20:38	10.0.0.7	49709	10.0.0.4	443	160	183,657 192,477	8916410db85077a5460817142dcbc8de

Windows 10 Meterpreter HTTPS

Stager (160-170 packets) JA3:

`72a589da586844d7f0818ce684948eea`

Payload JA3:

`8916410db85077a5460817142dcbc8de`

Stager-SNI (160-170 packets) JA3:

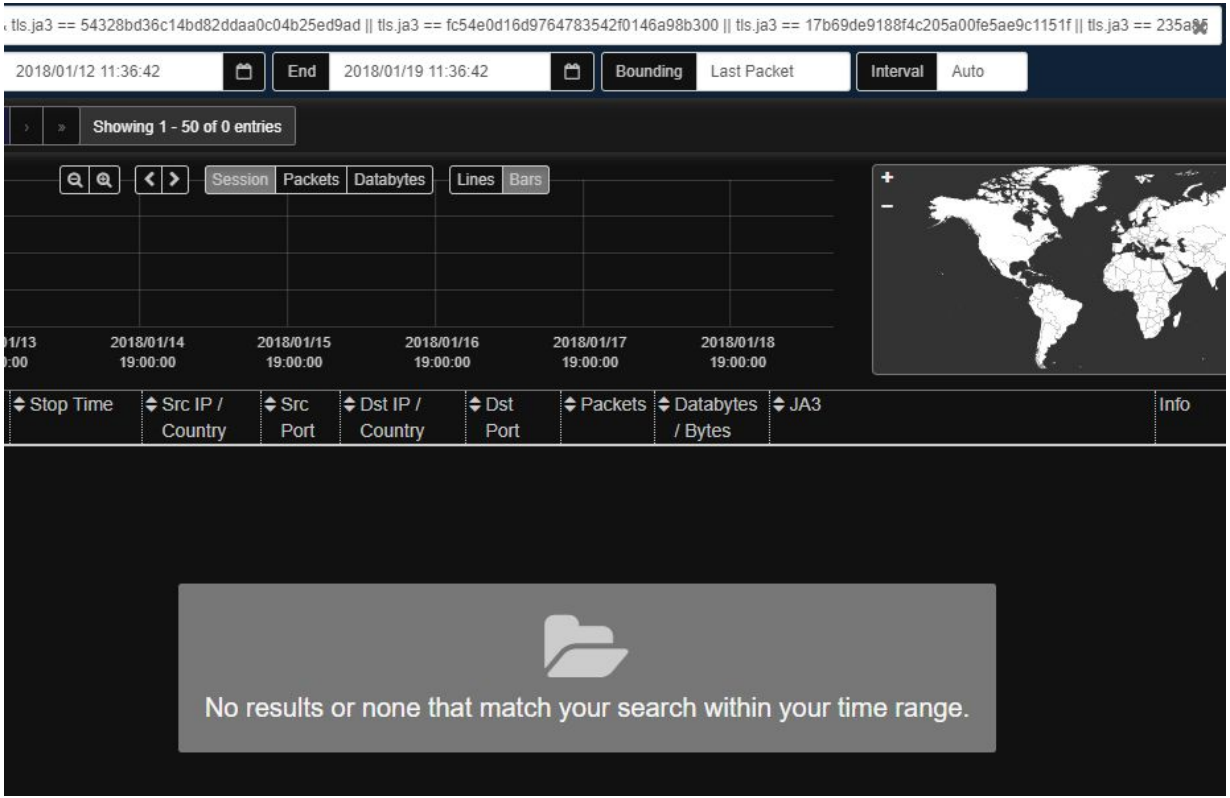
`a0e9f5d64349fb13191bc781f81f42e1`

Payload-SNI JA3:

