sales@candelatech.com
support@candelatech.com
+1 (360) 380-1618 [PST, GMT -8]

All LANforge-FIRE Cookbooks

The LANforge-FIRE Cookbook provides a set of high-level examples of how to setup useful test scenarios in LANforge-FIRE for traffic generation. Each example intends to give the reader a brief introduction to the test scenario and a set of step-by-step instructions on how to use the LANforge-GUI to configure the test.

All of the following examples will work on Linux systems running the LANforge software with the LANforge kernel and a sufficient license. If you are running another Linux kernel, you will not be able to exactly duplicate some of the examples, but there are usually work-arounds available to assist you. Please contact us at support@candelatech.com if you have any questions.

If you are using the Windows version of LANforge, you will have to modify ports using the Windows utilities. Everything else should work approximately the same, but the performance is limited to 10Mbps speeds.

LANforge-FIRE Traffic Generation

Before attempting the examples below, ensure that you have successfully followed these software installation guides:

- LANforge-GUI Installation
- LANforge Server Installation

It is also recommended that you back up your current running LANforge Server database so that you may safely return to your current operating state.

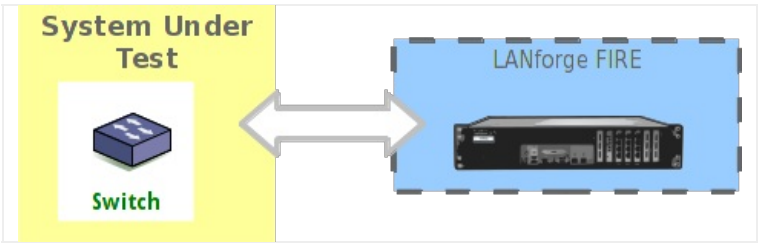
All LANforge-FIRE Cookbook Examples

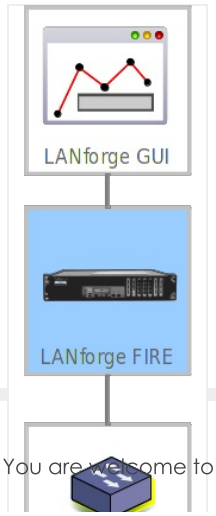
1. Chamber View: Dataplane test on 10G switched network
2. Switched Network Testing
3. Routed Network Testing
4. Firewall Testing
5. Webserver Testing
6. Introduction to Layer 4-7 Traffic
7. Scripted Cable Modem Testing
8. DSLAM Testing
9. VoIP Testing
10. Armageddon Testing (Accelerated UDP testing)
11. Armageddon Testing with Random MAC Addresses
12. Scripted Layer-3 Test (RFC-2544)
13. Scripted Armageddon Test (RFC-2544)
14. 5000 TCP Connections on unique IP addresses
15. Multiple Thousands of TCP Connections
16. Multiple Thousands of UDP Connections
17. Layer-3 UDP Traffic Generator
18. Armageddon Random MAC Generation
19. Dual Windows Host Layer 3 Testing
20. File IO Endpoint Testing
21. Providing HTTP service on a Port
22. Generating FTP Traffic Between Two Physical Ports
23. Table Report Builder
24. Displaying Sequence Gaps
25. Scalability Testing with NFS File-IO
26. Using the Port Monitor
27. Create Test Scripts With the Realm Class
28. Create Layer 4 Test Scripts With Python
29. Create Generic Test Scripts With Python
30. Layer 4 Upload Test
31. Install and Configure LANforge VMs on ESXi Server 8
32. Conference Call Testing

Use Dataplane to test throughput and latency at different packet sizes

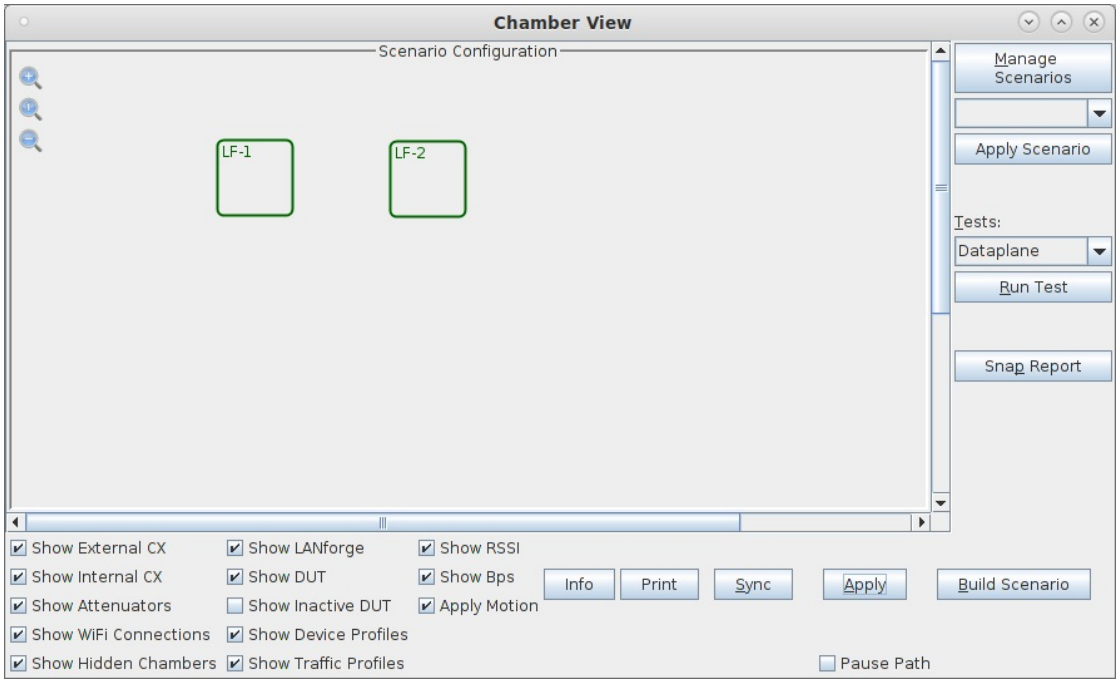
Goal: Setup and run a Dataplane test for an Ethernet network using two 10G LANforge systems.

In this test scenario, we use a system similar to the LANforge CT503-10g. The DUT is just a looped-back Ethernet cable. A real-world use of this test in the lab might have the DUT be a router, firewall, switch, or LAN. A field deployment could have one LANforge system at two different end points of a Wide Area Network. This feature requires LANforge version 5.4.1 or higher.

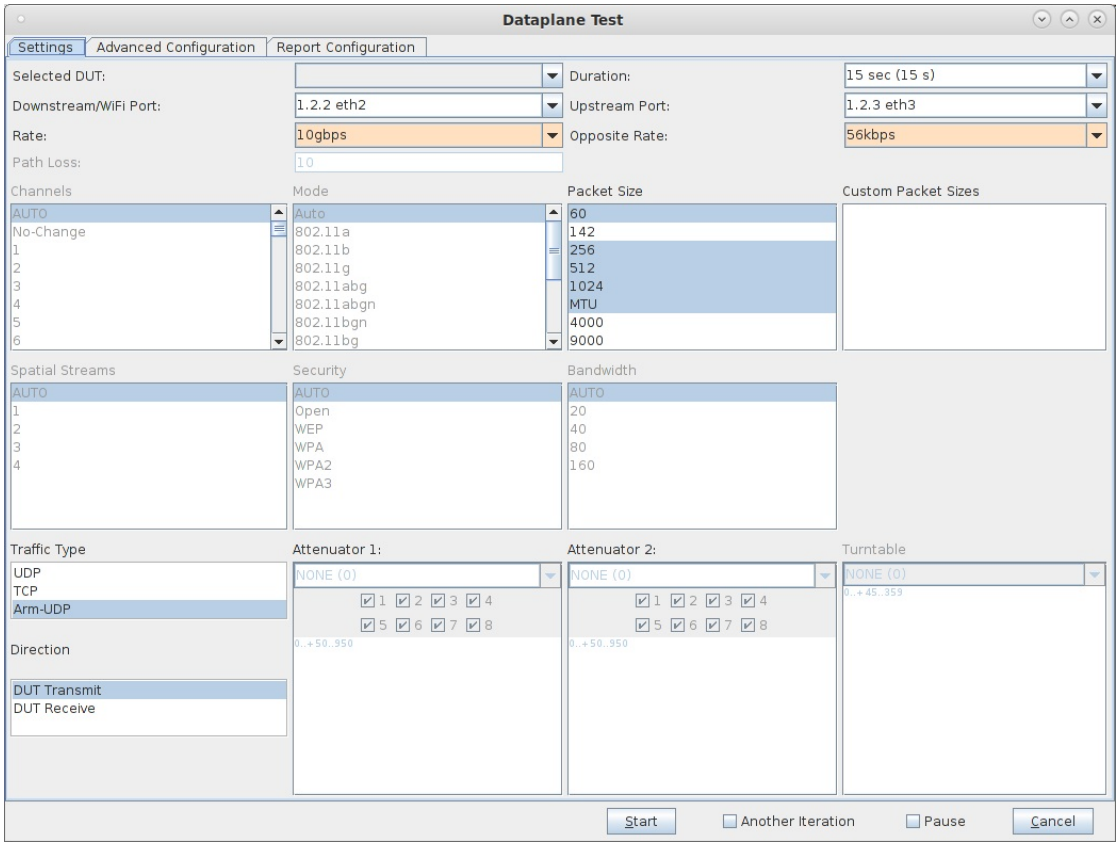




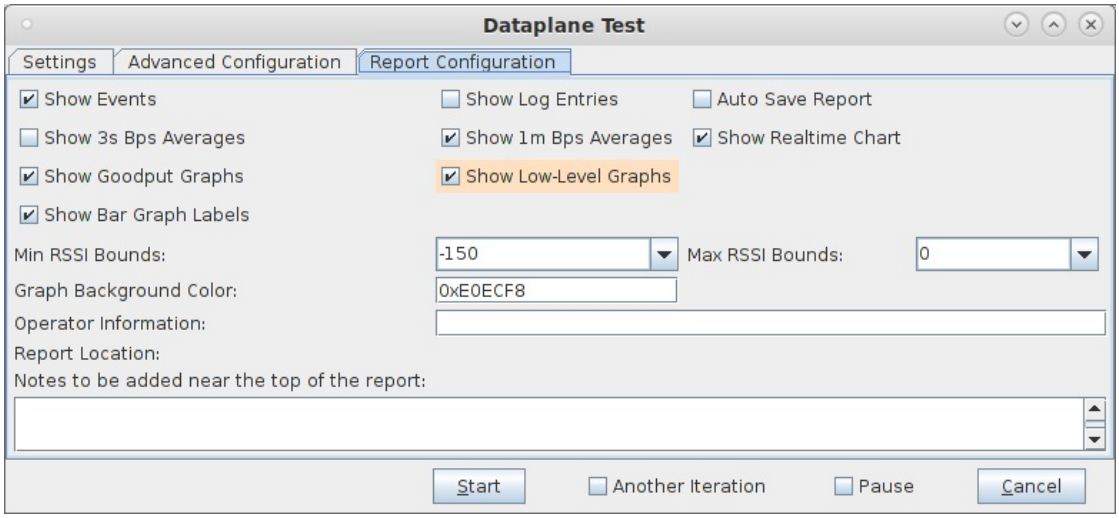
1. Use Chamber View to launch Dataplane test.
    A. Open Chamber View by clicking on the 'Chamber View' button in the LANforge-GUI. You are welcome to build a scenario, but for this particular test, a scenario is not required.



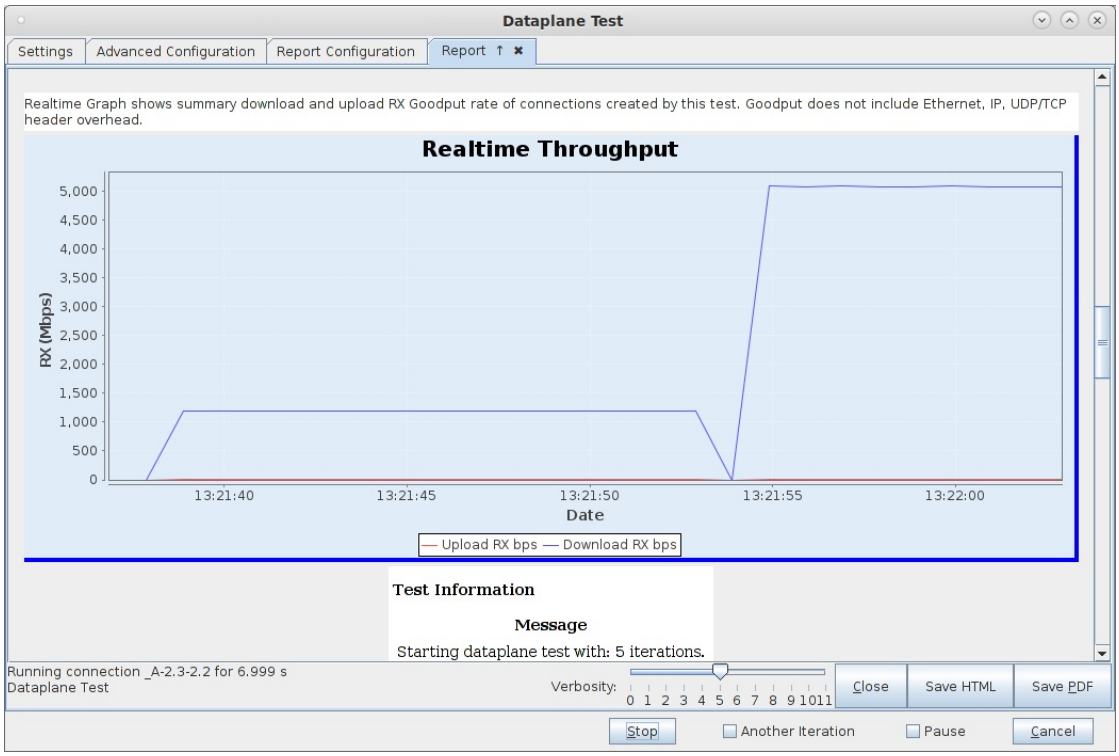
B. Select the **Dataplane** test and click **Run Test**. You should see the Dataplane Test configuration window pop up. It will remember the last configuration for most fields. Select the two 10G ports and the combinations of traffic types and packet sizes you wish to send:



C. You can select some details about how the report is generated on the 'Report Configuration' screen. In this case Low Level statistics is selected so that the on-wire (including IP, UDP, and Ethernet header) is reported:



D. When the configuration is complete, click the **Start** button (which will change to 'Stop' once start is clicked) to start the test. An interactive report window will be created and will be updated as the test runs.

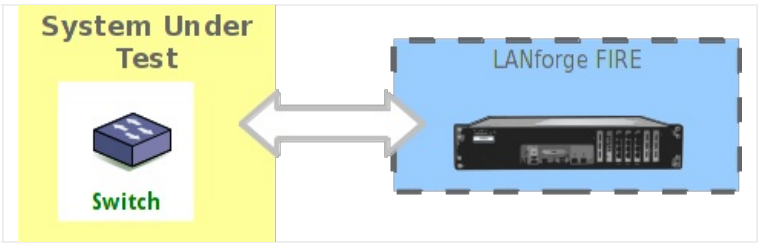


E. When the test is complete, click the **Save HTML** button to save an HTML report and generate the PDF. The PDF file will be linked from the HTML page. You can also click 'Save PDF' and the browser will be directed to open the pdf file directly. Please see this example 10G Dataplane Report.

Generating Traffic to a Switched Network

Goal: Set up and run traffic on a flat network.

In this test scenario, LANforge-FIRE is used to generate traffic between two ports on the same subnet (switch) in order to test throughput.

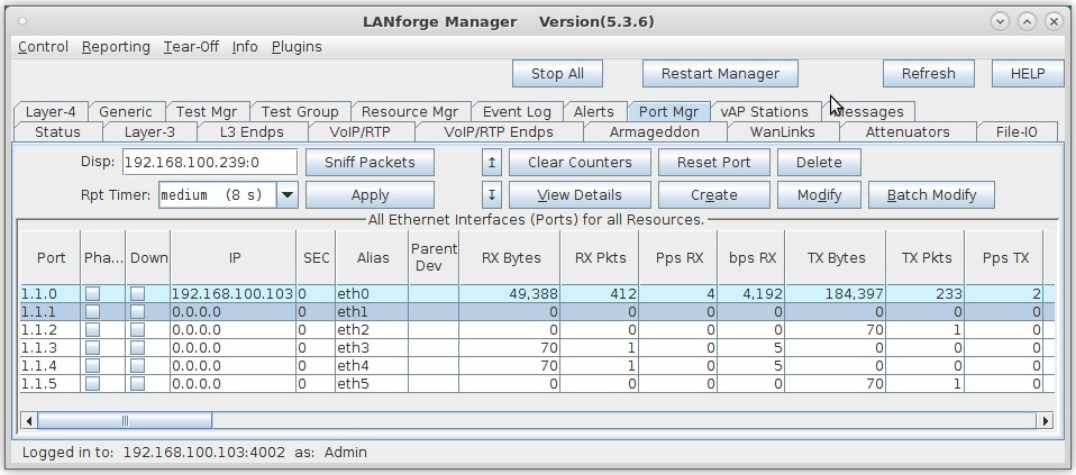


1. Connect two available data generating ports of your LANforge server to the DUT. In this example, the DUT is a switch.

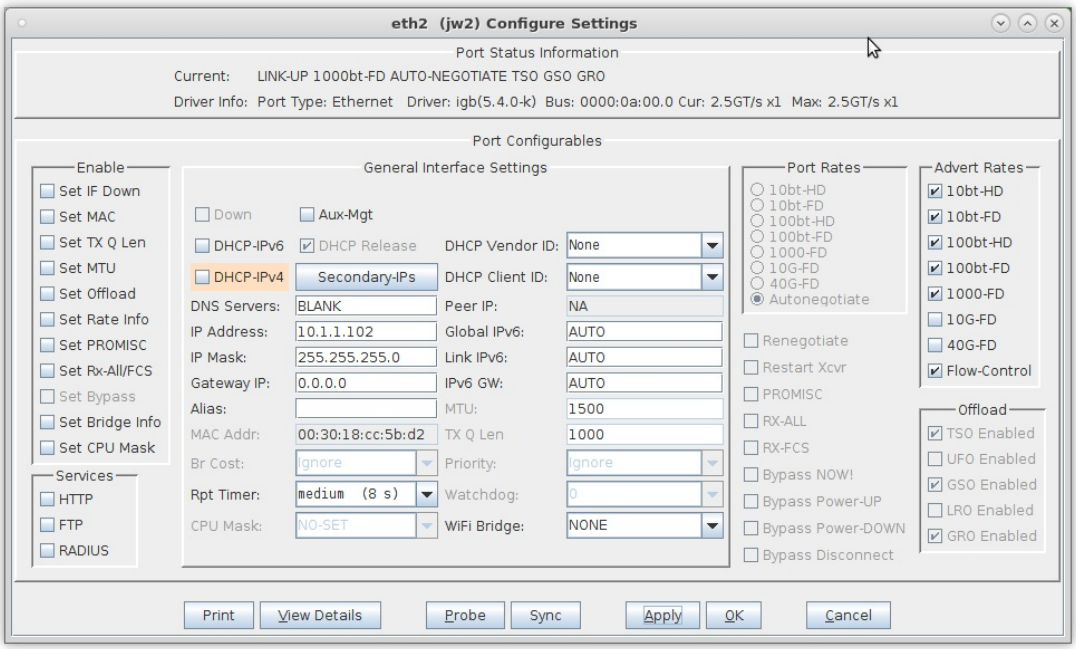
2. Set up the LANforge ports so that they have valid IP addresses. This example uses static IP addresses, but you

can also use DHCP if your network supports that.

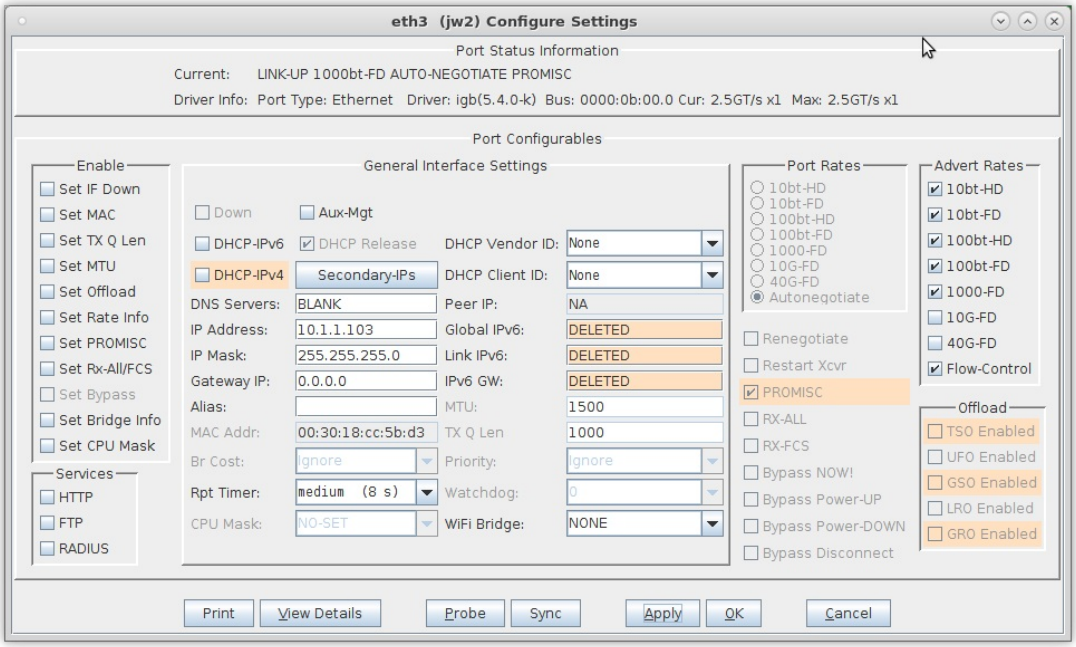
A. Go to the Port Manager



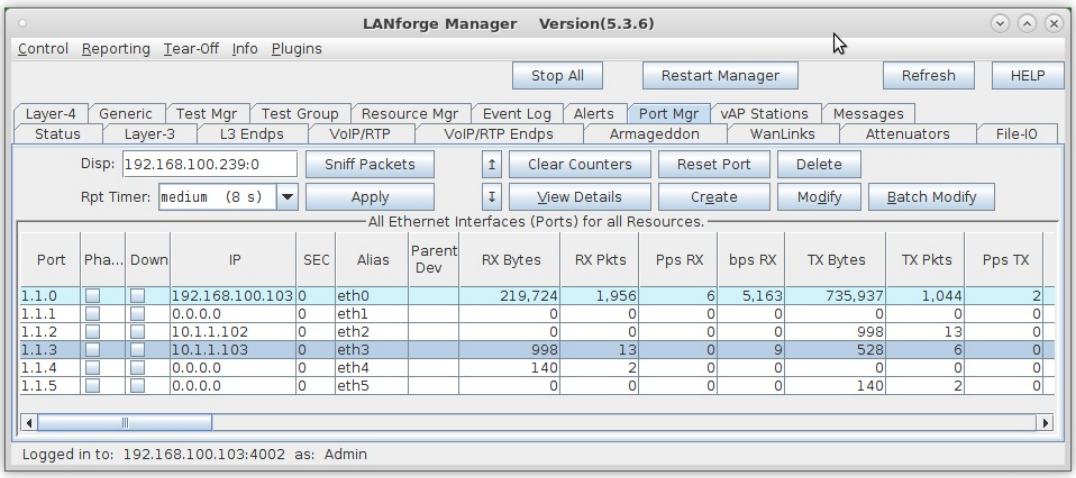
B. Modify port for Endpoint A (eth2)



C. Modify port for Endpoint B (eth3)

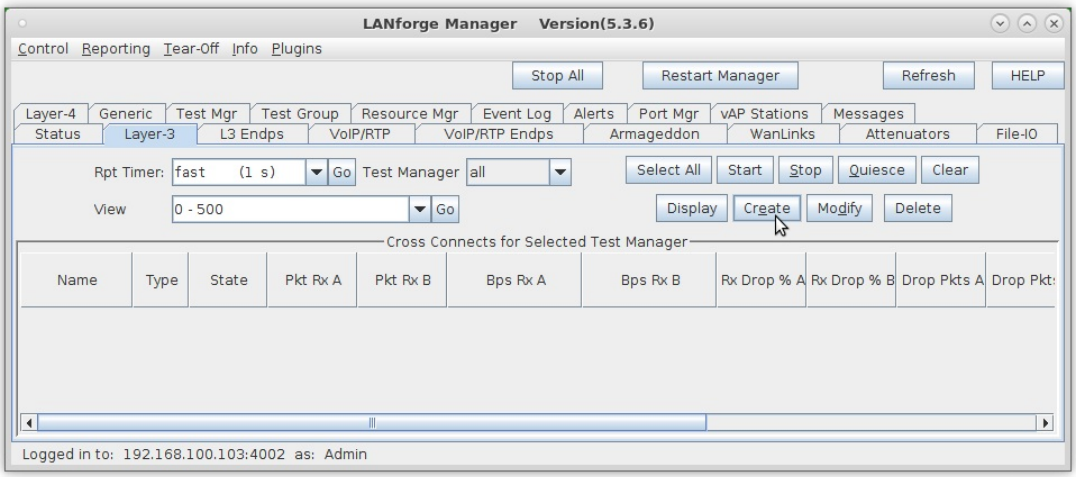


D. Verify the port configuration

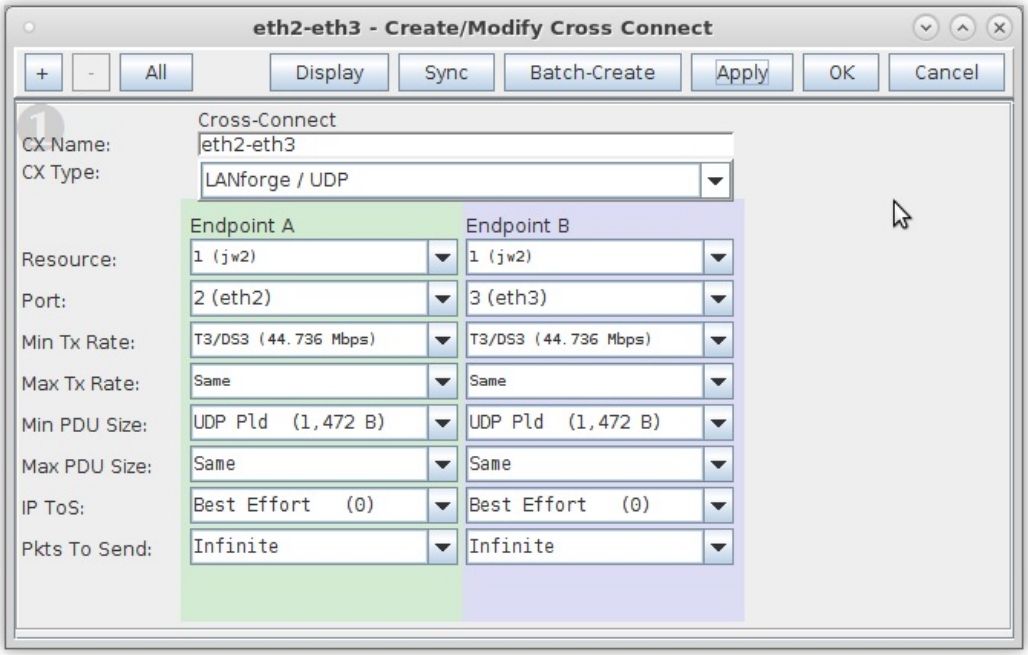


For more information see LANforge User's Guide: Ports (Interfaces)

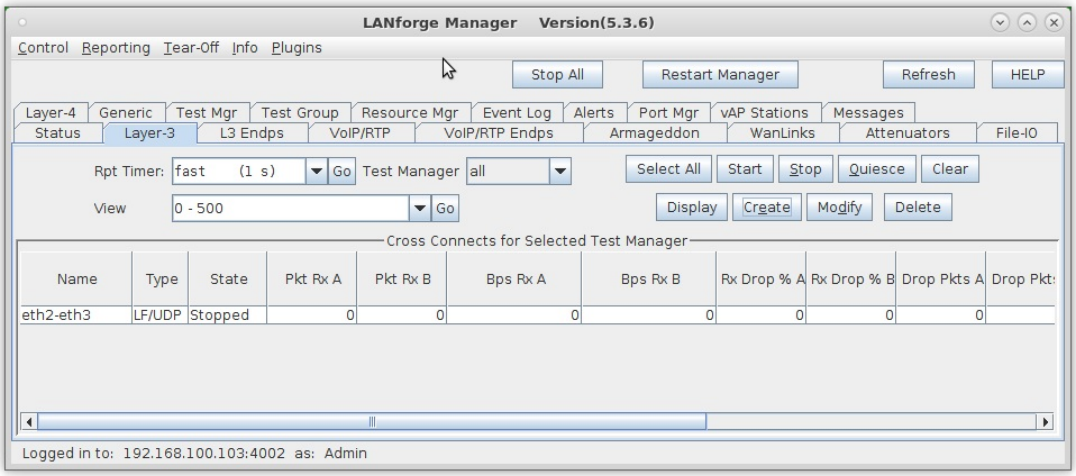
3. Create a Layer-3 connection using the two configured ports.
   A. Go to the **Layer-3** tab



B. Create a new Cross-Connect

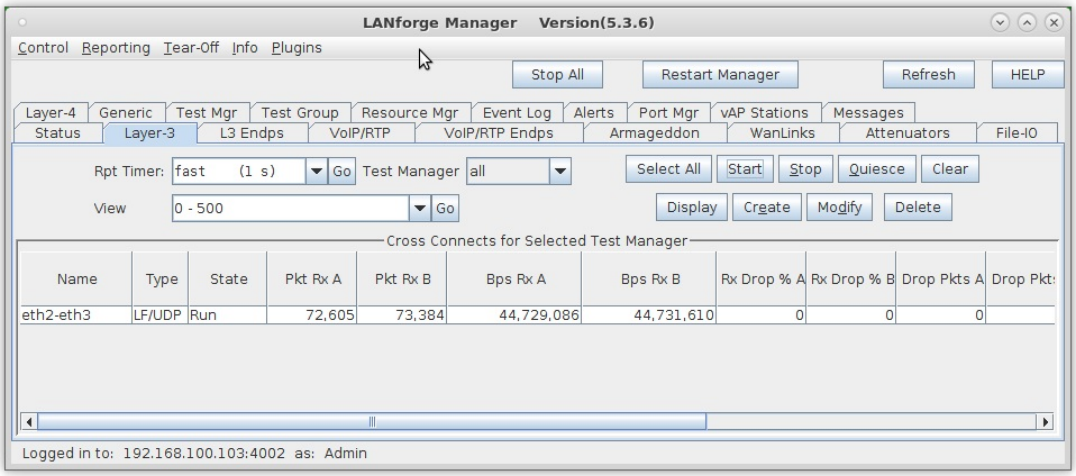


C. Verify the new Cross-Connect

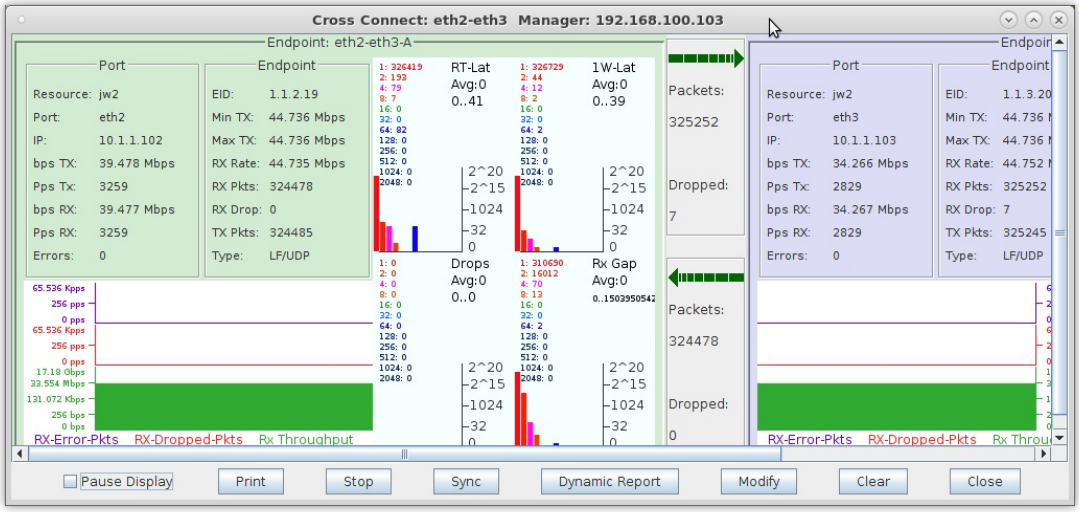


For more information see LANforge User's Guide: Layer-3 Cross-Connects (FIRE)

4. Run traffic and observe results.
   A. Select the cross-connect on the **Layer-3** tab, click **Start** and then **Display**

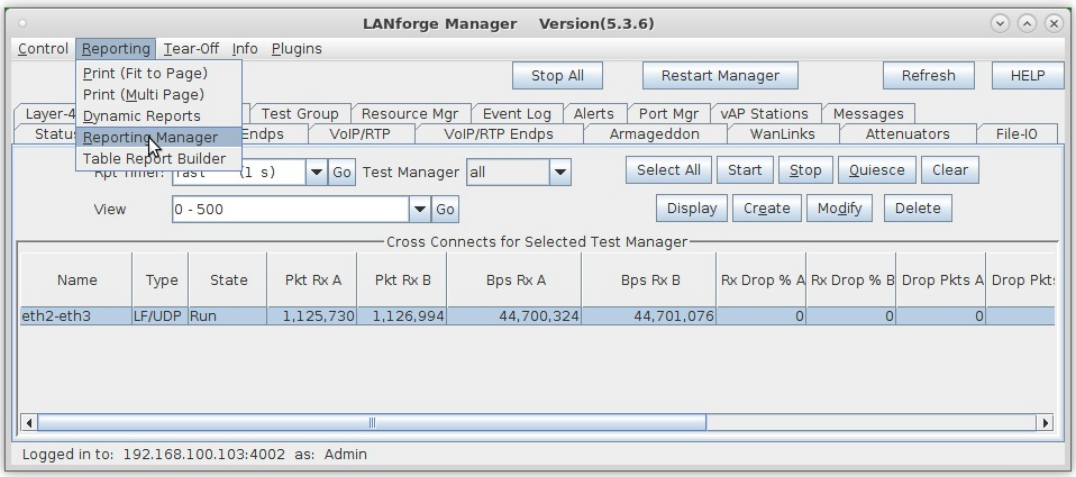


B. View the Layer-3 cross-connect display

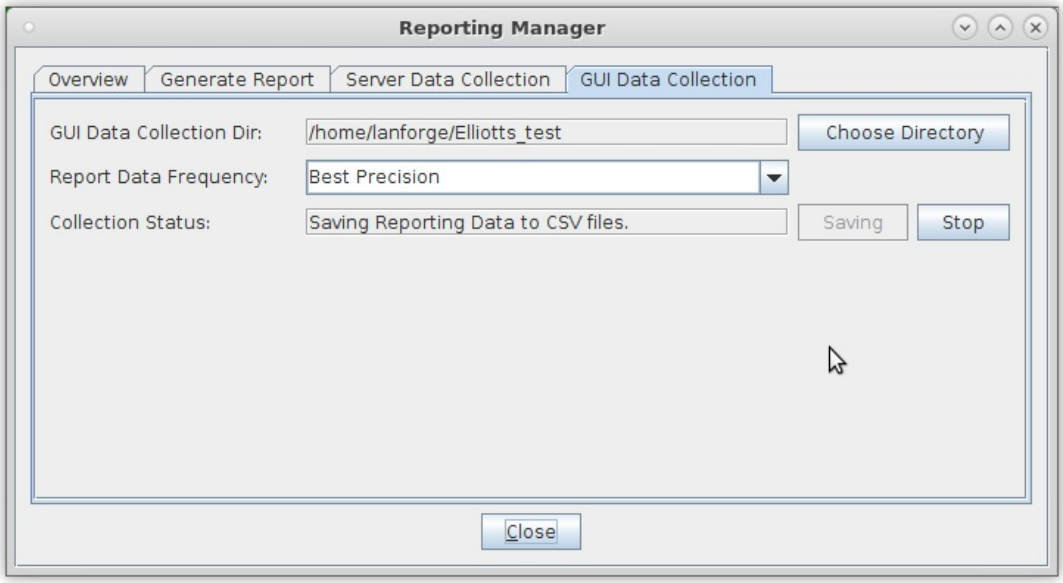


For more information see LANforge User's Guide: Layer-3 Cross-Connect Display

5. In this case, the overall throughput of the DUT is approximately 76Mbps. The Report Manager can also be used to record the throughput over time.
   A. Select **Reporting Manager** from the **Reporting** pull-down menu



B. Record test data by following these steps:



A. Select the **GUI Data Collection** tab.

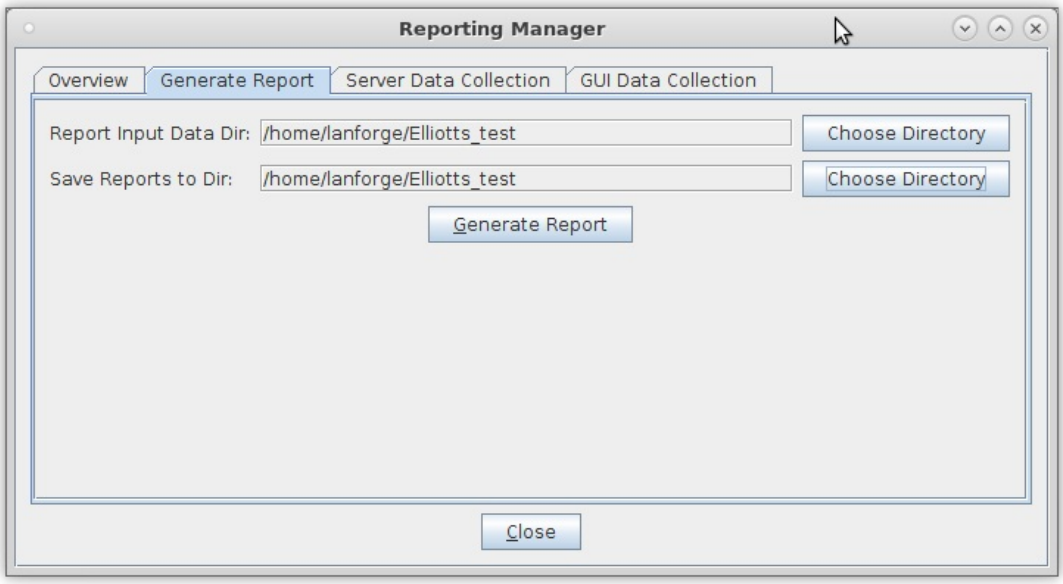
B. Choose a directory to save raw (.csv) data.

