

Lab Template – Ethan Roepke

1. Identify two additional architectures for which the Mirai dropper has been compiled. Don't give me just the abbreviations. You must give me the full names. Just because a device has this architecture, doesn't mean it is susceptible to Mirai.

(10 points)

Bins/dlr.arm – binaries dynamic language runtime

Bins/dlr.h4 – refers to the super H 4 processor

2. What service would need to be enabled to be susceptible to Mirai?

(10 points)

Telnet

3. Explain the line containing three commands. What are the commands doing?

(15 points)

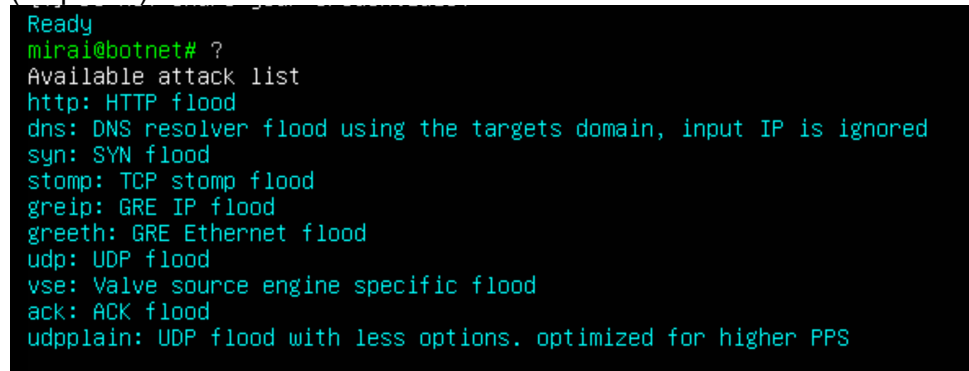
/bin/busybox wget... - Busybox is command calls the busybox executable and wget with the following URL retrieves the file mirai.x86 and drvHelper will save the output into a file.

/bin/busybox chmod... - Busybox command is called executable. Chmod will modify permission, given 777 therefor everyone will have r-w-e. And lastly drvHelper is the file they are referring to for the permission modifications.

/bin/busybox ECCHI – Busybox command calls the busybox executable and ECCHI is not a part of the standard busybox and a separate configurations so I am not sure what the role of it is used for.

4. Screenshot of help (?) from the CNC command line.

(10 points)



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

5. Screenshot of wireshark of successful syn flood attack

(10 points)

6590	9.540928610	192.168.1.1	192.168.1.4	TCP	74 7978 → 18614 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540929758	192.168.1.4	192.168.1.1	TCP	54 18614 → 7978 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540930347	192.168.1.1	192.168.1.4	TCP	74 36801 → 63320 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540931975	192.168.1.4	192.168.1.1	TCP	54 63320 → 36801 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540932189	192.168.1.1	192.168.1.4	TCP	74 3231 → 15410 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540933299	192.168.1.4	192.168.1.1	TCP	54 15416 → 3231 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540933827	192.168.1.1	192.168.1.4	TCP	74 53880 → 28999 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540934809	192.168.1.4	192.168.1.1	TCP	54 28999 → 53880 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540935489	192.168.1.1	192.168.1.4	TCP	74 18239 → 18927 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540936091	192.168.1.4	192.168.1.1	TCP	54 18927 → 18239 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540937173	192.168.1.1	192.168.1.4	TCP	74 38285 → 12157 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540938333	192.168.1.4	192.168.1.1	TCP	54 12157 → 38285 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540938812	192.168.1.1	192.168.1.4	TCP	74 59480 → 44166 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540939272	192.168.1.4	192.168.1.1	TCP	54 44166 → 59480 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540940420	192.168.1.1	192.168.1.4	TCP	74 63540 → 20425 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540941494	192.168.1.4	192.168.1.1	TCP	54 20425 → 63540 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540942175	192.168.1.1	192.168.1.4	TCP	74 10665 → 11899 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540943556	192.168.1.4	192.168.1.1	TCP	54 11899 → 10665 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540944149	192.168.1.1	192.168.1.4	TCP	74 30899 → 45455 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540945484	192.168.1.4	192.168.1.1	TCP	54 54545 → 59896 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540946119	192.168.1.1	192.168.1.4	TCP	74 19507 → 39463 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540947348	192.168.1.4	192.168.1.1	TCP	54 39463 → 19507 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540947819	192.168.1.1	192.168.1.4	TCP	74 64859 → 45468 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64
6590	9.540949063	192.168.1.4	192.168.1.1	TCP	54 45468 → 64859 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
6590	9.540949642	192.168.1.1	192.168.1.4	TCP	74 31014 → 31011 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=1134991599 TSecr=0 WS=64

