# How many Mirai variants are there?

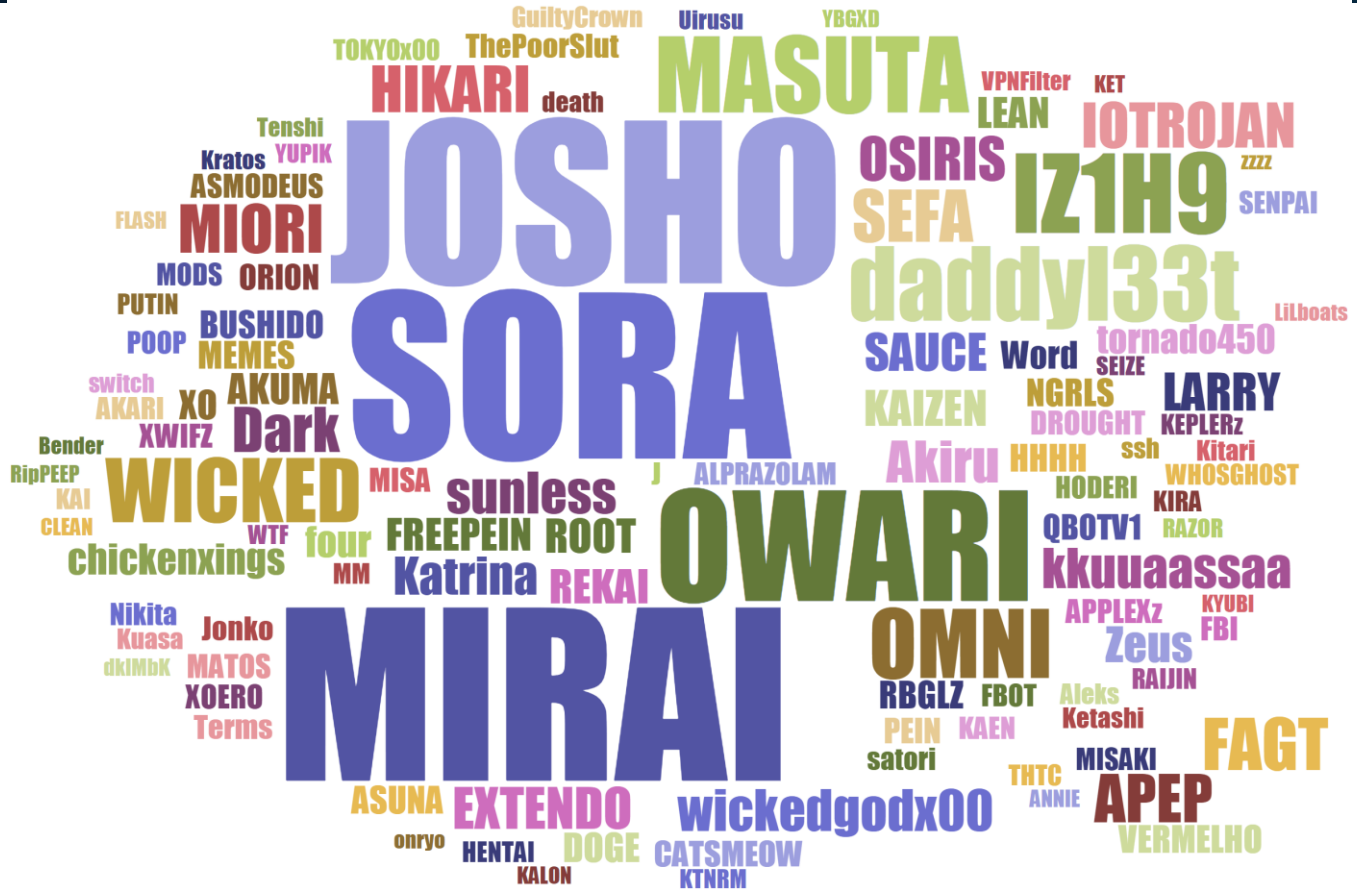
Ya Liu (*speaker*)

Hui Wang

# A short history

- Firstly analyzed by *@MalwareMustDie* in 2016-08
- Got known for crippling Krebsonsecurity, OVH, and DYN in autumn 2016
- Source code was released on Sep 30, 2016
- Some variants were also open sourced
  - e.g., MASUTA, OWARI, SORA, OMNI, …

# 116 branches from +21K samples




samples: 21,108

# The branch name

- An author-chosen command used in infection
  - *"/bin/busybox MIRAI"*
  - *"MIRAI: applet not found"*

- Later variant authors usually chose other meaningful words
  - *"/bin/busybox SORA"*
  - *"SORA: applet not found"*
  - *"/bin/busybox JOSHO"*
  - *"JOSHO: applet not found"*
  - *"/bin/busybox MASUTA"*
  - *"MASUTA: applet not found"*
  - *"/bin/busybox daddyl33t"*
  - *"daddyl33t: applet not found*

# Problems of branch based classification

- **Not accurate**: It's common that the same branch of samples vary a lot in features, e.g., supported attack methods

- **Confusing**: Other botnet family names (e.g., *zeus*, *QBOT*, *VPNFilter*) have been reused as branch names in some variants

- **Incomplete**: Not all samples include branch names

We suggest to classify Mirai samples based on Mirai genes

# The Mirai genes

- Encrypted configurations
  - A custom database storing running parameters of CNC, attack, scanner, killer, …
- Mirai-style attack methods
  - Starting with a large instruction block where attack options are parsed
  - To be installed to a table indexed with command codes
- (*Optional*) Telnet credentials and IoT exploits

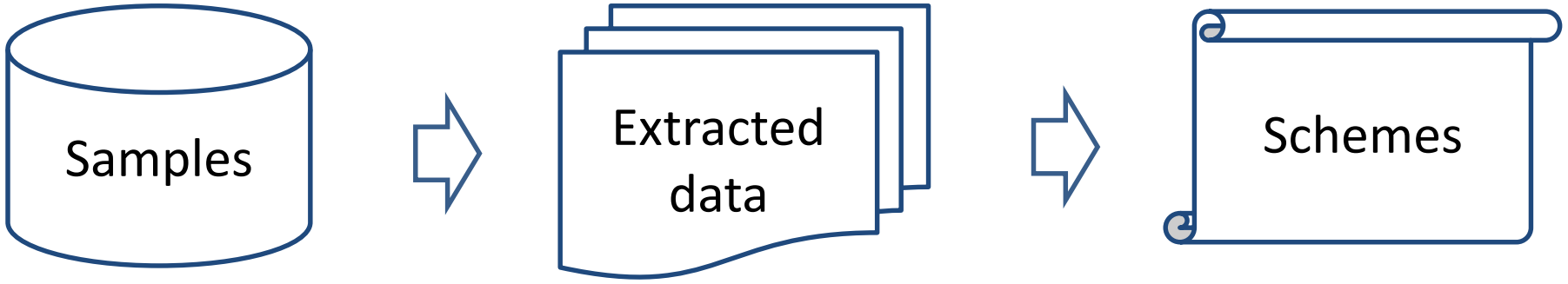
# Outline

- Background

- Data and methodology
  - Configuration
  - Supported attack methods

- Detailed analysis of branch *IZ1H9*
- Summary

# Our solution architecture




- **21,108** samples of x86 & ARM
- Configurations
- Attack methods
- **4** classification schemes

# Data extraction model

- Static analysis
  - To find target functions in sample

IDAPython

- Dynamic analysis
  - To emulate the found functions to obtain interested data

Unicorn

The ultimate CPU emulator

- Synthesis

# The default Mirai config (1/2)

```
[0x02]: "listening tun0\x00", size=15
[0x03]: "cnc.changeme.com", size=30
[0x04]: "\x00\x17", size=2
[0x05]: "https://youtu.be/dQw4w9WgXcQ\x00", size=29
[0x06]: "/proc/\x00", size=7
[0x07]: "/exe\x00", size=5
[0x08]: " (deleted)\x00", size=11
[0x09]: "/fd\x00", size=4
[0x0a]: ".anime\x00", size=7
[0x0b]: "/status\x00", size=8
[0x0c]: "REPORT %s:%s\x00", size=13
[0x0d]: "HTTPFLOOD\x00", size=10
[0x0e]: "LOLNOGTFO\x00", size=10
[0x0f]: "\x58\x4D\x4E\x4E\x43\x50\x46\x22\x00", size=33
[0x10]: "zollard\x00", size=8
[0x11]: "GETLOCALIP\x00", size=11
[0x12]: "report.changeme.com", size=29
[0x13]: "\xbb\xe5", size=2
[0x14]: "shell\x00", size=6
[0x15]: "enable\x00", size=7
[0x16]: "system\x00", size=7
[0x17]: "sh\x00", size=3
[0x18]: "/bin/busybox MIRAI\x00", size=19
[0x19]: "MIRAI: applet not found\x00", size=24
```

CNC with indexes of 3 and 4

killer

report with indexes of 0x12 and 0x13

scanner

# The default Mirai config (2/2)

scanner

```
[0x1a]: "ncorrect\x00", size=9
[0x1b]: "/bin/busybox ps\x00", size=16
[0x1c]: "/bin/busybox kill -9 \x00", size=22
```

attack

```
[0x1d]: "TSource Engine Query\x00", size=21
[0x1e]: "/etc/resolv.conf\x00", size=17
[0x1f]: "nameserver \x00", size=12
[0x20]: "Connection: keep-alive\x00", size=23
[0x21]: "Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\x00", size=83
[0x22]: "Accept-Language: en-US,en;q=0.8\x00", size=32
[0x23]: "Content-Type: application/x-www-form-urlencoded\x00", size=48
[0x24]: "setCookie('\x00", size=12
[0x25]: "refresh:\x00", size=9
[0x26]: "location:\x00", size=10
[0x27]: "set-cookie:\x00", size=12
[0x28]: "content-length:\x00", size=16
[0x29]: "transfer-encoding:\x00", size=19
[0x2a]: "chunked\x00", size=8
[0x2b]: "keep-alive\x00", size=11
[0x2c]: "connection:\x00", size=12
[0x2d]: "server: dosarrest\x00", size=18
[0x2e]: "server: cloudflare-nginx\x00", size=25
[0x2f]: "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103 Safari/537.36\x00", size=111
[0x30]: "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/52.0.2743.116 Safari/537.36\x00", size=111
[0x31]: "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103 Safari/537.36\x00", size=110
[0x32]: "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/52.0.2743.116 Safari/537.36\x00", size=110
[0x33]: "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/601.7.7 (KHTML, like Gecko) Version/9.1.2 Safari/601.7.7\x00", size=117
```

# Configuration related functions

- table_init() : to install the cipher-text items when bot starts running

- table_unlock_val()/table_retrieve_val() /table_lock_val() : to be consecutively called when referencing a config item

```
table_unlock_val(TABLE_CNC_DOMAIN);
entries = resolv_lookup(table_retrieve_val(TABLE_CNC_DOMAIN, NULL));
table_lock_val(TABLE_CNC_DOMAIN);
```