C. Click **Save** to start the data collection.

D. After desired test duration, click **Stop**.

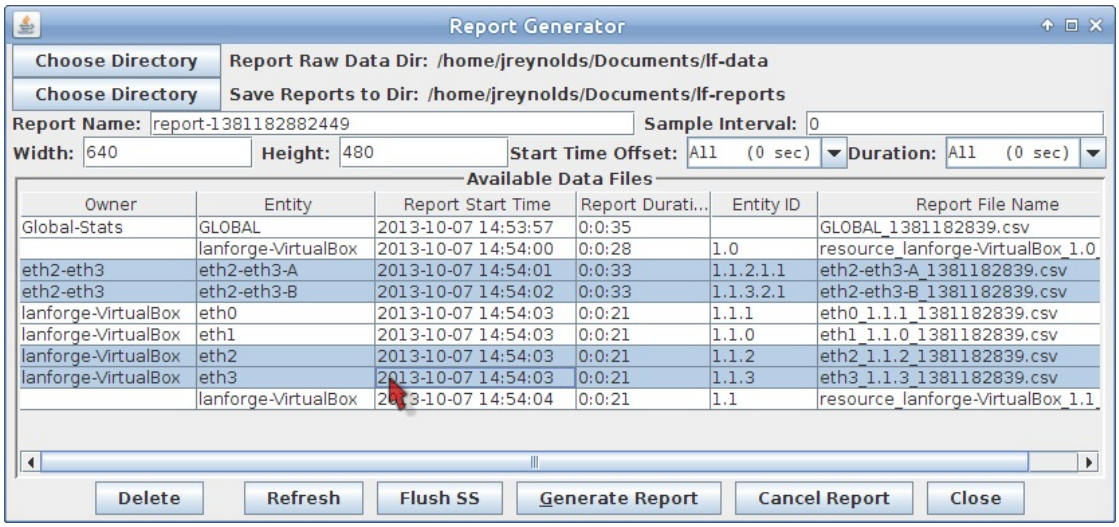
E. **NOTE**: Your report will look cleaner if you stop traffic about 2 minutes before you stop saving data.

C. In the Generate Report tab, choose a directory to save the HTML report. **NOTE:** Make sure your directory name does not include any special charicters such as apostrophes.

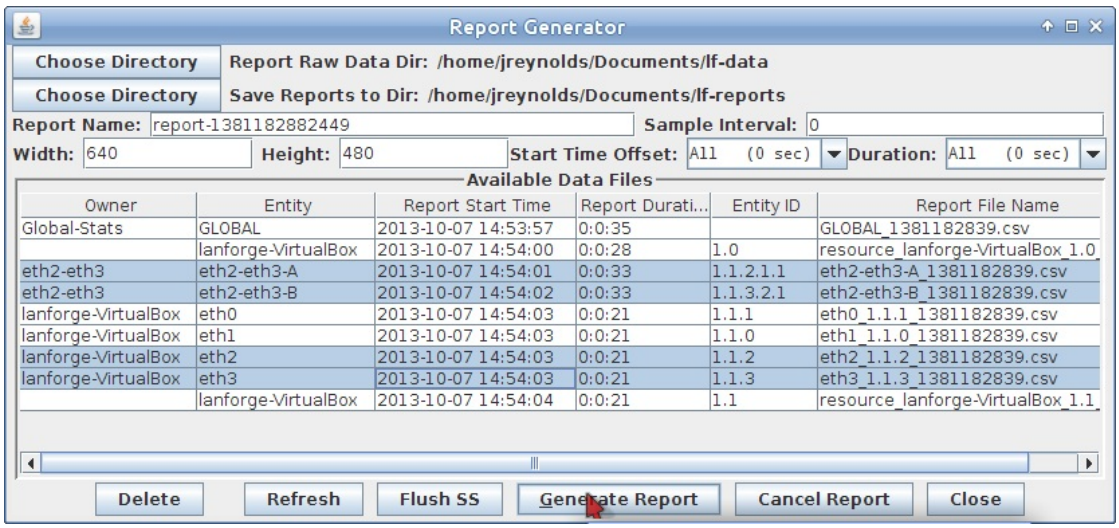


A. Click **Generate Report**.

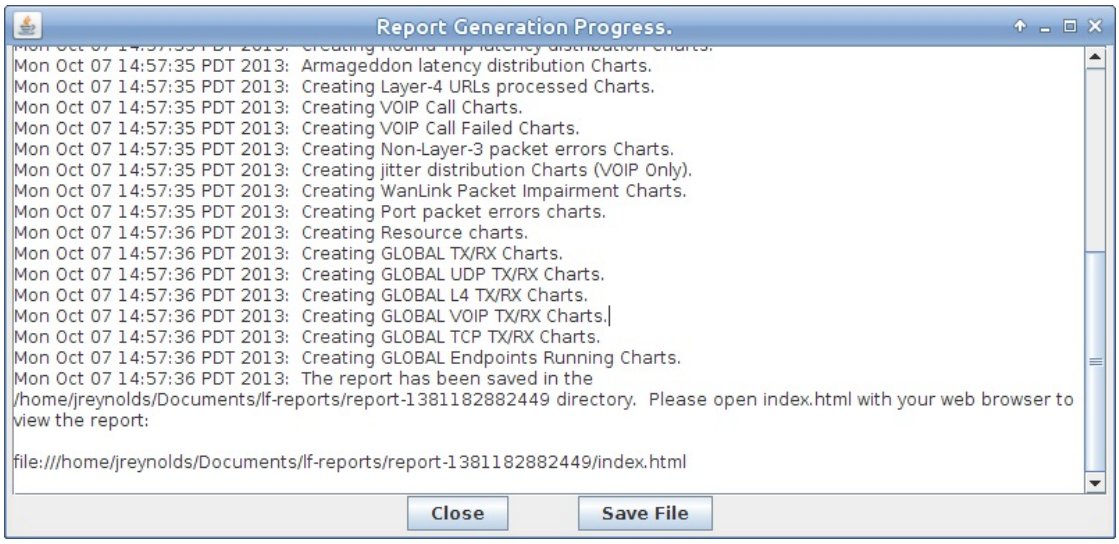
D. Select the entities to be included



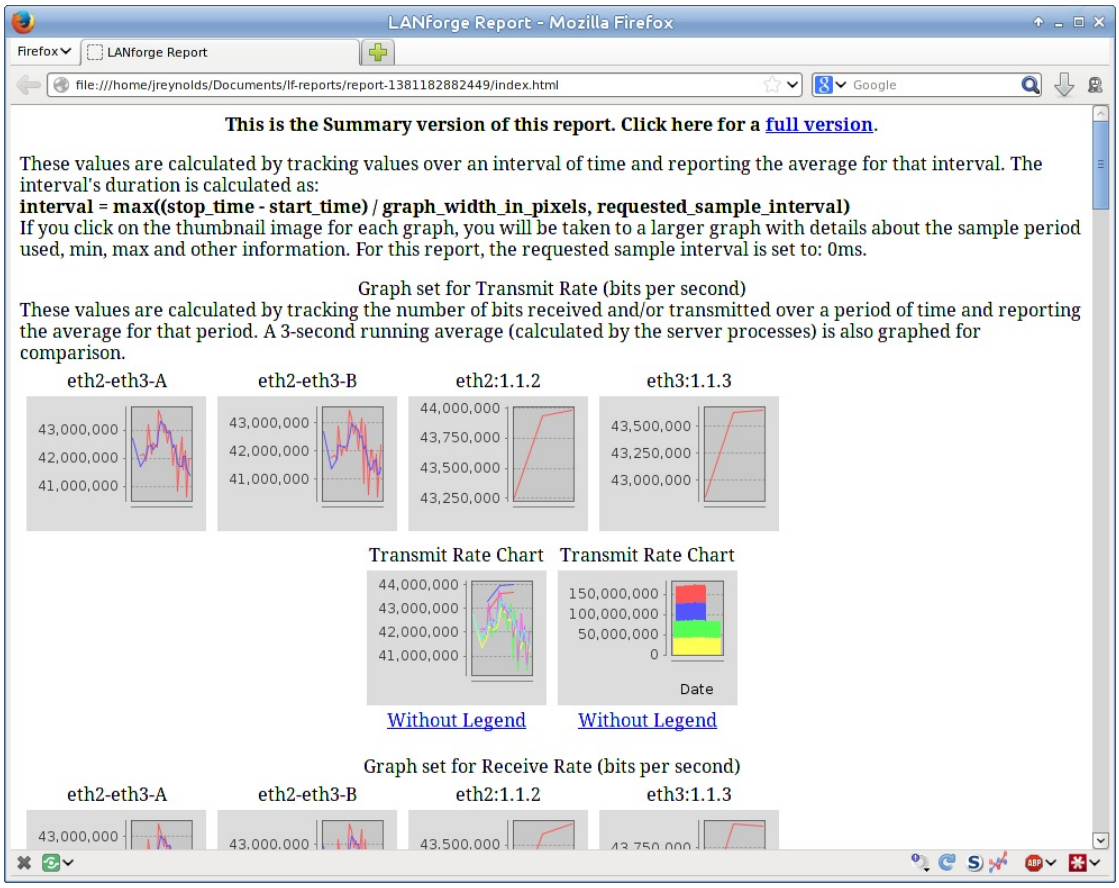
E. Click **Generate Report** to start the report generation



F. This window lists all the reports being generated. The last line indicates the location of the HTML report.



G. View the HTML report

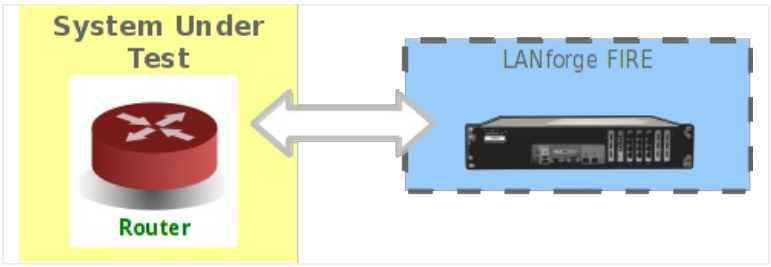


For more information see LANforge User's Guide: Reporting

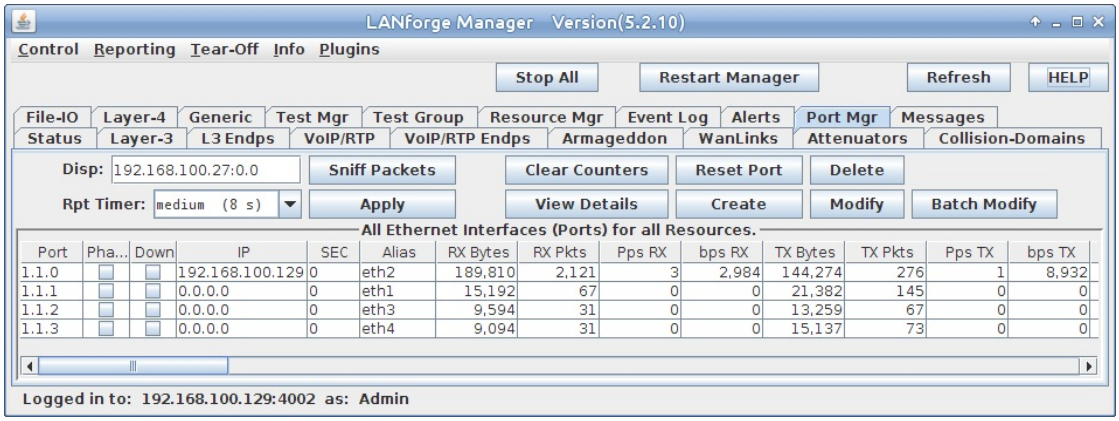
Generating Traffic to a Routed Network

Goal: Set up and run traffic on a routed network.

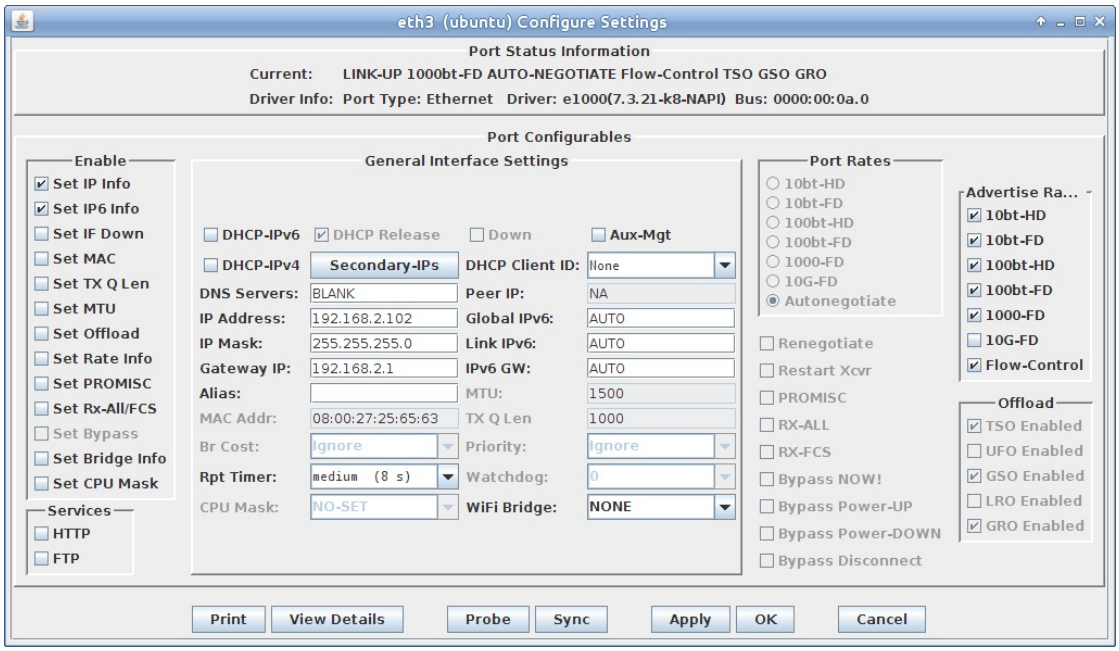
In this test scenario, LANforge-FIRE is used to generate traffic to a basic router in order to test throughput.



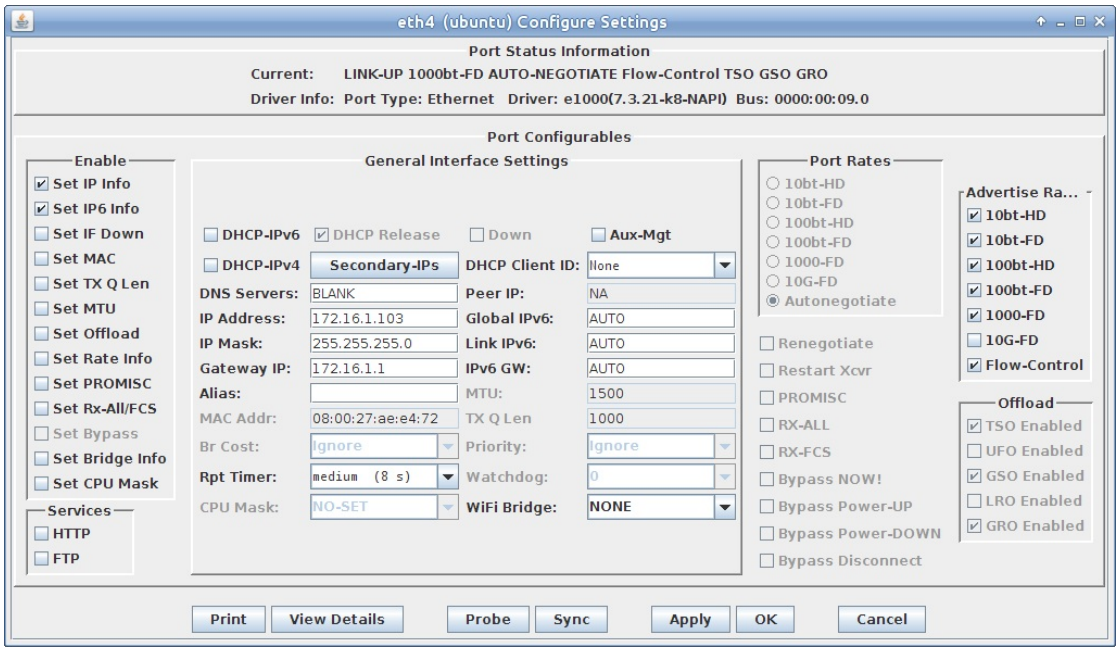
1. Connect one LANforge-FIRE port to the router's LAN port.
2. Connect another LANforge-FIRE port to the router's WAN port.
3. Set up the LANforge ports so that they have valid IP addresses. You can also use DHCP if the DUT supports it.
   A. Go to the Port Manager



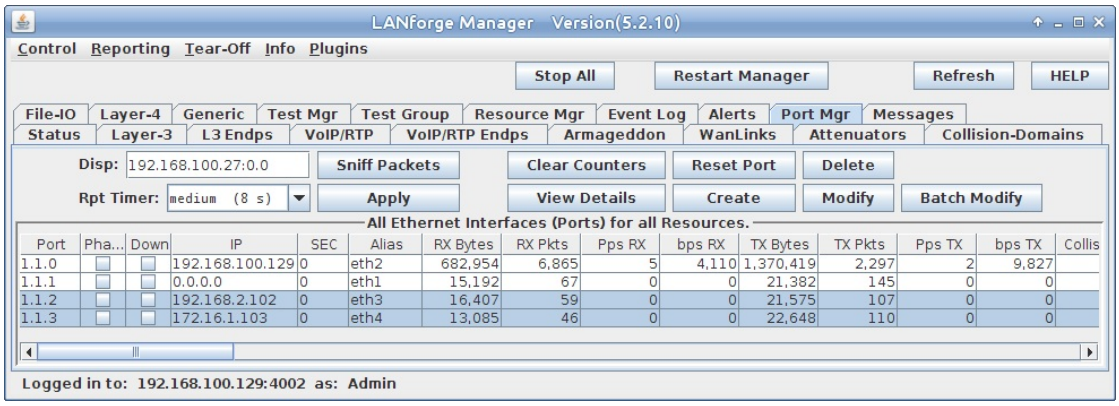
B. Modify port for Endpoint A (eth3). Set a valid network IP Address and Gateway IP.



C. Modify port for Endpoint B (eth4). Set a valid network IP Address and Gateway IP.

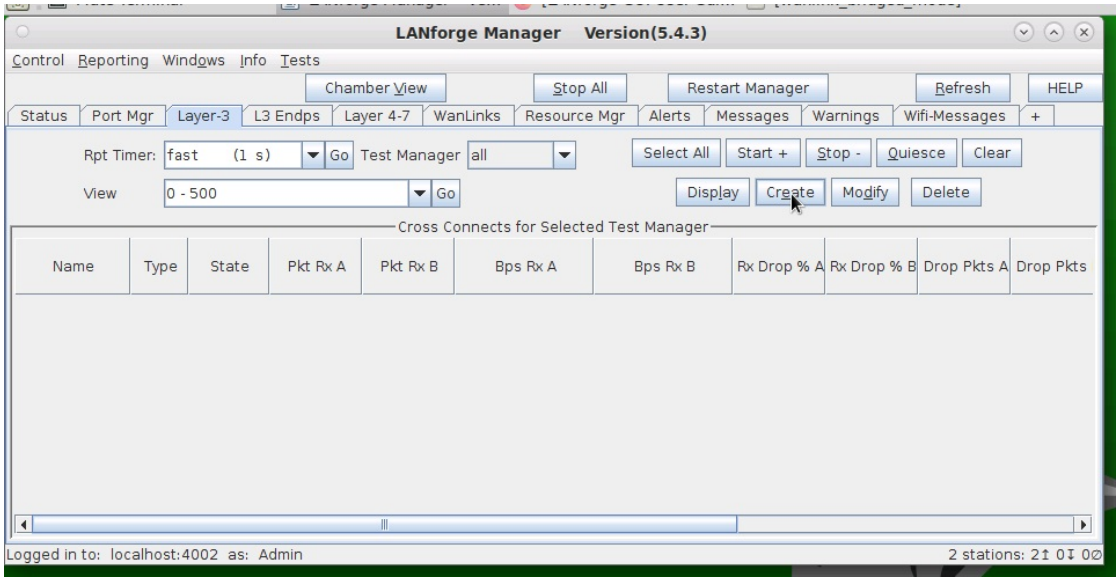


D. Verify the port configuration

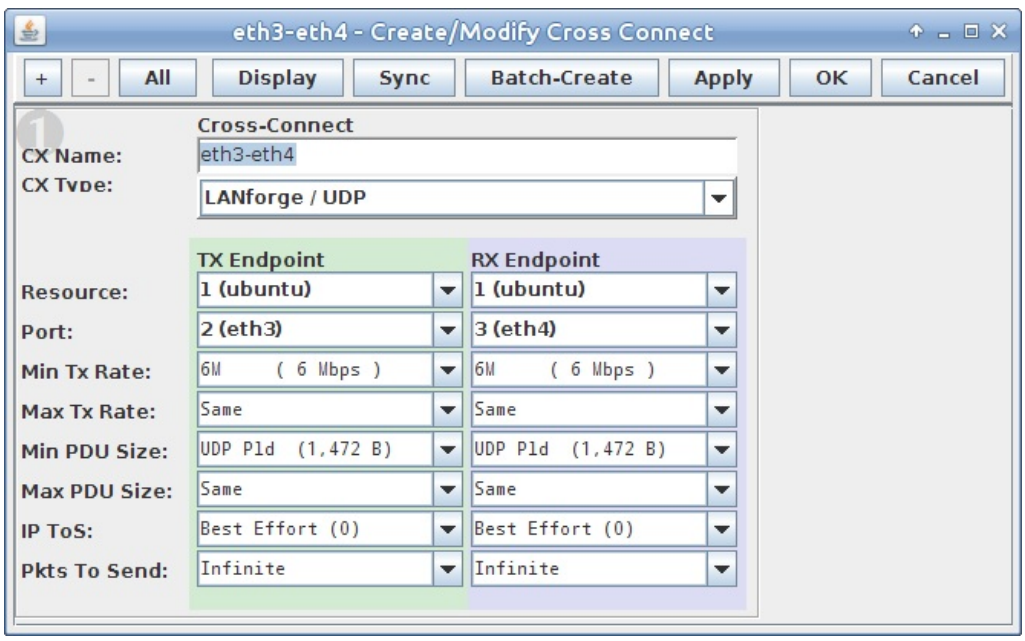


For more information see LANforge User's Guide: Ports(Interfaces)

4. Create a Layer-3 connection using the two configured ports.
   A. Go to the **Layer-3** tab



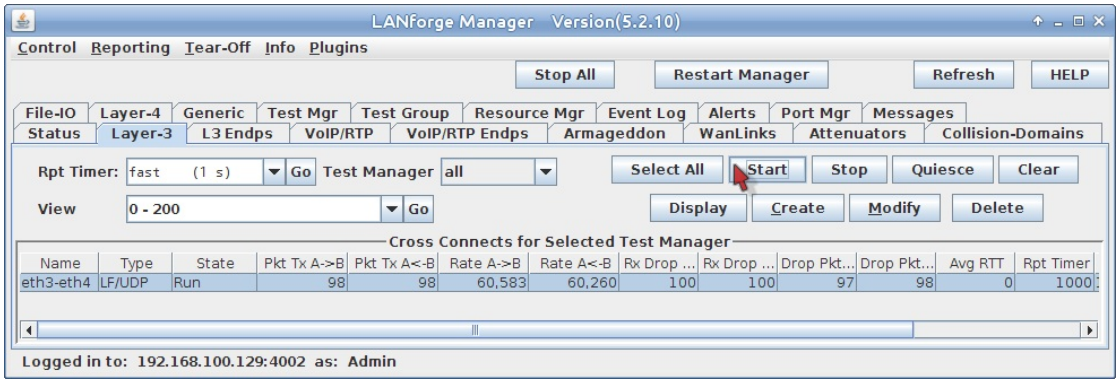
   B. Create a new Cross-Connect



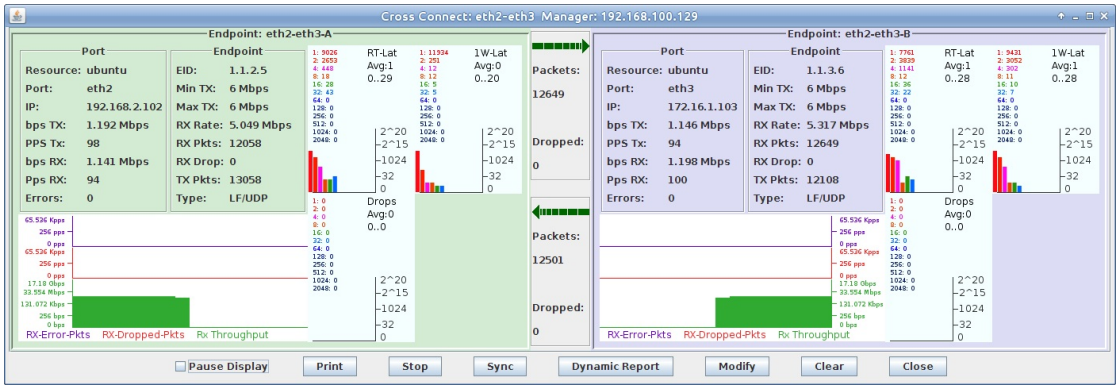
   C. Verify the new Cross-Connect

For more information see LANforge User's Guid: Layer-3 Cross-Connects (FIRE)

5. Run traffic and determine router throughput.
   A. Select the cross-connect on the **Layer-3** tab, click **Start** and then **Display**

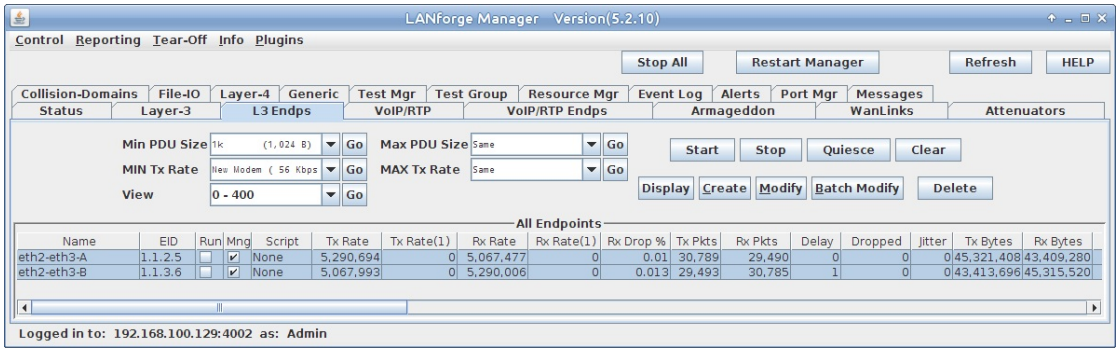


   B. View the Layer-3 cross-connect display

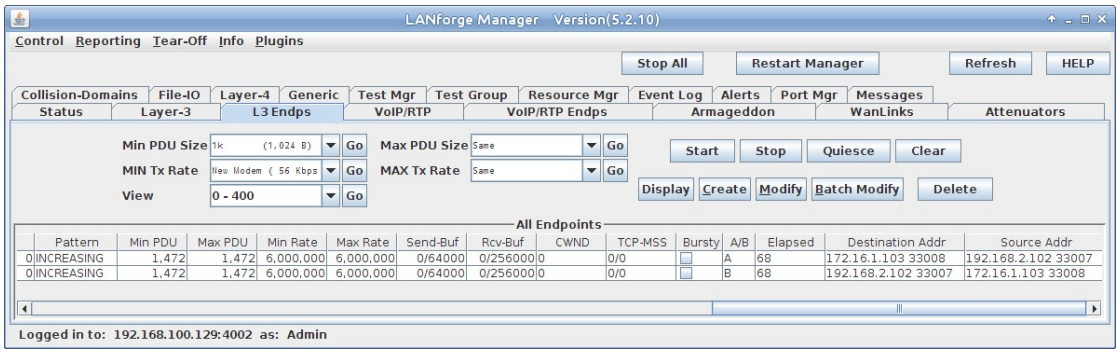


For more information see LANforge User's Guid: Layer-3 Cross-Connect Display

6. For this example, a low performance router was used to illustrate poor throughput, variable latency, and dropped packets.
   A. Go to the **L3 Endps** tab



B. Scroll to the right to view Latency and Dropped Packets



For more information see LANforge User's Guid: Layer-3 Endpoints (FIRE)

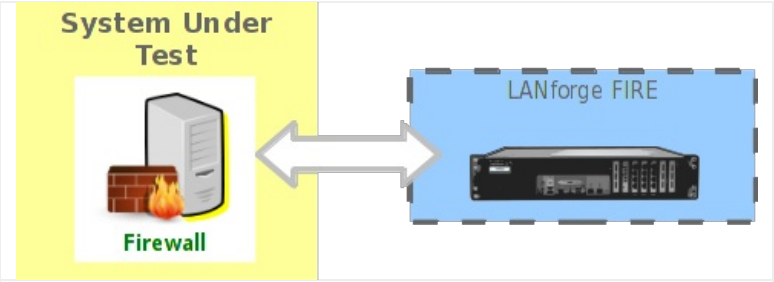
Generating Traffic to a Firewall

Goal: Set up and run traffic to test a firewall.

In this test scenario, LANforge-FIRE is used to generate traffic to a firewall DUT in order to measure the following benchmarks:

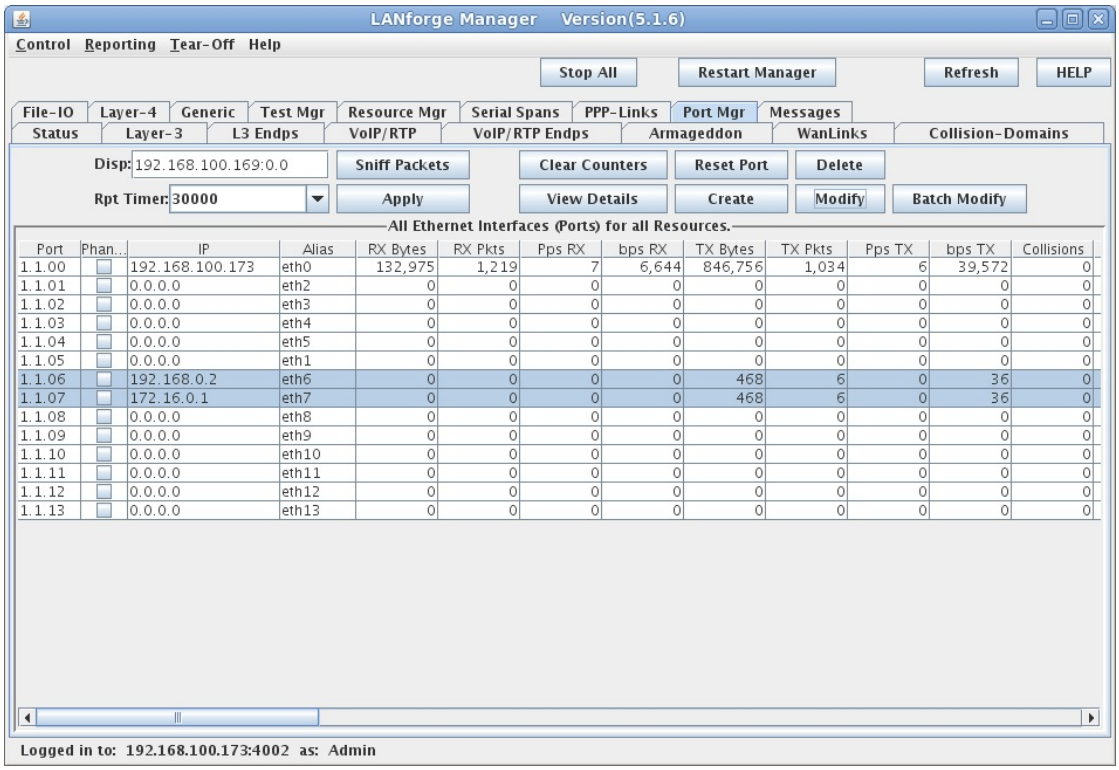
- **UDP Throughput** - Maximum payload bits per second with a UDP traffic flow.
- **TCP Throughput** - Maximum payload bits per second with a TCP traffic flow.
- **TCP Concurrent Connections** - Maximum number of simultaneous TCP connections.
- **TCP Connections per Second** - Maximum number of established TCP connections per second.

NOTE: If you are attempting to run this test scenario, you will need a LANforge license key that enables the correct number of ports and multi-connections. Please contact us at support@candelatech.com for assistance.



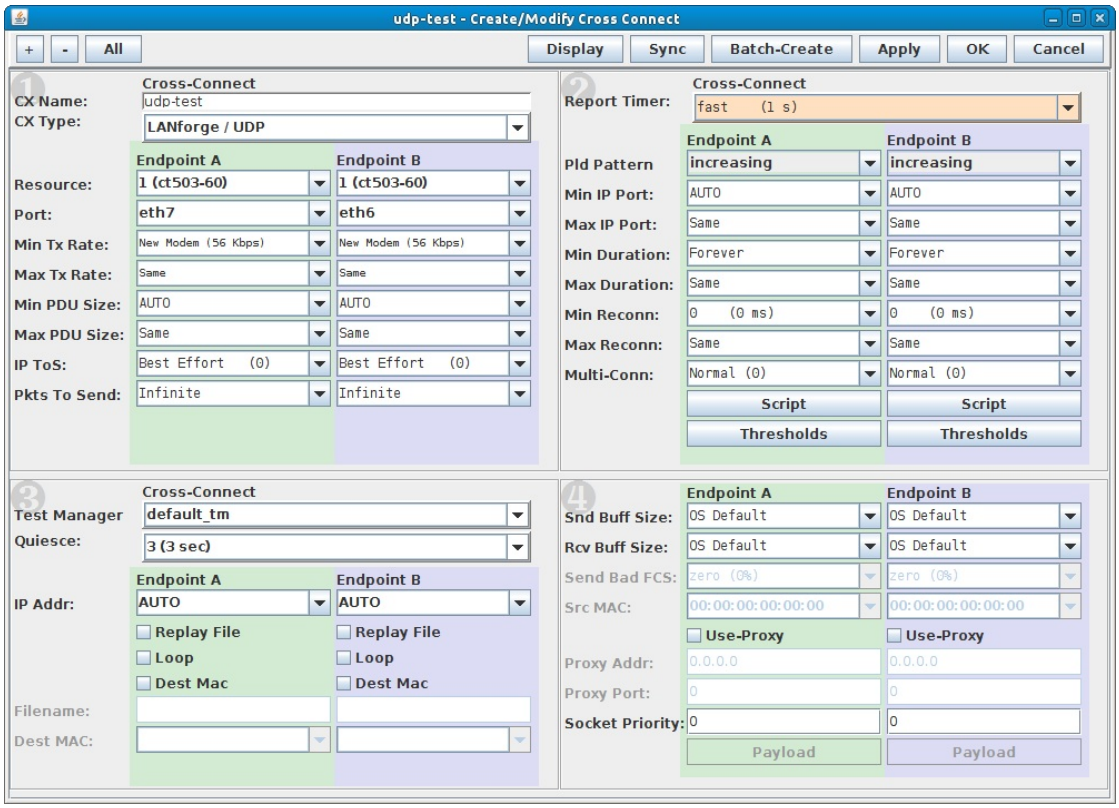
1. The **UDP Throughput** test will use a scripted Layer-3 connection to vary the rate and payload size to determine the bi-directional UDP throughput of the DUT across the scripted parameters.