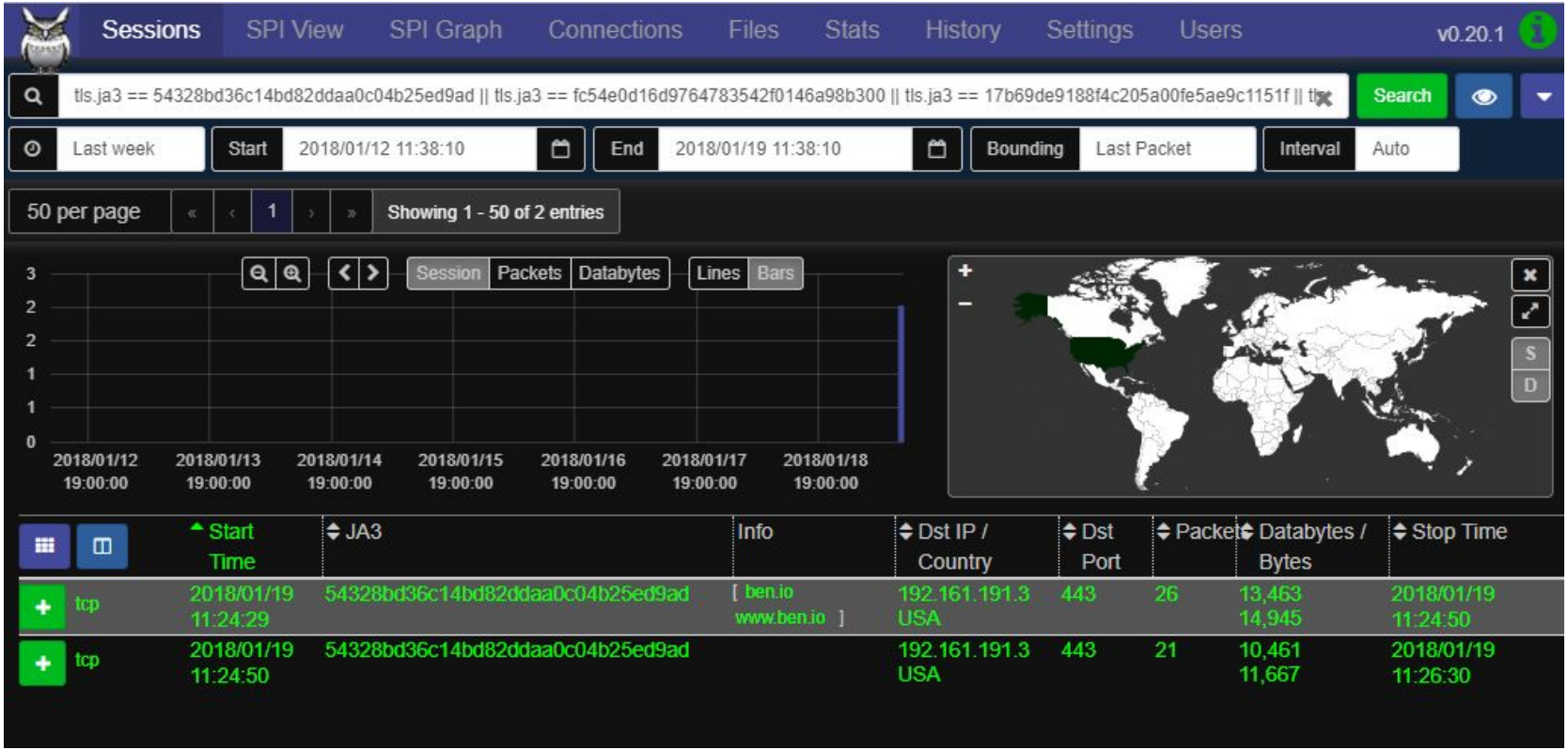
`ce5f3254611a8c095a3d821d44539877`

Powershell Exploit Kits

Powershell Tools (Empire)

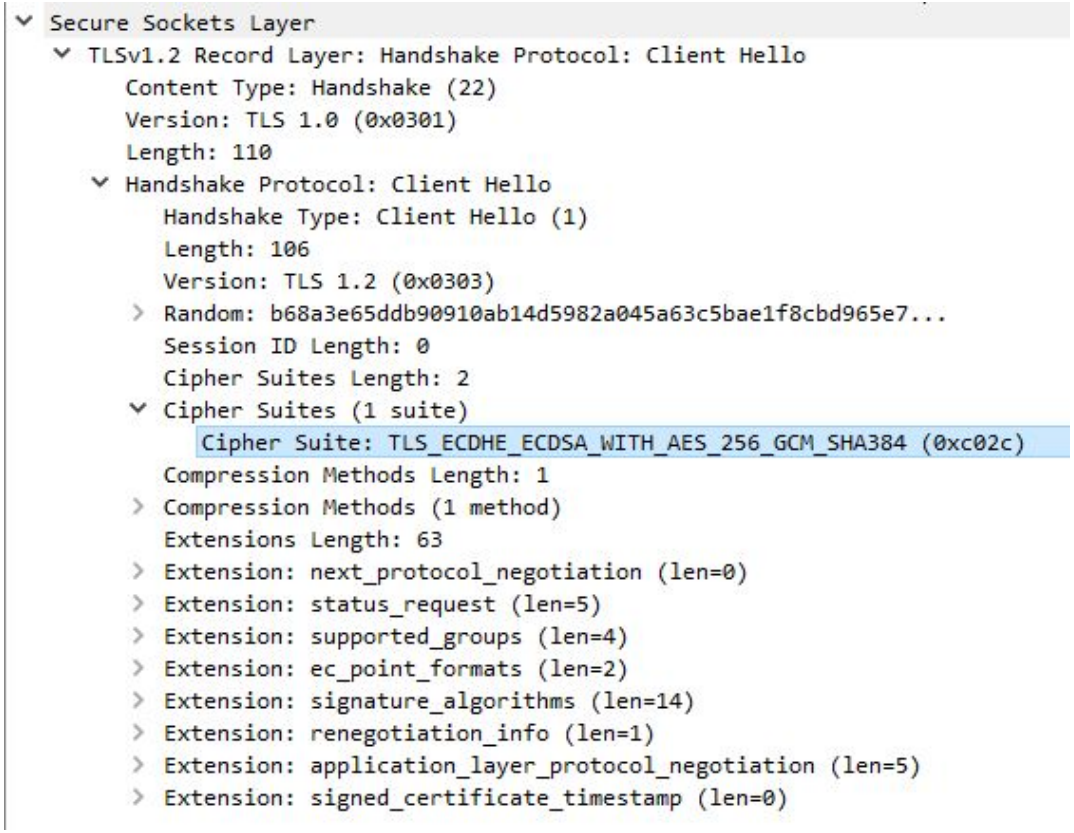


Powershell Tools (Empire)