6. Screenshot of wireshark of any other successful attack from Mirai's list of options (10 points)

2840	12.616492758	192.168.1.1	192.168.1.4	TCP	566 18914 → 62580 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2841	12.616494583	192.168.1.4	192.168.1.1	TCP	54 62580 → 18914 [RST] Seq=1 Win=0 Len=0
2842	12.616503197	192.168.1.1	192.168.1.4	TCP	566 59456 → 26187 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2843	12.616505264	192.168.1.4	192.168.1.1	TCP	54 26187 → 59456 [RST] Seq=1 Win=0 Len=0
2844	12.616511235	192.168.1.1	192.168.1.4	TCP	566 44703 → 51127 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2845	12.616512982	192.168.1.4	192.168.1.1	TCP	54 51127 → 44703 [RST] Seq=1 Win=0 Len=0
2846	12.616515034	192.168.1.1	192.168.1.4	TCP	566 11748 → 41428 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2847	12.616516750	192.168.1.4	192.168.1.1	TCP	54 41428 → 11748 [RST] Seq=1 Win=0 Len=0
2848	12.616521889	192.168.1.1	192.168.1.4	TCP	566 13287 → 28187 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2849	12.616523572	192.168.1.4	192.168.1.1	TCP	54 28187 → 13287 [RST] Seq=1 Win=0 Len=0
2850	12.616531771	192.168.1.1	192.168.1.4	TCP	566 19955 → 18526 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2851	12.616533099	192.168.1.4	192.168.1.1	TCP	54 18526 → 19955 [RST] Seq=1 Win=0 Len=0
2852	12.616545154	192.168.1.1	192.168.1.4	TCP	566 20979 → 53531 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2853	12.616547266	192.168.1.4	192.168.1.1	TCP	54 53531 → 20979 [RST] Seq=1 Win=0 Len=0
2854	12.616572294	192.168.1.1	192.168.1.4	TCP	566 25383 → 20189 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2855	12.616574282	192.168.1.4	192.168.1.1	TCP	54 20189 → 25383 [RST] Seq=1 Win=0 Len=0
2856	12.616581126	192.168.1.1	192.168.1.4	TCP	566 58558 → 5682 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2857	12.616583638	192.168.1.4	192.168.1.1	TCP	54 5682 → 58558 [RST] Seq=1 Win=0 Len=0
2858	12.616588959	192.168.1.1	192.168.1.4	TCP	566 64387 → 62871 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2859	12.616590847	192.168.1.4	192.168.1.1	TCP	54 62871 → 64387 [RST] Seq=1 Win=0 Len=0
2860	12.616592517	192.168.1.1	192.168.1.4	TCP	566 62734 → 23120 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2861	12.616594116	192.168.1.4	192.168.1.1	TCP	54 23120 → 62734 [RST] Seq=1 Win=0 Len=0
2862	12.616595589	192.168.1.1	192.168.1.4	TCP	566 25877 → 13886 [ACK] Seq=1 Ack=1 Win=40070 Len=512
2863	12.616597226	192.168.1.4	192.168.1.1	TCP	54 13886 → 25877 [RST] Seq=1 Win=0 Len=0
2864	12.616603603	192.168.1.1	192.168.1.4	TCP	566 8697 → 25813 [ACK] Seq=1 Ack=1 Win=40070 Len=512

7. What types of IoT devices (Fridges, Microwaves, etc...) were primarily exploited in Mirai attacks? You may refer to [this link](#). (10 points)

After reading the file, the most common compromised devices from the article were cameras and digital video recorders.

8. What are some of the commonalities found in the devices in your previous answer? (Think about OS, versions, services...) (10 points)

These two devices that became compromised were running a BusyBox on linux so Mirai are able exploit these devices vulnerabilities.

9. How did the developers of Mirai ensure that they didn't attack private IP addresses, the USPS, nor the DOD? Why would they want to do this? (15 points)

The developers developed a filter which would have the bots programmed to avoid specific IP addresses that they need to avoid. Doing this would decrease attention and avoid consequences from illegal activity. Being able to eliminate Private IP addresses would enhance the efficiency of the process because they are non-routable and inaccessible from external networks.