A. On the Port Manager tab, set up the LANforge ports with valid IP addresses.



Port	Phan...	IP	Alias	RX Bytes	RX Pkts	Pps RX	bps RX	TX Bytes	TX Pkts	Pps TX	bps TX	Collisions
1.1.00		192.168.100.173	eth0	132,975	1,219	7	6,644	846,756	1,034	6	39,572	0
1.1.01		0.0.0.0	eth2	0	0	0	0	0	0	0	0	0
1.1.02		0.0.0.0	eth3	0	0	0	0	0	0	0	0	0
1.1.03		0.0.0.0	eth4	0	0	0	0	0	0	0	0	0
1.1.04		0.0.0.0	eth5	0	0	0	0	0	0	0	0	0
1.1.05		0.0.0.0	eth1	0	0	0	0	0	0	0	0	0
1.1.06		192.168.0.2	eth6	0	0	0	0	468	6	0	36	0
1.1.07		172.16.0.1	eth7	0	0	0	0	468	6	0	36	0
1.1.08		0.0.0.0	eth8	0	0	0	0	0	0	0	0	0
1.1.09		0.0.0.0	eth9	0	0	0	0	0	0	0	0	0
1.1.10		0.0.0.0	eth10	0	0	0	0	0	0	0	0	0
1.1.11		0.0.0.0	eth11	0	0	0	0	0	0	0	0	0
1.1.12		0.0.0.0	eth12	0	0	0	0	0	0	0	0	0
1.1.13		0.0.0.0	eth13	0	0	0	0	0	0	0	0	0

B. On the Layer-3 tab, create a UDP connection.

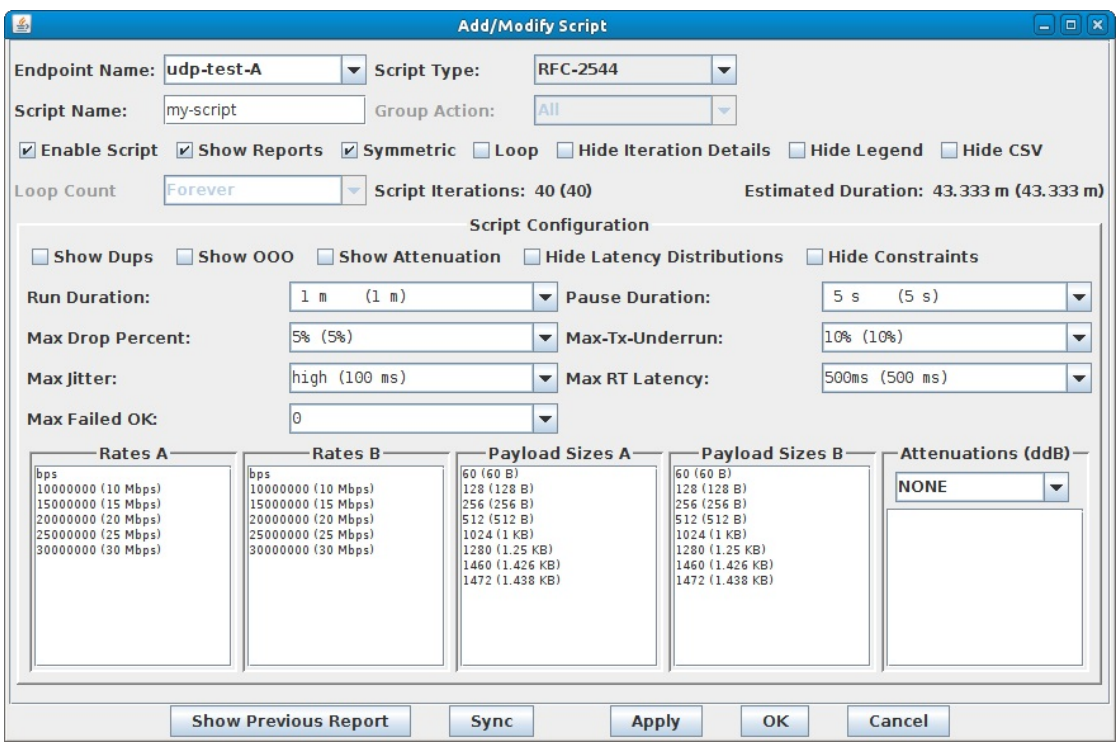


A. Set Endpoint A to use the DUT WAN port.

B. Set Endpoint B to use the DUT LAN port.

C. Make sure CX Type is `LANforge / UDP` then select **Apply**.

C. Select the **Script** button to set up the scripting parameters.

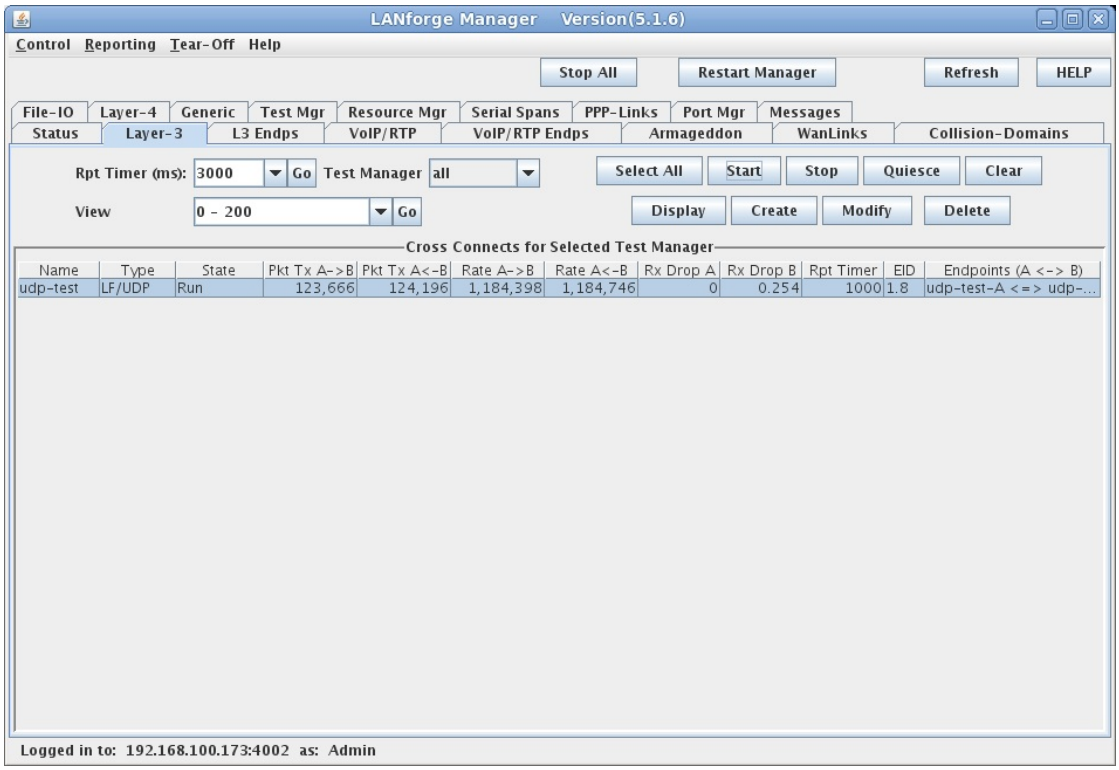


A. Set up the script to iterate over the rates.

B. Specify the payload sizes to be tested.

For more information see LANforge FIRE Cookbook example: Scripted Layer-3 Test

D. Highlight the connection and select the **Start** button.



E. The final test report shows the results of the test run. Here we can see that the DUT has the best bi-directional throughput with 1460Byte payloads at 24.9Mbps.

Script Report for Endpoint: udp-test-A

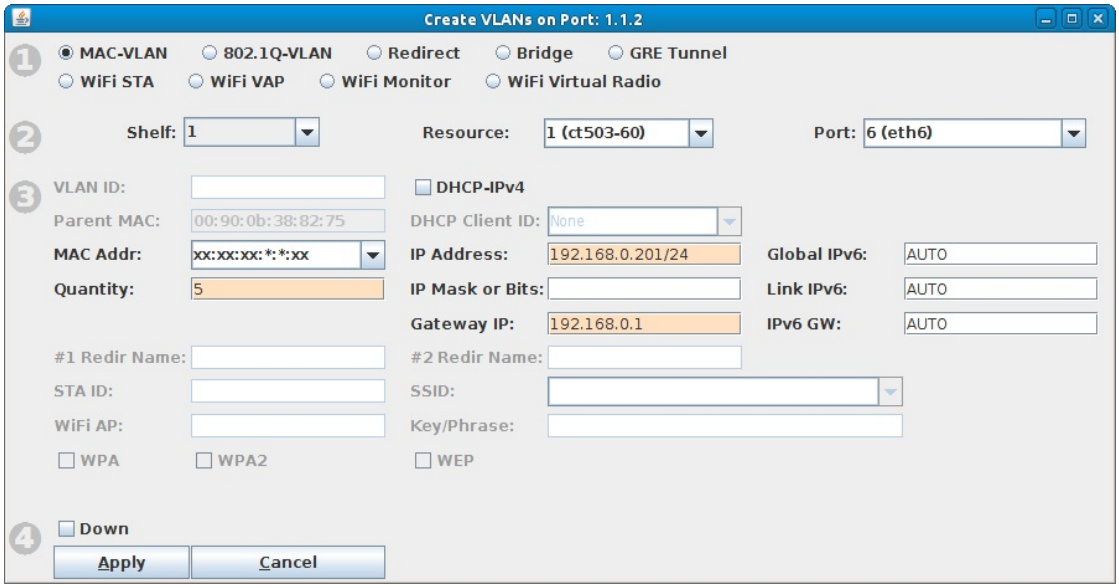
```
Started test at: Thu Sep 23 07:28:03 2010
Iteration Duration: 60000ms   Pause Duration: 5000ms
Number of running endpoints at end of first iteration: 2
System Load at end of first iteration: 0.05

Endpoint Information:
  Endpoint ID: udp-test-A     Type: LANFORGE_UDP     Peer Endpoint ID: udp-test-B


Summary data for each iteration:
```

## -	pld-size (bytes)	cfg-rate (bps)	tx-bps -	rx-bps peer	rx-bps-LL peer	tx-pps -	rx-pps peer	tx-pkts -	rx-pkts peer	cx-drops peer	drop% peer	rx-lat(ms) peer
0	60	10000000	1313968	1313832	0	2737	2737	164246	164229	17	0.010	319
1	128	10000000	2787447	2787447	0	2722	2722	163327	163327	0	0.000	263
2	256	10000000	5516322	5516322	0	2694	2694	161611	161611	0	0.000	205
3	512	10000000	9993967	9993967	0	2440	2440	146396	146396	0	0.000	0
4	1024	10000000	9993967	9993967	0	1220	1220	73198	73198	0	0.000	0
5	1280	10000000	9994581	9994581	0	976	976	58562	58562	0	0.000	0
6	1460	10000000	9993019	9993019	0	856	856	51334	51334	0	0.000	0
7	1472	10000000	9994487	9994487	0	849	849	50923	50923	0	0.000	0
8	60	15000000	1317586	1317586	0	2745	2745	164701	164701	0	0.000	443
9	128	15000000	2792550	2792550	0	2727	2727	163626	163626	0	0.000	370
10	256	15000000	5523149	5523149	0	2697	2697	161811	161811	0	0.000	288
11	512	15000000	10815556	10815556	0	2641	2641	158431	158431	0	0.000	204
12	1024	15000000	14991497	14991497	0	1830	1830	109801	109801	0	0.000	0
13	1280	15000000	14991019	14991019	0	1464	1464	87838	87838	0	0.000	0
14	1460	15000000	14990696	14990696	0	1283	1283	77007	77007	0	0.000	0
15	1472	15000000	14989082	14989082	0	1273	1273	76371	76371	0	0.000	0
16	60	20000000	19984952	1319512	0	41635	2749	2498119	164939	2333180	93.397	471
17	128	20000000	19985510	2825114	0	19517	2759	1171026	165534	1005492	85.864	449
18	256	20000000	5532058	5532058	0	2701	2701	162072	162072	0	0.000	371
19	512	20000000	10823885	10823885	0	2643	2643	158553	158553	0	0.000	263
20	1024	20000000	19985203	19985203	0	2440	2440	146376	146376	0	0.000	1
21	1280	20000000	19986261	19986261	0	1952	1952	117107	117107	0	0.000	0
22	1460	20000000	19987400	19987400	0	1711	1711	102675	102675	0	0.000	0
23	1472	20000000	19986620	19986620	0	1697	1697	101834	101834	0	0.000	0
24	60	25000000	24979984	1343464	0	52042	2799	3122498	167933	2954565	94.622	469
25	128	25000000	24983006	2851755	0	24397	2785	1463848	167095	1296753	88.585	451
26	256	25000000	19315678	5539908	0	9431	2705	565889	162302	403587	71.319	449
27	512	25000000	10834057	10834057	0	2645	2645	158702	158702	0	0.000	321
28	1024	25000000	20846592	16534323	0	2545	2018	152685	121101	31584	20.686	207
29	1280	25000000	24984576	24984576	0	2440	2440	146394	146394	0	0.000	1
30	1460	25000000	24985856	24985856	0	2139	2139	128352	128352	0	0.000	1
31	1472	25000000	24982588	24982588	0	2121	2121	127289	127289	0	0.000	1
32	60	30000000	29977504	1320720	0	62453	2752	3747188	165090	3582098	95.594	471
33	128	30000000	29979699	2772036	0	29277	2707	1756623	162424	1594199	90.754	457
34	256	30000000	29977498	5595648	0	14637	2732	878247	163935	714312	81.334	447
35	512	30000000	10844706	10844706	0	2648	2648	158858	158858	0	0.000	379
36	1024	30000000	20856757	16537188	0	2546	2019	152762	121124	31638	20.711	247
37	1280	30000000	25577301	18959360	0	2498	1852	149867	111090	38777	25.874	210
38	1460	30000000	28829939	19782611	0	2468	1694	148099	101623	46476	31.382	183
39	1472	30000000	28997811	19803503	0	2462	1682	147747	100901	46846	31.707	182

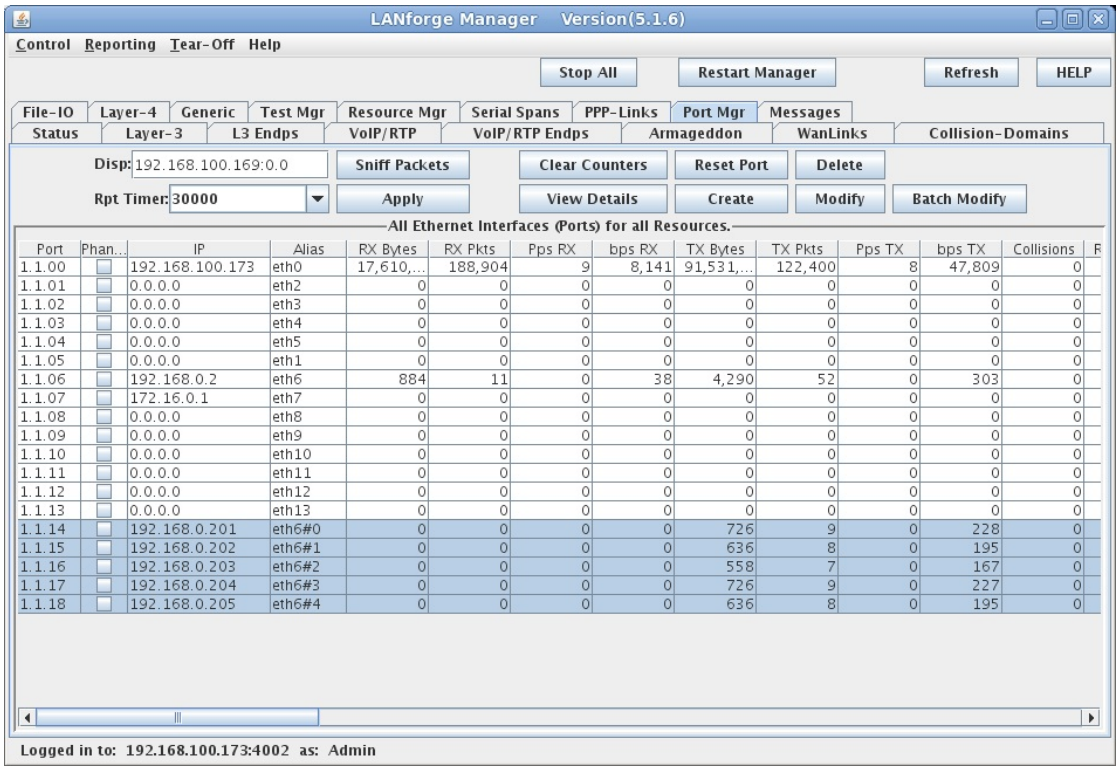
Close Save File

For more information see Full script report for the UDP test.

2. The **TCP Concurrent Connections** test will measure the maximum number of simultaneous TCP connections that the DUT can maintain at once.
   A. On the Port Manager tab, create 5 MAC-VLANs on the LANforge port connected to the DUT LAN port.

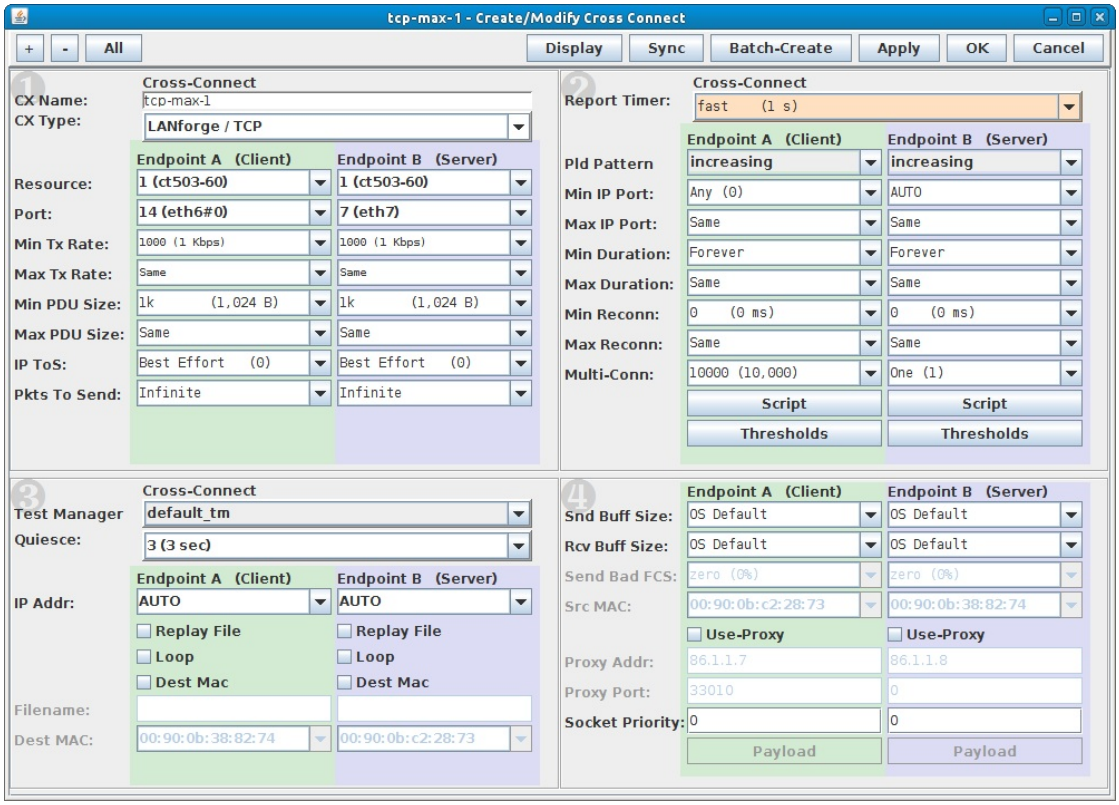


B. Verify that the MAC-VLANs have correct IP addresses.



Port	Phan...	IP	Alias	RX Bytes	RX Pkts	Pps RX	bps RX	TX Bytes	TX Pkts	Pps TX	bps TX	Collisions	R
1.1.00		192.168.100.173	eth0	17,610,...	188,904	9	8,141	91,531,...	122,400	8	47,809	0	
1.1.01		0.0.0.0	eth2	0	0	0	0	0	0	0	0	0	
1.1.02		0.0.0.0	eth3	0	0	0	0	0	0	0	0	0	
1.1.03		0.0.0.0	eth4	0	0	0	0	0	0	0	0	0	
1.1.04		0.0.0.0	eth5	0	0	0	0	0	0	0	0	0	
1.1.05		0.0.0.0	eth1	0	0	0	0	0	0	0	0	0	
1.1.06		192.168.0.2	eth6	884	11	0	38	4,290	52	0	303	0	
1.1.07		172.16.0.1	eth7	0	0	0	0	0	0	0	0	0	
1.1.08		0.0.0.0	eth8	0	0	0	0	0	0	0	0	0	
1.1.09		0.0.0.0	eth9	0	0	0	0	0	0	0	0	0	
1.1.10		0.0.0.0	eth10	0	0	0	0	0	0	0	0	0	
1.1.11		0.0.0.0	eth11	0	0	0	0	0	0	0	0	0	
1.1.12		0.0.0.0	eth12	0	0	0	0	0	0	0	0	0	
1.1.13		0.0.0.0	eth13	0	0	0	0	0	0	0	0	0	
1.1.14		192.168.0.201	eth6#0	0	0	0	0	726	9	0	228	0	
1.1.15		192.168.0.202	eth6#1	0	0	0	0	636	8	0	195	0	
1.1.16		192.168.0.203	eth6#2	0	0	0	0	558	7	0	167	0	
1.1.17		192.168.0.204	eth6#3	0	0	0	0	726	9	0	227	0	
1.1.18		192.168.0.205	eth6#4	0	0	0	0	636	8	0	195	0	

C. Create a Layer-3 connection that has a low-speed rate with **Multi-Conn** set to 10000 and **Min IP Port** to **0 (zero)** on Endpoint-A. **Multi-Conn** should be set to 1 on Endpoint-B.



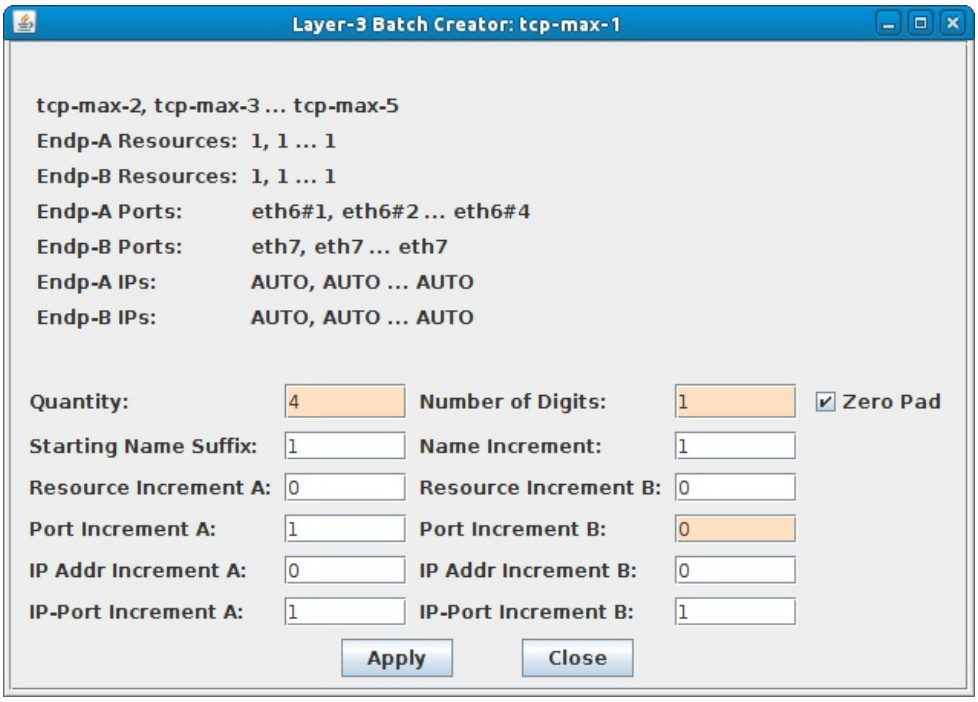
A. Endpoint-A will be one of the MAC-VLANs and Endpoint-B will be the port connected to the DUT WAN port. This setup will initiate the TCP sessions from the LAN side of the DUT.

B. Low-speed depends on the DUT, we could also set the rate to zero which would allow the TCP connections to be set up without payload data to be transmitted, but this would not give an accurate picture of the firewall performance. Here we are using 1Kbps connections with 1KB size payload.

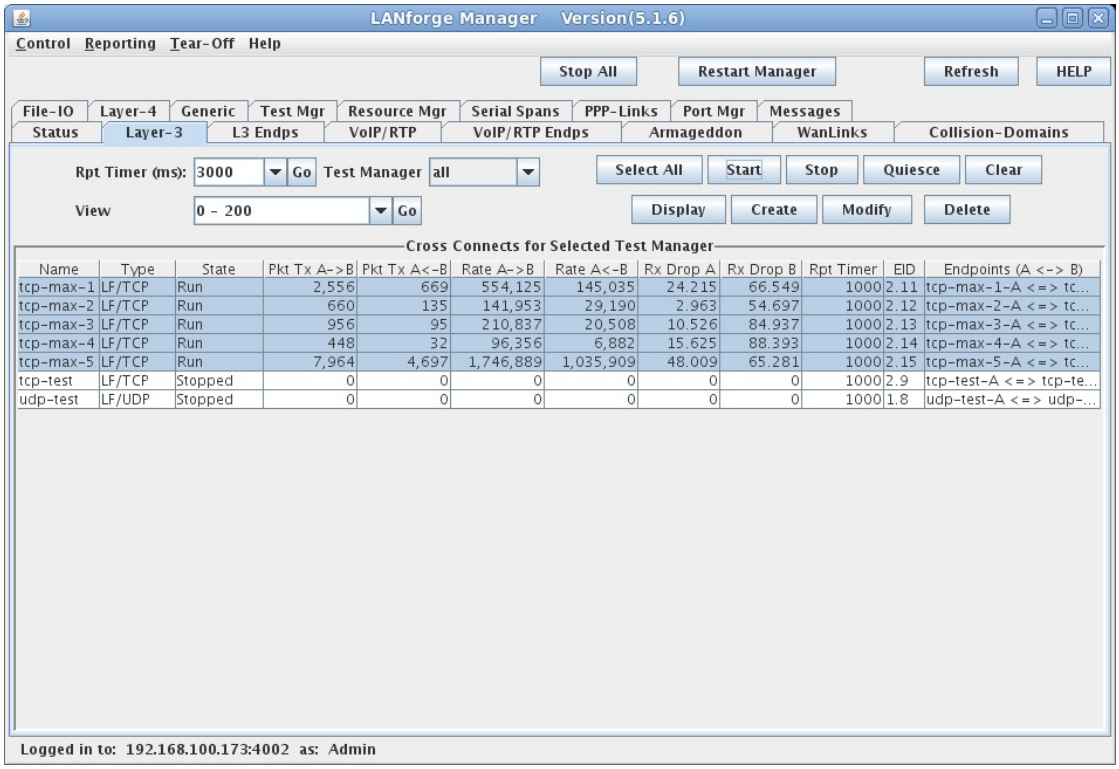
C. This is an iterative test, the number of TCP connections to use will depend on the DUT capabilities. Modify the number of connections as necessary to find the most accurate measurement.

D. The DUT should be power-cycled to reset it before each test run.

D. Select the Batch-Create button to create 4 more copies of this connection each with a new MAC-VLAN port.

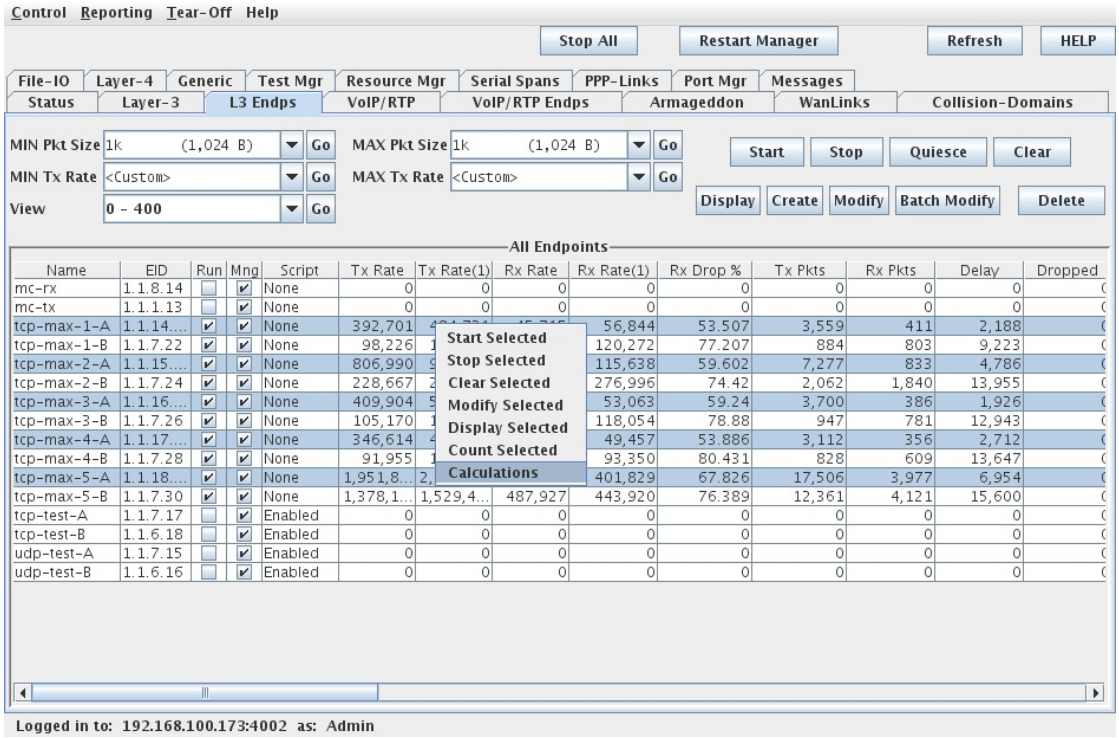


E. Highlight and start each set of 10000 connections until the target max simultaneous connections are running.



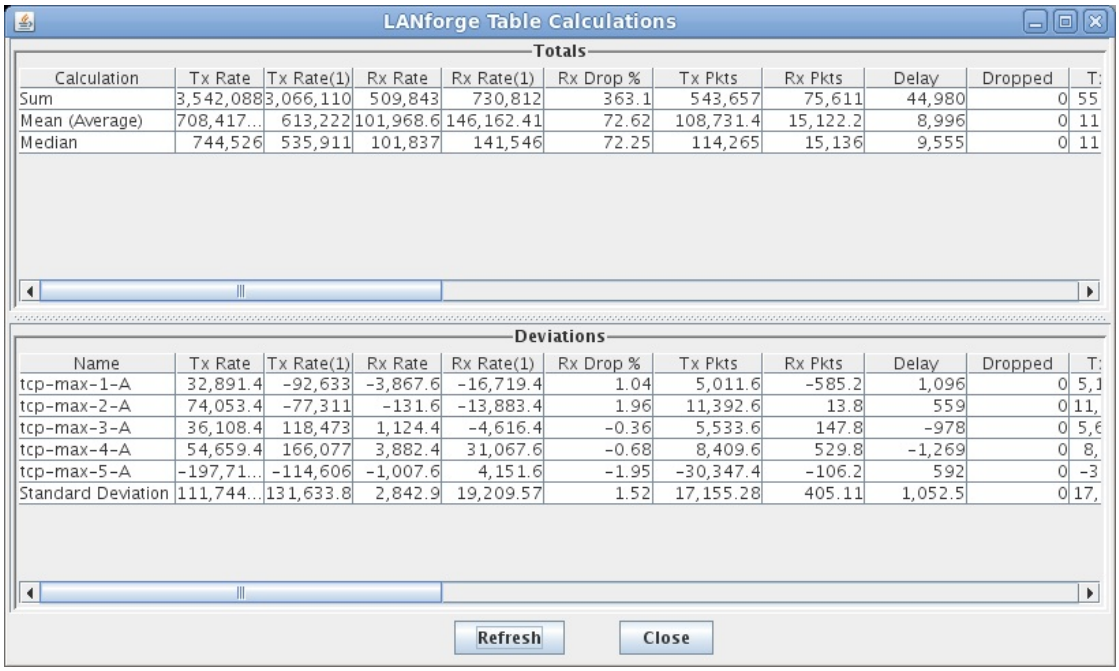
Name	Type	State	Pkt Tx A->B	Pkt Tx A<-B	Rate A->B	Rate A<-B	Rx Drop A	Rx Drop B	Rpt Timer	EID	Endpoints (A <-> B)
tcp-max-1	LF/TCP	Run	2,556	669	554,125	145,035	24.215	66.549	1000	2.11	tcp-max-1-A <=> tc...
tcp-max-2	LF/TCP	Run	660	135	141,953	29,190	2.963	54.697	1000	2.12	tcp-max-2-A <=> tc...
tcp-max-3	LF/TCP	Run	956	95	210,837	20,508	10.526	84.937	1000	2.13	tcp-max-3-A <=> tc...
tcp-max-4	LF/TCP	Run	448	32	96,356	6,882	15.625	88.393	1000	2.14	tcp-max-4-A <=> tc...
tcp-max-5	LF/TCP	Run	7,964	4,697	1,746,889	1,035,909	48.009	65.281	1000	2.15	tcp-max-5-A <=> tc...
tcp-test	LF/TCP	Stopped	0	0	0	0	0	0	1000	2.9	tcp-test-A <=> tcp-te...
udp-test	LF/UDP	Stopped	0	0	0	0	0	0	1000	1.8	udp-test-A <=> udp-...

F. On the Layer-3 Endpoints tab, highlight the Running A-Side Endpoints, then right-click and select Calculations.



Name	EID	Run	Mng	Script	Tx Rate	Tx Rate(1)	Rx Rate	Rx Rate(1)	Rx Drop %	Tx Pkts	Rx Pkts	Delay	Dropped
mc-rx	1.1.8.14	☐	☑	None	0	0	0	0	0	0	0	0	0
mc-tx	1.1.1.13	☐	☑	None	0	0	0	0	0	0	0	0	0
tcp-max-1-A	1.1.14....	☑	☑	None	392,701	[illegible]	[illegible]	56,844	53.507	3,559	411	2,188	0
tcp-max-1-B	1.1.7.22	☑	☑	None	98,226	1	[illegible]	120,272	77.207	884	803	9,223	0
tcp-max-2-A	1.1.15....	☑	☑	None	806,990	9	[illegible]	115,638	59.602	7,277	833	4,786	0
tcp-max-2-B	1.1.7.24	☑	☑	None	228,667	2	[illegible]	276,996	74.42	2,062	1,840	13,955	0
tcp-max-3-A	1.1.16....	☑	☑	None	409,904	5	[illegible]	53,063	59.24	3,700	386	1,926	0
tcp-max-3-B	1.1.7.26	☑	☑	None	105,170	1	[illegible]	118,054	78.88	947	781	12,943	0
tcp-max-4-A	1.1.17....	☑	☑	None	346,614	4	[illegible]	49,457	53.886	3,112	356	2,712	0
tcp-max-4-B	1.1.7.28	☑	☑	None	91,955	1	[illegible]	93,350	80.431	828	609	13,647	0
tcp-max-5-A	1.1.18....	☑	☑	None	1,951,8...	2,	[illegible]	401,829	67.826	17,506	3,977	6,954	0
tcp-max-5-B	1.1.7.30	☑	☑	None	1,378,1...	1,529,4...	487,927	443,920	76.389	12,361	4,121	15,600	0
tcp-test-A	1.1.7.17	☐	☑	Enabled	0	0	0	0	0	0	0	0	0
tcp-test-B	1.1.6.18	☐	☑	Enabled	0	0	0	0	0	0	0	0	0
udp-test-A	1.1.7.15	☐	☑	Enabled	0	0	0	0	0	0	0	0	0
udp-test-B	1.1.6.16	☐	☑	Enabled	0	0	0	0	0	0	0	0	0

G. The top line, Sum, is what we are interested in for Maximum Concurrent TCP Connections.



LANforge Table Calculations

Totals

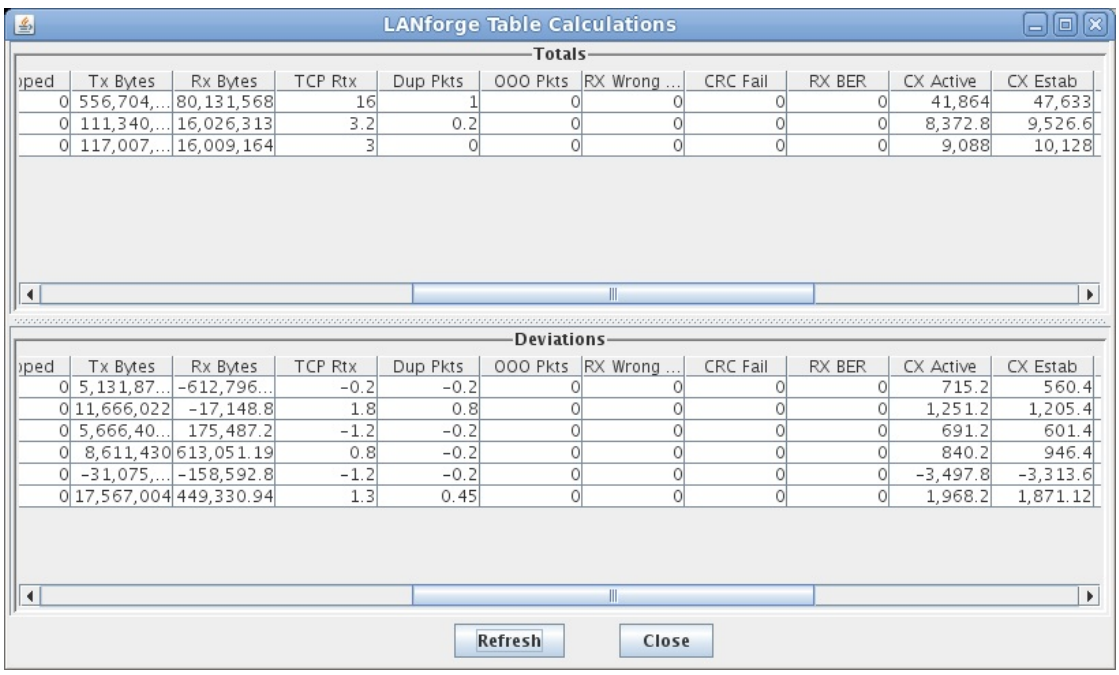
Calculation	Tx Rate	Tx Rate(1)	Rx Rate	Rx Rate(1)	Rx Drop %	Tx Pkts	Rx Pkts	Delay	Dropped	T:
Sum	3,542,088	3,066,110	509,843	730,812	363.1	543,657	75,611	44,980	0	55
Mean (Average)	708,417...	613,222	101,968.6	146,162.41	72.62	108,731.4	15,122.2	8,996	0	11
Median	744,526	535,911	101,837	141,546	72.25	114,265	15,136	9,555	0	11

Deviations

Name	Tx Rate	Tx Rate(1)	Rx Rate	Rx Rate(1)	Rx Drop %	Tx Pkts	Rx Pkts	Delay	Dropped	T:
tcp-max-1-A	32,891.4	-92,633	-3,867.6	-16,719.4	1.04	5,011.6	-585.2	1,096	0	5,1
tcp-max-2-A	74,053.4	-77,311	-131.6	-13,883.4	1.96	11,392.6	13.8	559	0	11,
tcp-max-3-A	36,108.4	118,473	1,124.4	-4,616.4	-0.36	5,533.6	147.8	-978	0	5,6
tcp-max-4-A	54,659.4	166,077	3,882.4	31,067.6	-0.68	8,409.6	529.8	-1,269	0	8,
tcp-max-5-A	-197,71...	-114,606	-1,007.6	4,151.6	-1.95	-30,347.4	-106.2	592	0	-3
Standard Deviation	111,744...	131,633.8	2,842.9	19,209.57	1.52	17,155.28	405.11	1,052.5	0	17,

Refresh Close

H. Scroll right to the CX Active and CX Established columns and select the **Refresh** button. This DUT can maintain a maximum of 41,864 simultaneous TCP connections.



