Noise EMI characteristics of Mirai botnet

Abhinav Narain

December 21, 2016

1 Mirai Activity and Results

1.1 Testbed Setup

The setup was first made in Virtual environment and then ported to actual machines on a private network in lab.

- Linux Box DNS server
- Linux Box CNC server
- Linux Box or Raspberry Pi - client

Figure 6 shows the testbed setup on the wireless network.

1.2 Breakdown of Mirai Bot(client) Activity

The following steps are performed by Mirai bot.

1. Sets up some certain data structures, and kills a previous instance is running
2. Connects to CNC server after resolving the IP address using a DNS packet to DNS server
3. executes a *fork()* system call to do a SYN scan for a range of IP addresses by transmitting TCP SYN frames using RAW sockets
4. Run an infinite loop waiting for SYN responses for further conducting a programmed TELNET login sequence.
5. Wait for attacks from a CNC server and executes a *fork()* system call to launch an attack for a period of time – one of which is a UDP flood attack conducted for 5 seconds 4. Following which the child process kills it's parent process (and thereby killing itself) and hence the attack is stopped.
6. The main thread sleeps for 1 second if it lost/couldn't connect or timed out on connecting to CNC server. It does sleep elsewhere also but those conditions are not true during the current test.

1.3 Expected Output of EMI

- Step 3 above should produce an EMI on spectrogram corresponding to SYN packets transmitted in bulk to certain subnets.
- Step 5 should show EMI on spectrogram corresponding to UDP flooding

1.4 Spectrogram of Lenovo Laptop

Experiments conducted by running Mirai bot on Lenovo laptop.

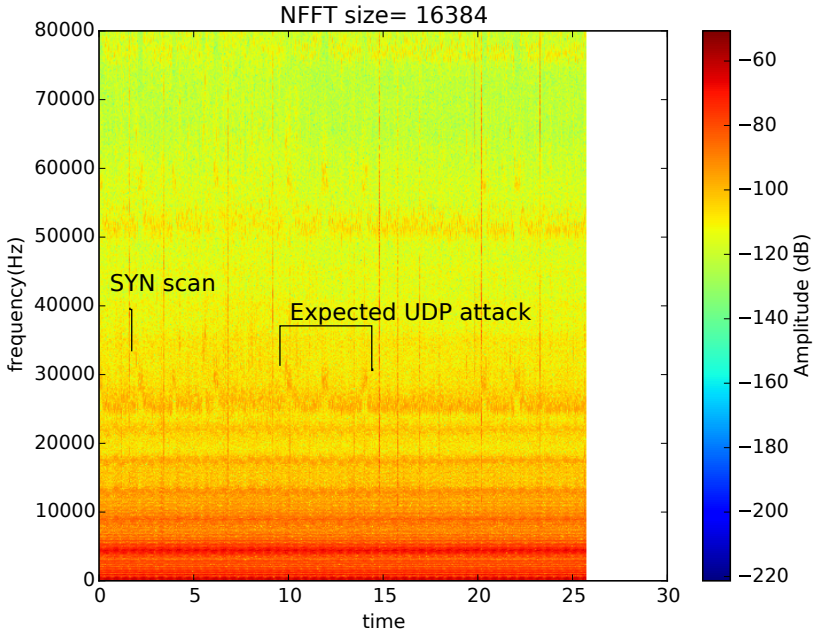

Figure 1: Mirai noise profile on Lenovo laptop. Sampling rate= $1e6$, $N = 2^{14}$, Resolution (fs/N)=61.035 Hz

1.4.1 Comments on Lenovo spectrogram

- Figure 1 shows EMI generated by EMI on Lenovo laptop. As described previously, the spectrogram is annotated with the initial SYN packet spew by the bot and the attack activated for 5 minutes. Unfortunately, it does not show a continuous increase in the frequency when UDP flood attack is conducted as expected by our previous experiments. I see the increase in the frequency when I run the bot in **DEBUG** mode as shown in Figure 3,

which is the same code, but without any *fork* system calls, hence running as a single process.

1.5 Spectrogram of Raspberry Pi

Experiments conducted by running Mirai bot on Raspberry Pi.

