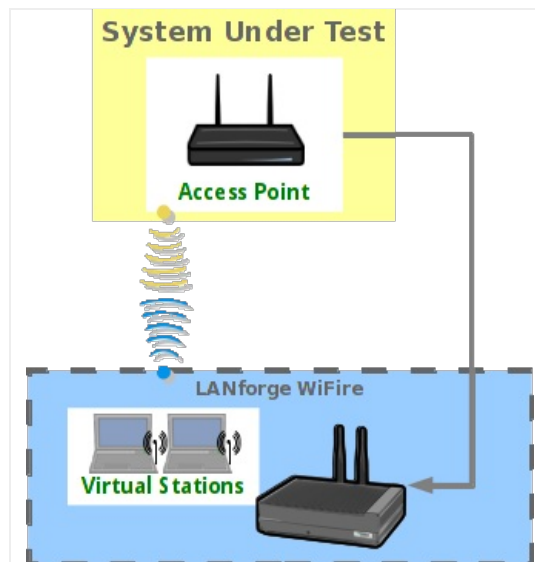


Using LANforge python script to generate a mix of traffic and capture it.

Goal: Create WiFi stations with specified a/b/g/n/AC/AX modes and spatial streams, data connections with different QoS, packet size and offered load settings and report on throughput and latency. Captures will be made of the offered traffic and also the RF packet capture for processing by external tools.

This script automates creating data connections with BK, BE, VI, VO and/or other QoS settings, as well as packet sizes, protocol type, and offered load. It then starts packet captures on the Ethernet port and RF monitor interfaces. The traffic and packet captures run for a specified amount of time and XLSX and CSV reports are generated to report latency and throughput. The packet captures are placed into a directory for post-processing by user-supplied tools. This script requires LANforge 5.4.2 or higher.



1. This test does not require much initial configuration. In this example, I have removed all existing WiFi station and data connection configuration before running the script.
2. Run `lf_tos_plus_test.py` script.

- A. Change directory to the /home/lanforge/scripts directory (or other location if you have installed scripts elsewhere), and run the lf_tos_plus_tst.py script with the --help argument to understand your options..

```

root@TR-398:/home/lanforge/lanforge-scripts
File Edit View Search Terminal Help
[root@TR-398 lanforge-scripts]# ./lf_tos_plus_test.py --help
usage: lf_tos_plus_test.py [-h] [--cx CX] [--radio RADIO] [--lfmgr LFMGR]
                          [--outfile OUTFILE] [--duration DURATION]
                          [--ssid SSID] [--passwd PASSWD] [--txpkts TXPKTS]
                          [--sniffer_radios SNIFFER_RADIOS]
                          [--wait_sniffer WAIT_SNIFFER]

ToS++ report Script

optional arguments:
  -h, --help            show this help message and exit
  --cx CX               Connection tuple: station-radio station-port mode
                        upstream-port protocol pkt-size speed_ul speed_dl QoS
  --radio RADIO         Radio tuple: radio nss channel
  --lfmgr LFMGR         LANforge Manager IP address
  --outfile OUTFILE     Output file for csv data
  --duration DURATION   Duration to run traffic, in minutes. If txpkts is
                        specified, that may stop the test earlier.
  --ssid SSID           AP's SSID
  --passwd PASSWD       AP's password if using PSK authentication, skip this
                        argement for OPEN
  --txpkts TXPKTS       Optional: Packets (PDUs) to send before stopping data
                        connections Default (0) means infinite
  --sniffer_radios SNIFFER_RADIOS
                        Optional: list of radios to sniff wifi traffic
                        "1.wiphy2 1.wiphy4"
  --wait_sniffer WAIT_SNIFFER
                        Optional: 1 means wait on sniffer to finish before
                        existing script"
[root@TR-398 lanforge-scripts]#

```

- B. Run the script with arguments for your test case. The basic idea is to provide a list of connection definitions you wish to use for traffic, along with some additional test-wide configuration.