LANforge Table Calculations

Totals

)ped	Tx Bytes	Rx Bytes	TCP Rtx	Dup Pkts	OOO Pkts	RX Wrong ...	CRC Fail	RX BER	CX Active	CX Estab
0	556,704,...	80,131,568	16	1	0	0	0	0	41,864	47,633
0	111,340,...	16,026,313	3.2	0.2	0	0	0	0	8,372.8	9,526.6
0	117,007,...	16,009,164	3	0	0	0	0	0	9,088	10,128

Deviations

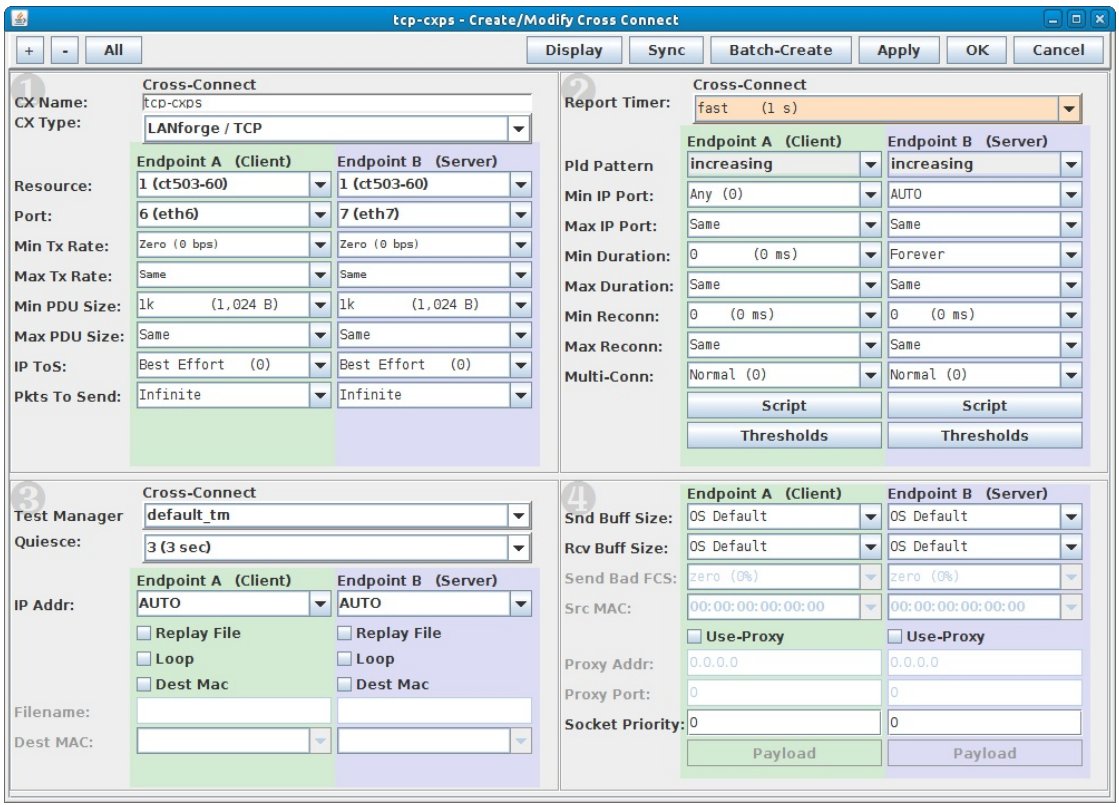
)ped	Tx Bytes	Rx Bytes	TCP Rtx	Dup Pkts	OOO Pkts	RX Wrong ...	CRC Fail	RX BER	CX Active	CX Estab
0	5,131,87...	-612,796...	-0.2	-0.2	0	0	0	0	715.2	560.4
0	11,666,022	-17,148.8	1.8	0.8	0	0	0	0	1,251.2	1,205.4
0	5,666,40...	175,487.2	-1.2	-0.2	0	0	0	0	691.2	601.4
0	8,611,430	613,051.19	0.8	-0.2	0	0	0	0	840.2	946.4
0	-31,075,...	-158,592.8	-1.2	-0.2	0	0	0	0	-3,497.8	-3,313.6
0	17,567,004	449,330.94	1.3	0.45	0	0	0	0	1,968.2	1,871.12

Refresh Close

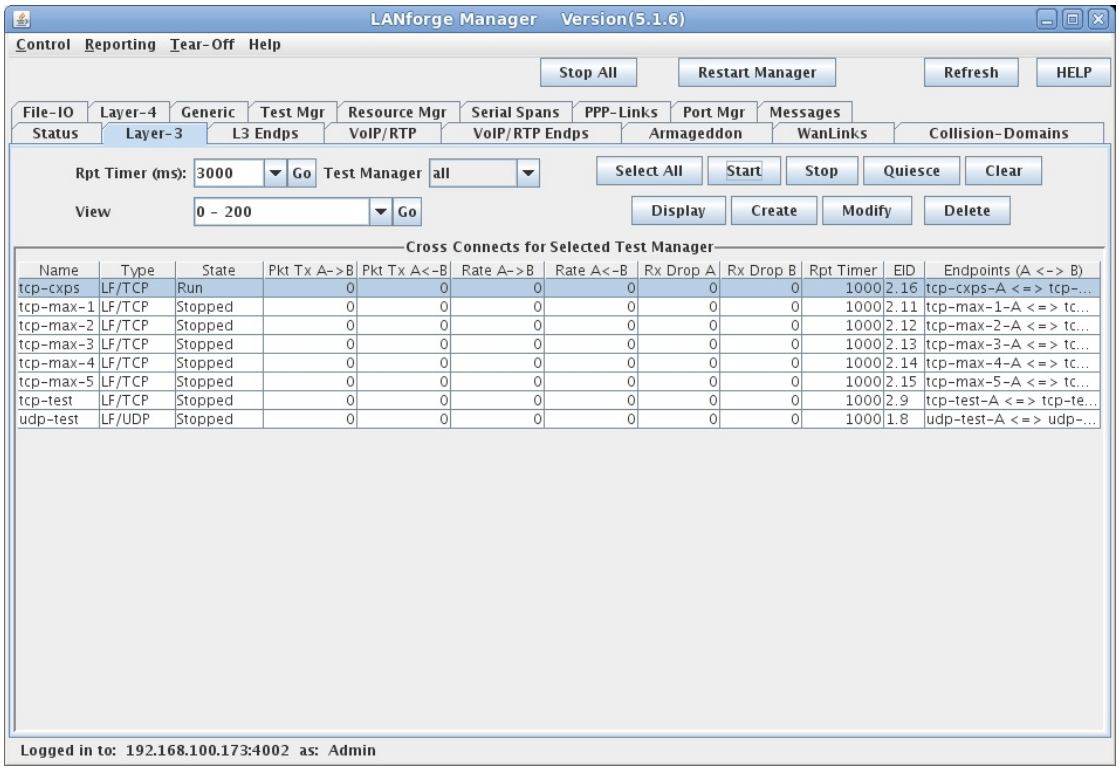
A. **CX Active** is the metric we are attempting to measure for Maximum Simultaneous TCP Connections. It will fluctuate with the DUT's ability to maintain the number of active TCP connections.

B. **CX Established** is the number of TCP connections LANforge has established since the start of the test. It will continue to increase as the DUT closes the TCP connections it cannot maintain.

3. The **TCP Connections per Second** test will measure the rate of TCP connections that can be set up through the DUT.
   A. Create a Layer-3 TCP connection with the Duration and IP Port set to zero.

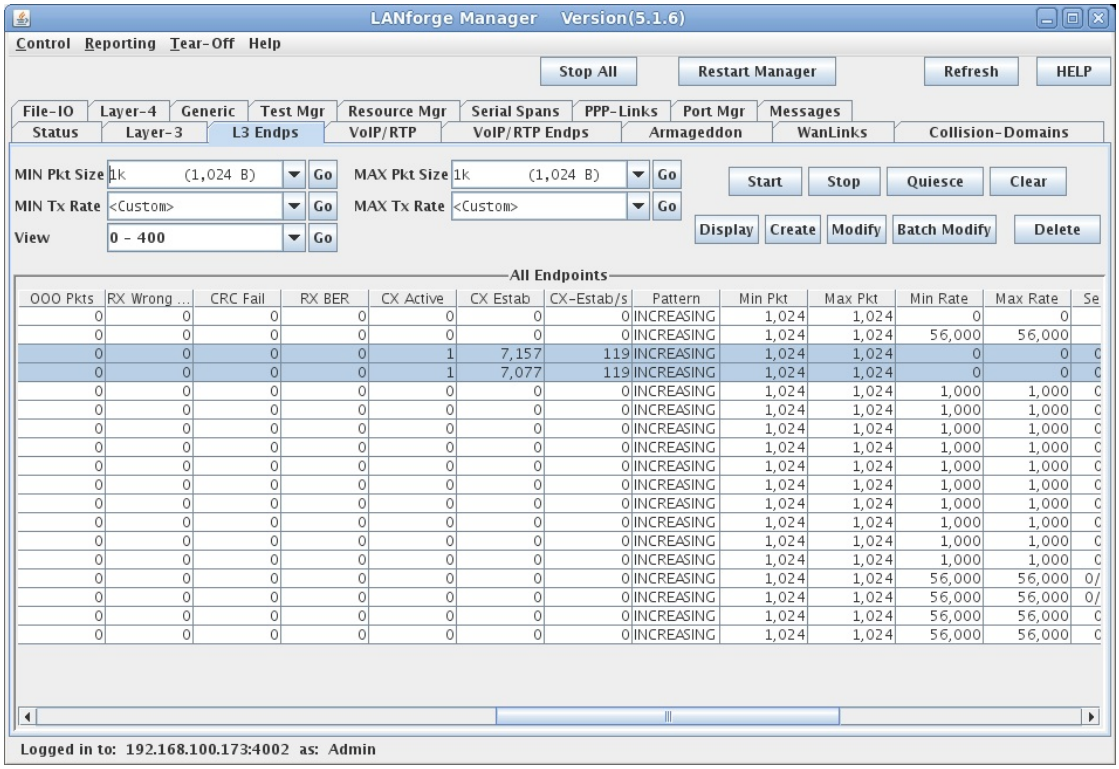


B. Highlight and Start the connection.



Name	Type	State	Pkt Tx A->B	Pkt Tx A<-B	Rate A->B	Rate A<-B	Rx Drop A	Rx Drop B	Rpt Timer	EID	Endpoints (A <-> B)
tcp-cxps	LF/TCP	Run	0	0	0	0	0	0	1000	2.16	tcp-cxps-A <=> tcp-...
tcp-max-1	LF/TCP	Stopped	0	0	0	0	0	0	1000	2.11	tcp-max-1-A <=> tc...
tcp-max-2	LF/TCP	Stopped	0	0	0	0	0	0	1000	2.12	tcp-max-2-A <=> tc...
tcp-max-3	LF/TCP	Stopped	0	0	0	0	0	0	1000	2.13	tcp-max-3-A <=> tc...
tcp-max-4	LF/TCP	Stopped	0	0	0	0	0	0	1000	2.14	tcp-max-4-A <=> tc...
tcp-max-5	LF/TCP	Stopped	0	0	0	0	0	0	1000	2.15	tcp-max-5-A <=> tc...
tcp-test	LF/TCP	Stopped	0	0	0	0	0	0	1000	2.9	tcp-test-A <=> tcp-te...
udp-test	LF/UDP	Stopped	0	0	0	0	0	0	1000	1.8	udp-test-A <=> udp-...

C. View the **CX-Estab/s** rate on the Layer-3 Endpoints tab. This DUT can set up about 120 connections per second.



OOO Pkts	RX Wrong ...	CRC Fail	RX BER	CX Active	CX Estab	CX-Estab/s	Pattern	Min Pkt	Max Pkt	Min Rate	Max Rate	Se
0	0	0	0	0	0	0	INCREASING	1,024	1,024	0	0	
0	0	0	0	0	0	0	INCREASING	1,024	1,024	56,000	56,000	
0	0	0	0	1	7,157	119	INCREASING	1,024	1,024	0	0	0
0	0	0	0	1	7,077	119	INCREASING	1,024	1,024	0	0	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	1,000	1,000	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	1,000	1,000	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	1,000	1,000	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	1,000	1,000	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	1,000	1,000	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	1,000	1,000	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	1,000	1,000	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	1,000	1,000	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	1,000	1,000	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	1,000	1,000	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	56,000	56,000	0/
0	0	0	0	0	0	0	INCREASING	1,024	1,024	56,000	56,000	0/
0	0	0	0	0	0	0	INCREASING	1,024	1,024	56,000	56,000	0
0	0	0	0	0	0	0	INCREASING	1,024	1,024	56,000	56,000	0

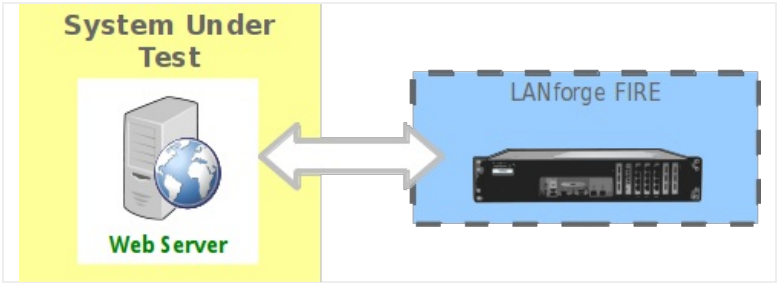
For more information see LANforge GUI User's Guide

Generating Traffic to a Web Server

Goal: Set up and run traffic to a web server.

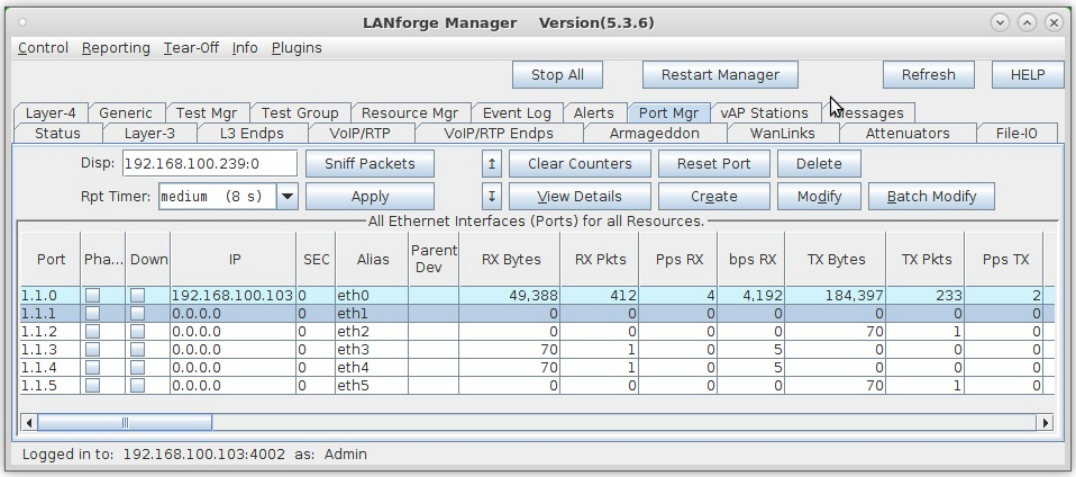
In this test scenario, LANforge-FIRE is used to generate traffic in the form of URL requests in order to determine the maximum number of URLs/second the web server can process.

Please note that the web server used in this example is an isolated Linux system running Apache.

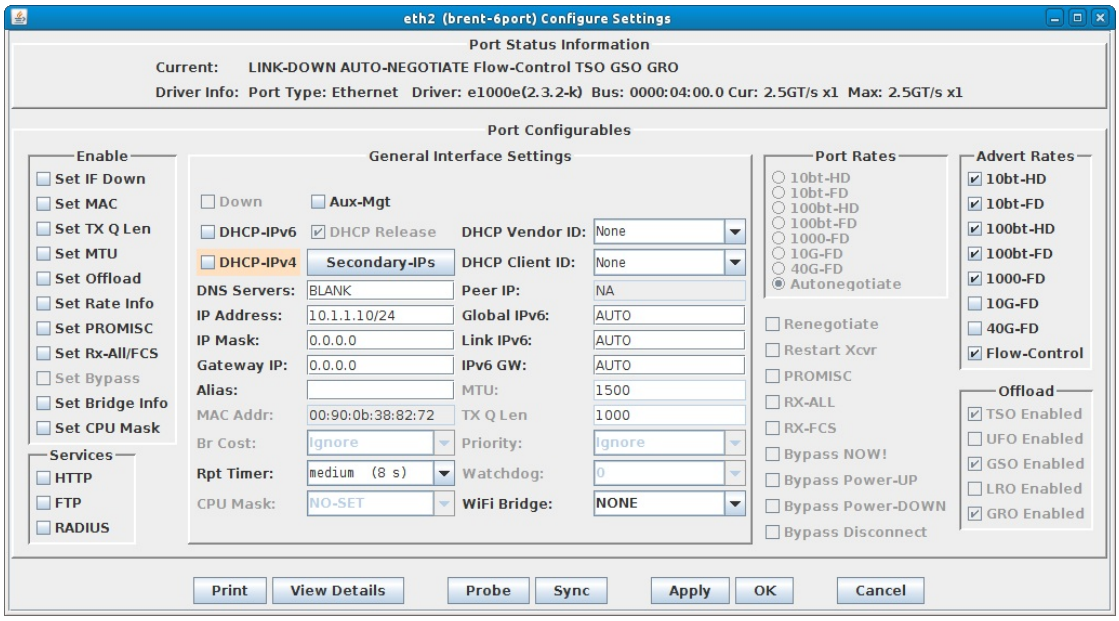


1. Connect one LANforge-FIRE port to the web server's network.
2. Set up the LANforge port so that it has a valid IP address.

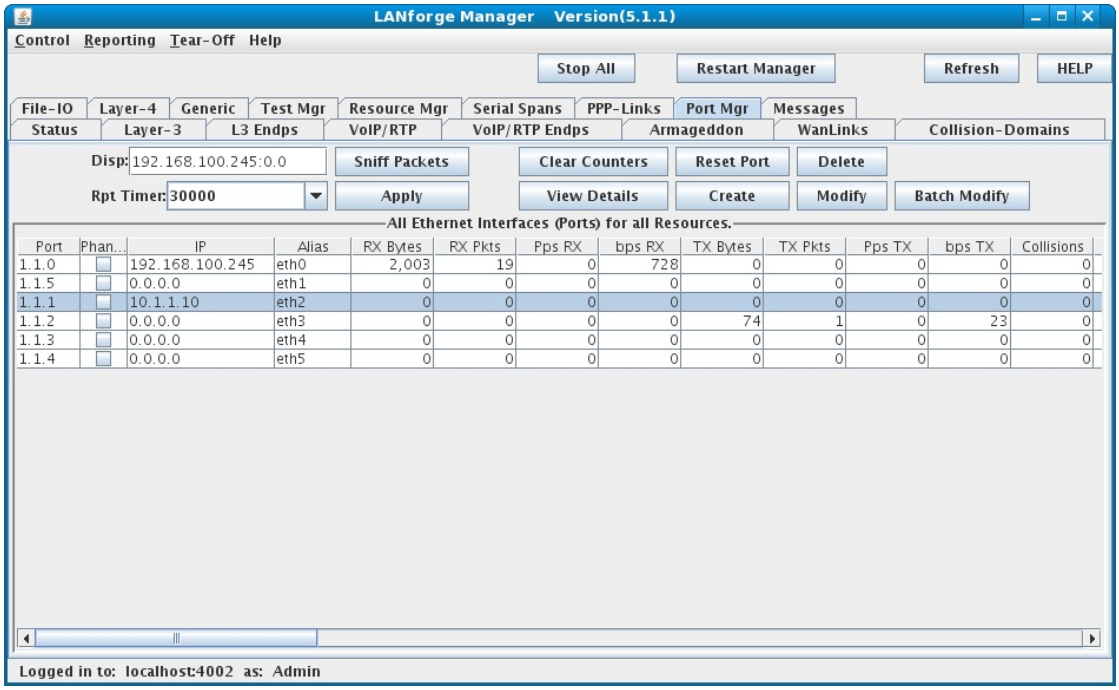
A. Go to the Port Manager



B. Modify the port connected to the web server. Set a valid network IP address and Gateway IP.

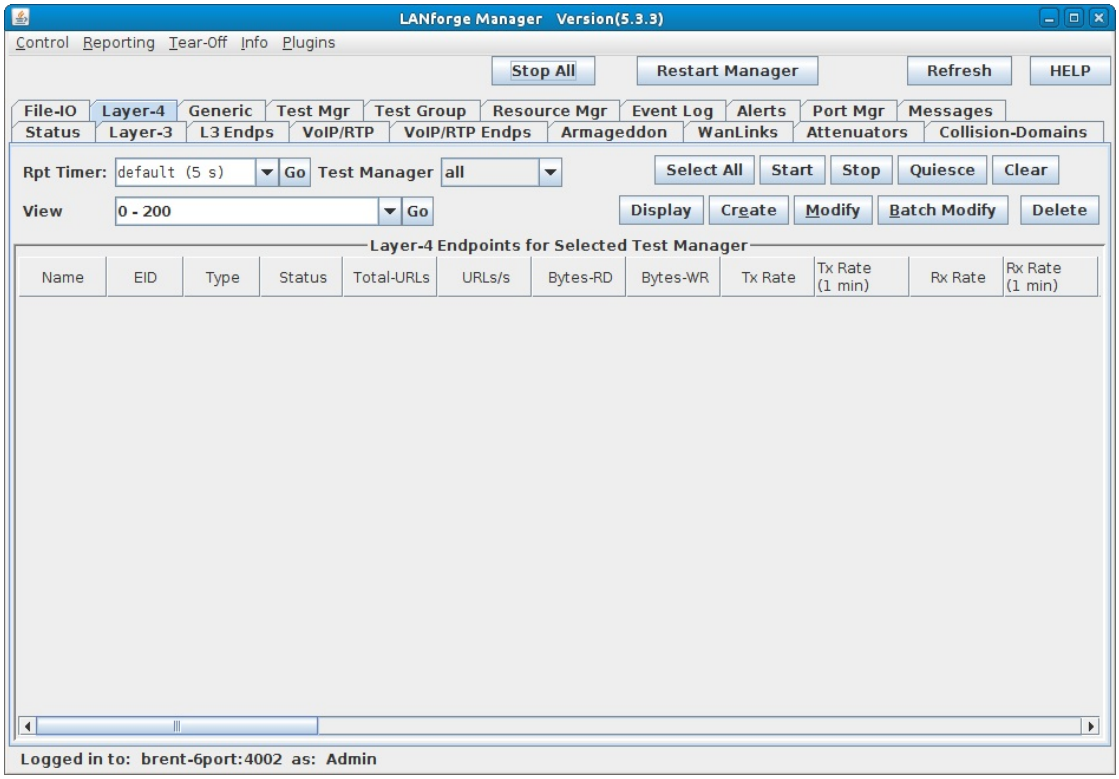


C. Verify the port configuration

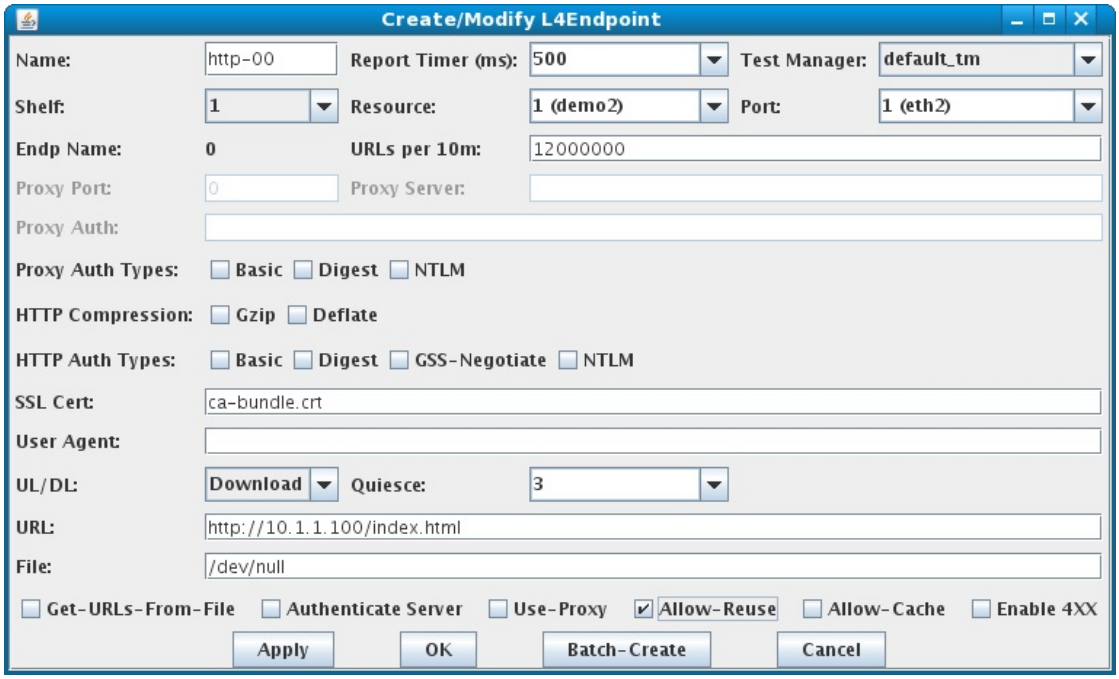


For more information see LANforge User's Guide: Ports (Interfaces)

3. Set up the URL requests.
   A. Go to the **Layer 4-7** tab

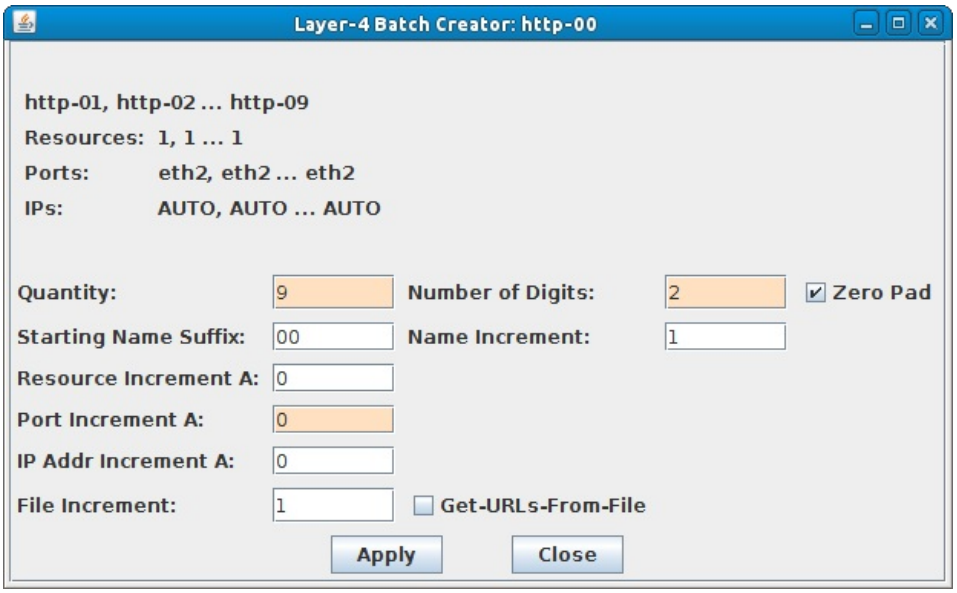


   B. Create a Layer 4-7 Endpoint:



   A. Enter a name and select the port configured in the previous step
   B. URLs per 10m should be set to around 1,200,000 which is 2,000 URLs/second
   C. Leave the SSL Cert alone as we are not using it in this example
   D. UL/DL should be set to Download
   E. URL should be set to the web server's IP address and file that you wish to download. In this example, index.html is a small test file.
   F. File is the designated location that the downloaded file will be stored. In this example, we use /dev/null which will essentially throw the file away so that it does not slow down the LANforge system with writing files.
   G. Select the 'Allow-Reuse' checkbox to maximize the number of URL requests LANforge can make per connection
   H. Click **Apply** or **OK** to create the Layer 4-7 endpoint

C. Create 9 more Layer 4-7 endpoints by opening the previously created endpoint and clicking **Batch-Create**

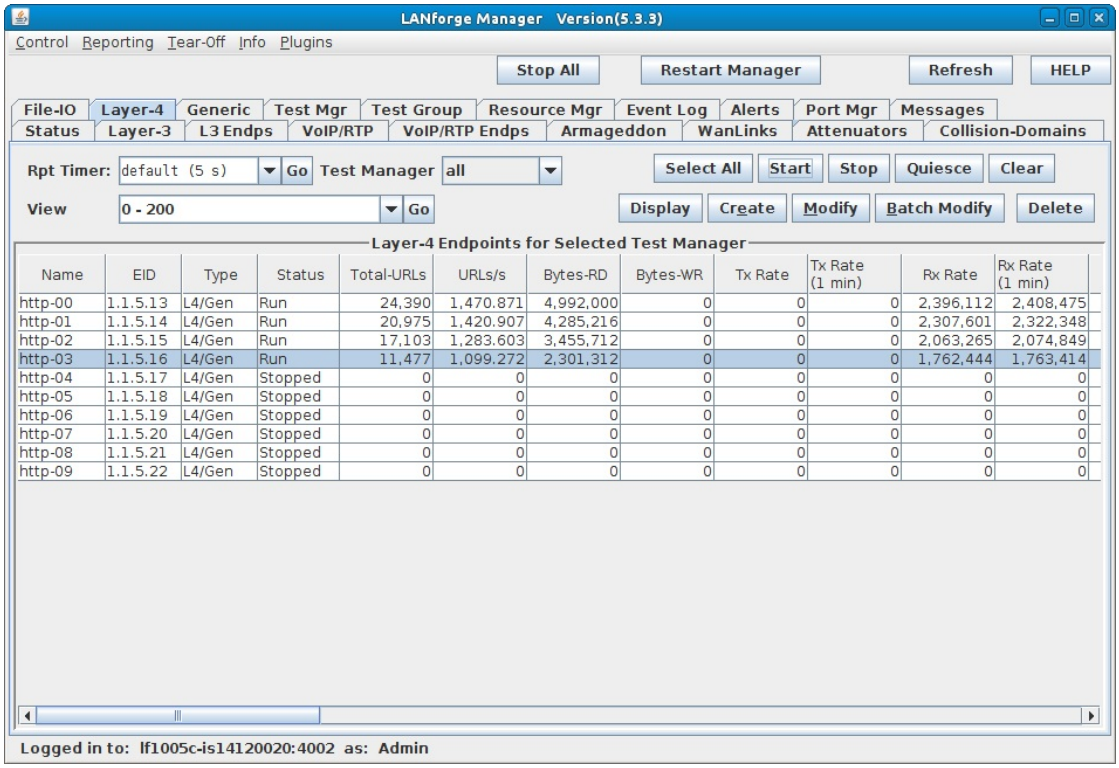


A. Set the **Quantity** to 9, **Number of Digits** to 2, and **Port Increment A** to 0. Then click **Apply**.

B. Each of the 10 Layer 4-7 Endpoints will attempt to generate 2000 URLs/second, effectively generating 20,000 URLs/second

For more information see LANforge User's Guide: Layer 4-7 Endpoints

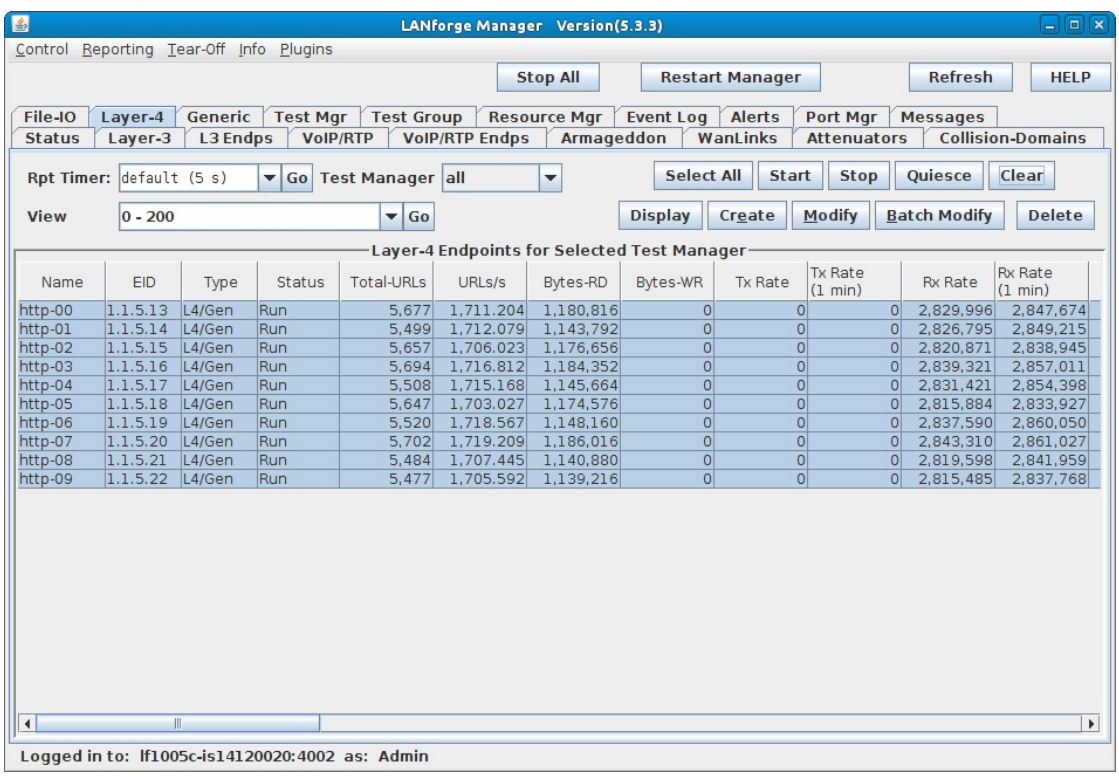
4. Run traffic and determine web server performance.
   A. On the **Layer 4-7** tab, select one Layer 4-7 endpoint, click **Start**, then repeat for all 10 Layer 4-7 endpoints:



Name	EID	Type	Status	Total-URLs	URLs/s	Bytes-RD	Bytes-WR	Tx Rate	Tx Rate (1 min)	Rx Rate	Rx Rate (1 min)
http-00	1.1.5.13	L4/Gen	Run	24,390	1,470.871	4,992,000	0	0	0	2,396,112	2,408,475
http-01	1.1.5.14	L4/Gen	Run	20,975	1,420.907	4,285,216	0	0	0	2,307,601	2,322,348
http-02	1.1.5.15	L4/Gen	Run	17,103	1,283.603	3,455,712	0	0	0	2,063,265	2,074,849
http-03	1.1.5.16	L4/Gen	Run	11,477	1,099.272	2,301,312	0	0	0	1,762,444	1,763,414
http-04	1.1.5.17	L4/Gen	Stopped	0	0	0	0	0	0	0	0
http-05	1.1.5.18	L4/Gen	Stopped	0	0	0	0	0	0	0	0
http-06	1.1.5.19	L4/Gen	Stopped	0	0	0	0	0	0	0	0
http-07	1.1.5.20	L4/Gen	Stopped	0	0	0	0	0	0	0	0
http-08	1.1.5.21	L4/Gen	Stopped	0	0	0	0	0	0	0	0
http-09	1.1.5.22	L4/Gen	Stopped	0	0	0	0	0	0	0	0

      A. As each endpoint is started, the rate of URLs/second will start to converge on a rate that the web server is capable of providing. Finding the final web server performance rate is a matter of adding up the rates of all running Layer 4-7 endpoints.
      B. Several Layer 4-7 endpoints (10 in this case) are used so each endpoint can make an independently large number of URL requests without having to wait for too many replies. Each URL request is waiting for a reply from the web server, so if only one Layer 4-7 endpoint was making requests, it would spend too much time waiting for replies instead of generating more requests. Spreading the URL requests over several endpoints allows each LANforge connection to the web server to maximize its rate of URL requests.

   B. Layer 4-7 Endpoint Results:



Name	EID	Type	Status	Total-URLs	URLs/s	Bytes-RD	Bytes-WR	Tx Rate	Tx Rate (1 min)	Rx Rate	Rx Rate (1 min)
http-00	1.1.5.13	L4/Gen	Run	5,677	1,711.204	1,180,816	0	0	0	2,829,996	2,847,674
http-01	1.1.5.14	L4/Gen	Run	5,499	1,712.079	1,143,792	0	0	0	2,826,795	2,849,215
http-02	1.1.5.15	L4/Gen	Run	5,657	1,706.023	1,176,656	0	0	0	2,820,871	2,838,945
http-03	1.1.5.16	L4/Gen	Run	5,694	1,716.812	1,184,352	0	0	0	2,839,321	2,857,011
http-04	1.1.5.17	L4/Gen	Run	5,508	1,715.168	1,145,664	0	0	0	2,831,421	2,854,398
http-05	1.1.5.18	L4/Gen	Run	5,647	1,703.027	1,174,576	0	0	0	2,815,884	2,833,927
http-06	1.1.5.19	L4/Gen	Run	5,520	1,718.567	1,148,160	0	0	0	2,837,590	2,860,050
http-07	1.1.5.20	L4/Gen	Run	5,702	1,719.209	1,186,016	0	0	0	2,843,310	2,861,027
http-08	1.1.5.21	L4/Gen	Run	5,484	1,707.445	1,140,880	0	0	0	2,819,598	2,841,959
http-09	1.1.5.22	L4/Gen	Run	5,477	1,705.592	1,139,216	0	0	0	2,815,485	2,837,768

      A. After starting all 10 endpoints, and letting them run for at least 1 minute, the overall URLs/second rate converges to around 17,000 URLs/second.