Custom Targeted Malware

[REDACTED]


Secure Sockets Layer
TLSv1.2 Record Layer: Handshake Protocol: Client Hello
Content Type: Handshake (22)
Version: TLS 1.0 (0x0301)
Length: 110
Handshake Protocol: Client Hello
Handshake Type: Client Hello (1)
Length: 106
Version: TLS 1.2 (0x0303)
Random: b68a3e65ddb90910ab14d5982a045a63c5bae1f8cbd965e7...
Session ID Length: 0
Cipher Suites Length: 2
Cipher Suites (1 suite)
Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
Compression Methods Length: 1
Compression Methods (1 method)
Extensions Length: 63
Extension: next_protocol_negotiation (len=0)
Extension: status_request (len=5)
Extension: supported_groups (len=4)
Extension: ec_point_formats (len=2)
Extension: signature_algorithms (len=14)
Extension: renegotiation_info (len=1)
Extension: application_layer_protocol_negotiation (len=5)
Extension: signed_certificate_timestamp (len=0)

[REDACTED]

JA3:

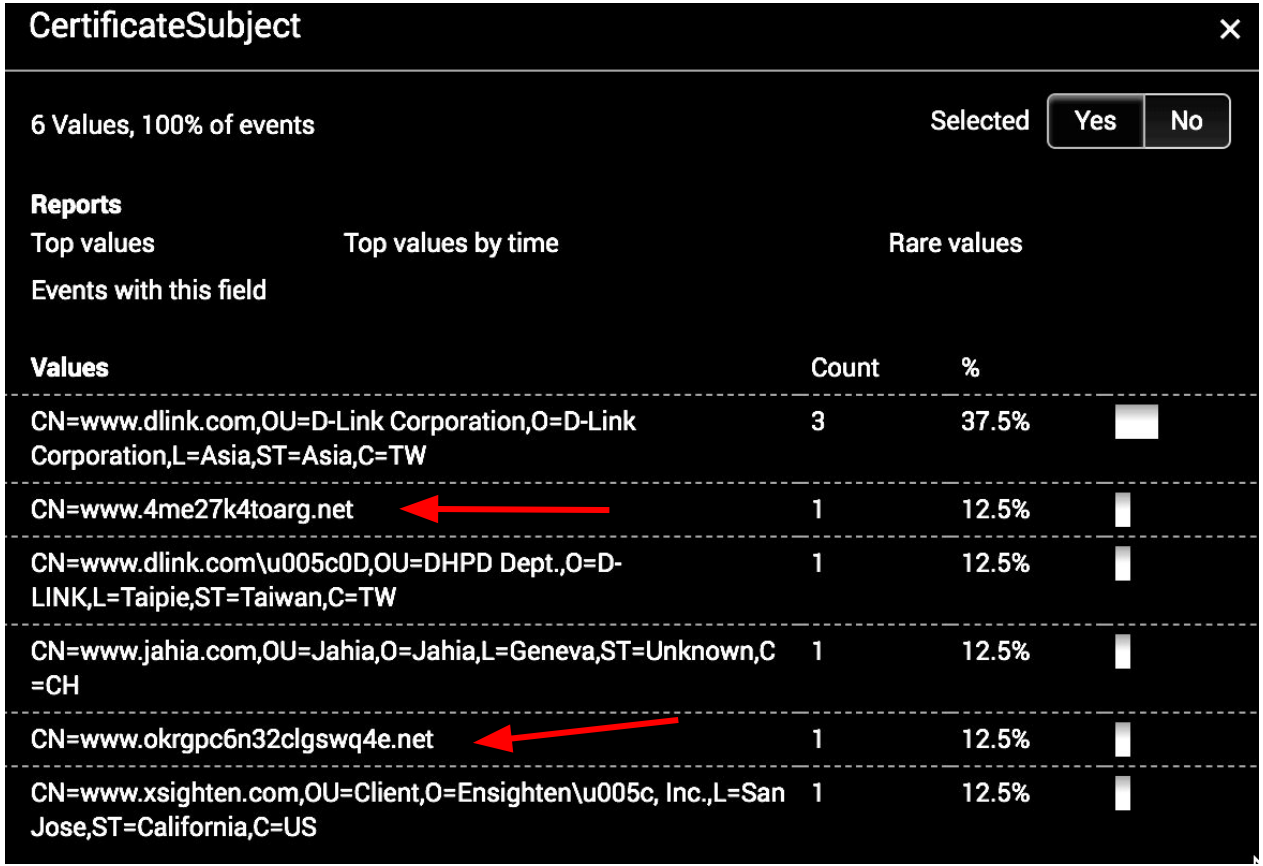
87bb7d3dcf10752c52ebeb53f0a57700

Fingerprint String:

771,49196,13172-5-10-11-13-65281-16-18,24,0

Hunting | Alerts | Analysis

Hunting weird Cert Subjects



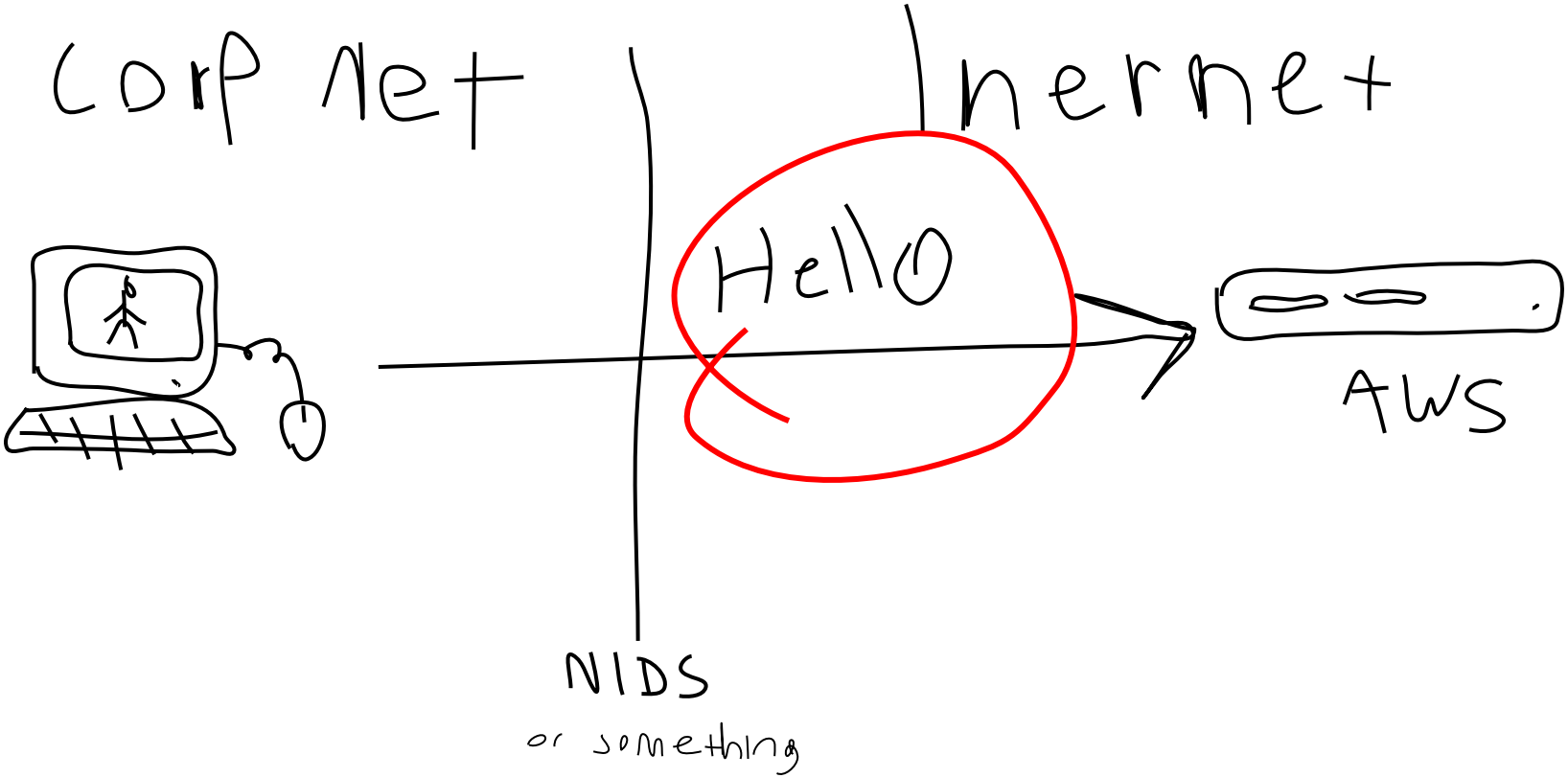
CN=www.okrgpc6n32clgswq4e.net

JA3=2d8794cb7b52b777bee2695e79c15760

100 Per Page / Format Preview

JA3	Domain	PortDestination	count	percent
2d8794cb7b52b777bee2695e79c15760	www.zjjm5e3itoyqpxy6kaedx2.com	443	1	8.333333
2d8794cb7b52b777bee2695e79c15760	www.w3vieagrffbogxrg6jt5.com	443	1	8.333333
2d8794cb7b52b777bee2695e79c15760	www.tvci4b2adx2j4yojj7kywdy5.com	443	1	8.333333
2d8794cb7b52b777bee2695e79c15760	www.pqqnumjtf7do.com	443	1	8.333333
2d8794cb7b52b777bee2695e79c15760	www.pmcn7w.com	443	1	8.333333
2d8794cb7b52b777bee2695e79c15760	www.okj3mbtcogv6jy6.com	443	1	8.333333
2d8794cb7b52b777bee2695e79c15760	www.mxdhm3yg7boyw33q7xr3.com	9001	1	8.333333
2d8794cb7b52b777bee2695e79c15760	www.kxxmnp3.com	443	1	8.333333
2d8794cb7b52b777bee2695e79c15760	www.ehmmkpvufe.com	443	1	8.333333
2d8794cb7b52b777bee2695e79c15760	www.okxqg.com	443	1	8.333333
2d8794cb7b52b777bee2695e79c15760	www.2gqcnd4a6.com	443	1	8.333333
2d8794cb7b52b777bee2695e79c15760	www.2ftdlovu2k2p3h.com	9001	1	8.333333