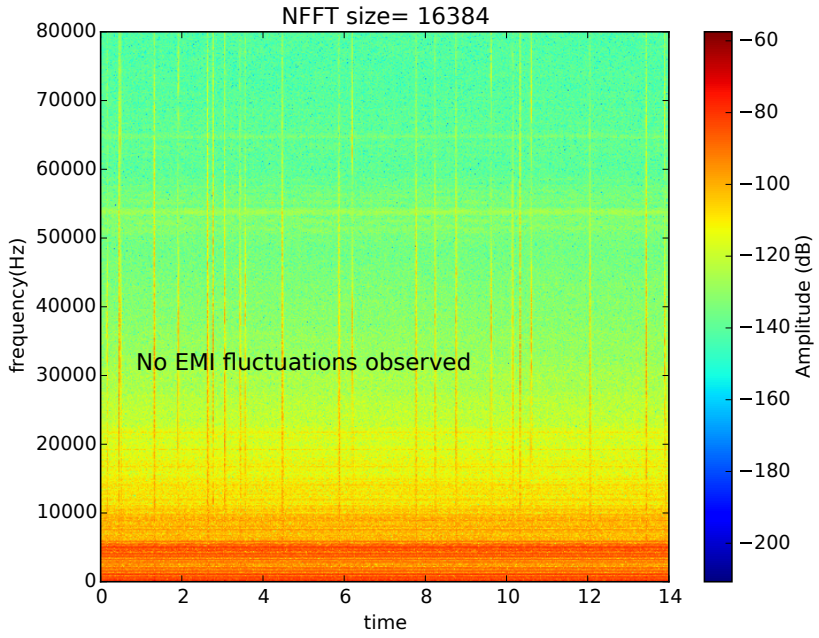

Figure 2: Mirai noise profile on Raspberry Pi. Sampling rate= $1e6$, $N = 2^{14}$, Resolution (fs/N)=61.035 Hz

1.5.1 comments on Raspberry Pi

I did the experiments with Raspberry Pi as the client. As with previous experiments of UDP blasts and computation, I was not able to see change in EMI. There are possibly two reasons for it.

- It uses the USB power-supply 5. I am not clear about how the conversion of power supply takes place.
- The change in power consumed by Raspberry Pi might not be significant. I don't think this should be the case. A busy loop should have changed the noise profile in our previous experiments and also showed some change in noise on power-line when Mirai was executed.

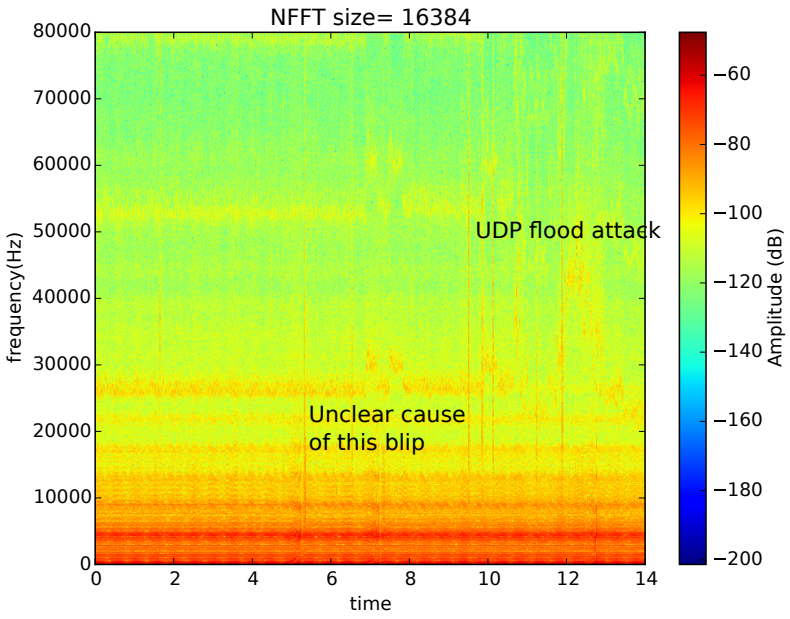

Figure 3: Mirai bot running in **DEBUG** mode as a single process shows increase in Noise EMI as expected.

2 Supplementary Images

The 4 shows the CNC server Mirai interface over localhost. One attack was conducted – UDP flood attack for 5 seconds.