- Items will resume to cipher-text state after using

# About table_init()

```
void table_init(void)
{
    add_entry(TABLE_CNC_DOMAIN, "\x41\x4C\x41\x0C\x41\x4A\x43\x4C\
    add_entry(TABLE_CNC_PORT, "\x22\x35", 2);    // 23

    add_entry(TABLE_SCAN_CB_DOMAIN, "\x50\x47\x52\x4D\x50\x56\x0C\
    add_entry(TABLE_SCAN_CB_PORT, "\x99\xC7", 2);          // 48101
```

index

cipher-text

item-size

```
static void add_entry(uint8_t id, char *buf, int buf_len)
{
    char *cpy = malloc(buf_len);

    util_memcpy(cpy, buf, buf_len);

    table[id].val = cpy;
    table[id].val_len = (uint16_t)buf_len;
#ifdef DEBUG
    table[id].locked = TRUE;
#endif
}
```

cipher text is copied to a new memory block

config DB

slot is determined by the index

# Binary table_init()

## A function with a single and big instruction block

```
table_init proc near

var_1C= dword ptr -1Ch

push    ebx
sub     esp, 14h
push    11h
call    malloc
add     esp, 0Ch
mov     ebx, eax
push    11h
push    offset unk_8050E20
push    eax
call    util_memcpy
mov     ds:dword_80526E8, ebx
mov     [esp+1Ch+var_1C], 2
mov     ds:word_80526EC, 11h
call    malloc
add     esp, 0Ch
mov     ebx, eax
push    2
push    offset unk_8050E32
push    eax
call    util_memcpy
mov     ds:dword_80526F0, ebx
mov     [esp+1Ch+var_1C], 11h
mov     ds:word_80526F4, 2
call    malloc
add     esp, 0Ch
mov     ebx, eax
push    11h
push    offset unk_8050E20
push    eax
call    util_memcpy
mov     ds:dword_80526F8, ebx
mov     [esp+1Ch+var_1C], 2
mov     ds:word_80526EC, 11h
call    malloc
add     esp, 0Ch
mov     ebx, eax
push    2
```

Repeatedly calling malloc/util_memcpy to save individual configuration items

```
push    11h
call    malloc
add     esp, 0Ch
mov     ebx, eax
push    11h
push    offset unk_8050E20
push    eax
call    util_memcpy
mov     ds:dword_80526E8, ebx
```

item size

cipher text address

slot address

# Recovering indexes and key

- The initial result is an array of {item_addr, cipher-text, size}
- Key is brute-force searched in the space of 1~256
- Indexes are calculated based on item addresses
  - item_index=(item_addr-table_addr)/8
- The final result is an array of {index, plain-text, size}

# Configuration example 1

```
[0x01]: "dCRAvvNuE1O5H18jX9TU\x00", size=21
[0x02]: "shell\x00", size=6
[0x03]: "enable\x00", size=7
[0x04]: "system\x00", size=7
[0x05]: "sh\x00", size=3
[0x06]: "/bin/busybox IZ1H9\x00", size=19
[0x07]: "IZ1H9: applet not found\x00", size=24
[0x08]: "ncorrect\x00", size=9
[0x09]: "assword\x00", size=8
[0x0a]: "ogin\x00", size=5
[0x0b]: "enter\x00", size=6
[0x0c]: "POST /ctrlt/DeviceUpgrade_1 HTTP/1.1\x00", size=37
[0x0d]: "Content-Length: 430\x00", size=20
[0x0e]: "Connection: keep-alive\x00", size=23
[0x0f]: "Accept: */*\x00", size=12
[0x10]: "<?xml version="1.0" ?><s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" s:encodingStyle="http://schemas.xmlsoap.org/soap/
encoding/"><s:Body><u:Upgrade xmlns:u="urn:schemas-upnp-org:service:WANPPPConnection:1"><NewStatusURL>\x00", size=238
[0x11]: "</NewStatusURL><NewDownloadURL>$(echo HUAWEIUPNP)</NewDownloadURL></u:Upgrade></s:Body></s:Envelope>\x00", size=101
[0x13]: "/proc/\x00", size=7
[0x14]: "/exe\x00", size=5
[0x15]: "/fd\x00", size=4
[0x16]: "/maps\x00", size=6
[0x17]: "/proc/net/tcp\x00", size=14
[0x18]: "dvrHelper\x00", size=10
[0x19]: "TSource Engine Query\x00", size=21
[0x1a]: "/etc/resolv.conf\x00", size=17
[0x1b]: "nameserver\x00", size=11
[0x1c]: "/dev/watchdog\x00", size=14
[0x1d]: "/dev/misc/watchdog\x00", size=19
[0x1e]: "/dev/FTWDT101_watchdog\x00", size=23
[0x1f]: "/dev/FTWDT101\ watchdog\x00", size=24
[0x20]: "abcdefghijklmnopqrstuvwxyz1234567890\x00", size=37
```

branch: IZ1H9

- 31 items in total
- No CNC
- No report server
- No HTTP agents

Exploits related configuration

MD5=0407a5c2d4d2afaff91c14b63aaa668c

# Configuration example 2 (1/2)

```
[0x01]: "\x91\xfc", size=2
[0x02]: "\xdc\xf4", size=2
[0x03]: "dCRAvvNuE1O5Hl8jX9TU\x00", size=21
[0x04]: "shell\x00", size=6
[0x05]: "enable\x00", size=7
[0x06]: "system\x00", size=7
[0x07]: "sh\x00", size=3
[0x08]: "/bin/busybox IZ1H9\x00", size=19
[0x09]: "IZ1H9: applet not found\x00", size=24
[0x0a]: "ncorrect\x00", size=9
[0x0b]: "assword\x00", size=8
[0x0c]: "ogin\x00", size=5
[0x0d]: "enter\x00", size=6
[0x0e]: "POST /ctrlt/DeviceUpgrade_1 HTTP/1.1\x00", size=37
[0x0f]: "Content-Length: 430\x00", size=20
[0x10]: "Connection: keep-alive\x00", size=23
[0x11]: "Accept: */*\x00", size=12
[0x13]: "<?xml version="1.0" ?><s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" s:encodingStyle="http://schemas.xmlsoa
p.org/soap/encoding/"><s:Body><u:Upgrade xmlns:u="urn:schemas-upnp-org:service:WANPPPConnection:1"><NewStatusURL>\x00", size=238
[0x14]: "</NewStatusURL><NewDownloadURL>$(echo HUAWEIUPNP)</NewDownloadURL></u:Upgrade></s:Body></s:Envelope>\x00", size=101
[0x15]: "/proc/\x00", size=7
[0x16]: "/exe\x00", size=5
[0x17]: "/fd\x00", size=4
[0x18]: "/maps\x00", size=6
[0x19]: "/proc/net/tcp\x00", size=14
[0x1a]: "UPX!\x00", size=5
[0x1a]: "dvrHelper\x00", size=10
[0x1b]: "foAxi102kxe\x00", size=12
[0x1c]: "yakuv4vxc\x00", size=10
[0x1d]: "mdeCrtCDRcdr\x00", size=13
[0x1e]: "X19I239124UIU\x00", size=14
```

branch: IZ1H9

- No CNC and report server
- No HTTP agents
- 62 items
- more killer parameters

Exploits related configuration

Killer related configs

MD5=5db7c47a33bfec2574af94c0b6a50cbe

# Configuration example 2 (2/2)

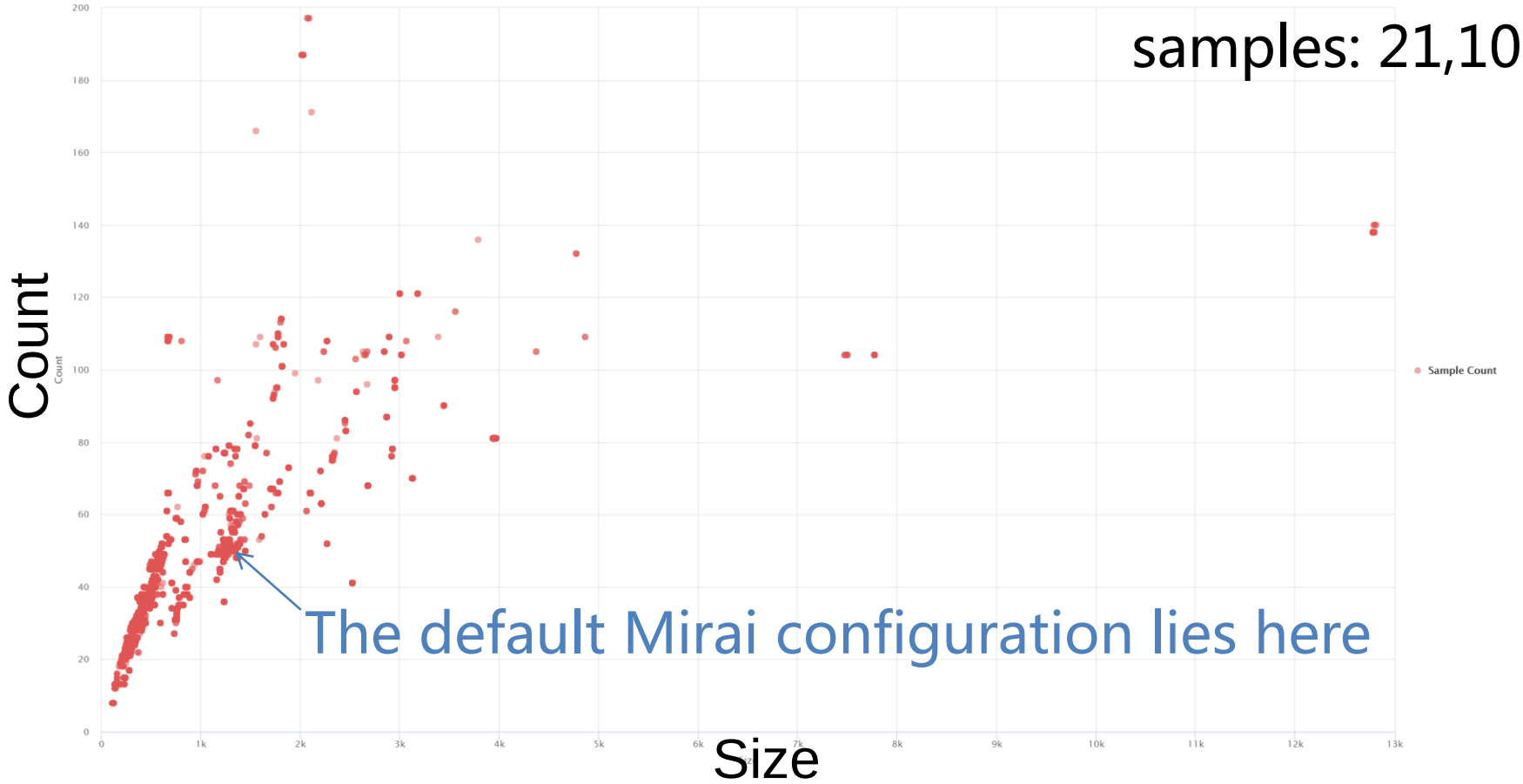
```
[0x1f]: "OaF3\x00", size=5
[0x20]: "SAIAKINA\x00", size=9
[0x21]: "WsGA4@F6F\x00", size=10
[0x22]: "19ju3d\x00", size=7
[0x23]: "NiGGeR69xd\x00", size=11
[0x24]: "BoatGangTsuki\x00", size=14
[0x25]: "0x766f6964\x00", size=11
[0x26]: "93OfjHZ2z\x00", size=10
[0x27]: "IuYgujeIqn\x00", size=11
[0x28]: "frgege\x00", size=7
[0x29]: "poilkjmnb\x00", size=10
[0x2a]: "elfLoad\x00", size=8
[0x2b]: "AbAd\x00", size=5
[0x2c]: "HOHO-U79OL\x00", size=11
[0x2d]: "IuYgujeIqn\x00", size=11
[0x2e]: "BzSxLxBxeY\x00", size=11
[0x2f]: "ccAD\x00", size=5
[0x30]: "Katrina32\x00", size=10
[0x31]: "SlaVLav12\x00", size=10
[0x32]: "5aA3\x00", size=5
[0x33]: "0DnAzepd\x00", size=9
[0x34]: "mioribitches\x00", size=13
[0x35]: "QBotBladeSPOOKY\x00", size=16
[0x36]: "OnrYoXd666\x00", size=11
[0x37]: "TSource Engine Query\x00", size=21
[0x38]: "/etc/resolv.conf\x00", size=17
[0x39]: "nameserver\x00", size=11
[0x3a]: "/dev/watchdog\x00", size=14
[0x3b]: "/dev/misc/watchdog\x00", size=19
[0x3c]: "/dev/FTWDT101_watchdog\x00", size=23
[0x3d]: "/dev/FTWDT101\ watchdog\x00", size=24
[0x3e]: "abcdefghijklmnopqrstuvwxyz1234567890\x00", size=37
```