--cx

Use multiple '--cx' arguments to create as many data connections as you wish. Each cx argument takes a list of configuration input:

```
station-radio station-port mode upstream-port protocol pkt-size speed_ul speed_dl QoS
```

For instance:

```
--cx "1.wiphy0 1.wlan0 an 1.eth1 udp 1024 10000 500000000 BK"
```

The example above creates a wlan0 interface in /AN mode on radio wiphy0. The eth1 port is the upstream side of the data connection. The protocol type is UDP, with 1024 byte payloads. Upload speed is 10kbps, download speed is 500Mbps, and ToS/QoS is Background.

--ssid

The --ssid specifies the SSID for the AP under test.

--passwd

The --passwd specifies the DUT password (PSK). If this option is not specified, then OPEN authentication will be used.

--radio

The '--radio' option allows one to configure the number of spatial streams and channel for a radio used in this test. In this example, we are forcing the radio to use 2 spatial streams and using channel 0 (which means use ANY available channel).

Example: `--radio "1.wiphy0 2 0"`

--dur

The --dur option determines the duration in minutes for the traffic connections and packet capture.

--txpkts

The --txpkts option configures data connections to generate a certain amount of PDUs and then quiesce the test. The test will complete when all connections are finished or the duration timer has expired, whichever is less.

--sniffer_radios

The --sniffer_radios allows you to select a list of radios to be used as packet capture devices. The first radio selected will capture on the first WiFi interface configured in the --cx section, the second for the second --cx, etc. The script automatically takes care of configuring the sniffer for the proper channel, and in the case of OFDMA radios, it will be configured for the specific AID so that it can capture OFDMA traffic.

--wait_sniffer

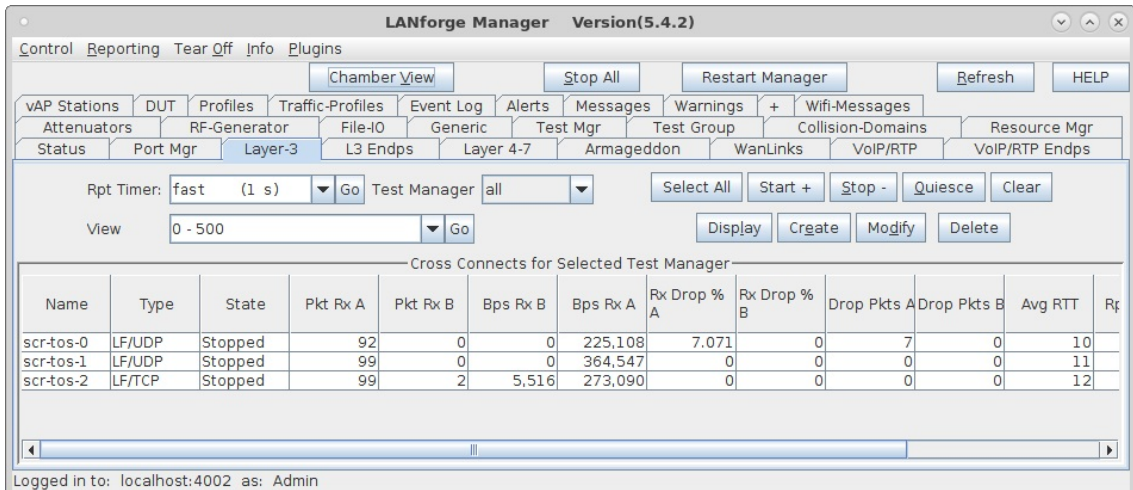
The --wait_sniffer option causes the script to pause until the sniffer programs have completed before exiting the script. This makes automation easier since you then know the captures are complete when the script exits.