For more information see LANforge User's Guide: Layer 4-7 Endpoints

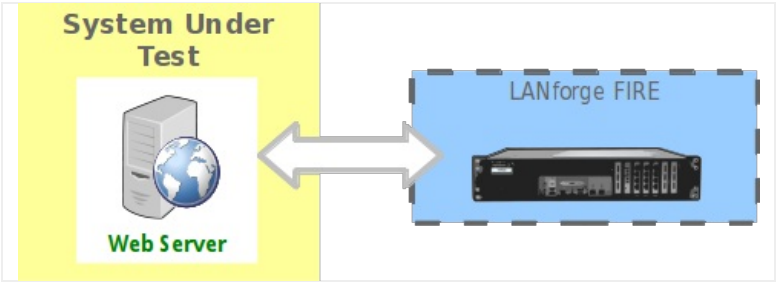
Introduction to Layer 4-7 Traffic Generation

Goal: Use LANforge to emulate layer 4-7 download traffic

Creating web browsing and movie watching emulation is a common task with LANforge. This cookbook will introduce HTTP download examples and describe the difference between the Layer 4-7 and Generic traffic generation techniques. We will begin on a CT520a with 20 stations.There are two methods of emulating web browsing:

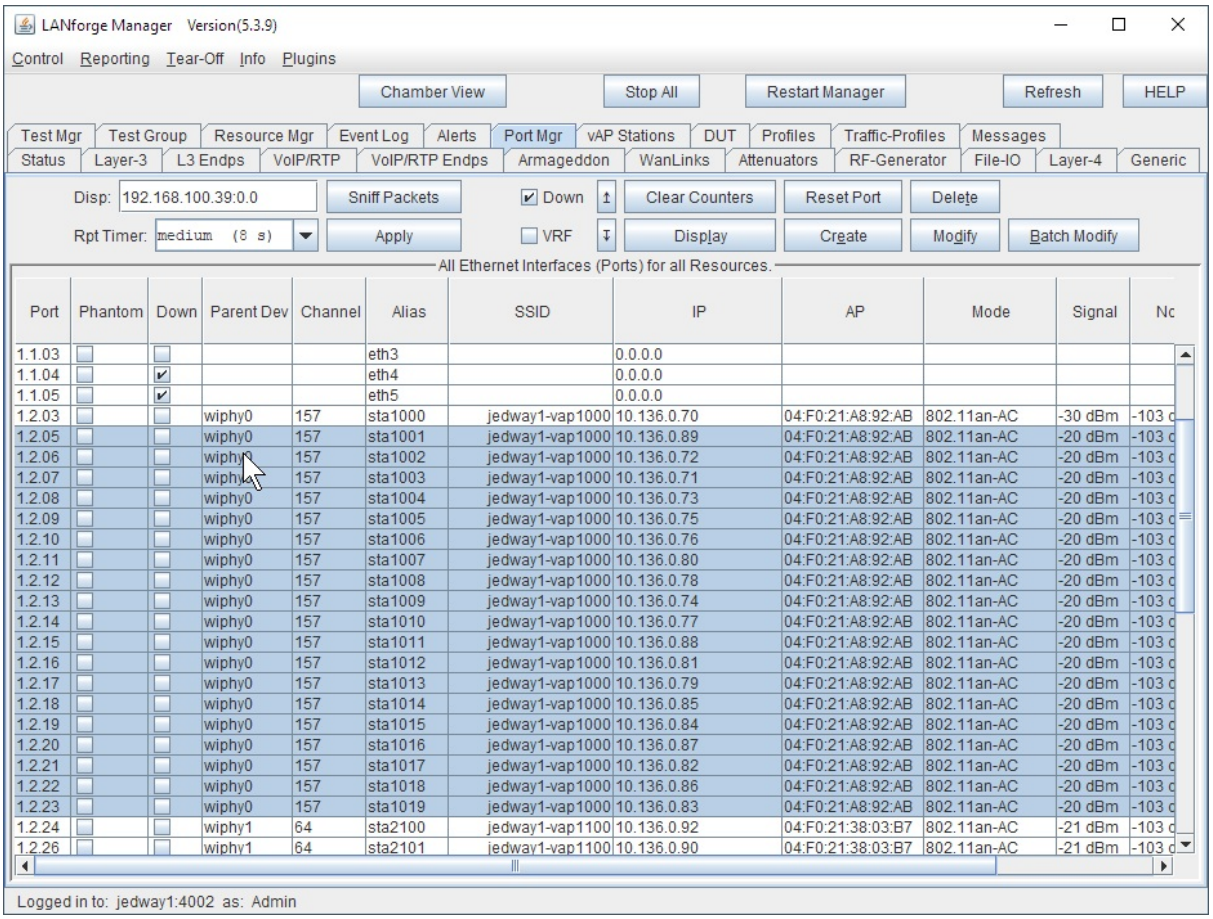
1. **Layer 4-7 connections**: these are workers doing downloads using `curl`
2. **Generic connections**: these are scripts called programs that can download or upload. Not just any program can be used, these programs need to be able to bind to a specified network interface.

If you want to:
...**load test** a web server, you will use mostly use **Layer 4-7** connections.
...**multiple video streams**, you would use **Generic** connections that call `c-clive`
...**Upload files**, you would use **Generic** connections that call `curl`
The following examples do not assume your LANforge has a route to the Internet, but are pointing at a webserver IP address. We are going to show making a common mistake in the first example.

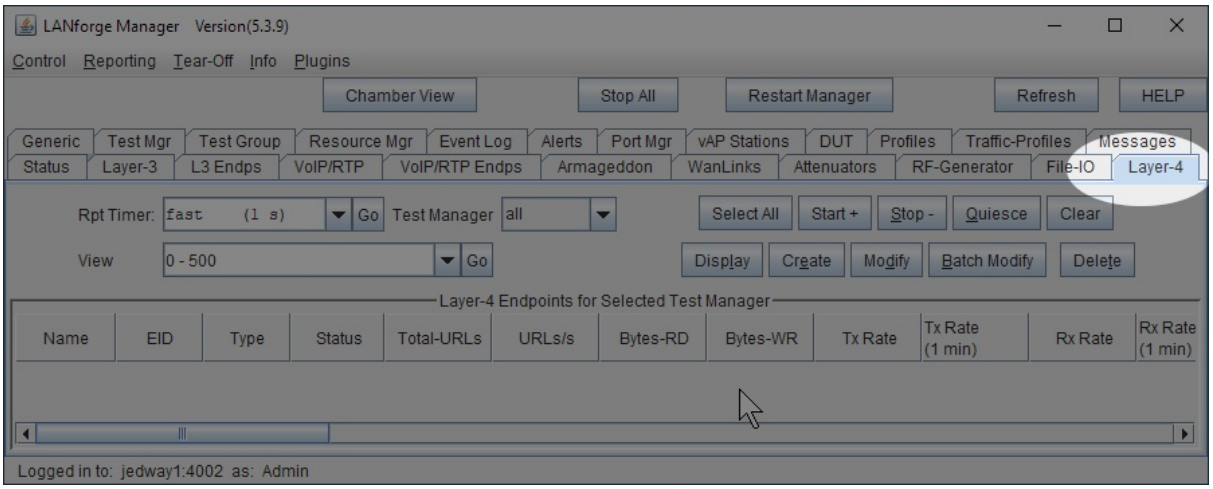


1. ## Simple HTTP Get

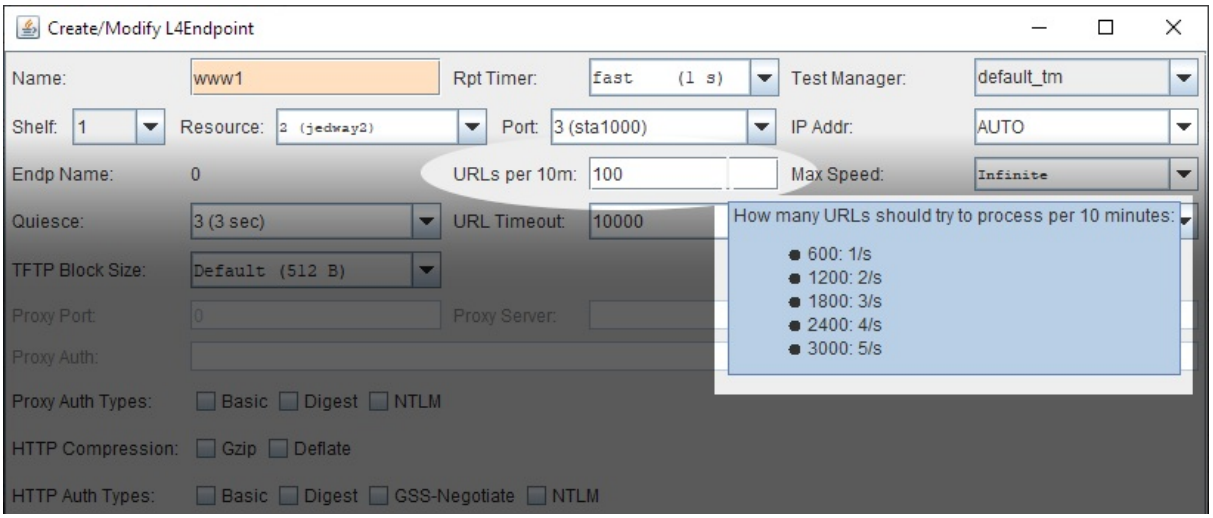
2. Check your Port Mgr tab and note we have 20 stations on resource 2, radio `wiphy0`.



3. Move to the Layer 4-7 tab

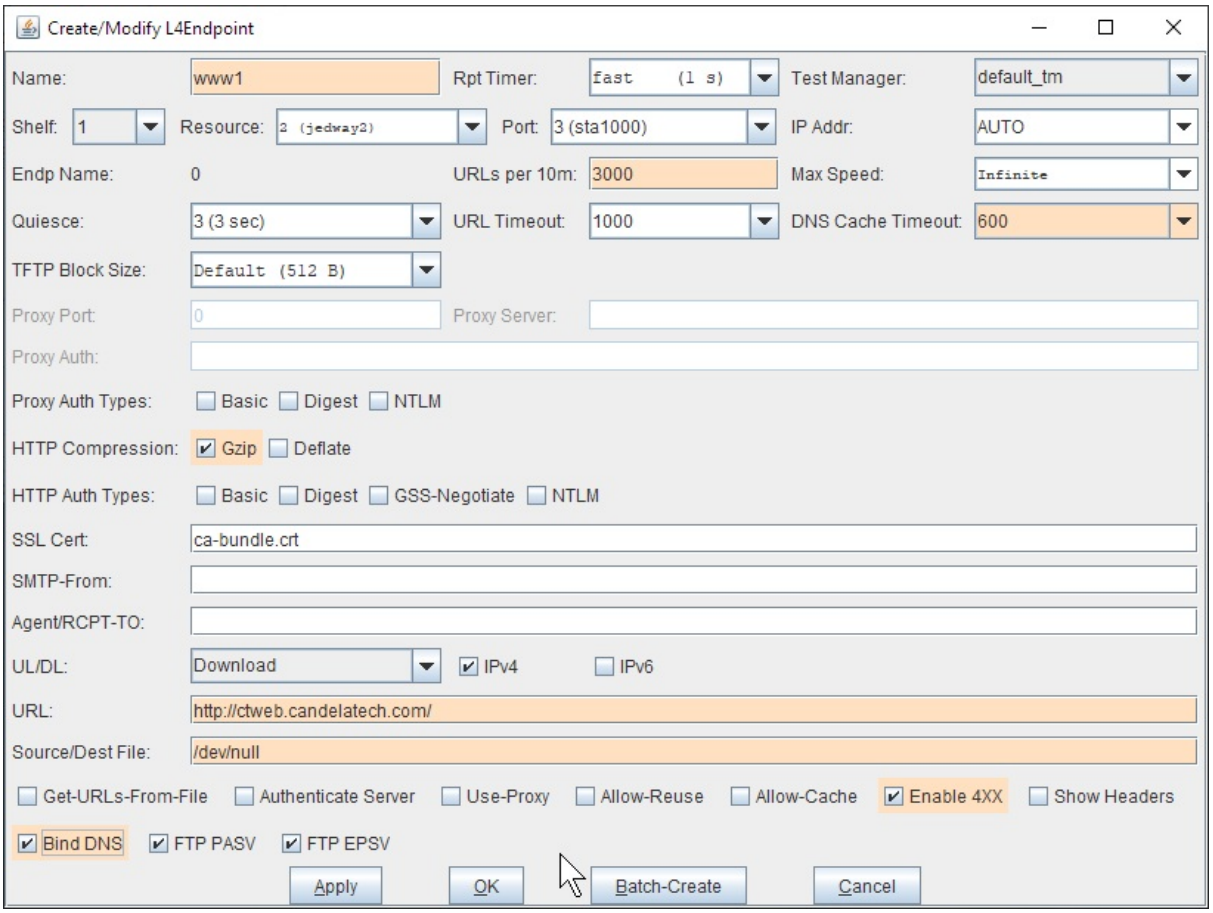


4. Create a new connection, this example is named `www1`. on resource 2, WiFi station `sta1000` Next, consider the **URLs per 10 minute** field.



A. Hover your mouse over the URLs per 10 min field to see the tool tip. If you wanted to create five connections per second, that would be **3000** connections in 10 minutes. This differs from the Max Speed field, which sets the maximum bit per second for the connection.

5. Continue with other settings. We are purposely going to make an error and diagnose it.



A. Set the request rate, choose the number of URLs per 10 min to **3000**

B. DNS Cache Timeout: 600

C. Select **Gzip**

D. Set URL to `http://10.136.0.1/`

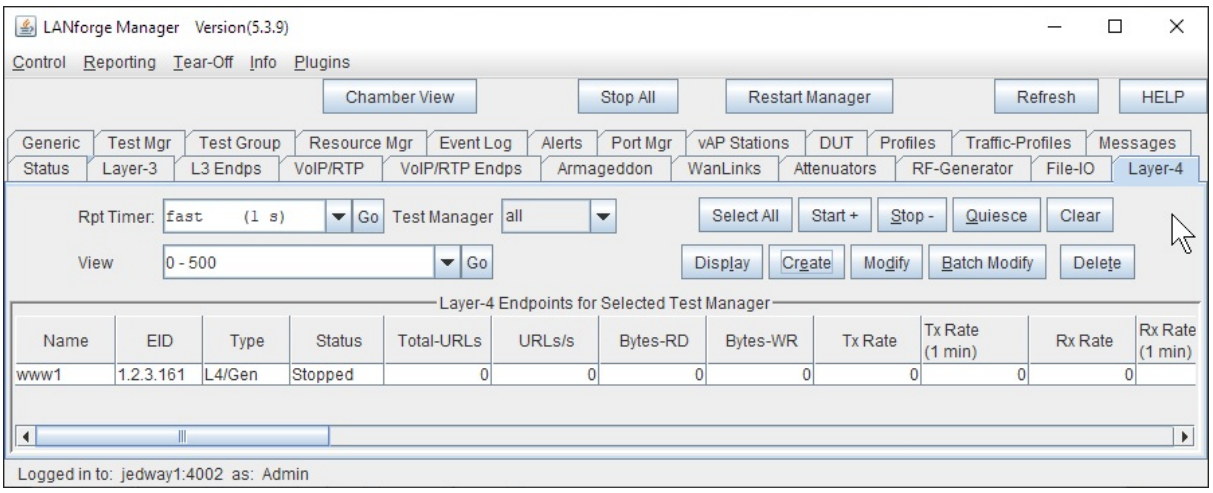
E. Set Destination file or directory to `/dev/null` if you are on Linux. (If you are on Windows, you need to use `NUL`)

F. Select **Enable 4XX** to show errors in logs

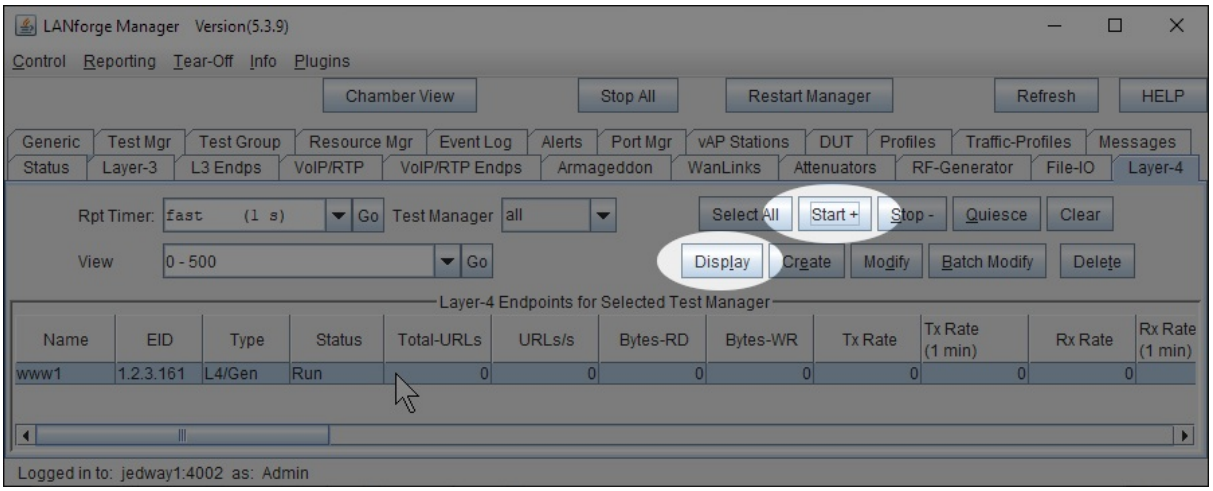
G. Select **Bind DNS** so that hostnames are resolved over `sta1000` (not over our management port)

H. Click **Apply**. Do not close this window.

6. Back in the Layer 4-7 tab, select the connection

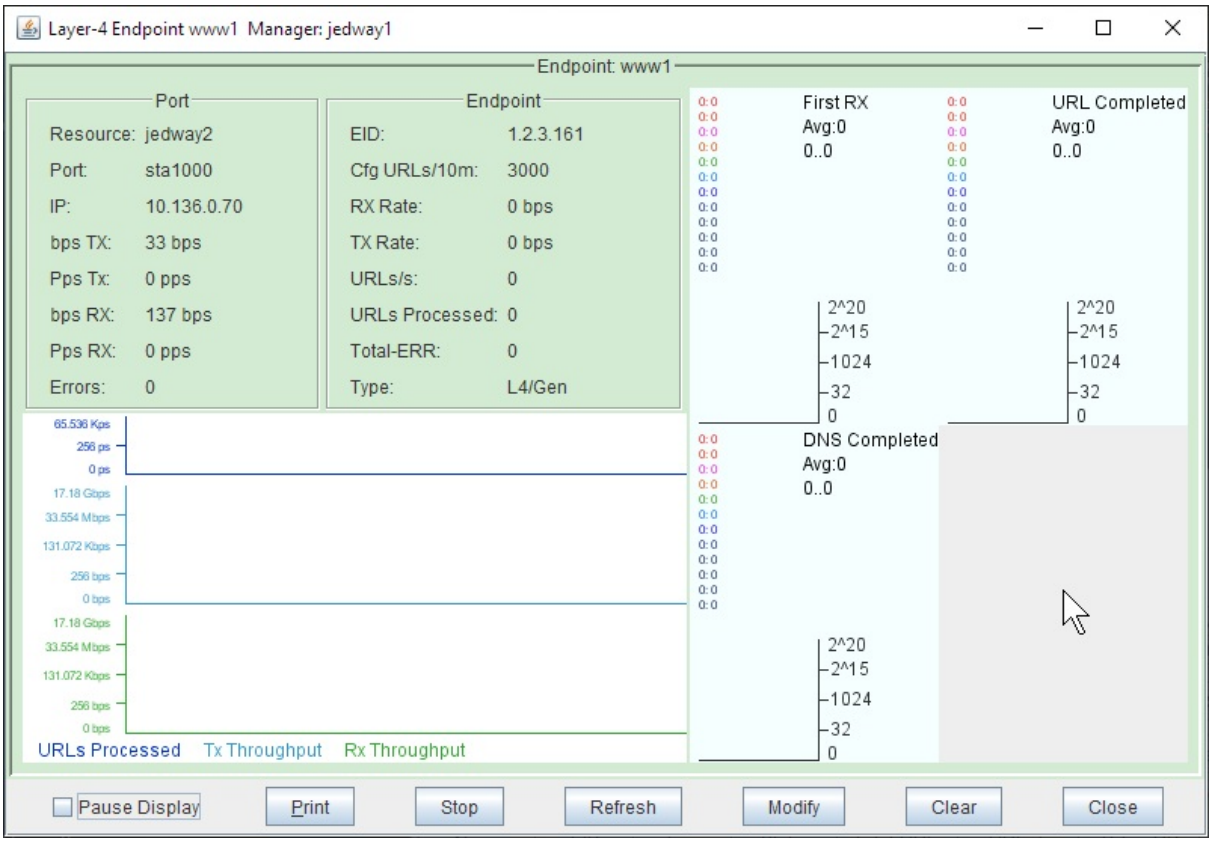


7. **Start** the connection



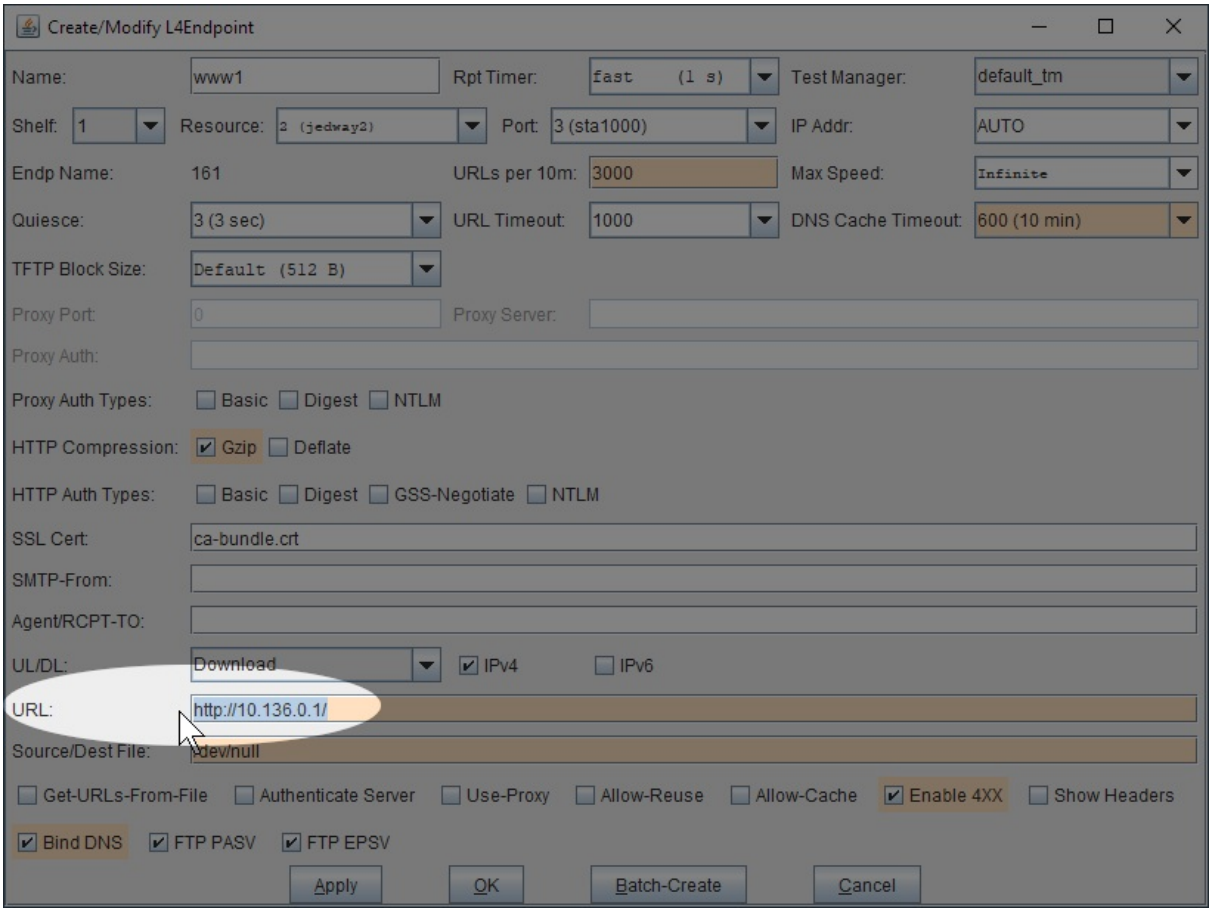
A. ... then click **Display**

8. Watch the `www1` endpoint window...



A. Notice how there is no traffic to `ctweb`: because the hostname will not resolve over our test network. This is a common mistake. Let's use the IP of our other LANforge hosting our test network instead.

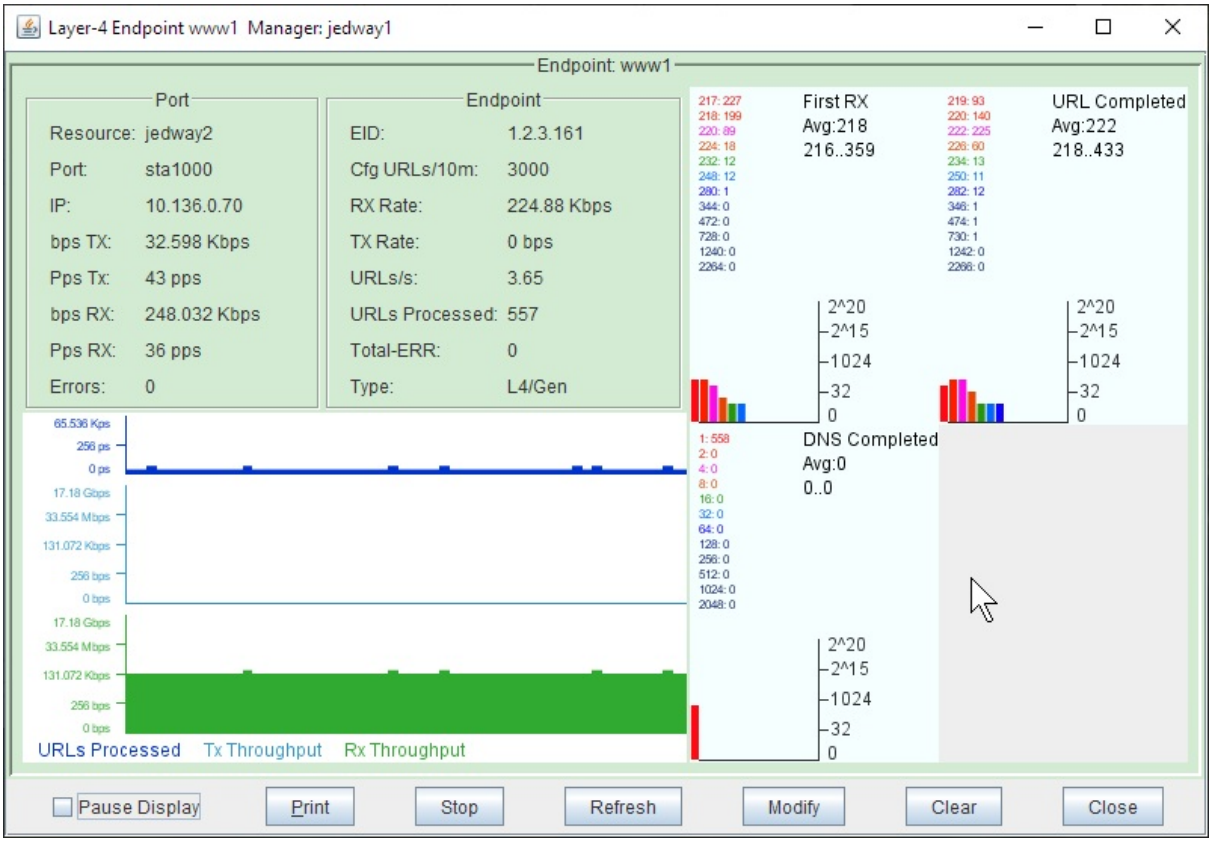
9. Change the URL to use an IP address: `http://10.136.0.1/`



A. Click **Apply**

B. Do not close the window.

10. The endpoint display now shows traffic.

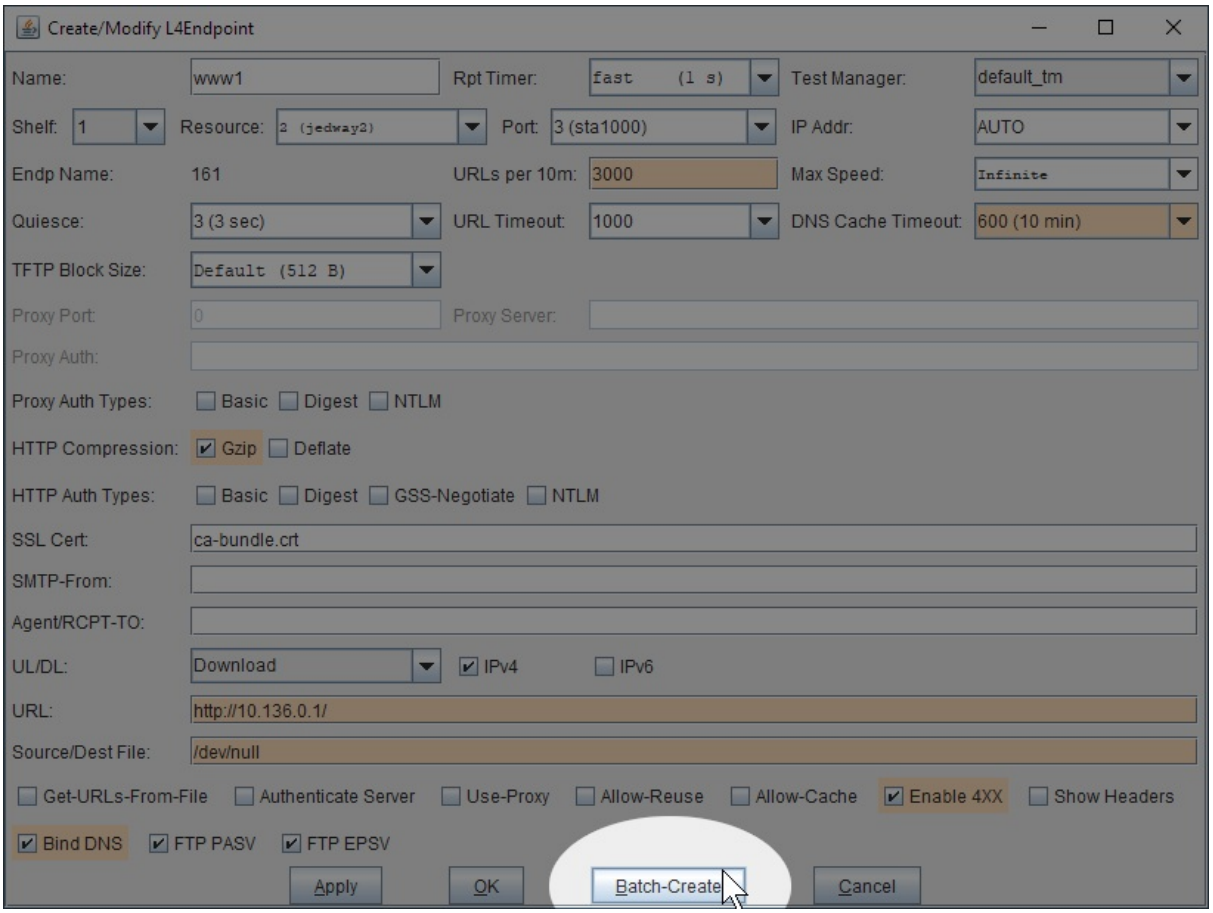


A. **Not having DNS resolution is a common problem in test networks.**

B. It can cause problems when connecting to a https website and getting certificate errors.

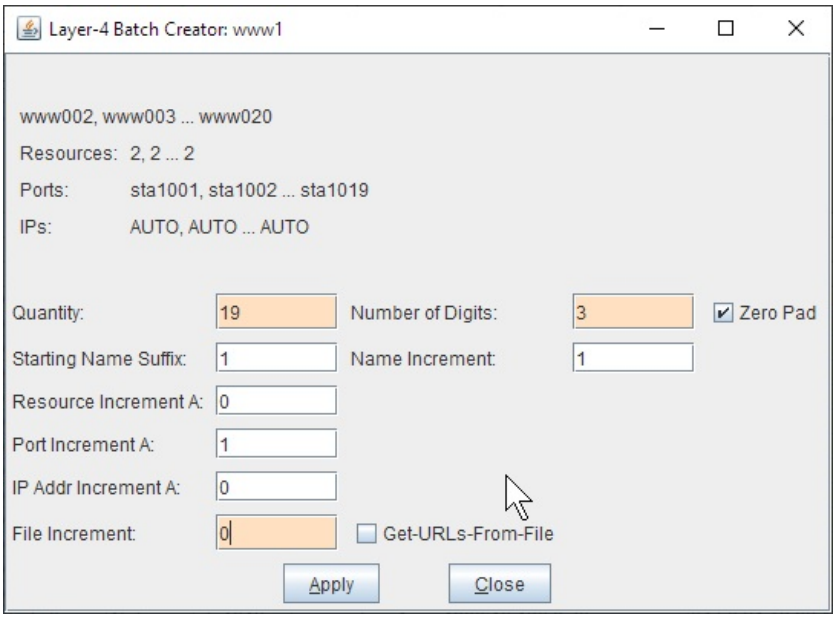
C. Click **Close**

11. It is pretty simple to create a connection per WiFi station on our LANforge. Return to our **Create/Modify L4Endpoint** window for `www1`



A. Click **Batch-Create**

12. In the **Batch Create** window, create 19 more connections



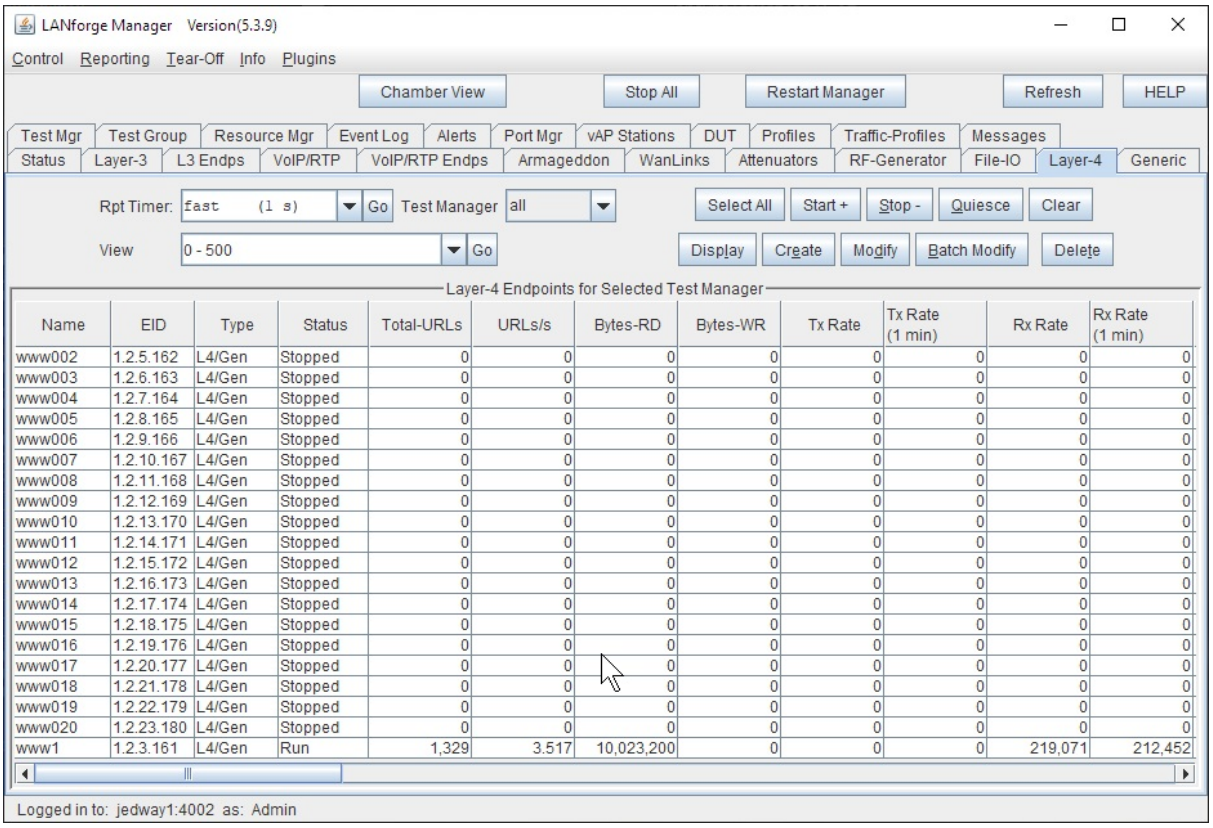
A. Quantity: **19**

B. Number of Digits **3**

C. File Increment: **0** (because we don't have multiple `/dev/null` files)

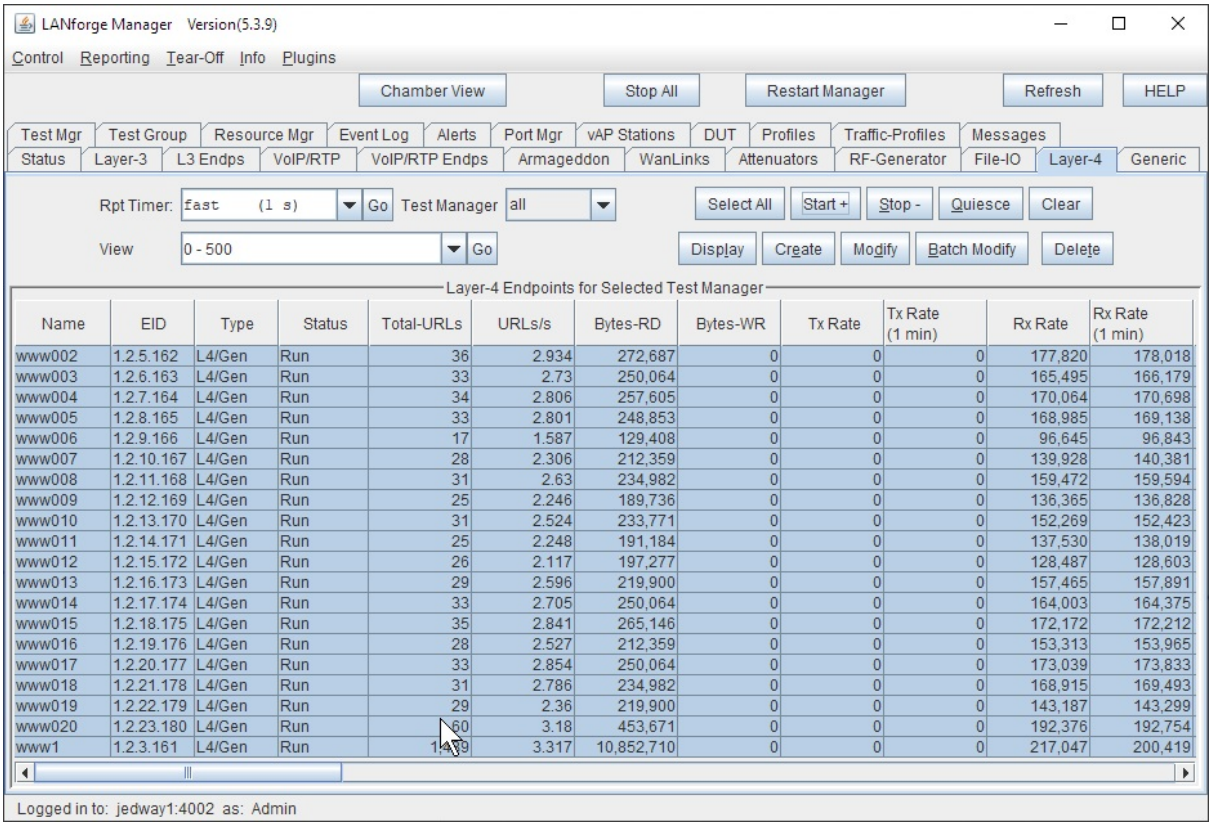
D. Click **Apply**

13. Back in the Layer 4-7 tab, observe all our connections



Name	EID	Type	Status	Total-URLs	URLs/s	Bytes-RD	Bytes-WR	Tx Rate	Tx Rate (1 min)	Rx Rate	Rx Rate (1 min)
www002	1.2.5.162	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www003	1.2.6.163	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www004	1.2.7.164	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www005	1.2.8.165	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www006	1.2.9.166	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www007	1.2.10.167	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www008	1.2.11.168	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www009	1.2.12.169	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www010	1.2.13.170	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www011	1.2.14.171	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www012	1.2.15.172	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www013	1.2.16.173	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www014	1.2.17.174	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www015	1.2.18.175	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www016	1.2.19.176	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www017	1.2.20.177	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www018	1.2.21.178	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www019	1.2.22.179	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www020	1.2.23.180	L4/Gen	Stopped	0	0	0	0	0	0	0	0
www1	1.2.3.161	L4/Gen	Run	1,329	3.517	10,023,200	0	0	0	219,071	212,452

