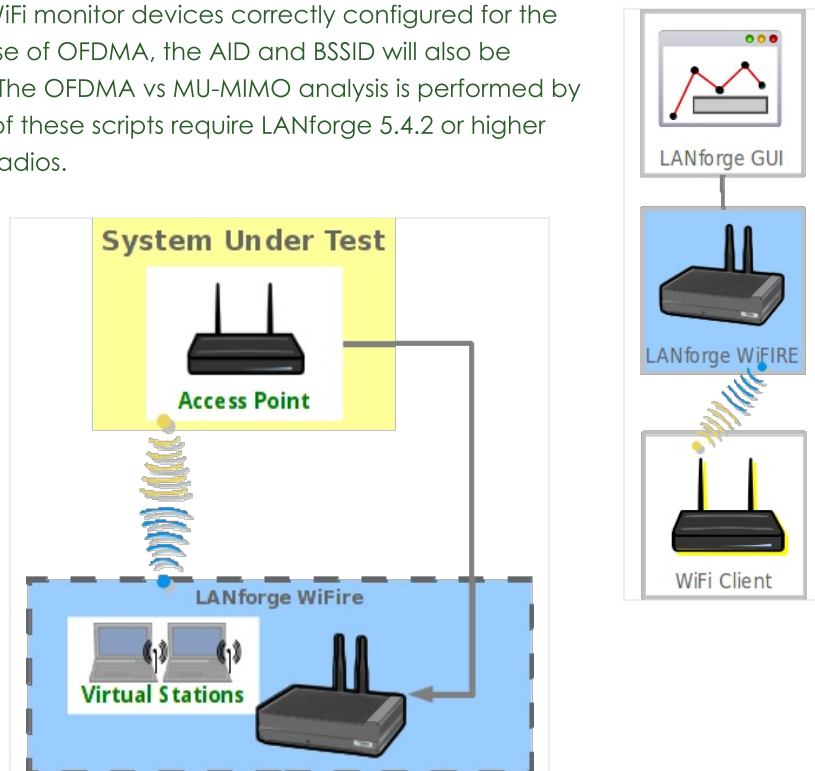


Using LANforge python script to do packet captures then analyze to determine if the traffic was OFDMA, MU-MIMO or Single User

Goal: Sniff wireless traffic to/from LANforge stations using additional LANforge Radios in monitor mode.

This script automates creating WiFi monitor devices correctly configured for the station's channel, and in the case of OFDMA, the AID and BSSID will also be configured in the sniffing radio. The OFDMA vs MU-MIMO analysis is performed by the wifi diagnostics script. Both of these scripts require LANforge 5.4.2 or higher and an appropriate set of WiFi radios.



1. Create and configure stations to talk to your DUT. You may create these through Chamber View scenarios or other methods. Please refer to introductory cookbooks for more information on this if you have questions.

2. Run lf_sniff.py script.

- A. Change directory to the /home/lanforge/scripts directory (or other location if you have installed scripts elsewhere), and run the lf_sniff.py script with the --help argument to understand your options..
- B. Run the script with arguments for your test case. The basic idea is to provide a list of stations you wish to sniff, and a list of matching radios that you wish to use as sniffers. The script will query the station to find the channel, AID and BSSID and will then create WiFi monitor devices on the sniffer radios. A command-line packet capture tool will then be started and write the captured packets to disk for the requested duration. When the duration is complete, the packet capture process will exit. You can then post-process the packet captures, open them with wireshark, etc.