The command used in this example is:

```
./lf_tos_plus_test.py --dur 0.5 --lfmgr 192.168.100.156 --ssid NETGEAR68-5G --passwd aquaticbug712 --radio "1.wiphy0 2 0" --txpkts 99 --cx "1.wiphy0 1.wlan0 an 1.eth1 udp 1024 10000 500000000 BK" --cx "1.wiphy0 1.wlan0 an 1.eth1 udp MTU 10000 500000000 VI" --cx "1.wiphy0 1.sta0 anAC 1.eth1 tcp 1472 56000 2000000 BK" --sniffer_radios "1.wiphy2" --wait_sniffer 1
```



C. The auto-created data connections can be found in the Layer-3 tab in the LANforge GUI.



D. Here are the details of the endpoints. Notice each sent 99 pdus as requested with the --txpkts option.

The screenshot shows the LANforge Manager interface. At the top, there are tabs for Control, Reporting, Tear Off, Info, and Plugins. Below these are various management buttons like Chamber View, Stop All, Restart Manager, Refresh, and HELP. A menu bar includes DUT, Profiles, Traffic-Profiles, Event Log, Alerts, Messages, Warnings, Test Group, Test Mgr, Test Group, Collision-Domains, Resource Mgr, and vAP Stations. A sub-menu bar includes Status, Port Mgr, Layer-3, L3 Endps, Layer 4-7, Armageddon, WanLinks, VoIP/RTP, VoIP/RTP Endps, and Attenuators. Configuration fields include Min PDU Size (AUTO), Max PDU Size (Same), MIN Tx Rate (New Modem (56 Kbps)), and MAX Tx Rate (Same). A View dropdown is set to 0-1000. Below the configuration is a table titled 'All Endpoints'.

Tx Rate	Tx Rate (1 min)	Tx Rate (last)	Tx Rate LL	Rx Rate	Rx Rate (1 min)	Rx Rate (last)	Rx Rate LL	Rx Drop %	Tx PDUs	Tx Pkts LL	PDU/s TX	Pps TX LL	Rx PDUs	F
0	0	0	0	225,108	226,169	0	235,448	7,071	0	0	0	0	0	92
242,236	243,376	243,472	0	0	0	0	0	0	99	99	29	29	0	0
0	0	0	0	364,547	366,261	365,462	376,712	0	0	0	0	0	0	99
364,433	366,141	368,931	0	0	0	0	0	0	99	99	30	30	0	0
5,516	5,550	0	0	273,090	274,829	0	296,419	0	2	4	0	0	0	99
273,090	274,825	0	0	5,516	5,550	0	6,161	0	99	198	23	46	2	2

Logged in to: localhost:4002 as: Admin

E. The script will run for the requested duration and then stop the traffic and generate the xlsx report.

The screenshot shows a terminal window titled 'root@TR-398:/home/lanforge/lanforge-scripts'. The terminal output shows the script checking station stats for '1.wlan0' and '1.sta0', setting the state of 'scr-tos-0', 'scr-tos-1', and 'scr-tos-2' to 'STOPPED', and finally saving a CSV report to 'tos+_results.xlsx.csv'. The script also indicates it is waiting for 34 seconds until the sniffer completes and that captures are found in the directory '2020-04-08-11:18:31'. The xlsx results are stored in 'tos+_results.xlsx'.

```

Checking station stats, key: 1.wlan0
sta-stats found: 1.wlan0, mode: 802.11an bw: 40
Checking station stats, key: 1.sta0
sta-stats found: 1.sta0, mode: 802.11an-AC bw: 80
default_tm: Setting state on CX: scr-tos-0 to STOPPED.

>>RSLT: 0 Cmd: 'set_cx_state' 'all' 'scr-tos-0' 'STOPPED'

default@
btbits>>
default_tm: Setting state on CX: scr-tos-1 to STOPPED.

>>RSLT: 0 Cmd: 'set_cx_state' 'all' 'scr-tos-1' 'STOPPED'

default@
btbits>>
default_tm: Setting state on CX: scr-tos-2 to STOPPED.

>>RSLT: 0 Cmd: 'set_cx_state' 'all' 'scr-tos-2' 'STOPPED'

default@
btbits>>
CSV report data saved to: tos+_results.xlsx.csv
Waiting 34 seconds until sniffer completes.
Captures are found in directory: 2020-04-08-11:18:31
Xlsx results stored in tos+_results.xlsx
[root@TR-398 lanforge-scripts]#