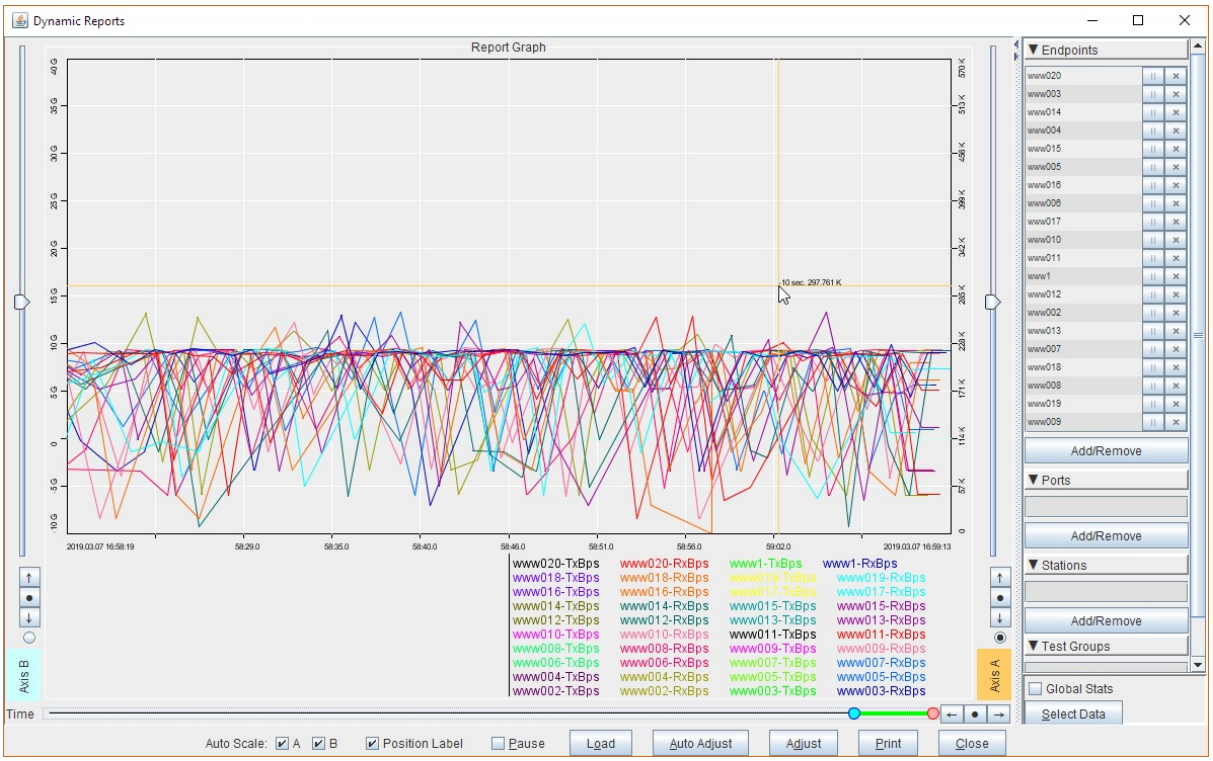
14. Hightlight them, and click **Start**



Name	EID	Type	Status	Total-URLs	URLs/s	Bytes-RD	Bytes-WR	Tx Rate	Tx Rate (1 min)	Rx Rate	Rx Rate (1 min)
www002	1.2.5.162	L4/Gen	Run	36	2.934	272,687	0	0	0	177,820	178,018
www003	1.2.6.163	L4/Gen	Run	33	2.73	250,064	0	0	0	165,495	166,179
www004	1.2.7.164	L4/Gen	Run	34	2.806	257,605	0	0	0	170,064	170,698
www005	1.2.8.165	L4/Gen	Run	33	2.801	248,853	0	0	0	168,985	169,138
www006	1.2.9.166	L4/Gen	Run	17	1.587	129,408	0	0	0	96,645	96,843
www007	1.2.10.167	L4/Gen	Run	28	2.306	212,359	0	0	0	139,928	140,381
www008	1.2.11.168	L4/Gen	Run	31	2.63	234,982	0	0	0	159,472	159,594
www009	1.2.12.169	L4/Gen	Run	25	2.246	189,736	0	0	0	136,365	136,828
www010	1.2.13.170	L4/Gen	Run	31	2.524	233,771	0	0	0	152,269	152,423
www011	1.2.14.171	L4/Gen	Run	25	2.248	191,184	0	0	0	137,530	138,019
www012	1.2.15.172	L4/Gen	Run	26	2.117	197,277	0	0	0	128,487	128,603
www013	1.2.16.173	L4/Gen	Run	29	2.596	219,900	0	0	0	157,465	157,891
www014	1.2.17.174	L4/Gen	Run	33	2.705	250,064	0	0	0	164,003	164,375
www015	1.2.18.175	L4/Gen	Run	35	2.841	265,146	0	0	0	172,172	172,212
www016	1.2.19.176	L4/Gen	Run	28	2.527	212,359	0	0	0	153,313	153,965
www017	1.2.20.177	L4/Gen	Run	33	2.854	250,064	0	0	0	173,039	173,833
www018	1.2.21.178	L4/Gen	Run	31	2.786	234,982	0	0	0	168,915	169,493
www019	1.2.22.179	L4/Gen	Run	29	2.36	219,900	0	0	0	143,187	143,299
www020	1.2.23.180	L4/Gen	Run	60	3.18	453,671	0	0	0	192,376	192,754
www1	1.2.3.161	L4/Gen	Run	1,439	3.317	10,852,710	0	0	0	217,047	200,419

15. Note that if we were to Right Click→Display to show dynamic report

16. Right Click → Dynamic Report to show dynamic report

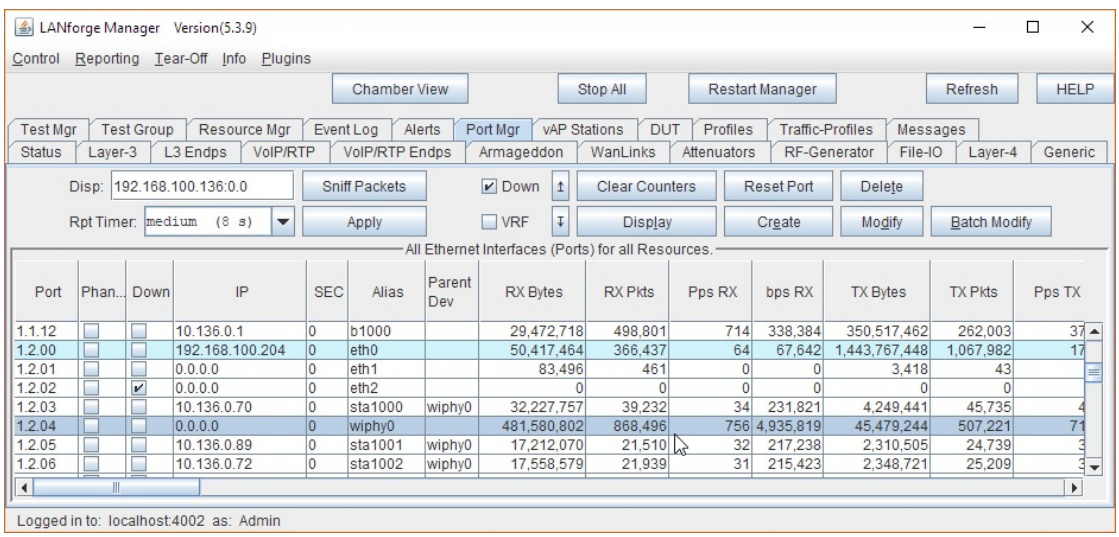


A. This shows throughput for all the connections.

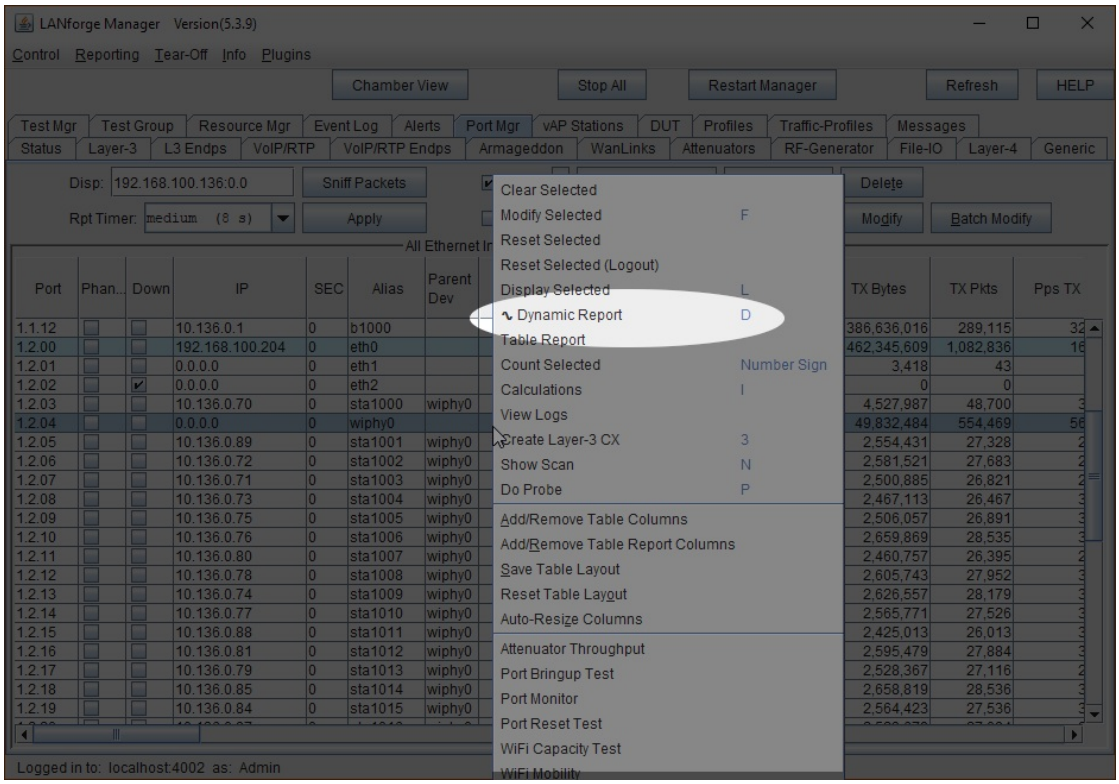
B. Notice how crowded this graph is. There is a simpler way to display overall throughput.

17. Next, select the Port Mgr tab.

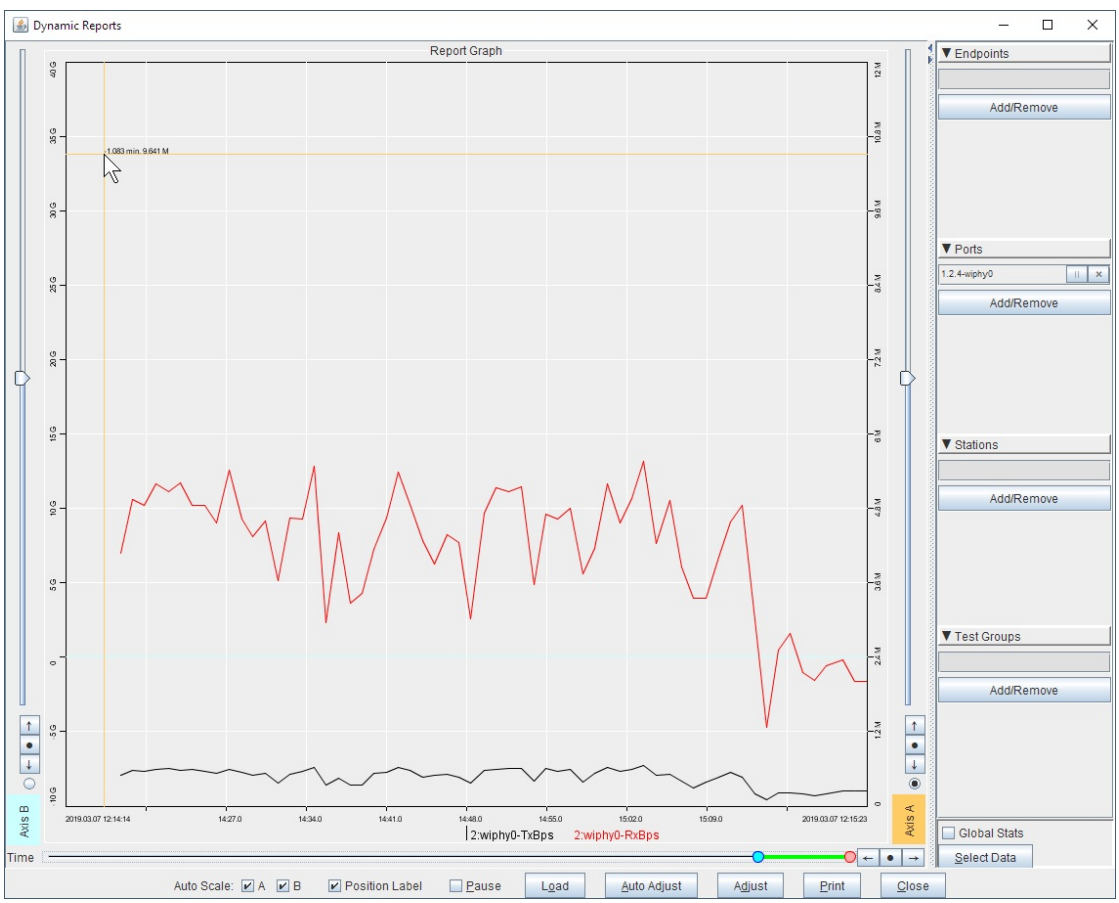
A. Select resource 2, radio ***wiphy0***



B. Right Click → Dynamic Report to show dynamic report



C. See the overall througput through the radio

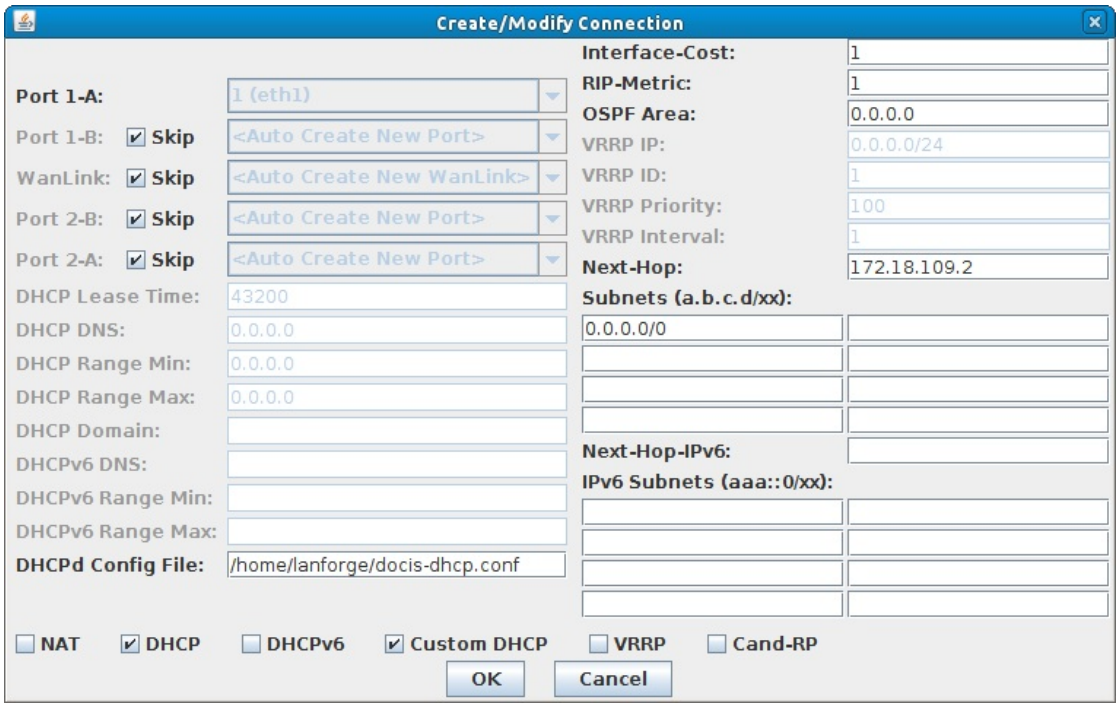


Scripted All-in-One Cable Modem Testing

Goal: Use LANforge to test CMTS and cable-modem network (DUT).

LANforge will serve DHCP and TFTP to the DUT, generate TCP traffic to measure upload and download speeds on each modem, and generate reports. In this example, eth0 is the management port, eth1 is the upstream network port (connects to CMTS), and ports eth2-eth9 are connected to cable modems. Eth2-9 are configured for DHCP, and eth1 is configured with static IP. Eth1 is also configured to serve DHCP requests to the cable modems and other LANforge ports.

1. Set up the network interfaces.
2. Configure eth1 to serve DHCP and TFTP.
   A. Go to Status panel in LANforge GUI. Click 'Netsmith' button on the resource.
   B. Right-click in empty space and select **New Router**. Use default values and click **OK**.
   C. Drag eth1 into the virtual router. Then double-click the virtual eth1 icon to configure DHCP. DHCP files for CMTS setups must be hand-written by the user. Configure LANforge to use this custom-dhcp config file.

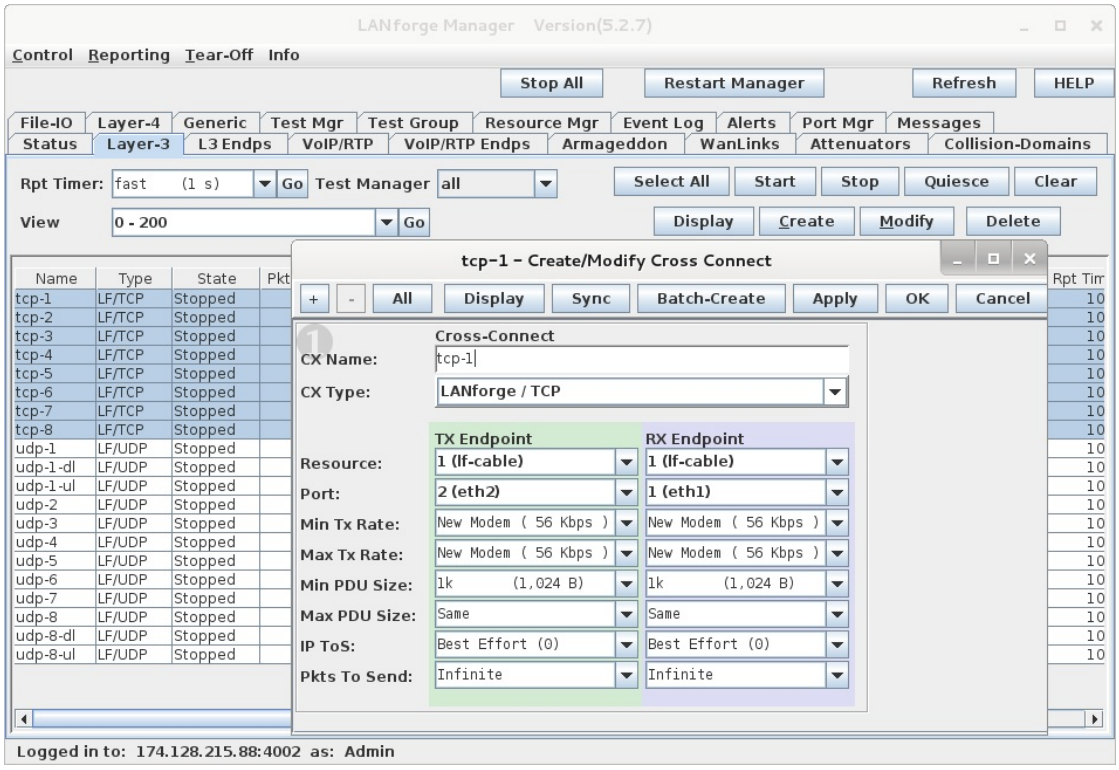


   D. When Netsmith setup is complete, click Apply on the main Netsmith window to start up the DHCP service, etc. This screenshot was taken after the Layer-3 connections were set up.

3. Configure eth2-9 to use DHCP.
   A. Go to Port-Mgr tab, double-click each port, and make sure the DHCP option is selected. Apply changes.

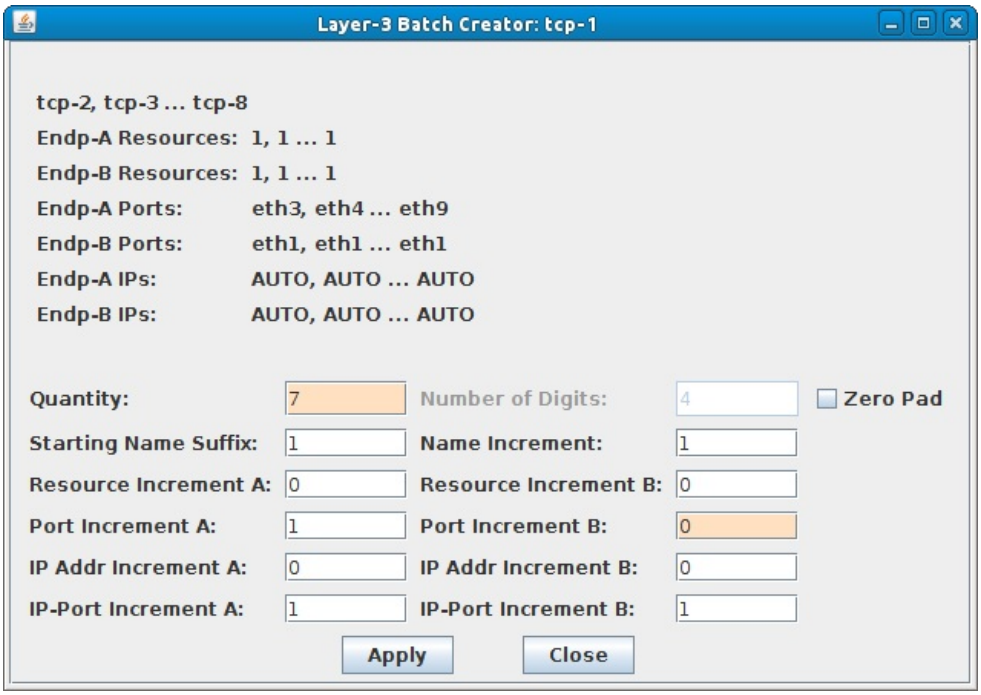
4. Set up Layer-3 connections to generate the throughput tests.

A. Go to the Layer-3 tab, and click Create. Many cable modems are configured to do NAT and/or some firewalling, so normally you will need the connections to be TCP and to originate out from the client-side port. So, choose the LANforge/TCP connection type, and make the 'B' side eth1. Everything else can stay at the default values because we will use a script to automate the settings.



B. Create 7 similar connections, always with B side port of eth1, and use A side ports eth2-eth9. In the Create/Modify window you can just change the name, change the port, and press apply. This will make copies of the connections.

A. You can also use **Batch-Create** (located in the Create/Modify window) to create these connections.

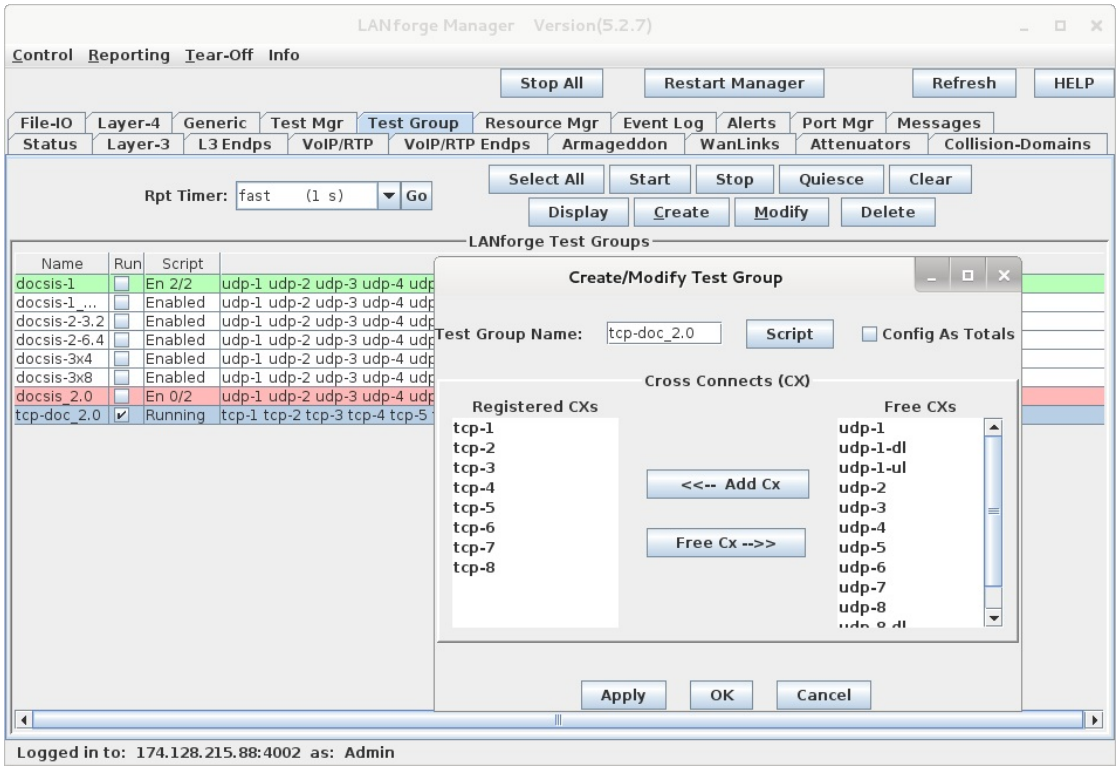


B. Set **Quantity** to 7, **Port Increment B** to 0, and deselect **Zero Pad**.

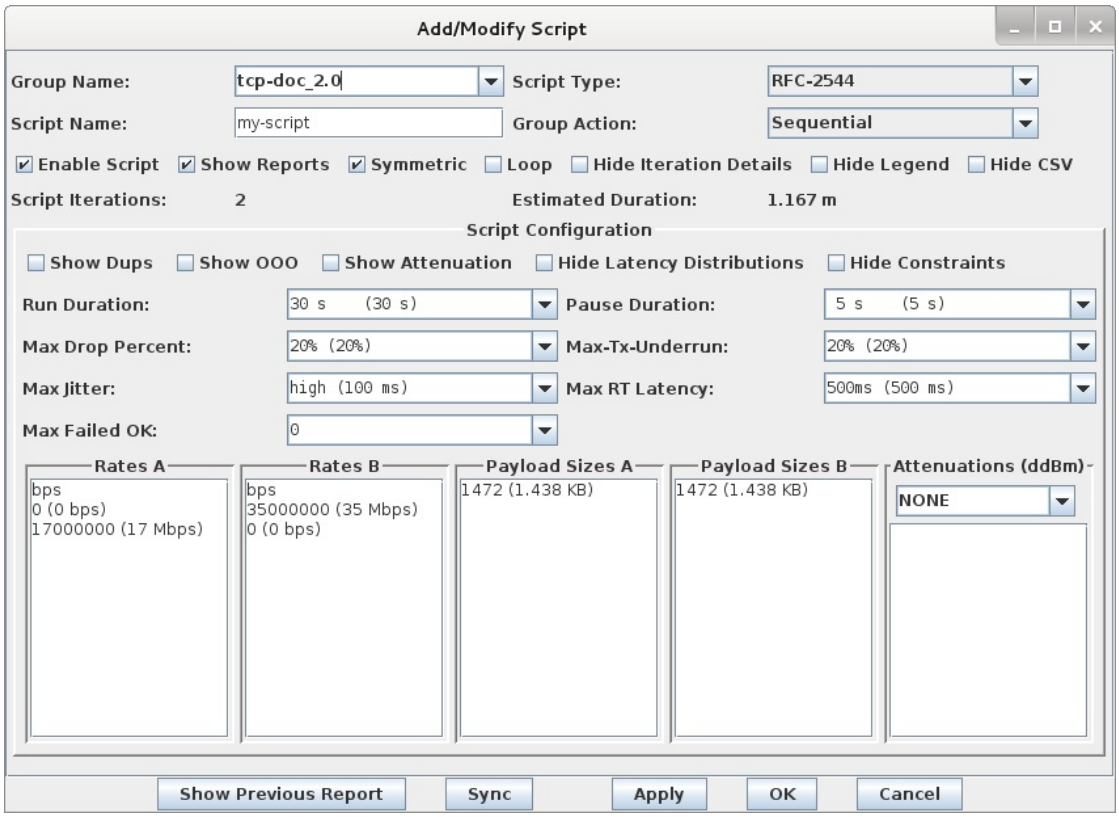
C. Click **Apply**.

5. Create Test-Group to control the 8 Layer-3 TCP connections.

A. Go to the Test Group tab, and click Create. Give your test group a name. Select the 8 Layer-3 TCP connections you just created in the previous step and add those to the Test Group. Click Apply and make sure the new group shows up in the Test Group table.

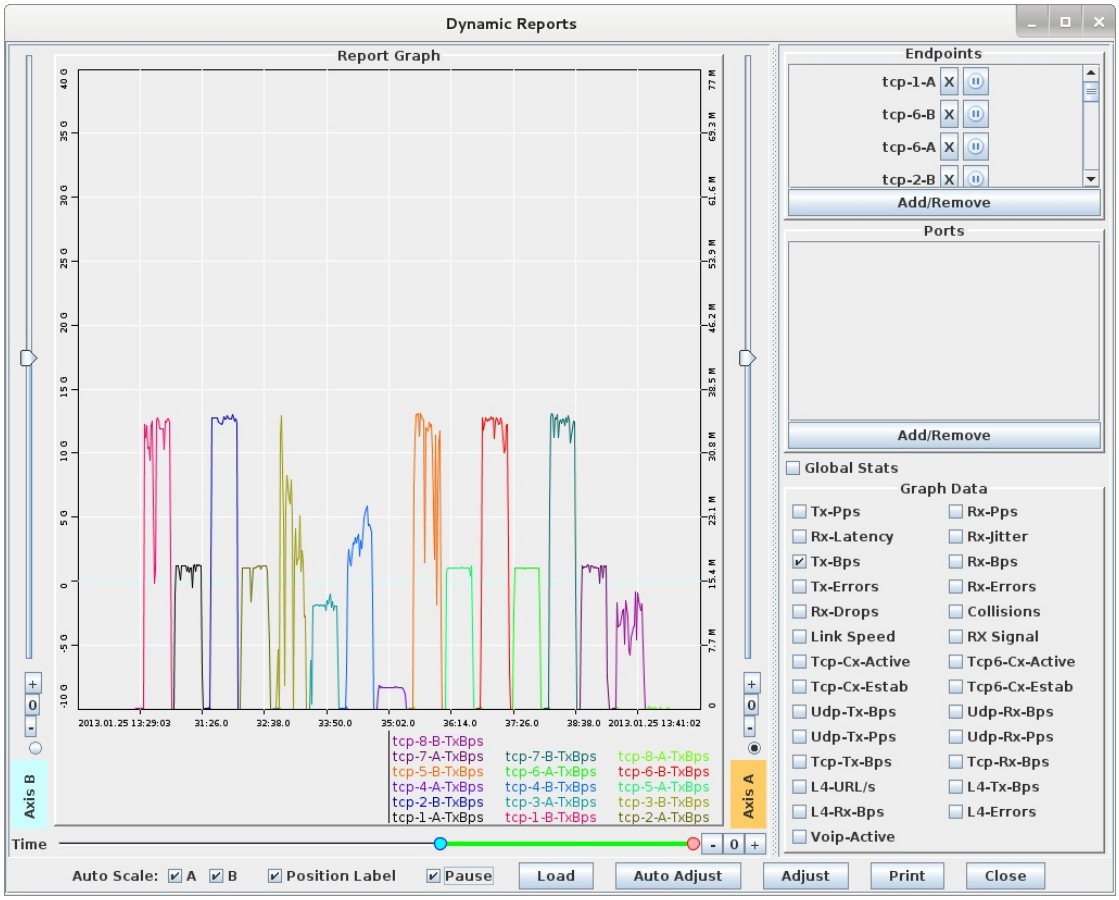


B. We need to add a script to automate the test. Click the Script button in the Test Group create/modify window. Choose RFC 2544 script type. In this scenario, we want to test each modem one at a time, so we select the Sequential option for Group Action. The rates sections determine the speeds for each iteration. In this case, the first iteration will send from B to A at 35Mbps. This is the download test. The next iteration will upload from A to B at 17Mbps. When the two iterations are complete, the script will repeat on the next Layer-3 connection.

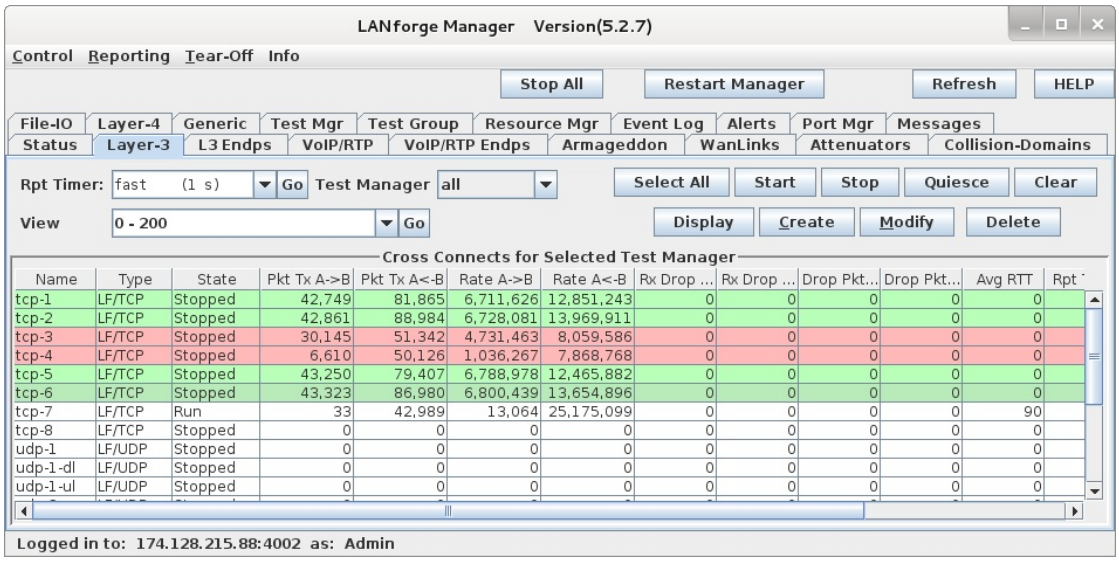


6. Run the test.

A. To start the test, select the Test Group in the table and click the Start button. You should see a script-report window pop up, and the state should go to running. To see a live report of throughput and other values, right-click on the Test Group table and select the Dynamic Report option.