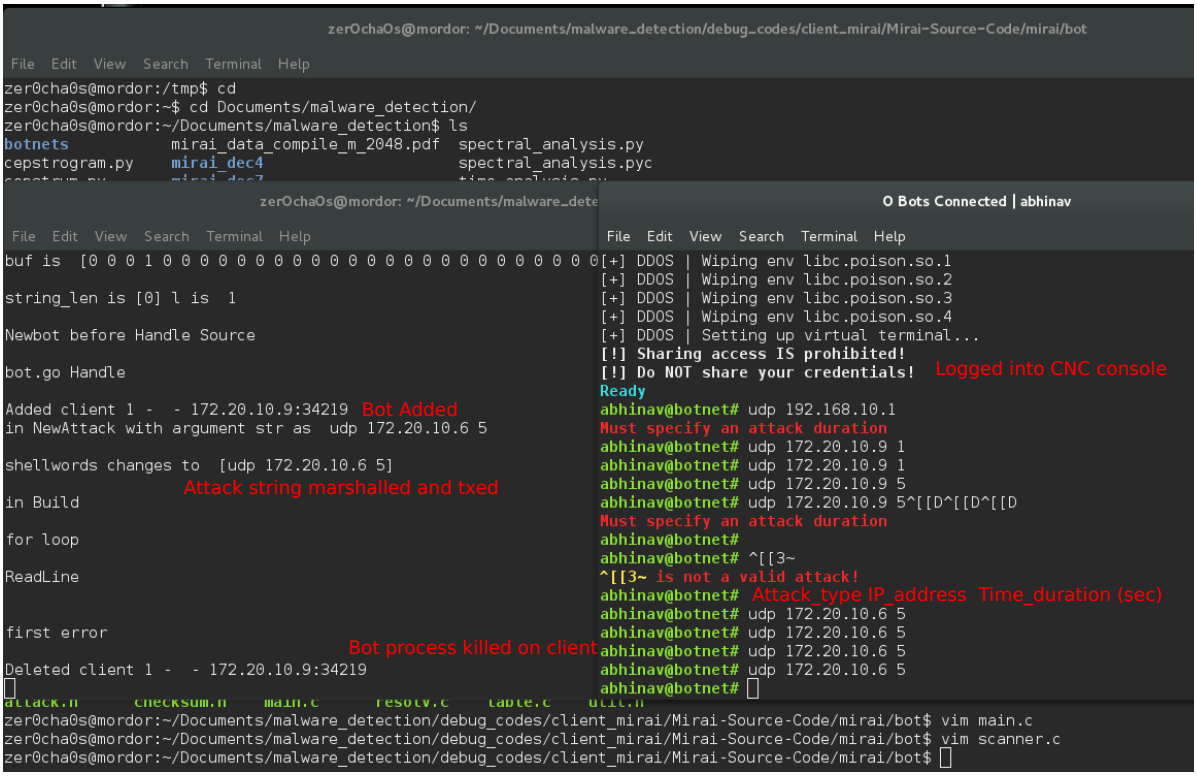

Figure 4: Attack issued on CNC window. UDP flood for 5 seconds

Figure 5: USB to power-line converter adapter for Raspberry Pi

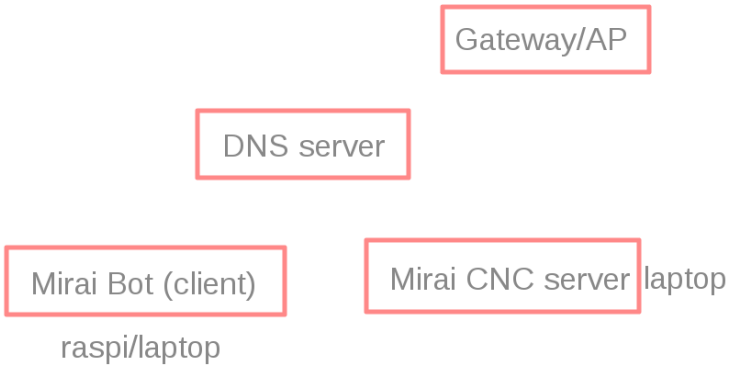

Figure 6: Lab testbed on Wireless Network for the devices used. Mirai Bot(client) is connected to the isolated Transformer for capturing Power-line traces.