No Server, No Problem



Enrich SSL/TLS Alerting

File Exfil Detection

Exfil Detection Bro Script

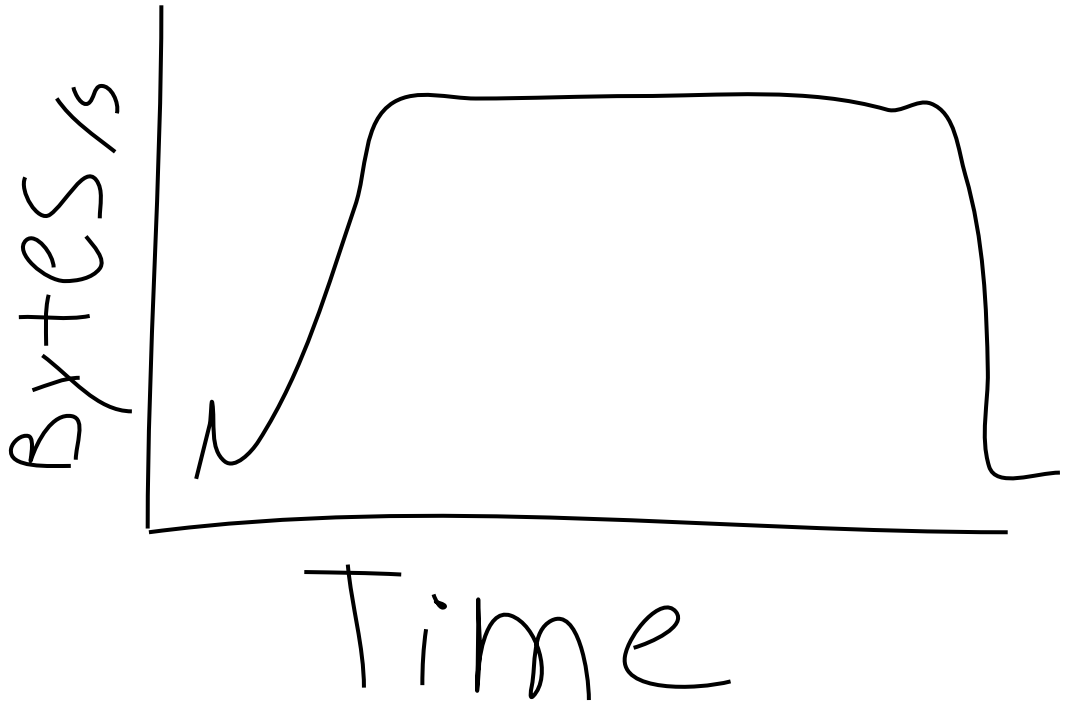
Exfil logging based off of:

https://github.com/reservoirlabs/bro-scripts

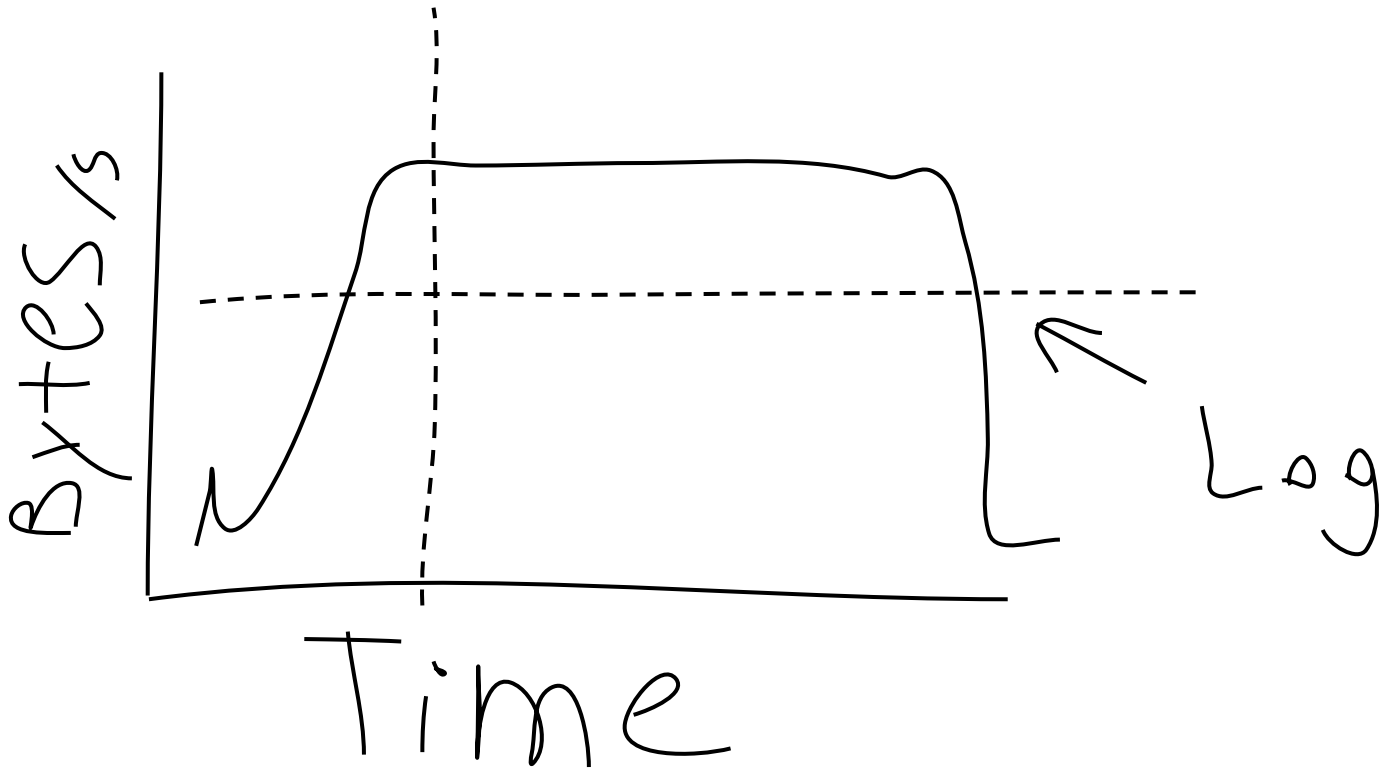
Bob Rotsted

Normal Outbound Traffic

File Transfer Outbound



Threshold Byte Count and Byte Rate



Exfil Logs

src	dst	port	service	cert.subject	bytes
10.1.2.3	50.1.2.3	443	HTTPS	CN=*.dropbox	2098220

Log File Transfers

Add to script:

Service

Domain

SSL Cert

JA3

JA3ClientApplication

```
type Info: record {

    ## Domain from SNI
    domain: string &log &optional;
    ## Subject of the X.509 certificate
    subject: string &log &optional;
    ## JA3 hash
    ja3: string &log &optional;
...
if (c?$ssl) rec$subject = c$ssl$subject;
if (c?$ssl) rec$domain = c$ssl$server_name;
if (c?$ssl) rec$ja3 = c$ssl$ja3;
```

End Result: Valuable Logs

Source IP:	10.1.2.3
Destination IPs:	50.1.2.3, 50.1.2.4, 50.1.2.5 …
Destination Port:	443
Service:	HTTPS
Destination Certificate:	CN=*.dropbox.com ...
Certificate Valid:	True
Files Transferred:	512
TotalBytes Transferred:	2,048MB
JA3:	fa030dbcb2e3c7141d3c2803780ee8db
JA3ClientApplication:	Dropbox