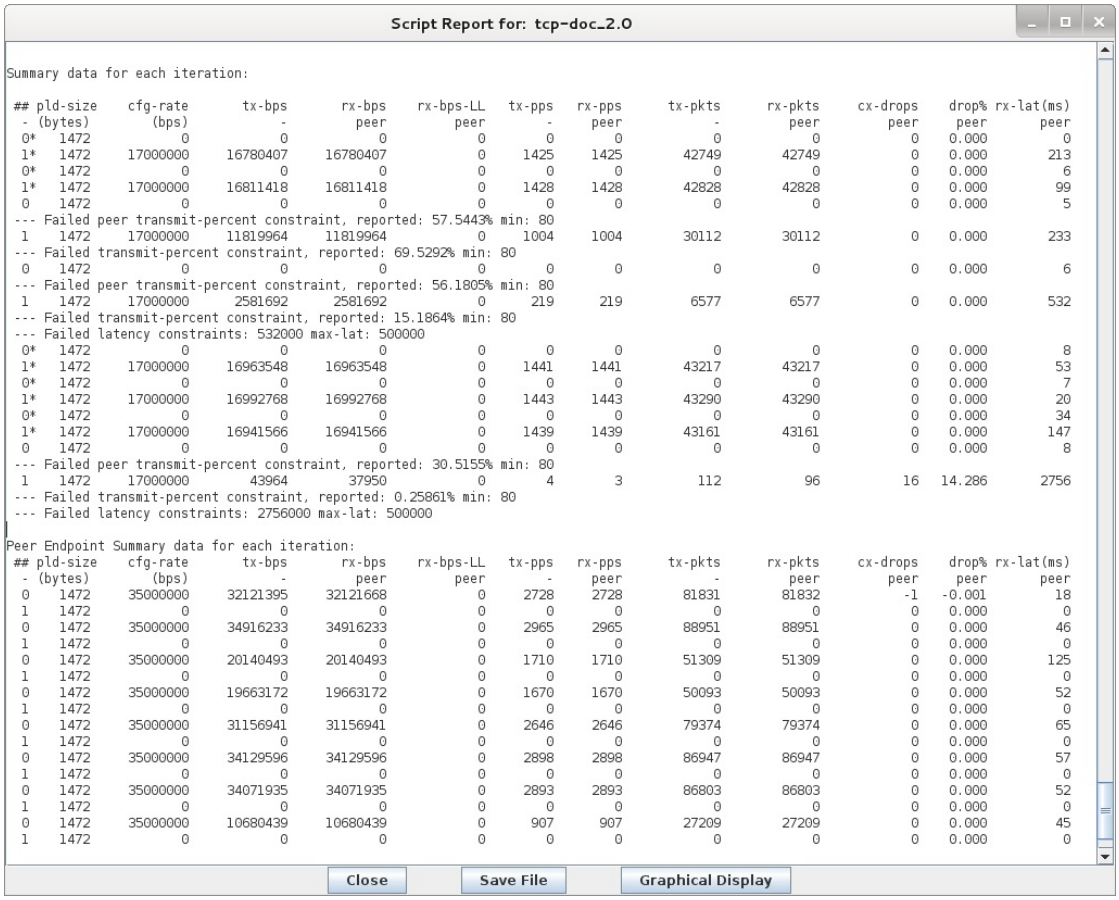
B. View the individual connections from the Layer-3 tab.



Name	Type	State	Pkt Tx A->B	Pkt Tx A<-B	Rate A->B	Rate A<-B	Rx Drop ...	Rx Drop ...	Drop Pkt...	Drop Pkt...	Avg RTT	Rpt
tcp-1	LF/TCP	Stopped	42,749	81,865	6,711,626	12,851,243	0	0	0	0	0	
tcp-2	LF/TCP	Stopped	42,861	88,984	6,728,081	13,969,911	0	0	0	0	0	
tcp-3	LF/TCP	Stopped	30,145	51,342	4,731,463	8,059,586	0	0	0	0	0	
tcp-4	LF/TCP	Stopped	6,610	50,126	1,036,267	7,868,768	0	0	0	0	0	
tcp-5	LF/TCP	Stopped	43,250	79,407	6,788,978	12,465,882	0	0	0	0	0	
tcp-6	LF/TCP	Stopped	43,323	86,980	6,800,439	13,654,896	0	0	0	0	0	
tcp-7	LF/TCP	Run	33	42,989	13,064	25,175,099	0	0	0	0	90	
tcp-8	LF/TCP	Stopped	0	0	0	0	0	0	0	0	0	
udp-1	LF/UDP	Stopped	0	0	0	0	0	0	0	0	0	
udp-1-dl	LF/UDP	Stopped	0	0	0	0	0	0	0	0	0	
udp-1-ul	LF/UDP	Stopped	0	0	0	0	0	0	0	0	0	

7. Gather reports.

A. When the script completes, a summary of each iteration will be shown in the Script Report window. The text may be converted to HTML, but for this particular script, the graphs are not useful because we are using Sequential mode, so just saving the text to a file or printing it is probably the best option. Note that some of these tests failed the constraints configured in the script. In this case, it is because those modems were not capable of the requested speeds.



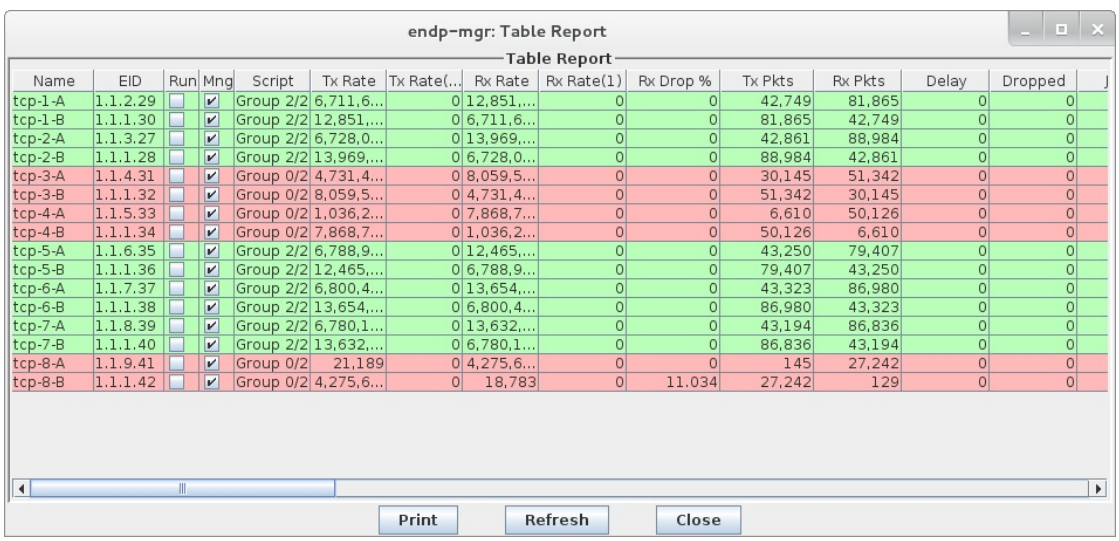
Script Report for: tcp-doc_2.0

```
Summary data for each iteration:

 ## pld-size    cfg-rate       tx-bps       rx-bps    rx-bps-LL   tx-pps   rx-pps      tx-pkts      rx-pkts     cx-drops    drop% rx-lat(ms)
  - (bytes)        (bps)            -         peer         peer        -     peer            -         peer         peer     peer       peer
  0*   1472            0            0            0            0        0        0            0            0            0    0.000          0
  1*   1472     17000000     16780407     16780407            0     1425     1425        42749        42749            0    0.000        213
  0*   1472            0            0            0            0        0        0            0            0            0    0.000          6
  1*   1472     17000000     16811418     16811418            0     1428     1428        42828        42828            0    0.000         99
  0    1472            0            0            0            0        0        0            0            0            0    0.000          5
 --- Failed peer transmit-percent constraint, reported: 57.5443% min: 80
  1    1472     17000000     11819964     11819964            0     1004     1004        30112        30112            0    0.000        233
 --- Failed transmit-percent constraint, reported: 69.5292% min: 80
  0    1472            0            0            0            0        0        0            0            0            0    0.000          6
 --- Failed peer transmit-percent constraint, reported: 56.1805% min: 80
  1    1472     17000000      2581692      2581692            0      219      219         6577         6577            0    0.000        532
 --- Failed transmit-percent constraint, reported: 15.1864% min: 80
 --- Failed latency constraints: 532000 max-lat: 500000
  0*   1472            0            0            0            0        0        0            0            0            0    0.000          8
  1*   1472     17000000     16963548     16963548            0     1441     1441        43217        43217            0    0.000         53
  0*   1472            0            0            0            0        0        0            0            0            0    0.000          7
  1*   1472     17000000     16992768     16992768            0     1443     1443        43290        43290            0    0.000         20
  0*   1472            0            0            0            0        0        0            0            0            0    0.000         34
  1*   1472     17000000     16941566     16941566            0     1439     1439        43161        43161            0    0.000        147
  0    1472            0            0            0            0        0        0            0            0            0    0.000          8
 --- Failed peer transmit-percent constraint, reported: 30.5155% min: 80
  1    1472     17000000        43964        37950            0        4        3          112           96           16   14.286       2756
 --- Failed transmit-percent constraint, reported: 0.25861% min: 80
 --- Failed latency constraints: 2756000 max-lat: 500000

Peer Endpoint Summary data for each iteration:
 ## pld-size    cfg-rate       tx-bps       rx-bps    rx-bps-LL   tx-pps   rx-pps      tx-pkts      rx-pkts     cx-drops    drop% rx-lat(ms)
  - (bytes)        (bps)            -         peer         peer        -     peer            -         peer         peer     peer       peer
  0    1472     35000000     32121395     32121668            0     2728     2728        81831        81832           -1   -0.001         18
  1    1472            0            0            0            0        0        0            0            0            0    0.000          0
  0    1472     35000000     34916233     34916233            0     2965     2965        88951        88951            0    0.000         46
  1    1472            0            0            0            0        0        0            0            0            0    0.000          0
  0    1472     35000000     20140493     20140493            0     1710     1710        51309        51309            0    0.000        125
  1    1472            0            0            0            0        0        0            0            0            0    0.000          0
  0    1472     35000000     19663172     19663172            0     1670     1670        50093        50093            0    0.000         52
  1    1472            0            0            0            0        0        0            0            0            0    0.000          0
  0    1472     35000000     31156941     31156941            0     2646     2646        79374        79374            0    0.000         65
  1    1472            0            0            0            0        0        0            0            0            0    0.000          0
  0    1472     35000000     34129596     34129596            0     2898     2898        86947        86947            0    0.000         57
  1    1472            0            0            0            0        0        0            0            0            0    0.000          0
  0    1472     35000000     34071935     34071935            0     2893     2893        86803        86803            0    0.000         52
  1    1472            0            0            0            0        0        0            0            0            0    0.000          0
  0    1472     35000000     10680439     10680439            0      907      907        27209        27209            0    0.000         45
  1    1472            0            0            0            0        0        0            0            0            0    0.000          0
```

Close | Save File | Graphical Display

B. When the script completes, you could also pause the dynamic report and print it. For an electronic copy, use a PDF printer to create PDF files instead of printing to paper.

C. The Layer-3 tab will color-code the 8 connections, with red meaning fail, and green meaning pass. You can print the connections you are interested in by selecting them in the table and using right-click → Table Report. This report can be printed or otherwise saved.



endp-mgr: Table Report

Table Report

Name	EID	Run	Mng	Script	Tx Rate	Tx Rate(...	Rx Rate	Rx Rate(1)	Rx Drop %	Tx Pkts	Rx Pkts	Delay	Dropped
tcp-1-A	1.1.2.29		✔	Group 2/2	6,711,6...	0	12,851,...	0	0	42,749	81,865	0	0
tcp-1-B	1.1.1.30		✔	Group 2/2	12,851,...	0	6,711,6...	0	0	81,865	42,749	0	0
tcp-2-A	1.1.3.27		✔	Group 2/2	6,728,0...	0	13,969,...	0	0	42,861	88,984	0	0
tcp-2-B	1.1.1.28		✔	Group 2/2	13,969,...	0	6,728,0...	0	0	88,984	42,861	0	0
tcp-3-A	1.1.4.31		✔	Group 0/2	4,731,4...	0	8,059,5...	0	0	30,145	51,342	0	0
tcp-3-B	1.1.1.32		✔	Group 0/2	8,059,5...	0	4,731,4...	0	0	51,342	30,145	0	0
tcp-4-A	1.1.5.33		✔	Group 0/2	1,036,2...	0	7,868,7...	0	0	6,610	50,126	0	0
tcp-4-B	1.1.1.34		✔	Group 0/2	7,868,7...	0	1,036,2...	0	0	50,126	6,610	0	0
tcp-5-A	1.1.6.35		✔	Group 2/2	6,788,9...	0	12,465,...	0	0	43,250	79,407	0	0
tcp-5-B	1.1.1.36		✔	Group 2/2	12,465,...	0	6,788,9...	0	0	79,407	43,250	0	0
tcp-6-A	1.1.7.37		✔	Group 2/2	6,800,4...	0	13,654,...	0	0	43,323	86,980	0	0
tcp-6-B	1.1.1.38		✔	Group 2/2	13,654,...	0	6,800,4...	0	0	86,980	43,323	0	0
tcp-7-A	1.1.8.39		✔	Group 2/2	6,780,1...	0	13,632,...	0	0	43,194	86,836	0	0
tcp-7-B	1.1.1.40		✔	Group 2/2	13,632,...	0	6,780,1...	0	0	86,836	43,194	0	0
tcp-8-A	1.1.9.41		✔	Group 0/2	21,189	0	4,275,6...	0	0	145	27,242	0	0
tcp-8-B	1.1.1.42		✔	Group 0/2	4,275,6...	0	18,783	0	11.034	27,242	129	0	0

Print | Refresh | Close

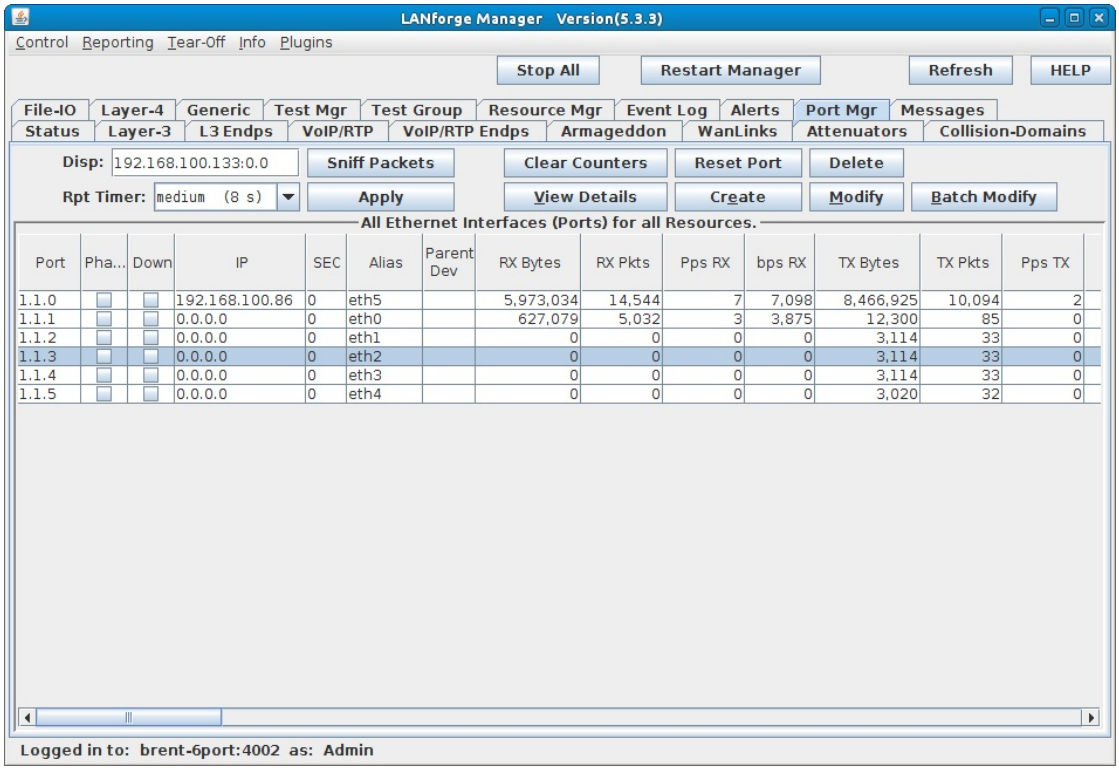
Generating Traffic for DSLAM Testing

Goal: Set up and run traffic to a DSLAM system as shown in the CT570 product description.

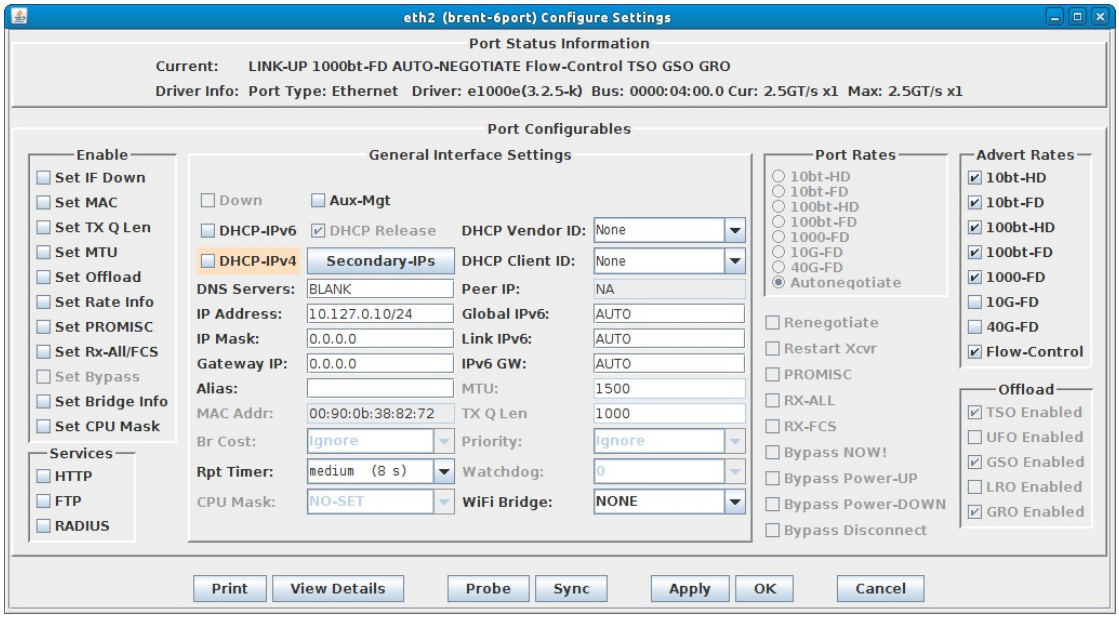
Please refer to the diagram in the CT570 link above. In this test scenario, the LANforge-FIRE Core is one interface on a LANforge system. The LANforge-FIRE Edge is another interface on the same LANforge system. The LF Core interface is connected to the upstream DSLAM interface either directly or through a switch. The CPEs are connected to a multi-port VLAN switch which has a VLAN trunk that is connected to the LF Edge. Traffic is sent between the LF Core and the LF Edge to achieve end-to-end network traffic testing of the DSLAM system.

1. Connect one LANforge-FIRE interface to the DSLAM upstream internet connection. This is the LF Core part of the diagram mentioned above.

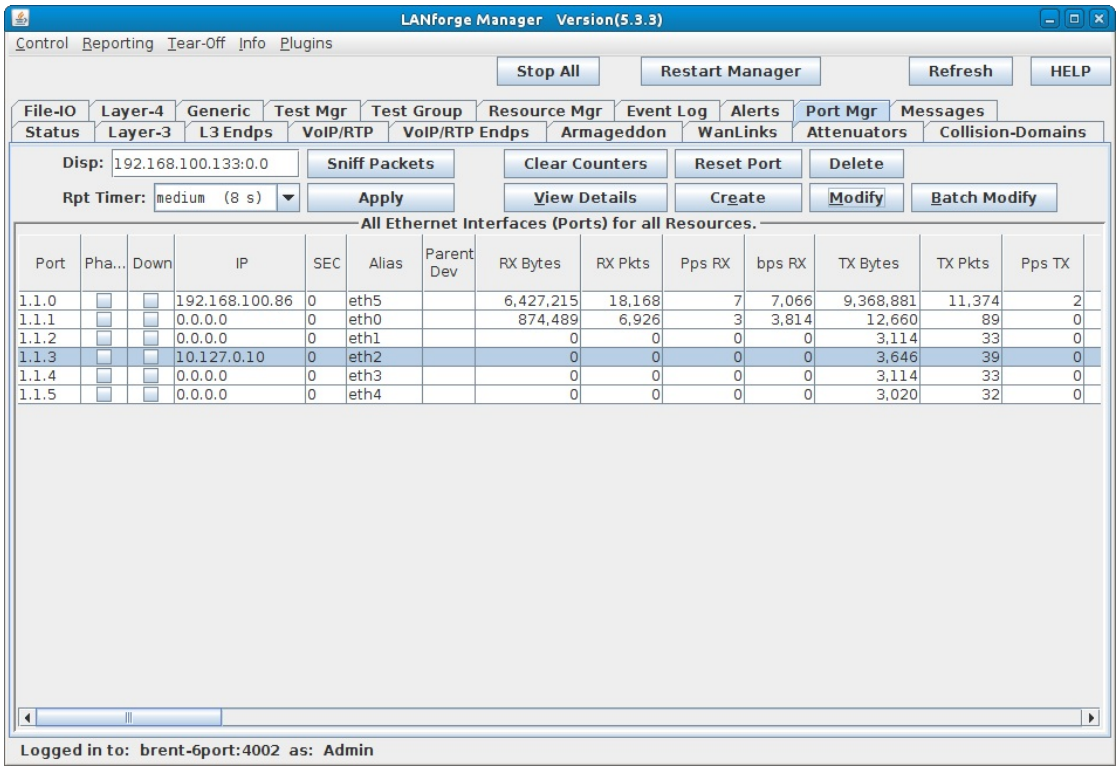
2. Set up the LANforge port so that it has a valid IP address and IP mask.
   A. Go to the Port Manager



   B. Modify the port connected to the DSLAM. Set a valid network IP address and IP mask.



C. Verify the port configuration

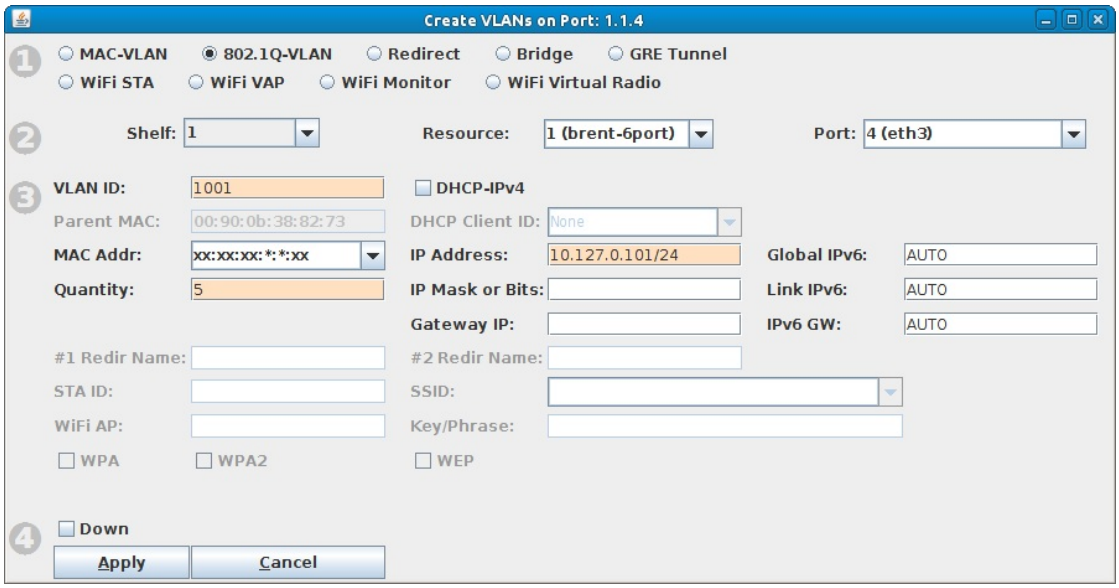


Port	Pha...	Down	IP	SEC	Alias	Parent Dev	RX Bytes	RX Pkts	Pps RX	bps RX	TX Bytes	TX Pkts	Pps TX
1.1.0			192.168.100.86	0	eth5		6,427,215	18,168	7	7,066	9,368,881	11,374	2
1.1.1			0.0.0.0	0	eth0		874,489	6,926	3	3,814	12,660	89	0
1.1.2			0.0.0.0	0	eth1		0	0	0	0	3,114	33	0
1.1.3			10.127.0.10	0	eth2		0	0	0	0	3,646	39	0
1.1.4			0.0.0.0	0	eth3		0	0	0	0	3,114	33	0
1.1.5			0.0.0.0	0	eth4		0	0	0	0	3,020	32	0

For more information see LANforge User's Guide: Ports (Interfaces)

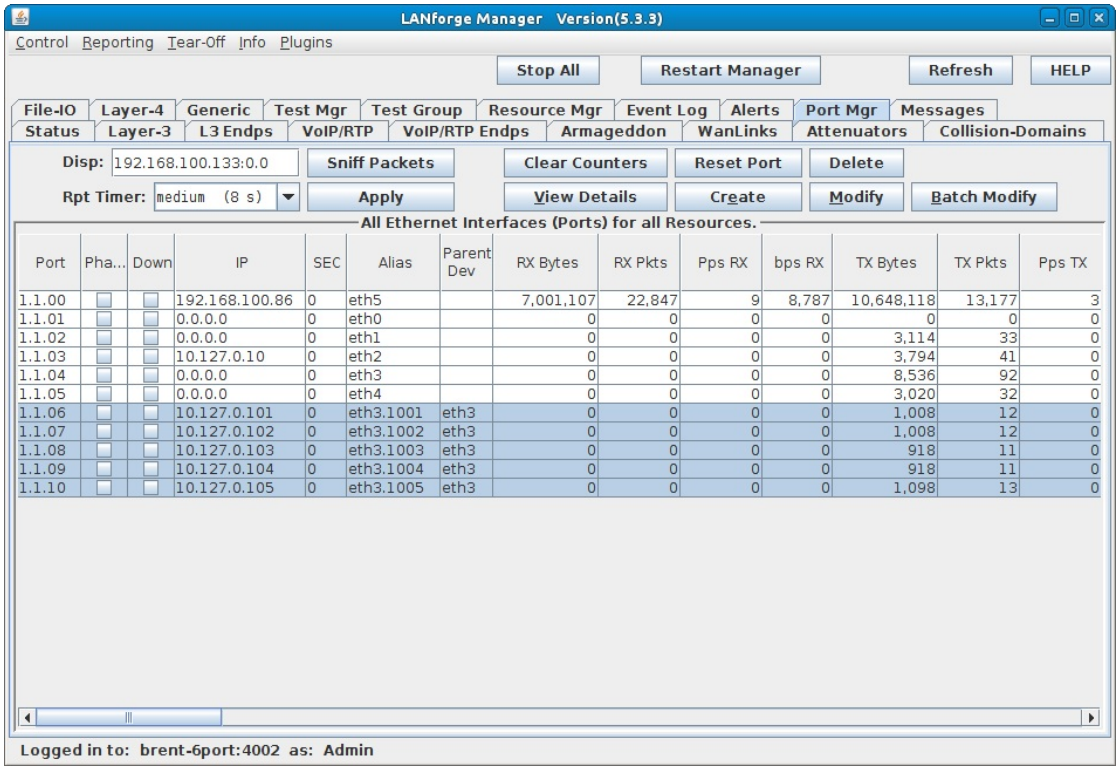
3. Set up virtual interfaces.
   A. On the **Port Mgr** tab, select the port that will connect to the multi-port switch and will also be your VLAN trunk. This is the LF Edge part of the diagram.

   B. Click the **Create** button:



   A. Select the **802.1Q-VLAN** virtual interface type
   B. The **VLAN ID** should correspond to your multi-port switch VLAN configuration
   C. Enter **Quantity** (number of virtual interfaces to be created)
   D. Enter the appropriate IP address and IP mask
   E. Click **Apply** to create the virtual interfaces

   C. Verify that the virtual interfaces are created and have the correct IP assignments

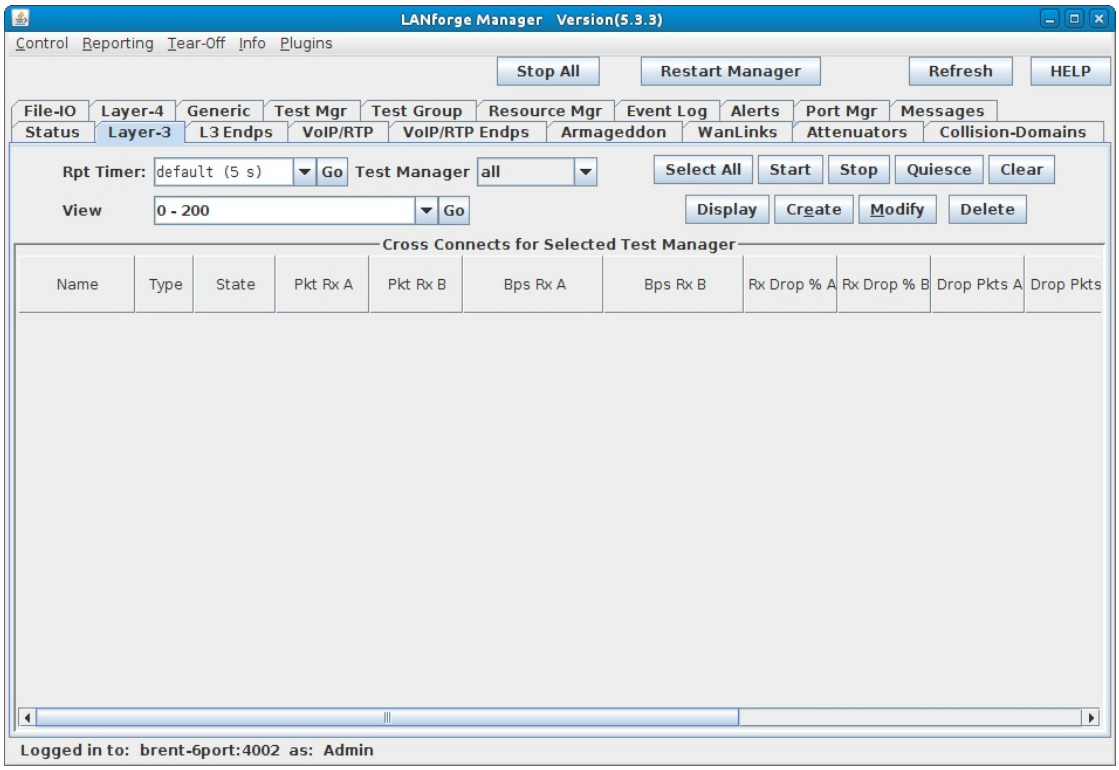


Port	Pha...	Down	IP	SEC	Alias	Parent Dev	RX Bytes	RX Pkts	Pps RX	bps RX	TX Bytes	TX Pkts	Pps TX
1.1.00			192.168.100.86	0	eth5		7,001,107	22,847	9	8,787	10,648,118	13,177	3
1.1.01			0.0.0.0	0	eth0		0	0	0	0	0	0	0
1.1.02			0.0.0.0	0	eth1		0	0	0	0	3,114	33	0
1.1.03			10.127.0.10	0	eth2		0	0	0	0	3,794	41	0
1.1.04			0.0.0.0	0	eth3		0	0	0	0	8,536	92	0
1.1.05			0.0.0.0	0	eth4		0	0	0	0	3,020	32	0
1.1.06			10.127.0.101	0	eth3.1001	eth3	0	0	0	0	1,008	12	0
1.1.07			10.127.0.102	0	eth3.1002	eth3	0	0	0	0	1,008	12	0
1.1.08			10.127.0.103	0	eth3.1003	eth3	0	0	0	0	918	11	0
1.1.09			10.127.0.104	0	eth3.1004	eth3	0	0	0	0	918	11	0
1.1.10			10.127.0.105	0	eth3.1005	eth3	0	0	0	0	1,098	13	0

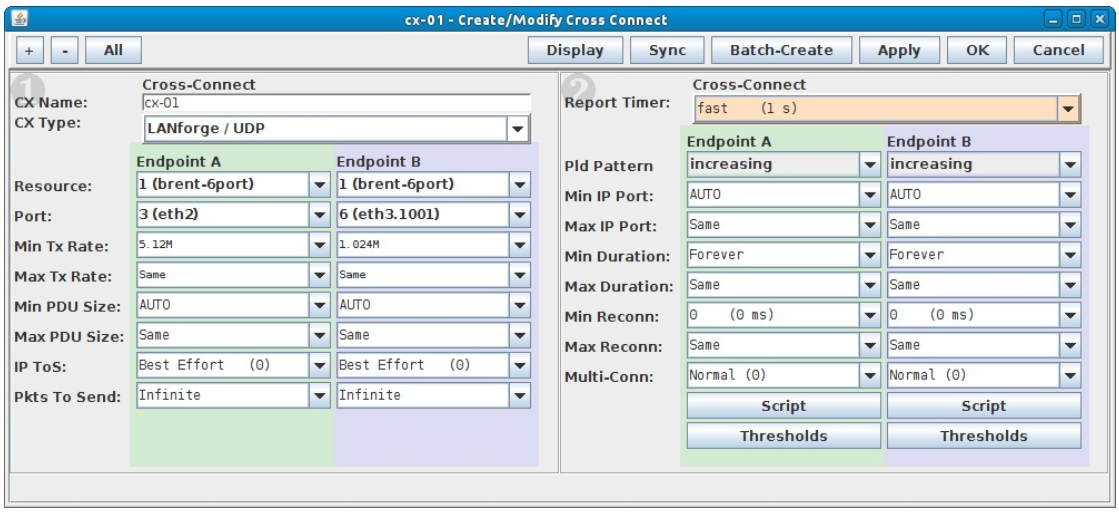
For more information see LANforge User's Guide: Creating & Deleting Virtual Interfaces

4. Create Layer-3 connections.

A. Go to the **Layer-3** tab



B. Click the **Create** button



A. Fill in a connection name, select a connection type and set the report timer

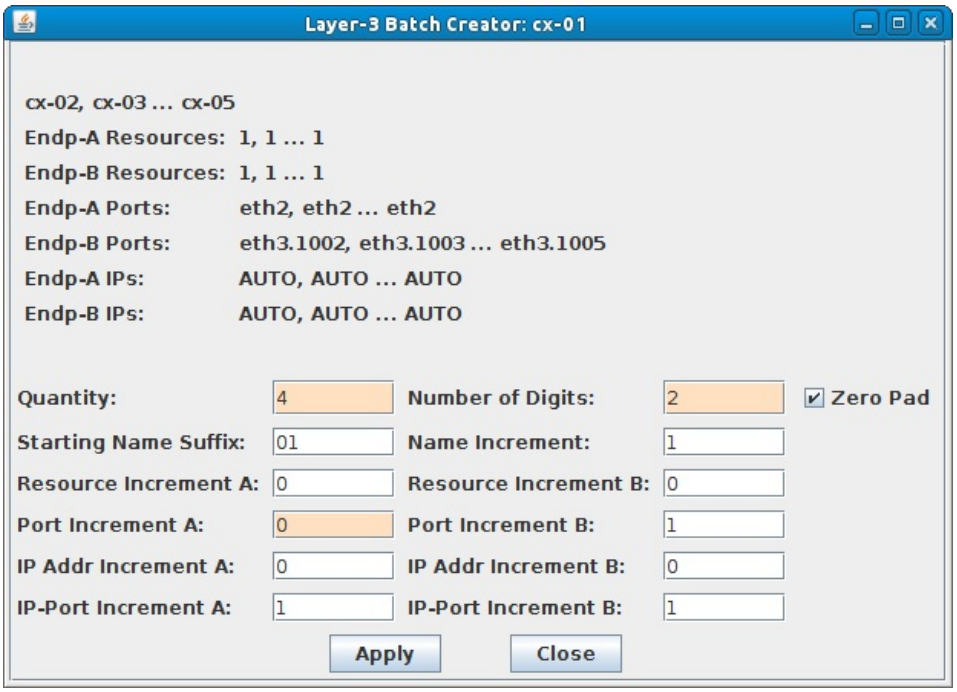
B. Make endpoint A the DSLAM upstream port or LF Core

C. Make endpoint B the CPE downstream port or LF Edge

D. You can make the TX/RX rates asymmetric for the connection to simulate how the connection would be used in a real-world scenario. Here we have about 5Mbps downstream (DSLAM to CPE) and 1Mbps upstream (CPE to DSLAM).