More killer related items

# How to use configuration for classification?

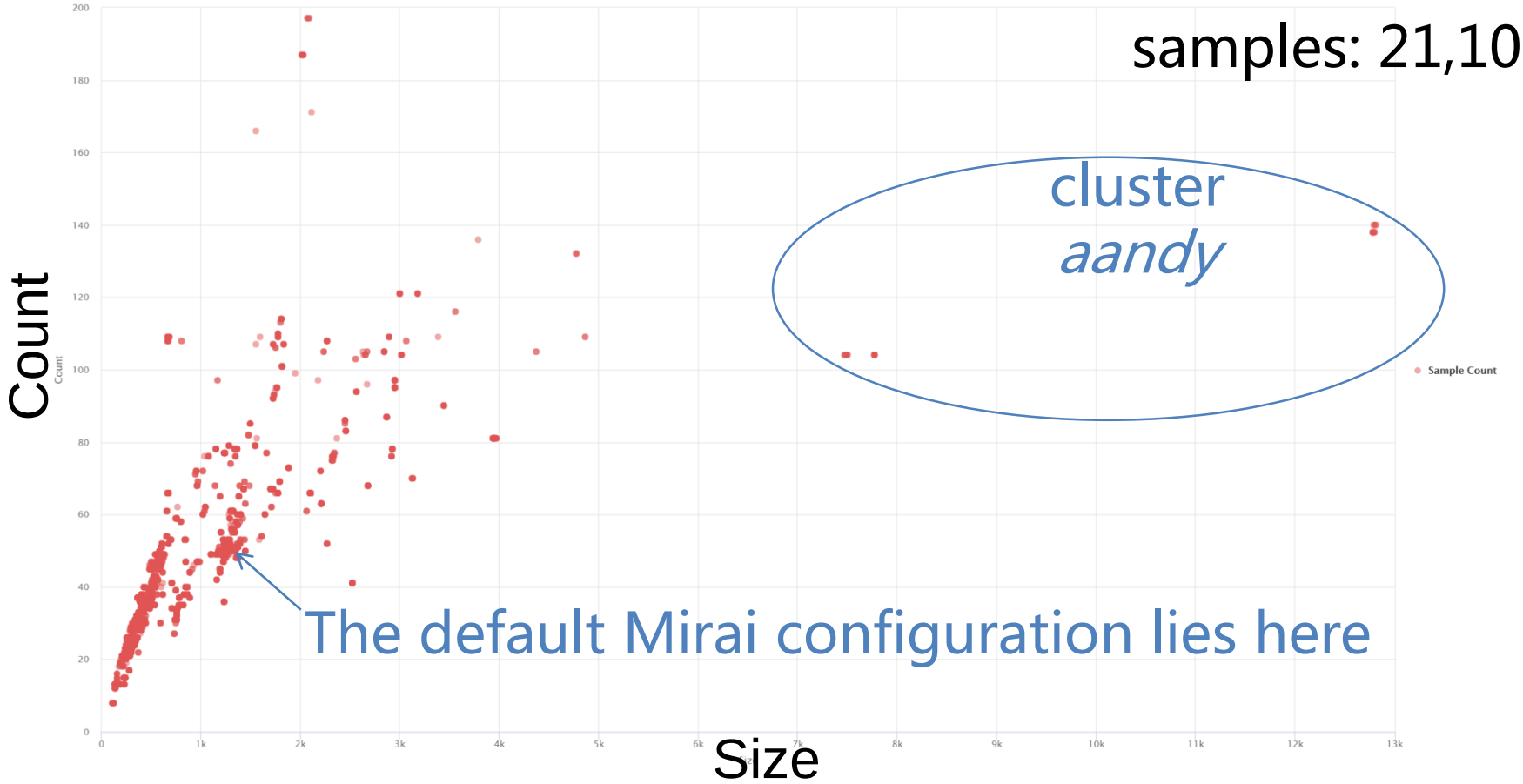
- There is too much useful information
  - E.g., item count, indexes, initialization order, item value, keys, semantics, …
- Considerations of scalability and universality
- 2 schemes to be introduced
  - Clustering samples based on config count/size
  - Classification based on encryption key