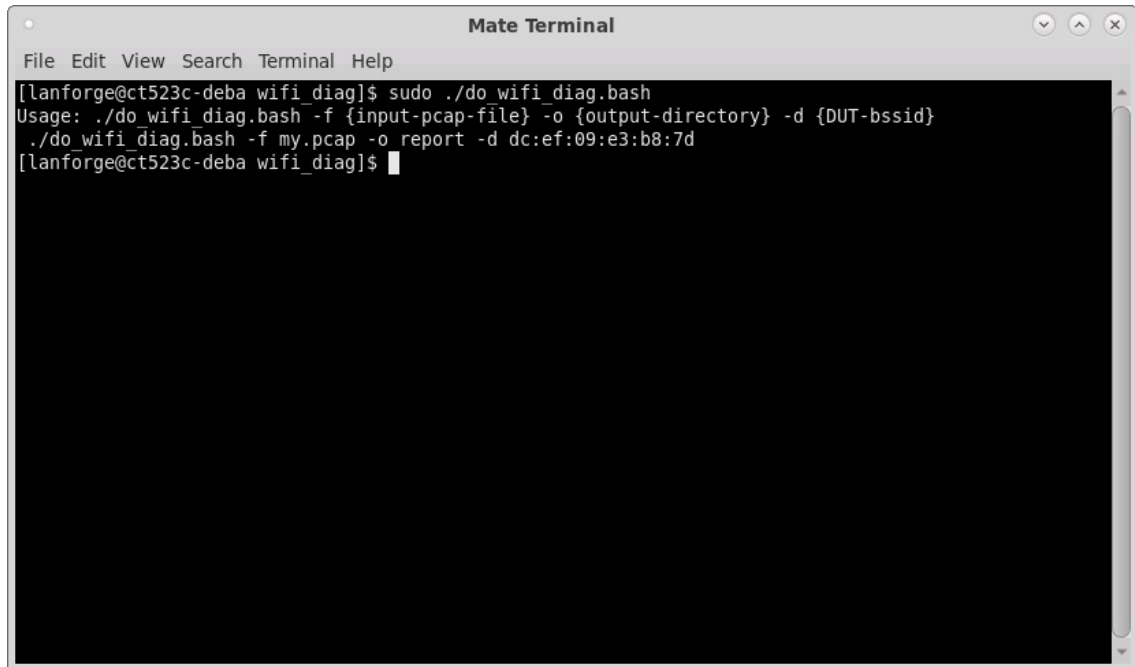
Examples

Sniff sta00000 on LANforge resource 1 using radio wiphy0 on LANforge resource 2 for 15 seconds:

```
./lf_sniff.py --lfmgr 192.168.100.238 --station "1.sta00000" \  
--sniffer_radios "2.wiphy0" --duration 0.25
```

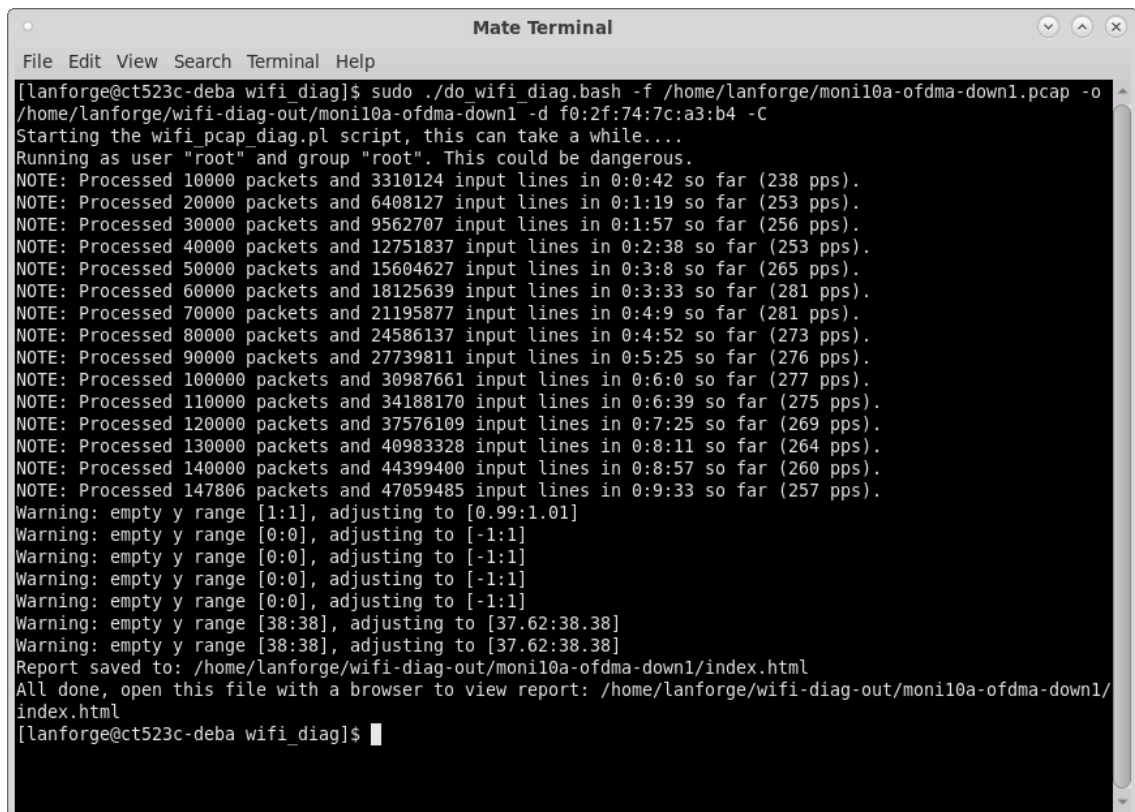
3. Run do_wifi_diag.bash script.