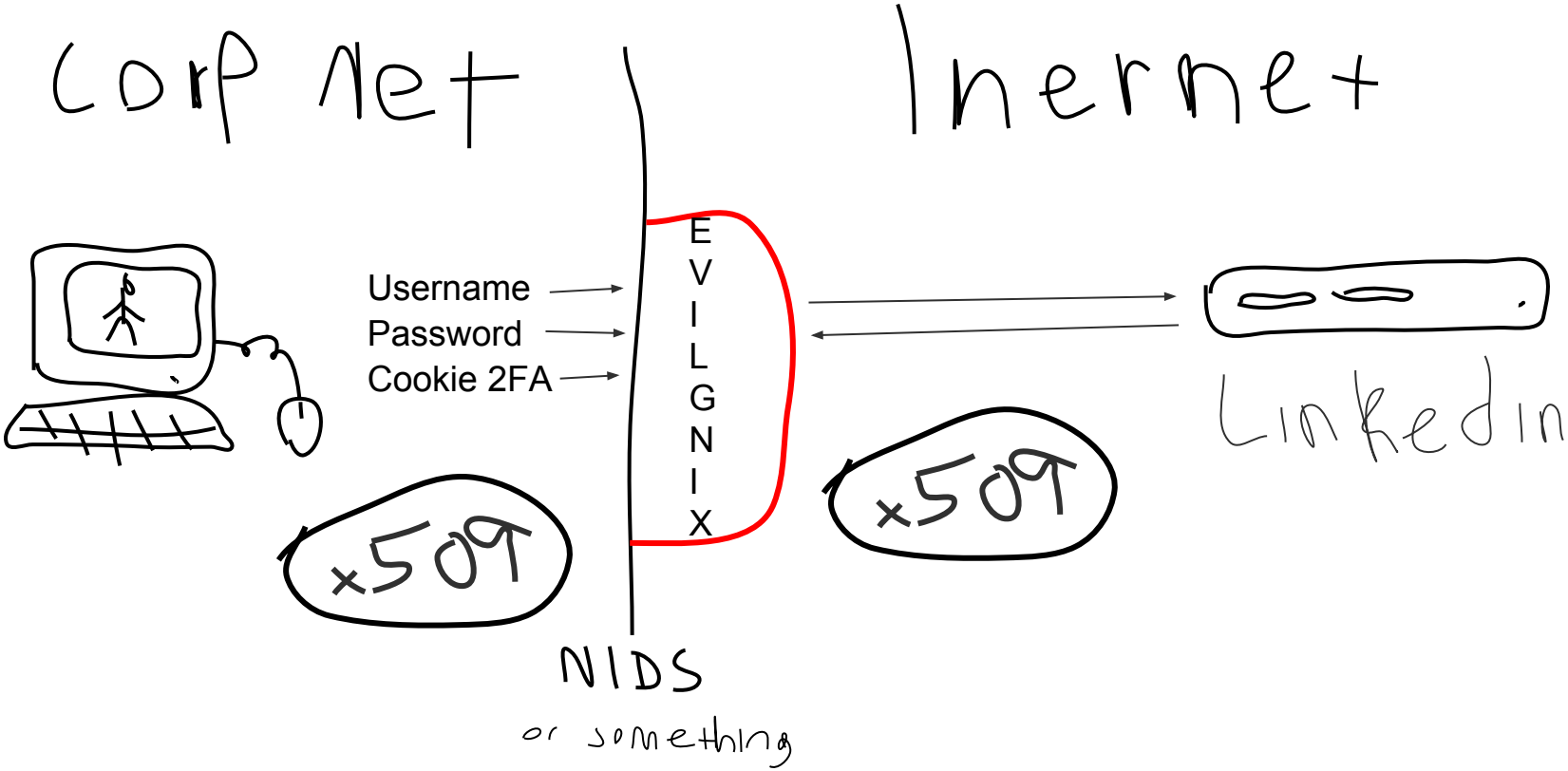
End Result: Valuable Logs

Source IP:	10.1.2.3
Destination IPs:	50.1.2.3, 50.1.2.4, 50.1.2.5 …
Destination Port:	443
Service:	HTTPS
Destination Certificate:	CN=*.dropbox.com ...
Certificate Valid:	True
Files Transferred:	512
TotalBytes Transferred:	2,048MB
JA3:	**17b69de9188f4c205a00fe5ae9c1151f**
JA3ClientApplication:	**Powershell**

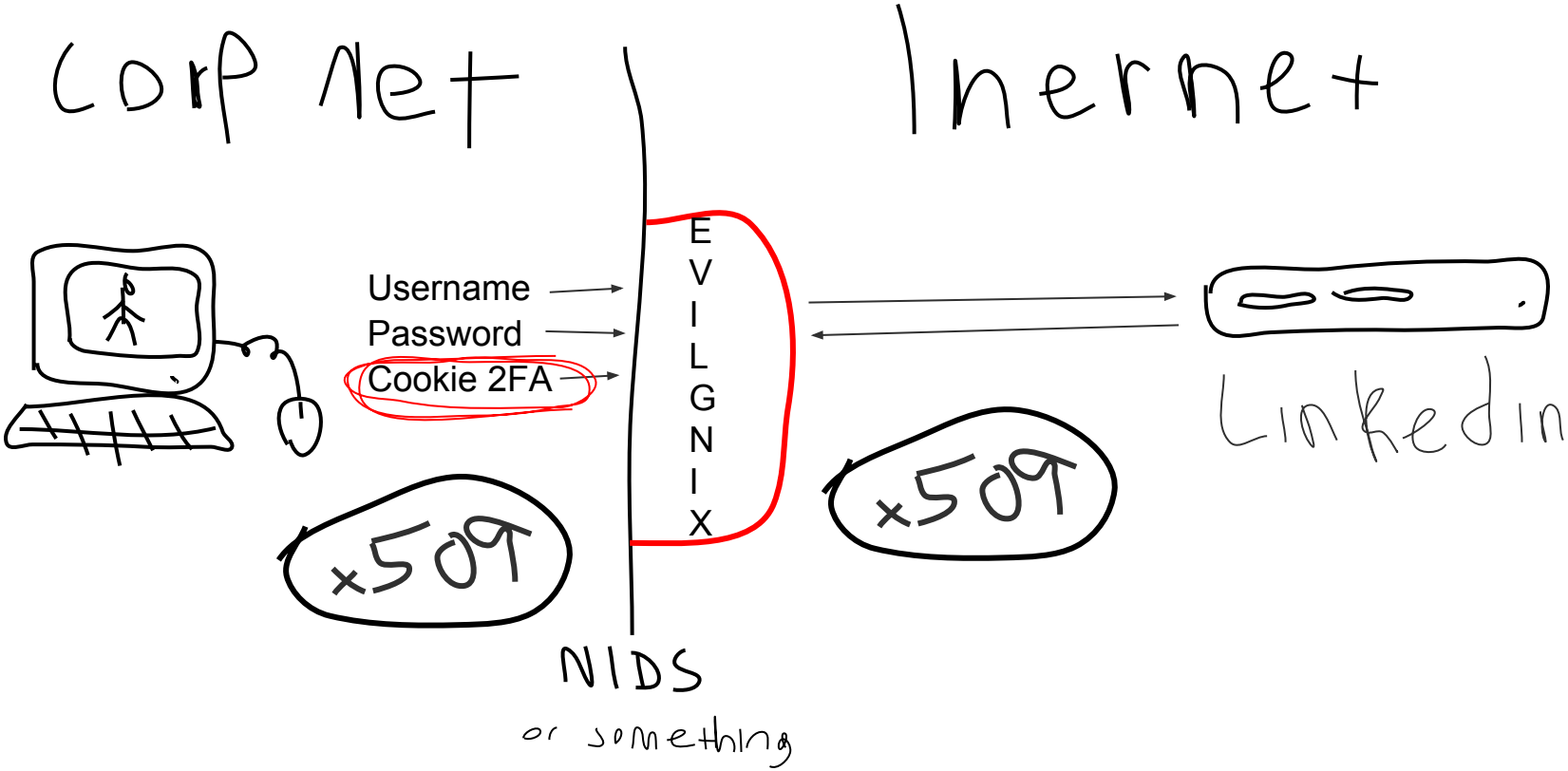
Enrich SSL/TLS Analysis

Evilginx - Phishing 2FA Tokens

Evilginx


Corp Net
Inernet
Username
Password
Cookie 2FA
EVILGNIX
x509
x509
LinkedIn
NIDS
or something