# Scheme-1: clustering samples based on configuration count and size



# Scheme-1: clustering samples based on configuration count and size

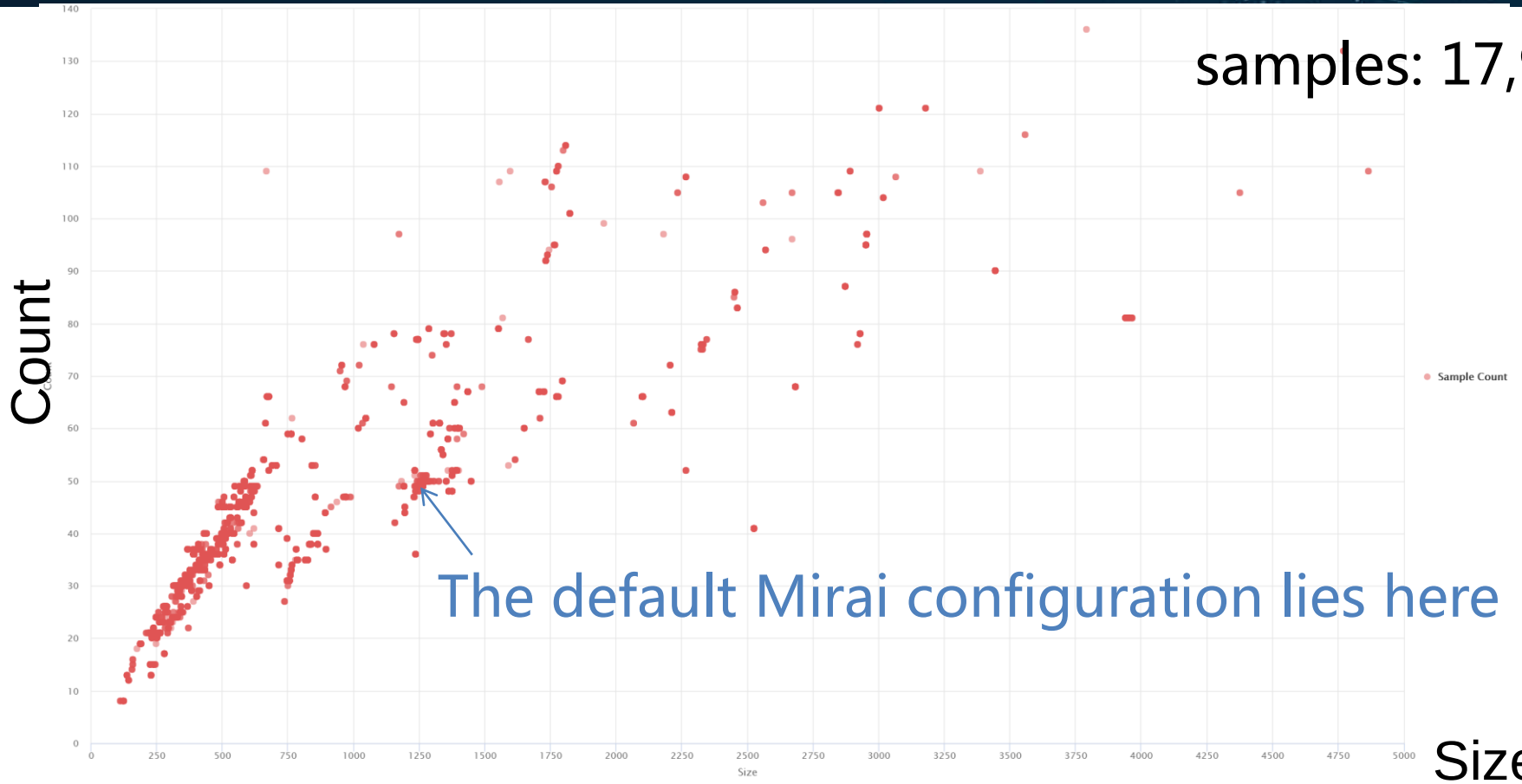
samples: 21,108



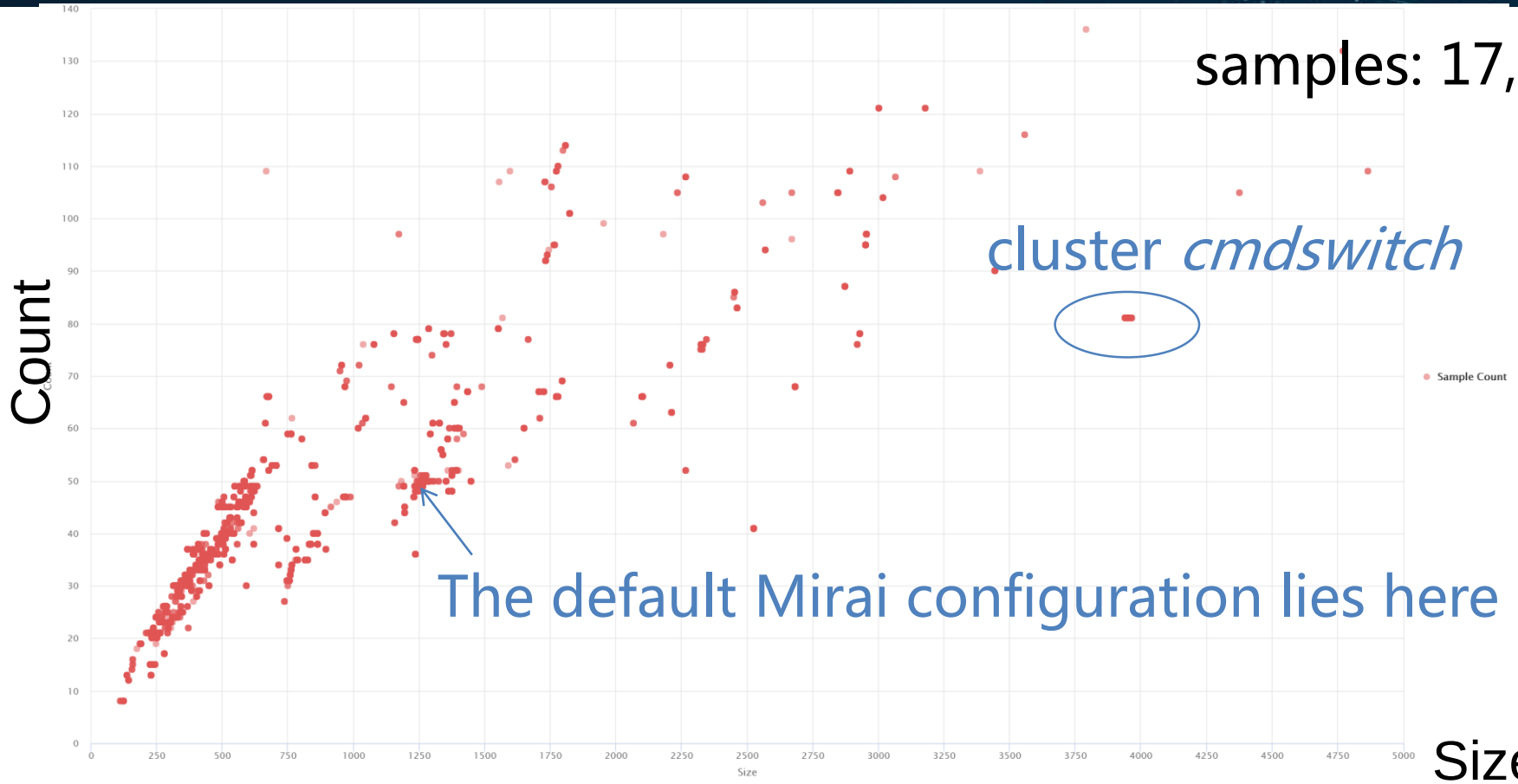
# Cluster *aandy*

| Branch name | Key | C2 | Samples |
|---|---|---|---|
| KYUBI | 0x34 | cnc.aandy.xyz | 4 |
| MIRAI | 0x34 | cnc.aandy.xyz | 8 |
| MIRAI | 0x34 | www.aandy.cf | 7 |
| MIRAI | 0x34 | www.askjasghasg.ru | 16 |
| | | 107.179.126.64 | |
| MIRAI | 0x22 | cnc.ttoww.com | 13 |

# Scheme-1 on samples emerged in 2018



# Scheme-1 on samples emerged in 2018



# Cluster *cmdswitch*

- Samples: 63
- C2 servers: 12
- Branches: *MIRAI* and ORION

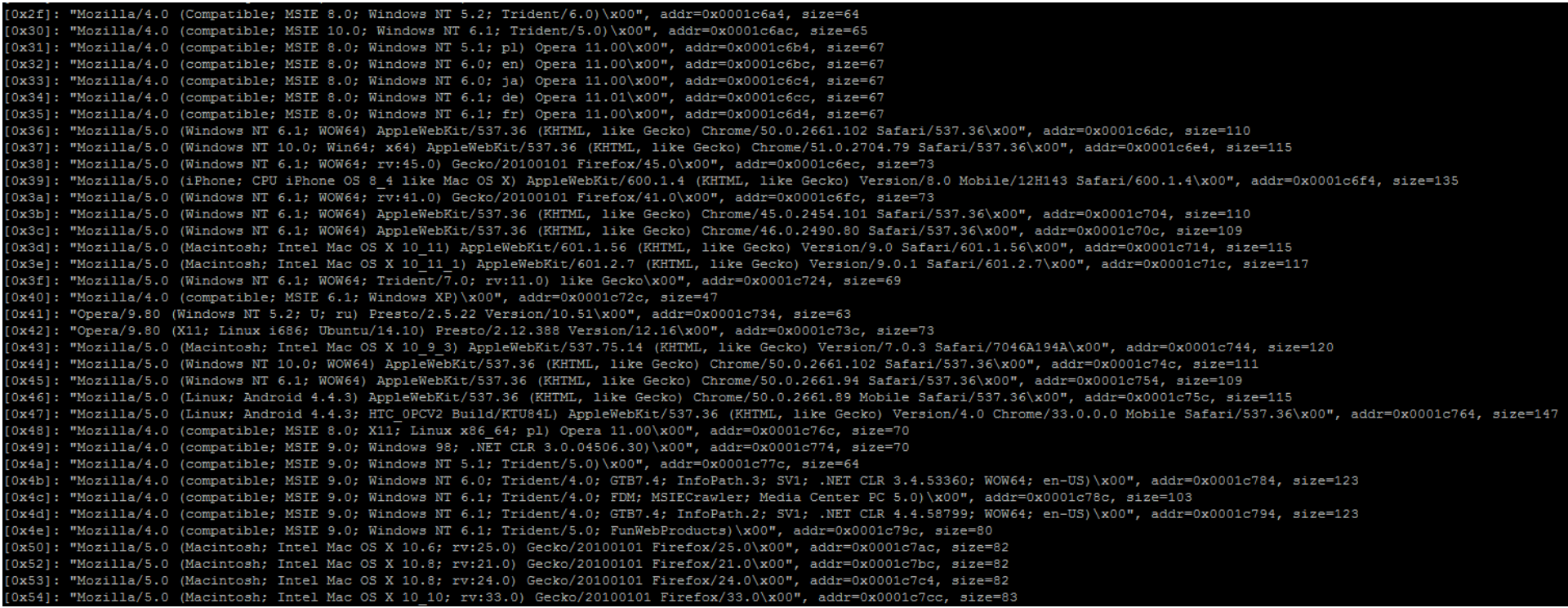
```
206.189.208.233
4ina.fastwars.ru
54.36.10.66
c0nr01ler.mirabot.top
center.cmdswitch.pw
center.cmdswitch.xyz
cmd.hubsg.net
cmd.spai3n.ru
cnc.bigboats.cf
cnc.mirabot.top
cnc.miraibot.top
help.d3ever.ml
```

**12** unique C2 servers

```
[0x12]: "206.189.208.233\x00", size=16
[0x12]: "54.36.10.66\x00\xd223\x00", size=16
[0x12]: "cnc.bigboats.cf\x00", size=16
[0x12]: "fp49dqklufsophuossx.mirabot.top\x00", size=32
[0x12]: "plusrepo4.fastwars.ru\x00", size=22
[0x12]: "rep.cmdswitch.pw\x00", size=17
[0x12]: "rep.cmdswitch.xyz\x00", size=18
[0x12]: "rep.mirabot.top\x00", size=16
[0x12]: "rep.miraibot.top\x00", size=17
[0x12]: "report.spai3n.ru\x00", size=17
[0x12]: "rep.spai3n.ru\x00", size=14
```

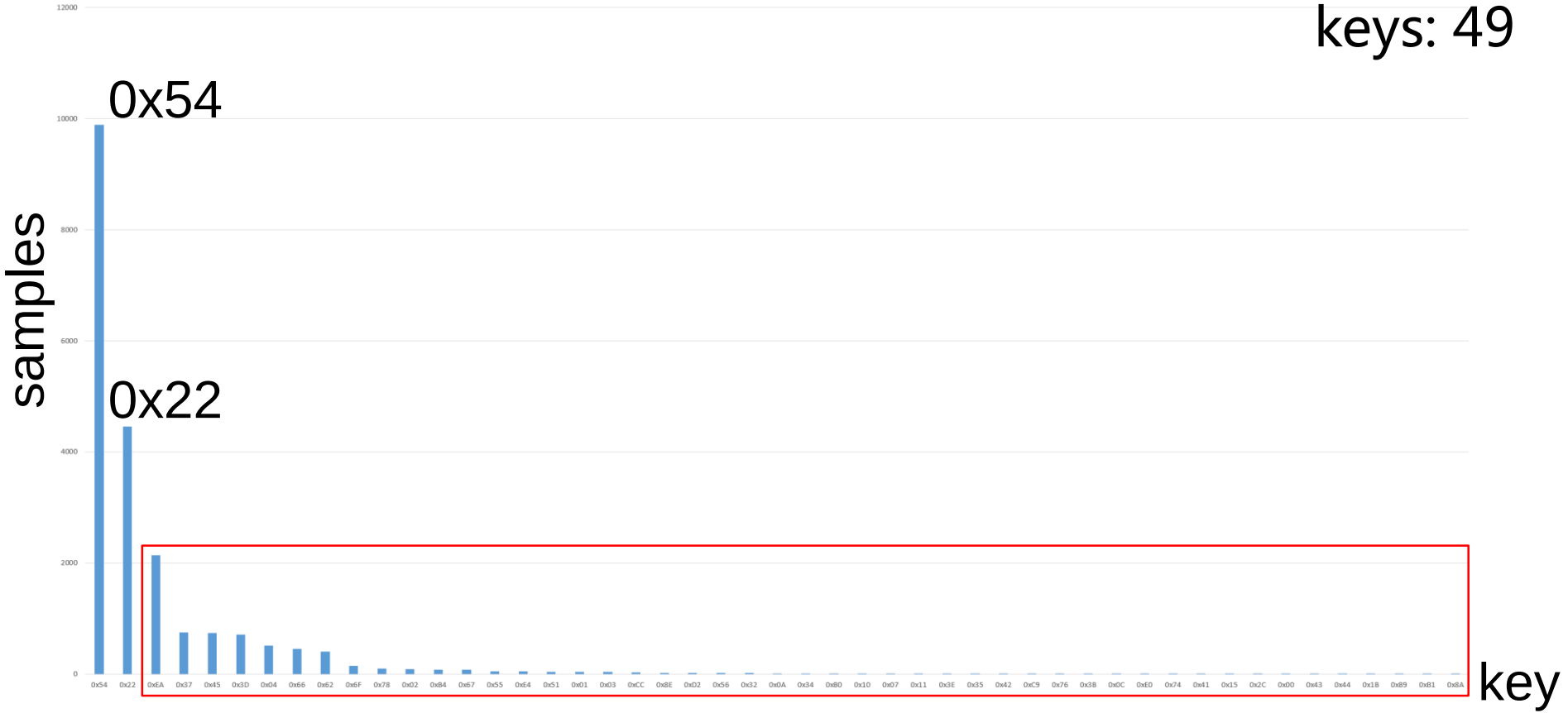
The item of index **0x12** points to a rep server