- A. Change directory to the /home/lanforge/scripts/wifi_diag directory, and run the bash script without any arguments to view the usage help.



```
Mate Terminal
File Edit View Search Terminal Help
[lanforge@ct523c-deba wifi_diag]$ sudo ./do_wifi_diag.bash
Usage: ./do_wifi_diag.bash -f {input-pcap-file} -o {output-directory} -d {DUT-bssid}
./do_wifi_diag.bash -f my.pcap -o report -d dc:ef:09:e3:b8:7d
[lanforge@ct523c-deba wifi_diag]$
```

- B. Run the bash script with correct location of the pcap file obtained with If_sniff.py and with the correct BSSID of the AP used as the DUT. Depending on your system processing power and the pcap file size, the wifi diag script will take some time to complete.
- C. When the wifi diag script is complete, you can open the index.html file to view the results.



```
Mate Terminal
File Edit View Search Terminal Help
[lanforge@ct523c-deba wifi_diag]$ sudo ./do_wifi_diag.bash -f /home/lanforge/monil10a-ofdma-down1.pcap -o /home/lanforge/wifi-diag-out/monil10a-ofdma-down1 -d f0:2f:74:7c:a3:b4 -C
Starting the wifi_pcap_diag.pl script, this can take a while...
Running as user "root" and group "root". This could be dangerous.
NOTE: Processed 10000 packets and 3310124 input lines in 0:0:42 so far (238 pps).
NOTE: Processed 20000 packets and 6408127 input lines in 0:1:19 so far (253 pps).
NOTE: Processed 30000 packets and 9562707 input lines in 0:1:57 so far (256 pps).
NOTE: Processed 40000 packets and 12751837 input lines in 0:2:38 so far (253 pps).
NOTE: Processed 50000 packets and 15604627 input lines in 0:3:8 so far (265 pps).
NOTE: Processed 60000 packets and 18125639 input lines in 0:3:33 so far (281 pps).
NOTE: Processed 70000 packets and 21195877 input lines in 0:4:9 so far (281 pps).
NOTE: Processed 80000 packets and 24586137 input lines in 0:4:52 so far (273 pps).
NOTE: Processed 90000 packets and 27739811 input lines in 0:5:25 so far (276 pps).
NOTE: Processed 100000 packets and 30987661 input lines in 0:6:0 so far (277 pps).
NOTE: Processed 110000 packets and 34188170 input lines in 0:6:39 so far (275 pps).
NOTE: Processed 120000 packets and 37576109 input lines in 0:7:25 so far (269 pps).
NOTE: Processed 130000 packets and 40983328 input lines in 0:8:11 so far (264 pps).
NOTE: Processed 140000 packets and 44399400 input lines in 0:8:57 so far (260 pps).
NOTE: Processed 147806 packets and 47059485 input lines in 0:9:33 so far (257 pps).
Warning: empty y range [1:1], adjusting to [0.99:1.01]
Warning: empty y range [0:0], adjusting to [-1:1]
Warning: empty y range [0:0], adjusting to [-1:1]
Warning: empty y range [0:0], adjusting to [-1:1]
Warning: empty y range [0:0], adjusting to [-1:1]
Warning: empty y range [38:38], adjusting to [37.62:38.38]
Warning: empty y range [38:38], adjusting to [37.62:38.38]
Report saved to: /home/lanforge/wifi-diag-out/monil10a-ofdma-down1/index.html
All done, open this file with a browser to view report: /home/lanforge/wifi-diag-out/monil10a-ofdma-down1/index.html
[lanforge@ct523c-deba wifi_diag]$
```

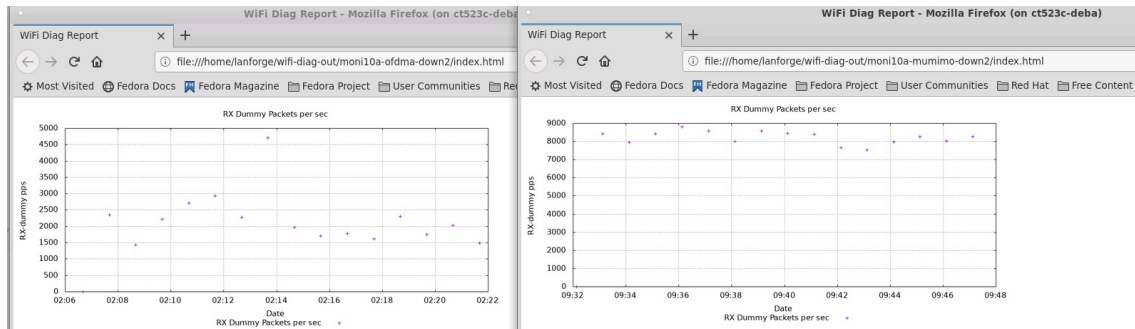
4. Interpreting the report for OFDMA or MU-MIMO operation.

- A. To determine if the packet capture contains OFDMA or MU-MIMO operation, scroll down in the report to the table labeled "RX Packet Type Histogram" and the element "ACK but not captured" which is a count of the number of Block ACKs captured without capturing the corresponding Data frames. If the report shows a large percentage of "ACK but not captured", this indicates MU-MIMO operation because the data frames are not able to be captured. If there is a lower percentage, than the traffic is likely OFDMA.

Type	Packets	Percentage
ACK but not Captured	33520	15.298182

Type	Packets	Percentage
ACK but not Captured	123843	85.135358

- B. "ACK but not captured" is also represented in the graph later in the report labeled "RX Dummy Packets".



- C. The graph labeled "RX RU Alloc over time" also indicates OFDMA operation because it shows when frames were assigned a Resource Unit allocation which is the main trait of OFDMA operation.