Evilginx


Corp Net
Inernet
Username
Password
Cookie 2FA
EVILGNIX
x509
x509
LinkedIn
NIDS
or something

Evilginx


no nginx - pure evil
by Kuba Gretzky (@mrgretzky) version 2.1.0
[03:46:32] [inf] loading phishlets from: /app/phishlets
[03:46:32] [inf] redirect parameter set to: rs
[03:46:32] [inf] verification parameter set to: xt
[03:46:32] [inf] verification token set to: ae9e
[03:46:32] [inf] unauthorized request redirection URL set to: https://www.youtube.com/watch?v=dQw4w9WgXcQ
[03:46:33] [inf] loaded phishlet 'amazon' made by @customsync from 'amazon.yaml'
[03:46:33] [inf] loaded phishlet 'facebook' made by @mrgretzky from 'facebook.yaml'
[03:46:33] [inf] loaded phishlet 'linkedin' made by @mrgretzky from 'linkedin.yaml'
[03:46:33] [inf] loaded phishlet 'outlook' made by @mrgretzky from 'outlook.yaml'
[03:46:33] [inf] loaded phishlet 'reddit' made by @customsync from 'reddit.yaml'
[03:46:33] [inf] loaded phishlet 'twitter-mobile' made by @white_fi from 'twitter-mobile.yaml'
[03:46:33] [inf] loaded phishlet 'twitter' made by @white_fi from 'twitter.yaml'
[03:46:33] [war] server domain not set! type: config domain <domain>
[03:46:33] [war] server ip not set! type: config ip <ip_address>
: config domain wat.phishing
[03:46:46] [inf] server domain set to: wat.phishing
[03:46:46] [war] server ip not set! type: config ip <ip_address>
: config ip 127.0.0.1
[03:46:52] [inf] server IP set to: 127.0.0.1
: phishlets hostname linkedin linkedin.wat.phishing
[03:47:06] [inf] phishlet 'linkedin' hostname set to: linkedin.wat.phishing
[03:47:06] [inf] disabled phishlet 'linkedin'
: phishlets enable linkedin
[03:47:16] [inf] enabled phishlet 'linkedin'
[03:47:16] [inf] developer mode is on - will use self-signed SSL/TLS certificates for phishlet 'linkedin'
: phishlets get-url linkedin https://www.linkedin.com
https://www.linkedin.wat.phishing/uas/login?xt=ae9e&rs=aHR0cHM6Ly93d3cubGlua2VkaW4uY29t
:

Evilginx


```
[03:29:35] [+++] [1] Password: [hello]
[03:29:35] [+++] [1] Username: [test@gmail.com]
: sessions

+-----+-----------+-----------------+----------+--------+-------------+---------------------+
| id  | phishlet  |    username     | password | tokens | remote ip   |        time         |
+-----+-----------+-----------------+----------+--------+-------------+---------------------+
| 1   | linkedin  |                 |          | none   | 172.17.0.1  | 2018-09-14 02:59    |
| 2   | linkedin  | test@gmail.com  | hello    | none   | 172.17.0.1  | 2018-09-14 03:29    |
+-----+-----------+-----------------+----------+--------+-------------+---------------------+
```

Evilginx

evilginx2 is a man-in-the-middle attack framework used for phishing login credentials along with session cookies, which in turn allows to bypass 2-factor authentication protection.

Released July and gaining momentum.

Written in GO, can be noisy.

JA3: **d3e1de2ca313c6c0a639f69cc3e924a4**

TODO:

- Need to combine with unencrypted URI for login page
- Need access to HTTP User Agents?

Remember:

Collisions can happen

There are OS APIs

JA3S
Fingerprinting Server Hellos

It’s a thing, but who’s got time for that today.

Conclusion

- JA3 is not a silver bullet
  - Collisions can happen
  - Applications can connect through OS APIs
  - There can be up to 5 JA3s for the same application
  - But it is always valuable as a pivot point for analysis
- JA3 is a silver bullet (sometimes)
  - Each environment is different
- JA3S adds even more context
- Please contribute and iterate
  - Let’s push the industry forward!

JA3 Support


splunk>
NGINX
ALIEN VAULT
VirusTotal
MISP
Threat Sharing
corelight
DARKTRACE
Nessus
Professional
ABUSE.ch
proofpoint
FireEye
python
Reservoir Labs
TRISUL
ICEBRG
GREY NOISE
SURICATA

Jeff Atkinson

neslog[at]gmail[dot]com
LinkedIn

https://github.com/salesforce/ja3

bro-pkg install ja3