# ~36 HTTP agents in *cmdswitch* samples



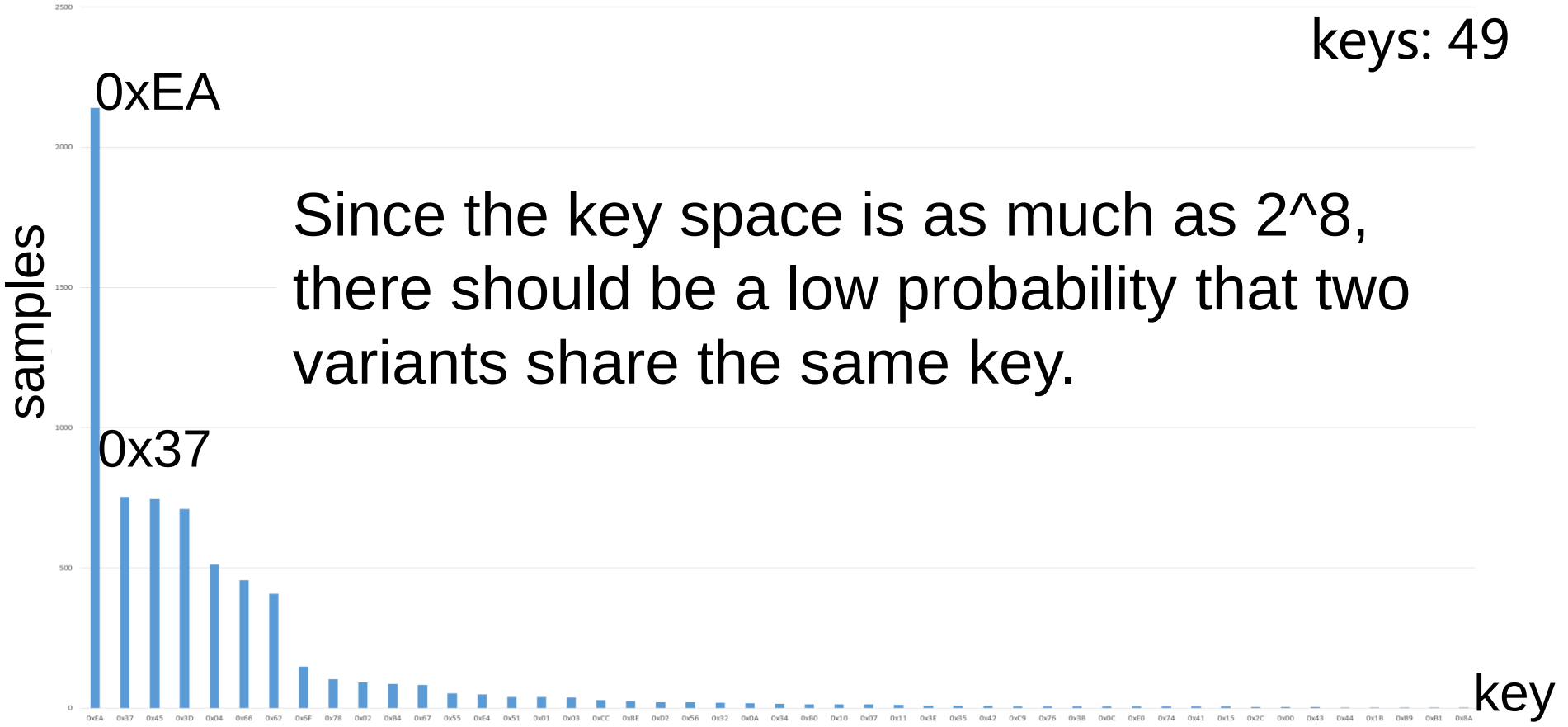
# Scheme-2: key based classification

keys: 49



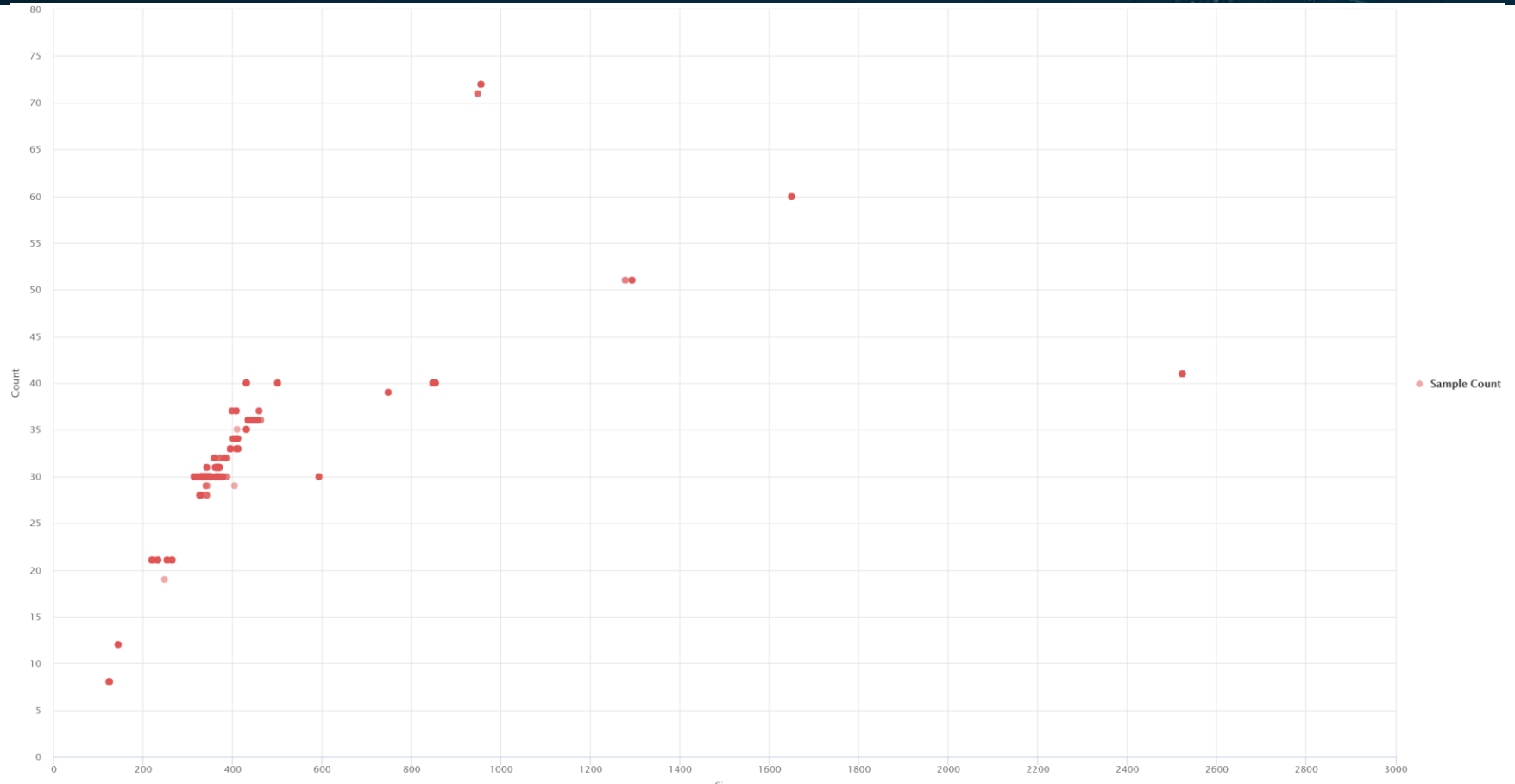
# Scheme-2: key based classification

keys: 49

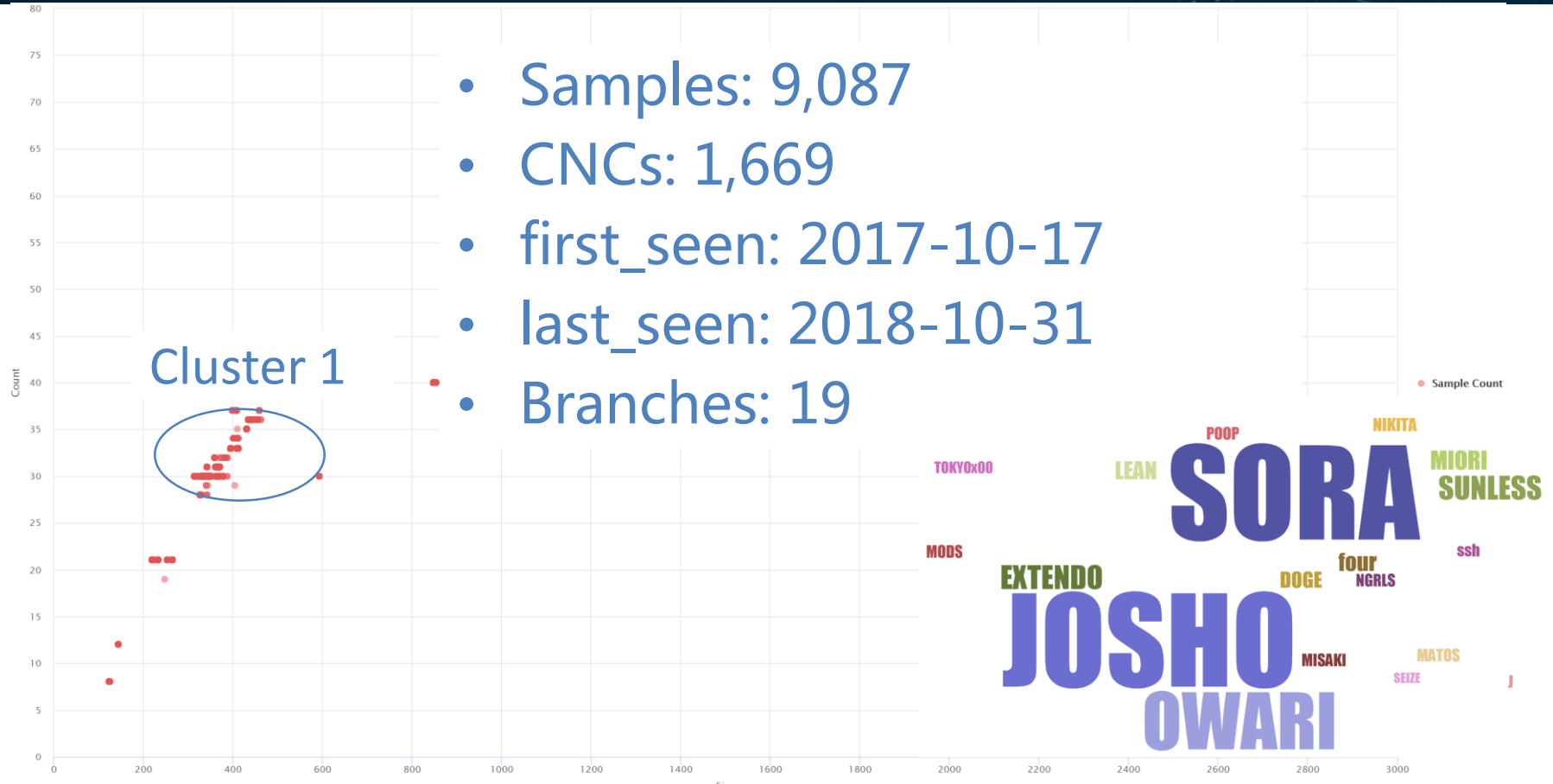



Since the key space is as much as 2^8, there should be a low probability that two variants share the same key.