```

F. You can open the xlsx report in your favorite spreadsheet tool. Here is the first part of the document.

The screenshot shows a LibreOffice Calc spreadsheet titled 'tos+_results.xlsx'. The spreadsheet contains a table with columns for CX-Name, Endp-Name, Port, Protocol, ToS, AP BSSID, Band width, Mode, Last MCS Rx, Combined RSI, Endpoint Tx Pkt Size, Endpoint Offered Load, Endpoint Rx Throughput, and Cx Offered Load. The data is as follows:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
	CX-Name	Endp-Name	Port	Protocol	ToS	AP BSSID	Band width	Mode	Last MCS Rx	Combined RSI	Endpoint Tx Pkt Size	Endpoint Offered Load	Endpoint Rx Throughput	Cx Offered Load
1	scr-tos-0	scr-tos-0-A	1.wlan0	udp	BK	78:d2:94:4f:21:78	40	802.11an	216000kbps	-35dBm	1024	0.00	0.25	0.25
2	scr-tos-0	scr-tos-0-B	1.eth1	udp	BK						1024	0.25	0.00	0.25
3	scr-tos-1	scr-tos-1-A	1.wlan0	udp	VI	78:d2:94:4f:21:78	40	802.11an	216000kbps	-35dBm	MTU	0.00	0.37	0.37
4	scr-tos-1	scr-tos-1-B	1.eth1	udp	VI						MTU	0.37	0.00	0.37
5	scr-tos-2	scr-tos-2-A	1.sta0	tcp	BK	78:d2:94:4f:21:78	80	802.11an-AC	650000kbps	-35dBm	1472	0.01	0.28	0.29
6	scr-tos-2	scr-tos-2-B	1.eth1	tcp	BK						1472	0.28	0.01	0.29

- G. The more interesting part of the report may be the throughput and latency results. The latency-range columns show the amount of packets received in certain latency ranges. This gives you an idea of jitter and other latency related information. The average latency is a good way to get a quick idea of the latency. In this case, we are only generating 99 packets, so the report is of limited use. When sending more data, this data becomes more useful.

	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AI
1	Cx Offered Load	Cx Rx Throughput	Avg Latency	Min Latency	Max Latency	Latency Range 0	Latency Range 1	Latency Range 2-3	Latency Range 4-7	Latency Range 8-15	Latency Range 16-31	Latency Range 32-63	Latency Range 64-127	Latency Range 128-255	Late Ran 256-
2	0.25	0.25	11	8	19	0	0	0	0	35	35	29	0	0	0
3	0.25	0.25	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0.37	0.37	7	2	14	0	0	16	54	0	29	0	0	0	0
5	0.37	0.37	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0.29	0.29	13	2	38	0	0	2	1	26	61	8	1	0	0
7	0.29	0.29	1	1	1	0	2	0	0	0	0	0	0	0	0

- H. This script is designed to work well with automation. It creates a file called TOS_PLUS.sh that holds some shell variables defining the name of the XLSX and CSV files, as well as the capture directory. The capture files may be opened with wireshark or your other favorite packet analyzer, as well as post-processed by other automated tools. For instance, you can find a LANforge packet in the eth1 capture and the same packet in the RF monitor pcap file and compare the latency. If you are capturing on multiple radios at once, you may wish to merge the captures with the merg pcap tool or similar tool.

```

root@TR-398:/home/lanforge/lanforge-scripts
File Edit View Search Terminal Help

[root@TR-398 lanforge-scripts]# cat TOS_PLUS.sh
CAPTURE_DIR=2020-04-08-11:18:31
CSV_FILE=tos+_results.csv
XLSX_FILE=tos+_results.xlsx
[root@TR-398 lanforge-scripts]# ls -l 2020-04-08-11:18:31
total 1168
-rw-r--r-- 1 root root 448572 Apr  8 11:18 eth1.pcap
-rw-r--r-- 1 root root 740384 Apr  8 11:18 moni4a.pcap
[root@TR-398 lanforge-scripts]#

```