# Samples of key 0x54 in scheme-1



# Samples of key 0x54 in scheme-1




- Samples: 9,087
- CNCs: 1,669
- first_seen: 2017-10-17
- last_seen: 2018-10-31
- Branches: 19

# Configurations of cluster 1

```
[0x01]: "\xff\xff", addr=0x0001d26c, size=2 [0x01]: "\x00-", addr=0x000209e8, size=2
[0x02]: "\x00U", addr=0x0001d274, size=2    [0x02]: "\x87\x98", addr=0x000209f0, size=2
[0x03]: "sunless infected your router\x00", [0x03]: "Connected To CNC\x00", addr=0x000209f8, size=17
[0x04]: "shell\x00", addr=0x0001d284, size=6[0x04]: "shell\x00", addr=0x00020a00, size=6
[0x05]: "enable\x00", addr=0x0001d28c, size=[0x05]: "enable\x00", addr=0x00020a08, size=7
[0x06]: "system\x00", addr=0x0001d294, size=[0x06]: "system\x00", addr=0x00020a10, size=7
[0x07]: "sh\x00", addr=0x0001d29c, size=3   [0x07]: "sh\x00", addr=0x00020a18, size=3
[0x08]: "/bin/busybox sunless\x00", addr=0x0[0x08]: "/bin/busybox SORA\x00", addr=0x00020a20, size=18
[0x09]: "sunless: applet not found\x00", add[0x09]: "SORA: applet not found\x00", addr=0x00020a28, size=23
[0x0a]: "ncorrect\x00", addr=0x0001d2b4, siz[0x0a]: "ncorrect\x00", addr=0x00020a30, size=9
[0x0b]: "/bin/busybox ps\x00", addr=0x0001d2[0x0b]: "/bin/busybox ps\x00", addr=0x00020a38, size=16
[0x0c]: "/bin/busybox kill -9 \x00", addr=0x[0x0c]: "/bin/busybox kill -9 \x00", addr=0x00020a40, size=22
[0x0d]: "/proc/\x00", addr=0x0001d2cc, size=[0x0d]: "/proc/\x00", addr=0x00020a48, size=7
[0x0e]: "/exe\x00", addr=0x0001d2d4, size=5 [0x0e]: "/exe\x00", addr=0x00020a50, size=5
[0x0f]: "/fd\x00", addr=0x0001d2dc, size=4  [0x0f]: "/fd\x00", addr=0x00020a58, size=4
[0x10]: "/maps\x00", addr=0x0001d2e4, size=6[0x10]: "/maps\x00", addr=0x00020a60, size=6
[0x11]: "/proc/net/tcp\x00", addr=0x0001d2ec[0x11]: "/proc/net/tcp\x00", addr=0x00020a68, size=14
[0x12]: "/proc/net/route\x00", addr=0x0001d2[0x12]: "/proc/net/route\x00", addr=0x00020a70, size=16
[0x13]: "assword\x00", addr=0x0001d2fc, size[0x13]: "assword\x00", addr=0x00020a78, size=8
[0x14]: "TSource Engine Query\x00", addr=0x0[0x14]: "TSource Engine Query\x00", addr=0x00020a80, size=21
[0x15]: "/etc/resolv.conf\x00", addr=0x0001d[0x15]: "/etc/resolv.conf\x00", addr=0x00020a88, size=17
[0x16]: "nameserver \x00", addr=0x0001d314, [0x16]: "nameserver \x00", addr=0x00020a90, size=12
[0x17]: "/dev/watchdog\x00", addr=0x0001d31c[0x17]: "/dev/watchdog\x00", addr=0x00020a98, size=14
[0x18]: "/dev/misc/watchdog\x00", addr=0x000[0x18]: "/dev/misc/watchdog\x00", addr=0x00020aa0, size=19
[0x19]: "assword\x00", addr=0x0001d32c, size[0x19]: "assword\x00", addr=0x00020aa8, size=8
[0x1a]: "ogin\x00", addr=0x0001d334, size=5 [0x1a]: "ogin\x00", addr=0x00020ab0, size=5
[0x1b]: "enter\x00", addr=0x0001d33c, size=6[0x1b]: "enter\x00", addr=0x00020ab8, size=6
[0x1c]: "1gba4cdom53nhp12ei0kfj\x00", addr=0[0x1c]: "1gba4cdom53nhp12ei0kfj\x00", addr=0x00020ac0, size=23
[0x1d]: "/status\x00", addr=0x0001d34c, size[0x1d]: "/status\x00", addr=0x00020ac8, size=8
[0x1e]: ".anime\x00", addr=0x0001d354, size=[0x1e]: ".anime\x00", addr=0x00020ad0, size=7
[0x20]: "sl.sunlessmods.xyz\x00", addr=0x0001d364, size=19
```

# Configurations of cluster 1

```
[0x01]: "\x06\x07", addr=0x0001db2c, size=2
[0x02]: "\x10\x06", addr=0x0001db34, size=2
[0x03]: "OWARI09123id9i123xd912\x00", addr=0x0001db
[0x04]: "shell\x00", addr=0x0001db44, size=6
[0x05]: "enable\x00", addr=0x0001db4c, size=7
[0x06]: "system\x00", addr=0x0001db54, size=7
[0x07]: "sh\x00", addr=0x0001db5c, size=3
[0x08]: "/bin/busybox OWARI\x00", addr=0x0001db64,
[0x09]: "OWARI: applet not found\x00", addr=0x0001d
[0x0a]: "ncorrect\x00", addr=0x0001db74, size=9
[0x0b]: "/bin/busybox ps\x00", addr=0x0001db7c, siz
[0x0c]: "/bin/busybox kill -9 \x00", addr=0x0001db8
[0x0d]: "/proc/\x00", addr=0x0001db8c, size=7
[0x0e]: "/exe\x00", addr=0x0001db94, size=5
[0x0f]: "/fd\x00", addr=0x0001db9c, size=4
[0x10]: "/maps\x00", addr=0x0001dba4, size=6
[0x11]: "/proc/net/tcp\x00", addr=0x0001dbac, size=
[0x12]: "/proc/net/route\x00", addr=0x0001dbb4, siz
[0x13]: "assword\x00", addr=0x0001dbbc, size=8
[0x14]: "TSource Engine Query\x00", addr=0x0001dbc4
[0x15]: "/etc/resolv.conf\x00", addr=0x0001dbcc, si
[0x16]: "nameserver \x00", addr=0x0001dbd4, size=12
[0x17]: "/dev/watchdog\x00", addr=0x0001dbdc, size=
[0x18]: "/dev/misc/watchdog\x00", addr=0x0001dbe4,
[0x19]: "assword\x00", addr=0x0001dbec, size=8
[0x1a]: "ogin\x00", addr=0x0001dbf4, size=5
[0x1b]: "enter\x00", addr=0x0001dbfc, size=6
[0x1c]: "9u123448u124au814d4x10\x00", addr=0x0001dc
[0x1d]: "/status\x00", addr=0x0001dc0c, size=8
[0x1e]: ".anime\x00", addr=0x0001dc14, size=7
[0x20]: "sl.sunlessmods.xyz\x00", addr=0x0001d364,
```

```
[0x01]: "\x00A", addr=0x0001e26c, size=2
[0x02]: "u0", addr=0x0001e274, size=2
[0x03]: "im an xbox modder lol\x00", addr=0x0001e27c, size=22
[0x04]: "shell\x00", addr=0x0001e284, size=6
[0x05]: "enable\x00", addr=0x0001e28c, size=7
[0x06]: "system\x00", addr=0x0001e294, size=7
[0x07]: "sh\x00", addr=0x0001e29c, size=3
[0x08]: "/bin/busybox LEAN\x00", addr=0x0001e2a4, size=18
[0x09]: "LEAN: applet not found\x00", addr=0x0001e2ac, size=23
[0x0a]: "ncorrect\x00", addr=0x0001e2b4, size=9
[0x0b]: "/bin/busybox ps\x00", addr=0x0001e2bc, size=16
[0x0c]: "/bin/busybox kill -9 \x00", addr=0x0001e2c4, size=22
[0x0d]: "/proc/\x00", addr=0x0001e2cc, size=7
[0x0e]: "/exe\x00", addr=0x0001e2d4, size=5
[0x0f]: "/fd\x00", addr=0x0001e2dc, size=4
[0x10]: "/maps\x00", addr=0x0001e2e4, size=6
[0x11]: "/proc/net/tcp\x00", addr=0x0001e2ec, size=14
[0x12]: "/proc/net/route\x00", addr=0x0001e2f4, size=16
[0x13]: "assword\x00", addr=0x0001e2fc, size=8
[0x14]: "TSource Engine Query\x00", addr=0x0001e304, size=21
[0x15]: "/etc/resolv.conf\x00", addr=0x0001e30c, size=17
[0x16]: "nameserver \x00", addr=0x0001e314, size=12
[0x17]: "/dev/watchdog\x00", addr=0x0001e31c, size=14
[0x18]: "/dev/misc/watchdog\x00", addr=0x0001e324, size=19
[0x19]: "assword\x00", addr=0x0001e32c, size=8
[0x1a]: "ogin\x00", addr=0x0001e334, size=5
[0x1b]: "enter\x00", addr=0x0001e33c, size=6
[0x1c]: "9u123448u124au814d4x10\x00", addr=0x0001e344, size=23
[0x1d]: "/status\x00", addr=0x0001e34c, size=8
[0x1e]: ".anime\x00", addr=0x0001e354, size=7
```

# Outline

- Background

- Data and methodology
  - Configuration
  - Supported attack methods

- Detailed analysis of branch IZ1H9
- Summary

# Supported attack methods

- It's reasonable to classify variants of a DDoS attacking purposed botnet family based on their supported attack methods

- Mirai variants did vary a lot in attack methods
  - 10 attack methods were found in the firstly released code
  - Dozens of new methods have been detected in later variants

# Attack method initialization

```
BOOL attack_init(void)
{
    int i;

    add_attack(ATK_VEC_UDP,       (ATTACK_FUNC)attack_udp_generic);
    add_attack(ATK_VEC_VSE,       (ATTACK_FUNC)attack_udp_vse);
    add_attack(ATK_VEC_DNS,       (ATTACK_FUNC)attack_udp_dns);
    add_attack(ATK_VEC_UDP_PLAIN, (ATTACK_FUNC)attack_udp_plain);

    add_attack(ATK_VEC_SYN,       (ATTACK_FUNC)attack_tcp_syn);
    add_attack(ATK_VEC_ACK,       (ATTACK_FUNC)attack_tcp_ack);
    add_attack(ATK_VEC_STOMP,     (ATTACK_FUNC)attack_tcp_stomp);

    add_attack(ATK_VEC_GREIP,     (ATTACK_FUNC)attack_gre_ip);
    add_attack(ATK_VEC_GREETH,    (ATTACK_FUNC)attack_gre_eth);

    //add_attack(ATK_VEC_PROXY, (ATTACK_FUNC)attack_app_proxy);
    add_attack(ATK_VEC_HTTP,      (ATTACK_FUNC)attack_app_http);
```

command code

attack function

# Static patterns of *attack_init*()

- It's composed of one single instruction block
- 1, or 2 in case of inline optimization, unique functions are repeatedly called
- Multiple callback functions, actually attack method functions, are referenced

By exploiting the above patterns, attack_init() function could be located in binary samples with IDAPython

# Dynamic patterns of add_attack()

```
static void add_attack(ATTACK_VECTOR vector, ATTACK_FUNC func)
{
    struct attack_method *method = calloc(1, sizeof (struct attack_method));
    method->vector = vector;
    method->func = func;

    methods = realloc(methods, (methods_len + 1) * sizeof (struct attack_method *));
    methods[methods_len++] = method;
}
```

each method is allocated a separate item

item is saved to method table

method table

- The core is the newly allocated item
  - Func-call: returned from a function
  - Mem-write1: be written with {command code, attack method}
  - Mem-write2: saved to a global table

# Scheme-3: command code based clustering

| Command code combination | Samples |
|---|---|
| 0_1_2_3_4_5_6_7_8_9_10 | 10746 |
| 0_1_2_3_4_5_6_7_9_10 | 3851 |
| 0_1_2_3_4_5_6_7_8 | 2031 |
| 0_1_2_3_4_5_6_7_8_9 | 806 |
| 0_1_2_3_6_7_8 | 670 |
| 0_1_2_3_4_5_6_7 | 250 |
| 1 | 247 |
| 0_1_2_3_4_5_6_7_9_10_11 | 200 |
| 1_2_3 | 157 |
| 0_1_2_3_4 | 125 |

# Same code, different method

## Mirai.1st

```
#define ATK_VEC_UDP          0
#define ATK_VEC_VSE          1
#define ATK_VEC_DNS          2
#define ATK_VEC_SYN          3
#define ATK_VEC_ACK          4
#define ATK_VEC_STOMP        5
#define ATK_VEC_GREIP        6
#define ATK_VEC_GREETH       7
//#define ATK_VEC_PROXY          8
#define ATK_VEC_UDP_PLAIN    9
#define ATK_VEC_HTTP         10
```

## Owari

```
#define ATK_VEC_UDP          0
#define ATK_VEC_VSE          1
#define ATK_VEC_DNS          2
#define ATK_VEC_SYN          3
#define ATK_VEC_ACK          4
#define ATK_VEC_STOMP        5
#define ATK_VEC_GREIP        6
#define ATK_VEC_GREETH       7
#define ATK_VEC_UDP_PLAIN    8
#define ATK_VEC_STD          9
#define ATK_VEC_XMAS         10
```

# Same code, different method

## Mirai.1st

```
#define ATK_VEC_UDP          0
#define ATK_VEC_VSE          1
#define ATK_VEC_DNS          2
#define ATK_VEC_SYN          3
#define ATK_VEC_ACK          4
#define ATK_VEC_STOMP        5
#define ATK_VEC_GREIP        6
#define ATK_VEC_GREETH       7
//#define ATK_VEC_PROXY        8
#define ATK_VEC_UDP_PLAIN    9
#define ATK_VEC_HTTP         10
```

## Omni

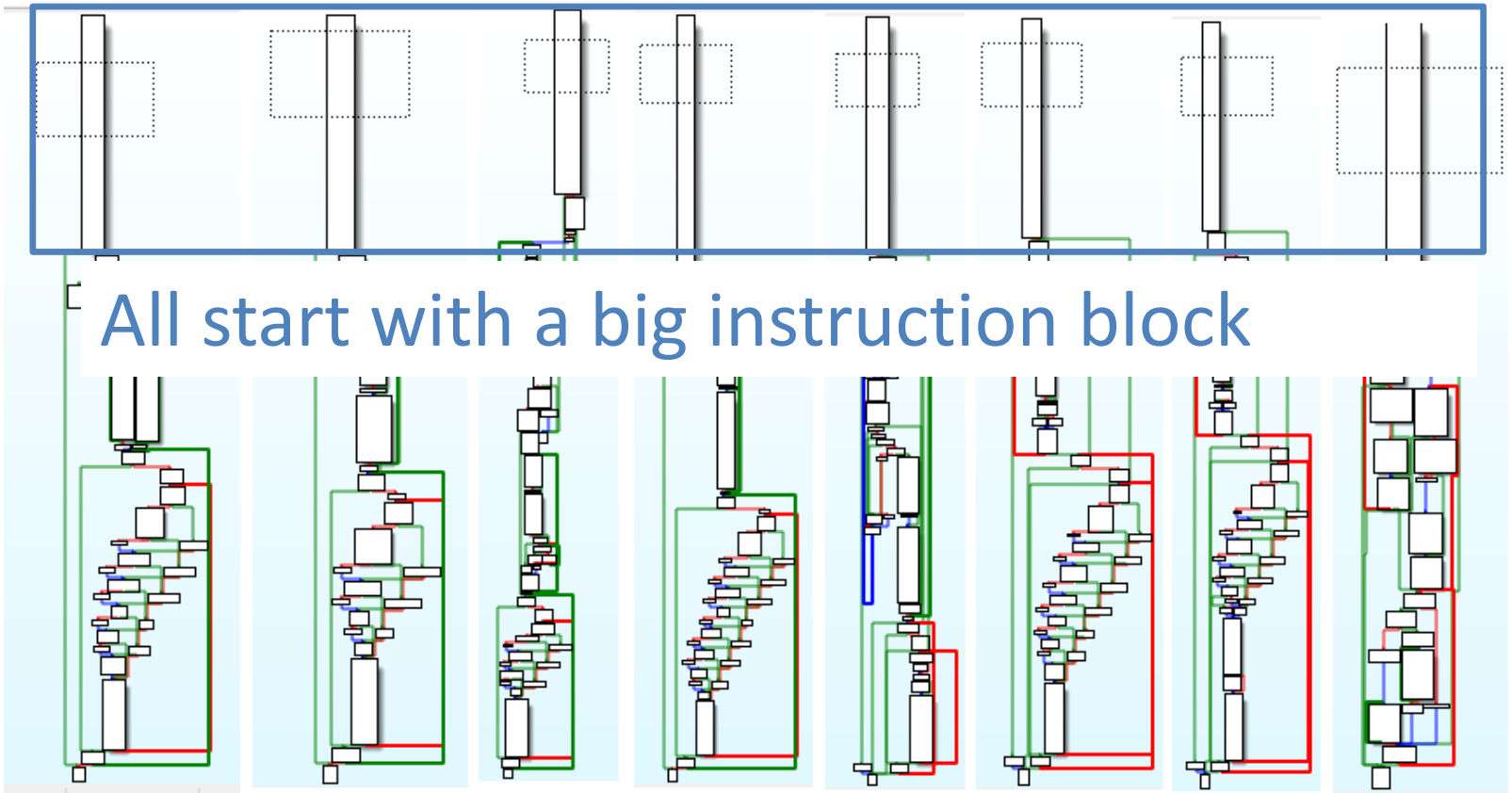
```
#define ATK_VEC_SYN          0
#define ATK_VEC_ACK          1
#define ATK_VEC_USYN         2
#define ATK_VEC_TCPALL       3
#define ATK_VEC_TCPFRAG      4
#define ATK_VEC_ASYN         5
#define ATK_VEC_GAME         6
#define ATK_VEC_UDPPLAIN     7
#define ATK_VEC_STOP         8
#define ATK_VEC_DESTRUCT     9
```

# Fingerprinting attack functions

- To figure out extracted attack functions' real semantics
  - E.g., SYN-/UDP-/HTTP-flood

- It's inspired by the following 2 findings:
  - A set of attack options, together with command codes, were defined to deliver attack parameters
  - Option sets are unique to different attack functions

# Mirai-style attack functions



All start with a big instruction block

MD5=652ba82411b745e5dac44cd15e314b25

# Attack option parsing

```
void attack_app_http(uint8_t targs_len, struct attack_target *targs, uint8_t opts_len, struct attack_option *opts)
{
    int i, ii, rfd, ret = 0;
    struct attack_http_state *http_table = NULL;
    char *postdata = attack_get_opt_str(opts_len, opts, ATK_OPT_POST_DATA, NULL);
    char *method = attack_get_opt_str(opts_len, opts, ATK_OPT_METHOD, "GET");
    char *domain = attack_get_opt_str(opts_len, opts, ATK_OPT_DOMAIN, NULL);
    char *path = attack_get_opt_str(opts_len, opts, ATK_OPT_PATH, "/");
    int sockets = attack_get_opt_int(opts_len, opts, ATK_OPT_CONNS, 1);
    port_t dport = attack_get_opt_int(opts_len, opts, ATK_OPT_DPORT, 80);
```

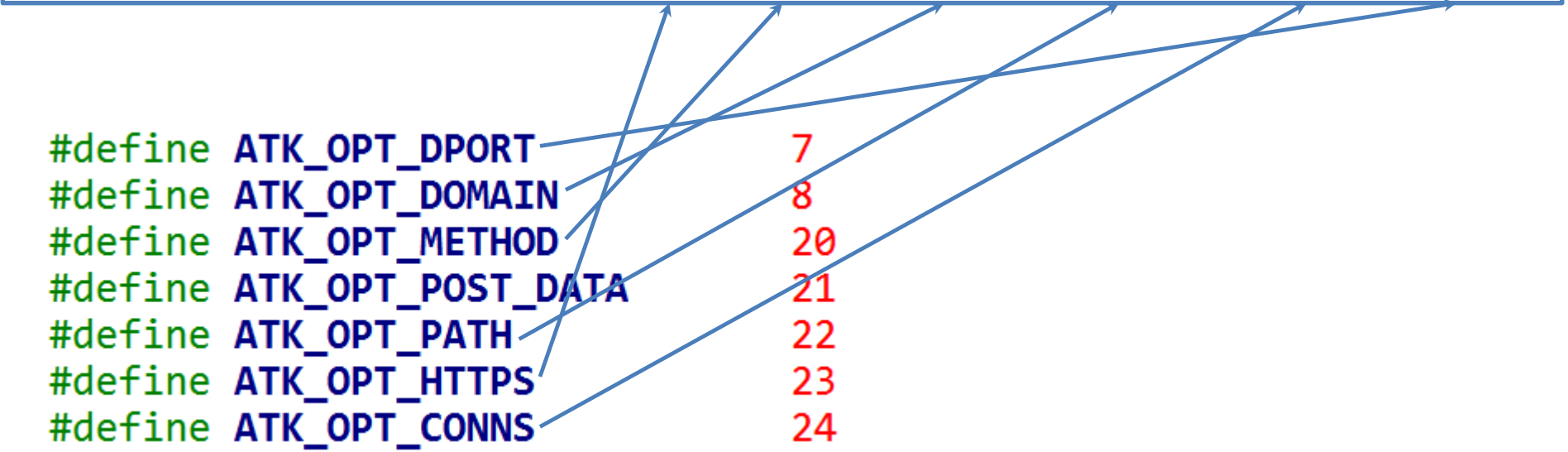
different functions,
different option sets

```
void attack_gre_ip(uint8_t targs_len, struct attack_target *targs, ... *opts)
{
    int i, fd;
    char **pkts = calloc(targs_len, sizeof (char *));
    uint8_t ip_tos = attack_get_opt_int(opts_len, opts, ATK_OPT_IP_TOS, 0);
    uint16_t ip_ident = attack_get_opt_int(opts_len, opts, ATK_OPT_IP_IDENT, 0xffff);
    uint8_t ip_ttl = attack_get_opt_int(opts_len, opts, ATK_OPT_IP_TTL, 64);
    BOOL dont_frag = attack_get_opt_int(opts_len, opts, ATK_OPT_IP_DF, TRUE);
    port_t sport = attack_get_opt_int(opts_len, opts, ATK_OPT_SPORT, 0xffff);
    port_t dport = attack_get_opt_int(opts_len, opts, ATK_OPT_DPORT, 0xffff);
    int data_len = attack_get_opt_int(opts_len, opts, ATK_OPT_PAYLOAD_SIZE, 512);
    BOOL data_rand = attack_get_opt_int(opts_len, opts, ATK_OPT_PAYLOAD_RAND, TRUE);
    BOOL gcip = attack_get_opt_int(opts_len, opts, ATK_OPT_GRE_CONSTIP, FALSE);
    uint32_t source_ip = attack_get_opt_int(opts_len, opts, ATK_OPT_SOURCE, LOCAL_ADDR);
```

# Fingerprinting definition

- FP(atk_func)={concatenation of option codes}

FP(atk_app_http)=0x15_0x14_0x08_0x16_0x18_0x07



# Summary of attack fingerprints

- In total **82** unique fingerprints have been found
  - Most of them are shared across variants

- Maps of {FP, atk_type} could be established by manual RE or using symbols from unstripped samples

# Scheme-4: attack type based classification

- A variant is defined as the coded attack types
  - E.g., {0-atk_udp1, 1-atk_udp_vse1, 2-atk_tcp_syn1, …}
- Information of method count, command codes, and attack types is fully exploited
- In total 206 unique combinations have been found
  - In other word, there are 206 variants under scheme-4

# Cluster *aandy* and *cmdswitch* in scheme-4

- Cluster *aandy* and *cmdswitch* belong to the same variant
  - [0-udp1, 1-udp_vse1, 10-http1, 2-udp_dns, 3-tcp_syn1, 4-tcp_ack1, 5-tcp_stomp_or_xmas1, 6-gre1, 7-gre1, 9-std_or_udp]

# Outline

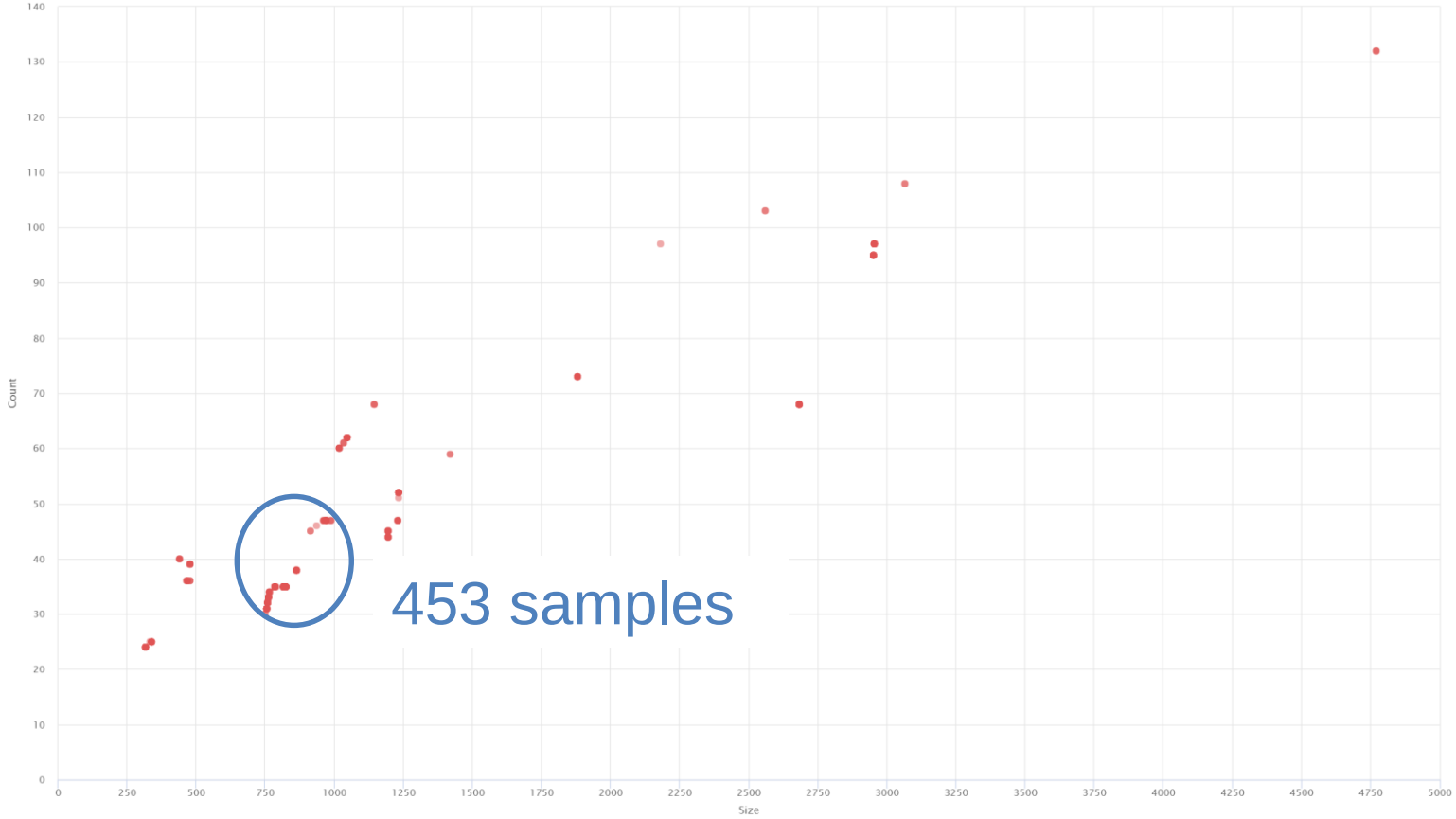
- Background
- Data and methodology
- Detailed analysis of branch *IZ1H9*
- Summary

# Summary of IZ1H9

- Samples: 709
- First_seen: 2018-08-09
- Last_seen: 2018-10-31
- CNCs: 96

| Samples | CNC |
|---|---|
| 143 | 185.244.25.176 |
| 27 | 145.239.117.244 |
| 26 | 128.199.222.37 |
| 20 | xnx.mariokartayy.com |
| 20 | 205.185.113.79 |
| 18 | 185.10.68.127 |
| 18 | 128.199.175.181 |
| 15 | 178.62.45.105 |
| 15 | 178.128.150.223 |
| 15 | 176.32.33.155 |

# IZ1H9 samples under scheme-1



# IZ1H9 samples under scheme-2

- **3** keys were found

| Variant | Samples | CNCs |
|---|---|---|
| IZ1H9+0xEA | 579 | 92 |
| IZ1H9+0x22 | 90 | 6 |
| IZ1H9+0x3D | 9 | 3 |

# 26 variants under scheme-4

| Samples | Combination of command code and method name |
|---|---|
| 405 | [0-atk_udp_or_gre2, 1-atk_udp_vse1, 2-atk_udp_dns, 3-atk_tcp_syn5, 4-atk_tcp_ack2, 5-atk_tcp_stomp_or_xmas2, 6-atk_gre2, 7-atk_gre2, 8-atk_std_or_udp] |
| 90 | [0-atk_udp1, 1-atk_udp_vse1, 10-atk_http1, 11-atk_cf, 2-atk_udp_dns, 3-atk_tcp_syn1, 4-atk_tcp_ack1, 5-atk_tcp_stomp_or_xmas1, 6-atk_gre1, 7-atk_gre1, 9-atk_std_or_udp] |
| 47 | [1-atk_tcp_syn1, 2-atk_std_or_udp, 3-atk_std_or_udp, 4-atk_udp_dns] |
| 37 | [0-atk_tcp_syn1, 1-atk_tcp_syn1, 2-atk_tcp_syn1, 3-atk_tcp_syn1, 4-atk_tcp_syn1, 5-atk_tcp_syn1, 6-atk_udp_vse1, 7-atk_std_or_udp, 8-atk_gre1, 9-atk_std_or_udp] |

# Summary

- Current branch name based classification is not enough to deal with the Mirai variant explosion problem
- Ideas of variant classification based on Mirai configuration and attack methods are introduced
  - Data extraction method
  - 4 schemes based on the extracted data
- Samples of the *IZ1H9* branch were investigated under the proposed data and schemes


Thank you