E. Click **Apply** to create the connection

C. Create multiple connections



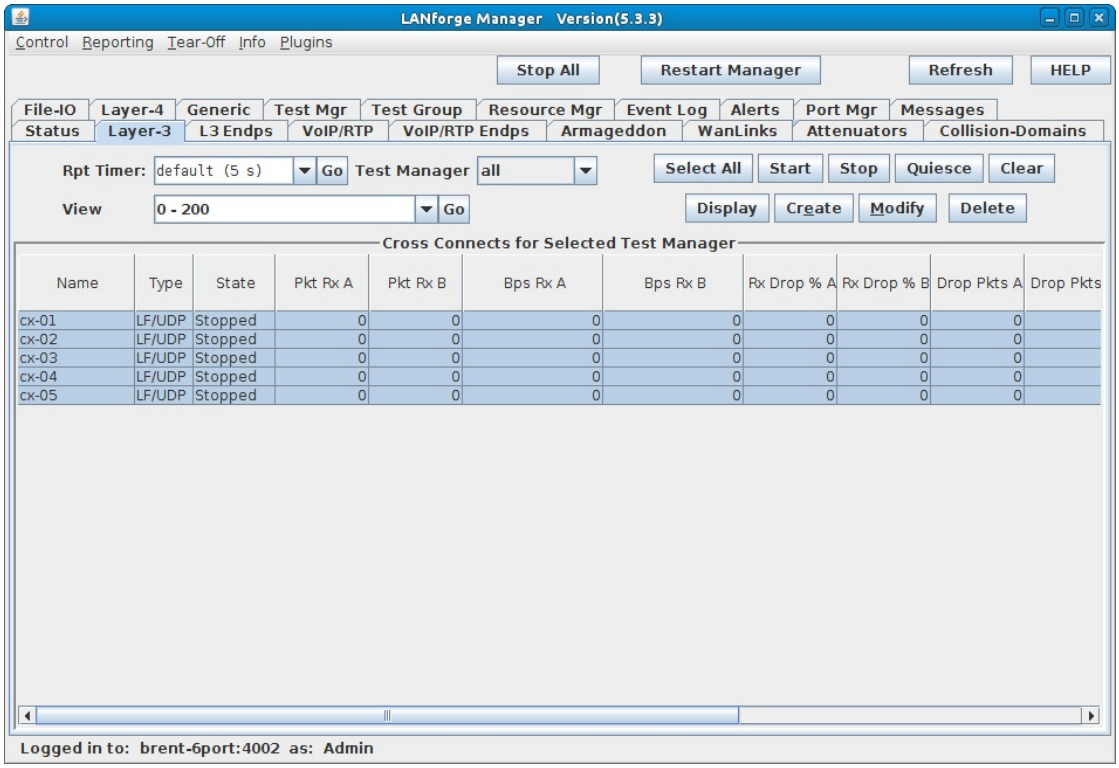
A. Click **Batch-Create** (located in the Layer-3 connection's Create/Modify window) to create four additional connections.

B. Set **Quantity** to 4, **Port Increment A** to 0.

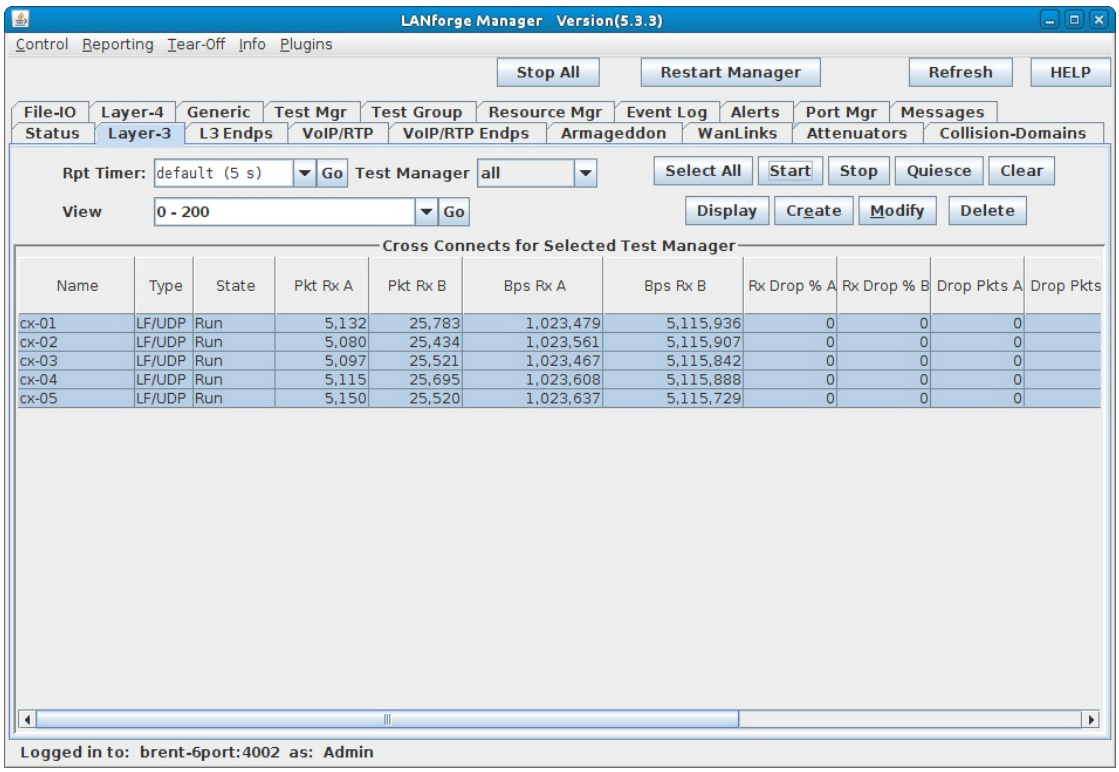
C. Click **Apply**.

For more information see LANforge User's Guide: Layer-3 Endpoints (FIRE)

5. Run traffic.
   A. Select the connections you want to start



   B. Click the **Start** button



For more information see LANforge User's Guide: Layer-3 Endpoints (FIRE)

Generating Traffic for VoIP Testing

Goal: Set up and run VoIP traffic.

In this example, LANforge-FIRE is used to set up two VoIP test calls that may be used as a basis for VoIP load testing or VoIP Gateway testing.

- **Test 1**: Directed VoIP call where a LANforge endpoint calls another LANforge endpoint.
- **Test 2**: Gateway VoIP call where two LANforge endpoints register with a VoIP Gateway so that the call from one endpoint to the other goes through the gateway. The VoIP Gateway used in this example is Asterisk.

> ⚠ This cookbook does not cover FXS/FXO ports (analog RJ11 lines). It is possible that those setup require special commercial gateway features.

---

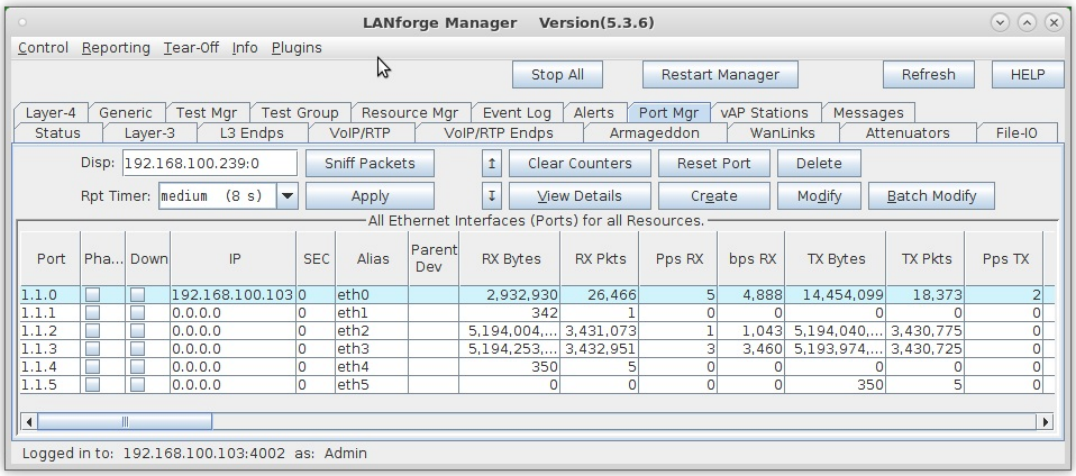
1. ## Set up the LANforge physical connections

   The same two ports are used for both tests. Connect eth1 and eth2 from the LANforge-FIRE system to a network switch that is also connected to the VoIP Gateway. This example assumes that your VoIP Gateway is set up properly. If you need assistance, you can contact us at support@candelatech.com.

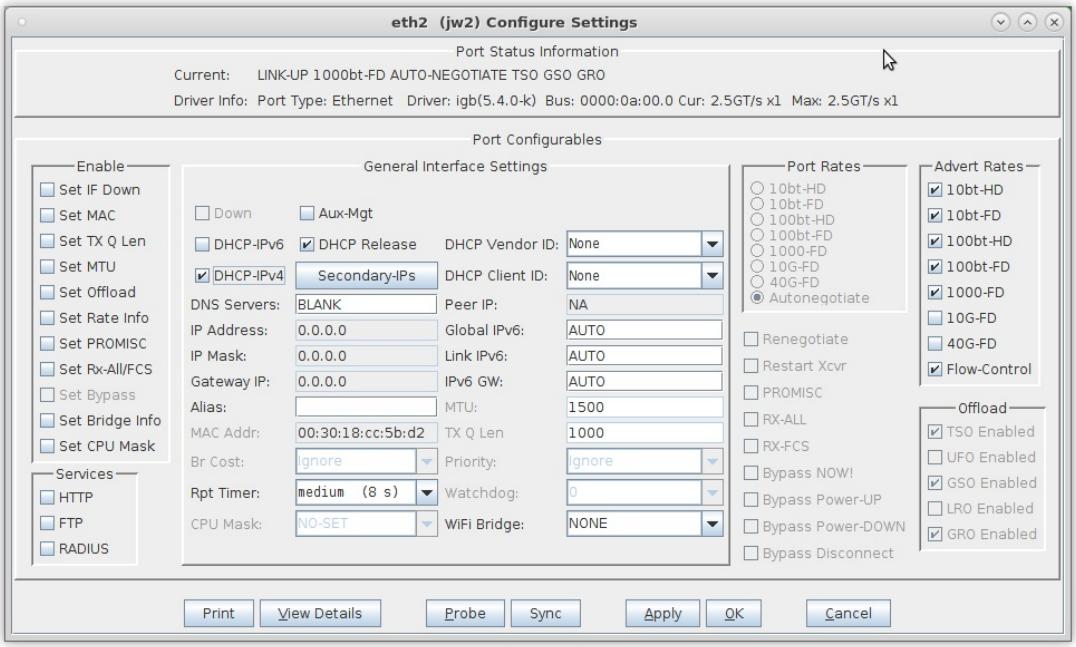
2. ## Configure LANforge ports

   Ports require valid IP addresses and IP masks.

   A. Go to the Port Manager

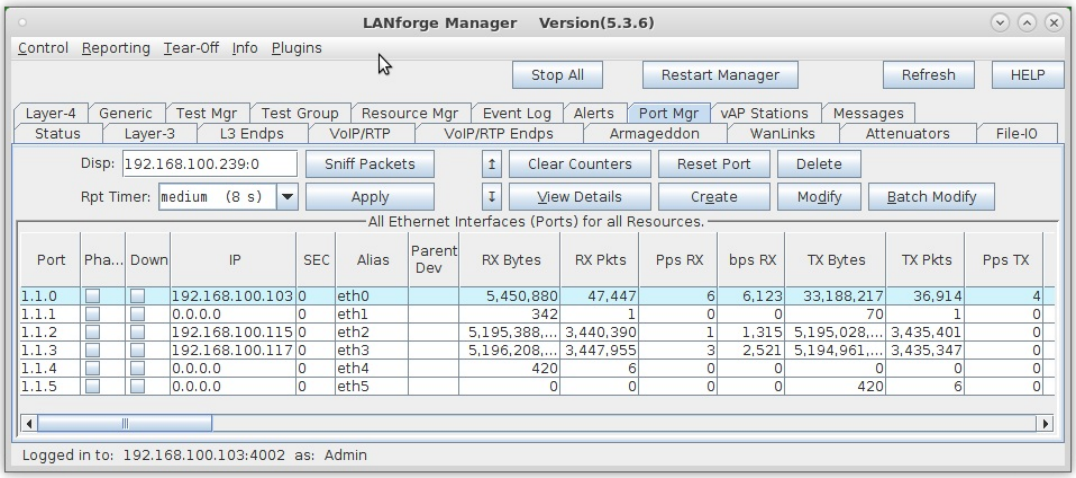


B. Modify eth2 and eth3 to set a valid network IP address and mask



A. If your network has DHCP service, you can select the 'DHCP-IPv4' checkbox so that each port is a DHCP client and will acquire its IP address from your DHCP server

C. Verify the port configuration

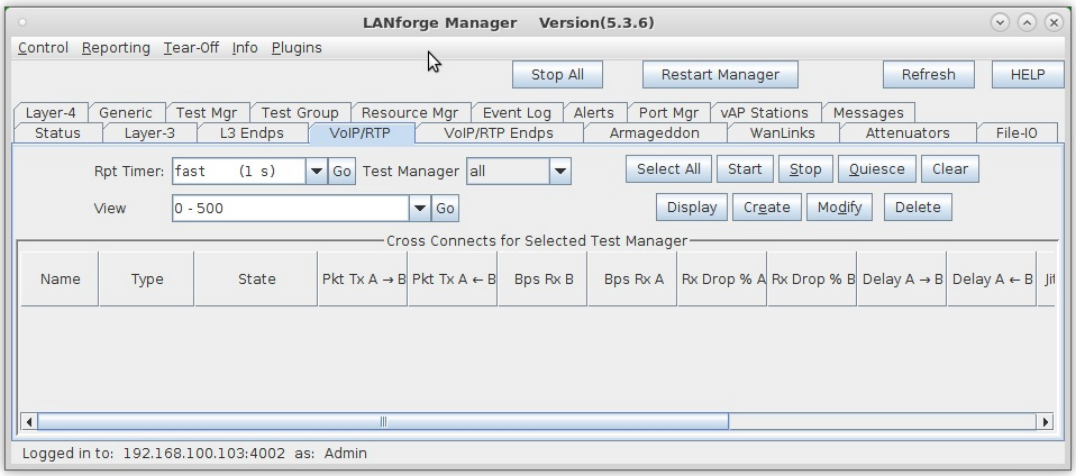


For more information see LANforge User's Guide: Ports (Interfaces)

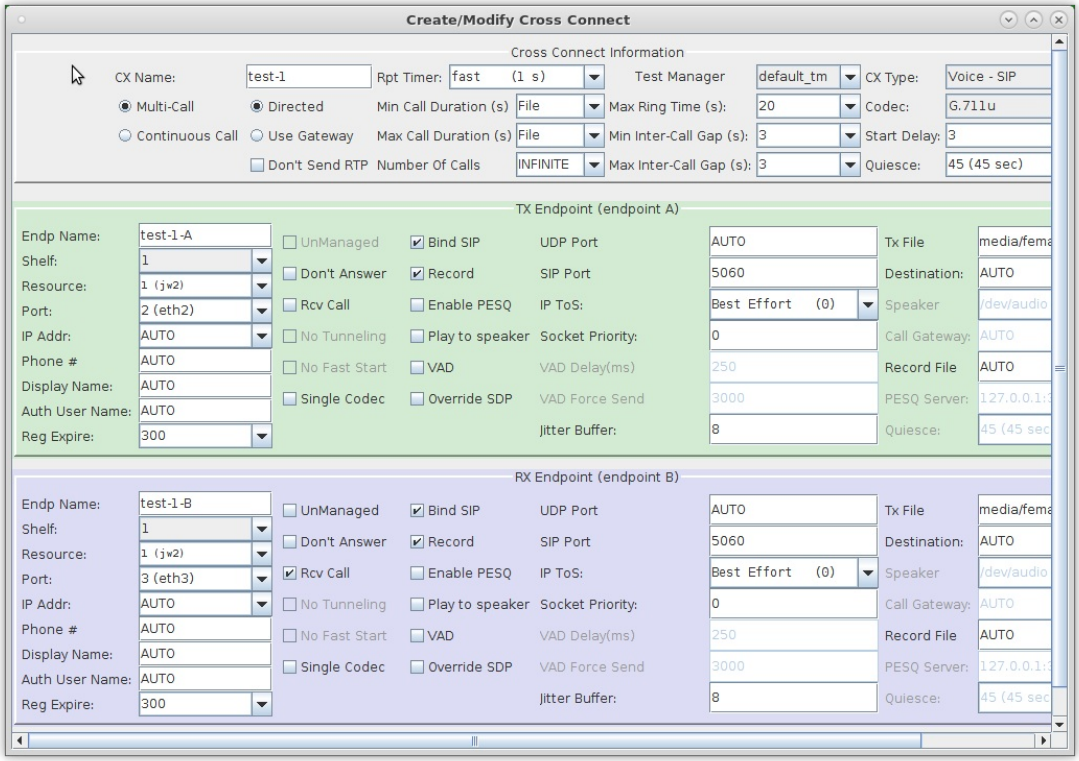
3.

Set up Test 1, a Directed VoIP call.

A. Go to the **VoIP/RTP** tab



B. Click the **Create** button:



A. Enter test-1 in the **CX Name** field

B. Select the **Multi-Call** and **Directed** buttons

C. Endpoint A is on port eth2 in this example. If you have a PESQ licensed server available, you can select Record and Enable PESQ.

D. Endpoint B is on port eth3. If you are using PESQ, be sure to enter a Record File and the IP address and port of your PESQ licensed server. Be sure to select the **Rcv Call** checkbox for this endpoint to receive the call.

E. Be careful of the VoIP phone number: you might have to format the number as extension@IP-address, E.G.: `5678@192.168.1.10`

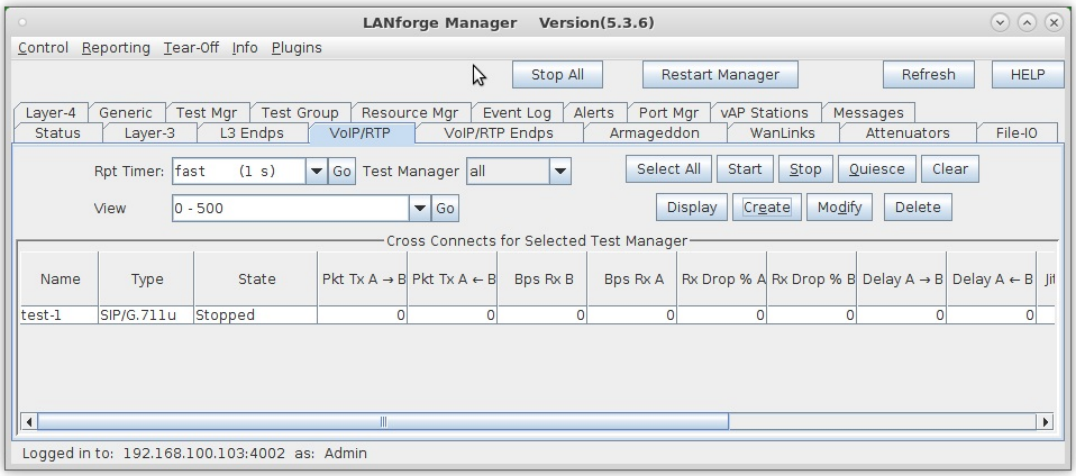
F. Click **OK** to create the VoIP Directed call

C. Verify that the test call is created

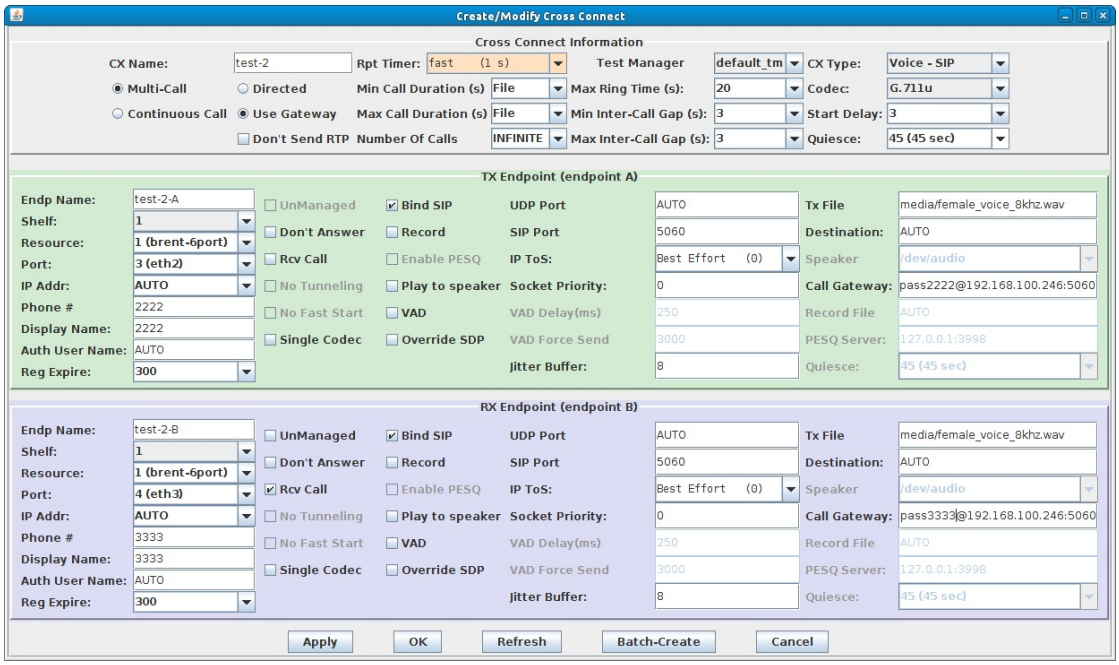
For more information see LANforge User's Guide: VoIP Call Generator

4. # Set up Test 2, a Gateway VoIP call.

A. Go to the VoIP/RTP tab



B. Click the Create button:



A. Enter test-2 in the **CX Name** field

B. Select the **Multi-Call** and **Use Gateway** buttons

C. Endpoint A is on port eth2 in this example. Be sure to enter the proper username and password for the endpoint so that it can authenticate with the VoIP Gateway if necessary.

D. Configure gateway authentication:

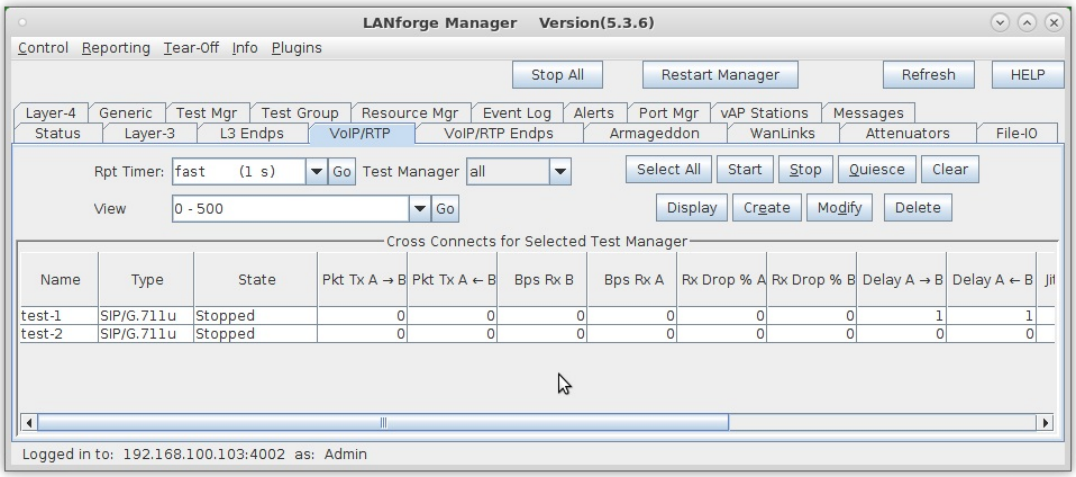
   I. To register with the gateway, often the **Auth User Name** is the phone extension (like `3333`).

   II. The **call gateway** begins with the extension password: `pass333@192.168.100.245:5060`

E. Endpoint B is on port eth3. Be sure to select the **Rcv Call** checkbox for this endpoint to receive the call.

F. Click **OK** to create the VoIP Gateway call

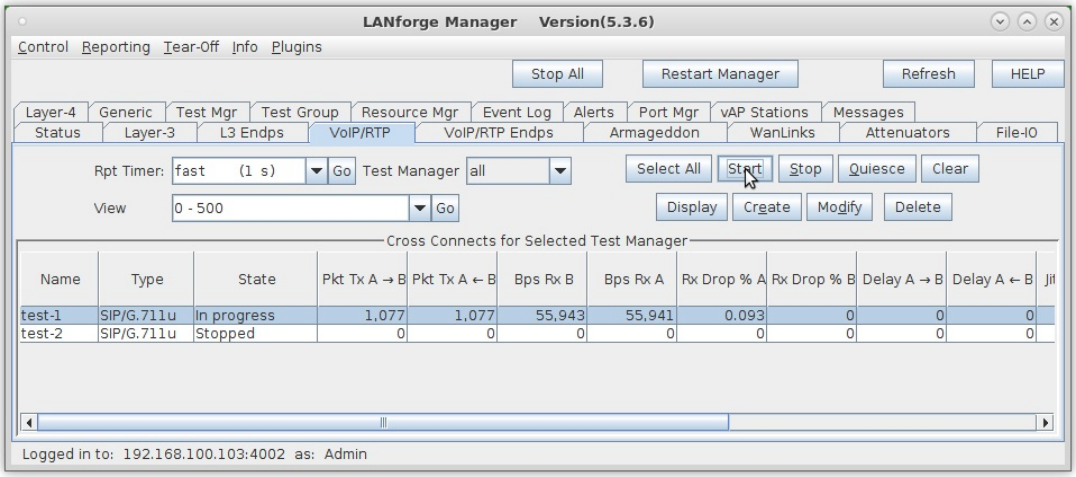
C. Verify that the test call is created



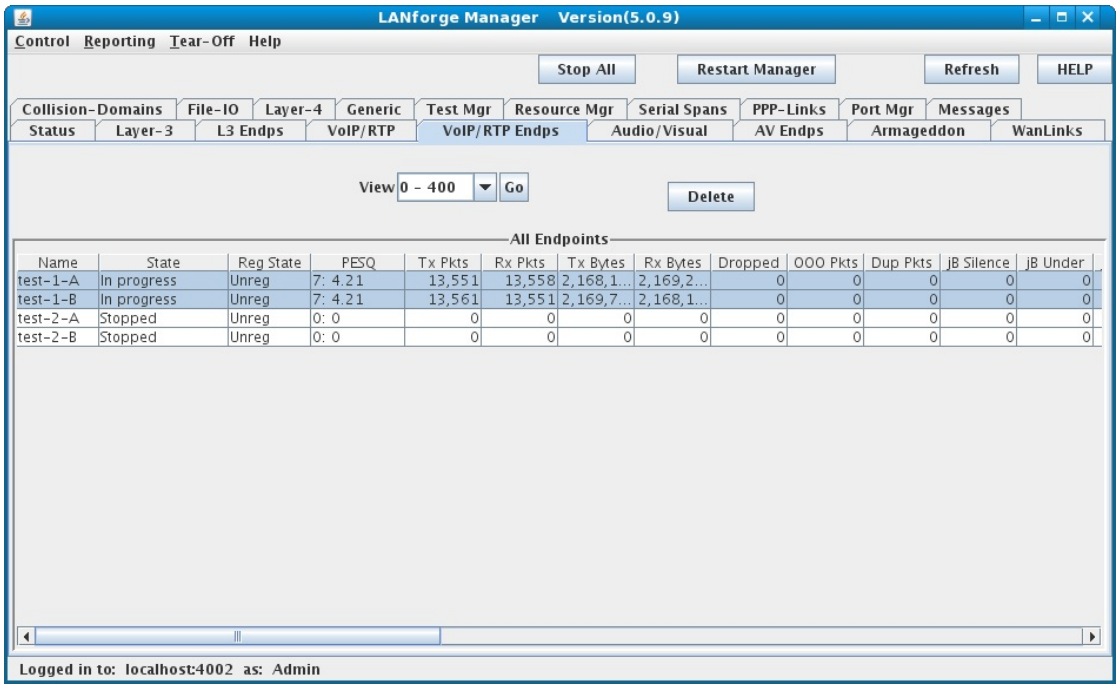
For more information see LANforge User's Guide: VoIP Call Generator

5. **Run test-1 and test-2 individually.**

A. Select test-1 and click the **Start** button



B. Go to the **VoIP/RTP Endps** tab to see detailed results:

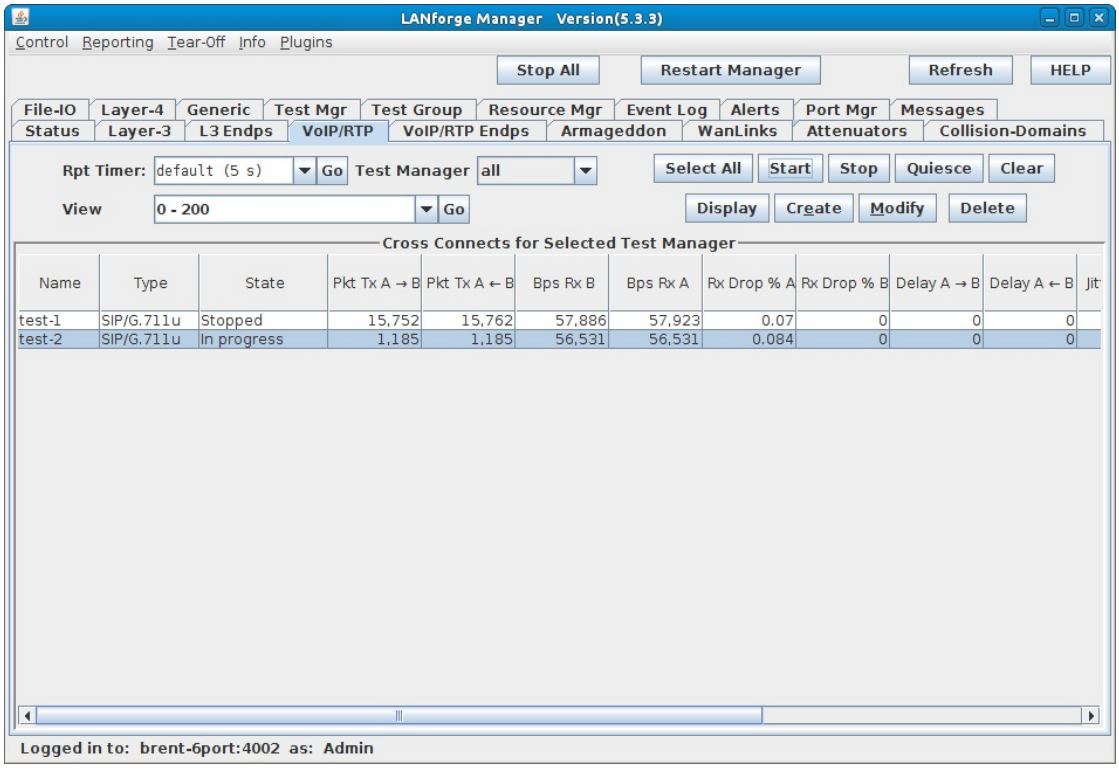


Name	State	Reg State	PESQ	Tx Pkts	Rx Pkts	Tx Bytes	Rx Bytes	Dropped	OOO Pkts	Dup Pkts	jB Silence	jB Under
test-1-A	In progress	Unreg	7: 4.21	13,551	13,558	2,168,1...	2,169,2...	0	0	0	0	0
test-1-B	In progress	Unreg	7: 4.21	13,561	13,551	2,169,7...	2,168,1...	0	0	0	0	0
test-2-A	Stopped	Unreg	0: 0	0	0	0	0	0	0	0	0	0
test-2-B	Stopped	Unreg	0: 0	0	0	0	0	0	0	0	0	0

A. The PESQ score will be reported after the first successful call is completed and updated after each subsequent call

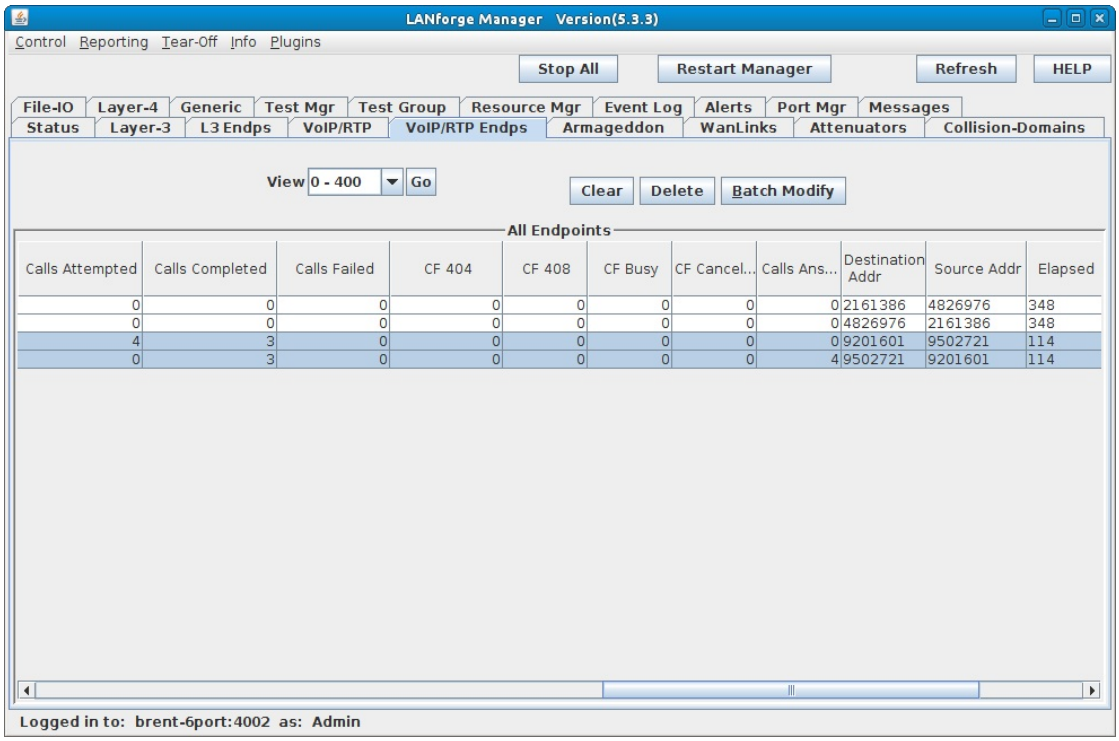
B. **NOTE**: Endpoints are unregistered while the call is in progress because they are not calling through the VoIP gateway

C. Stop test-1, select test-2 and click **Start**



Name	Type	State	Pkt Tx A → B	Pkt Tx A ← B	Bps Rx B	Bps Rx A	Rx Drop % A	Rx Drop % B	Delay A → B	Delay A ← B	Jit
test-1	SIP/G.711u	Stopped	15,752	15,762	57,886	57,923	0.07	0	0	0	
test-2	SIP/G.711u	In progress	1,185	1,185	56,531	56,531	0.084	0	0	0	

D. Go to the **VoIP/RTP Endps** tab to see detailed results:



Calls Attempted	Calls Completed	Calls Failed	CF 404	CF 408	CF Busy	CF Cancel...	Calls Ans...	Destination Addr	Source Addr	Elapsed
0	0	0	0	0	0	0	0	2161386	4826976	348
0	0	0	0	0	0	0	0	4826976	2161386	348
4	3	0	0	0	0	0	0	9201601	9502721	114
0	3	0	0	0	0	0	4	9502721	9201601	114

A. PESQ remains 0: 0 when it is disabled for the call in progress

B. **NOTE**: Endpoints are registered with the VoIP gateway while the call is in progress

C. Calls Attempted, Calls Completed and Calls Failed can be viewed by scrolling to the right on the **VoIP/RTP Endps** tab

For more information see LANforge User's Guide: VoIP Call Generator

6. ## Diagnosing Problems

A. If your VoIP endpoint is not going on-hook, check your VoIP gateway to see if the extension is failing to register.

B. Extensions failing to register might be missing name hostname or IP of the voice gateway they are calling.

C. Some gateways want **Auth User Name** to include the IP or hostname, E.G. `3333@grandstream` or `3333@192.168.100.245`. Make sure any hostnames are resolvable using `nslookup $name` or `host -v $name`.

D. Your VoIP gateway should log when extensions go on-hook, please check the gateway logs

E. Some gateways will not accept direct extension dialing but require extension@gateway style dialing, E.G. the phone number that extension 2222 would call wants to look like `3333@192.168.100.245`

Generating Traffic Using Armageddon

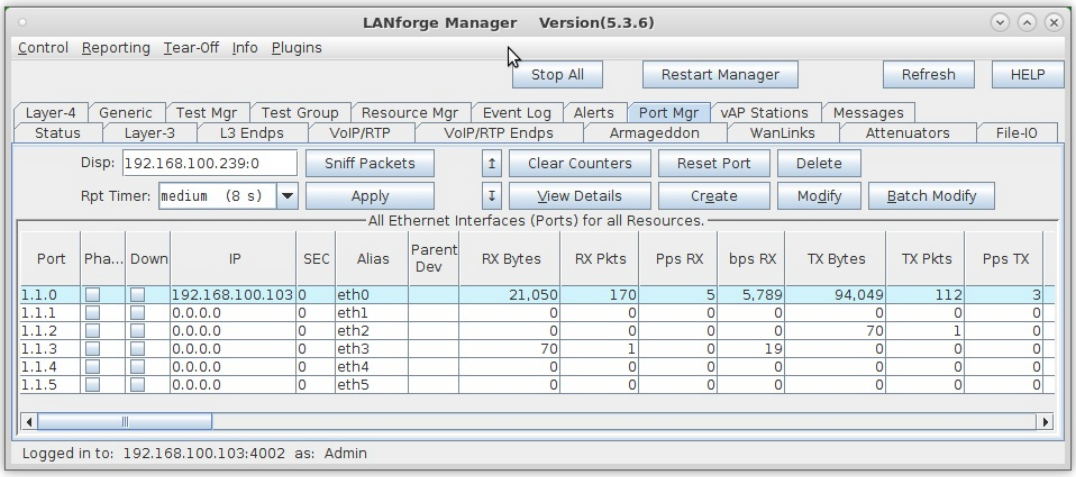
Goal: Set up and run near line-speed 1Gbps traffic using the LANforge Armageddon feature.

- **For more information, see the LANforge User's Guide: Armageddon (Accelerated UDP)**

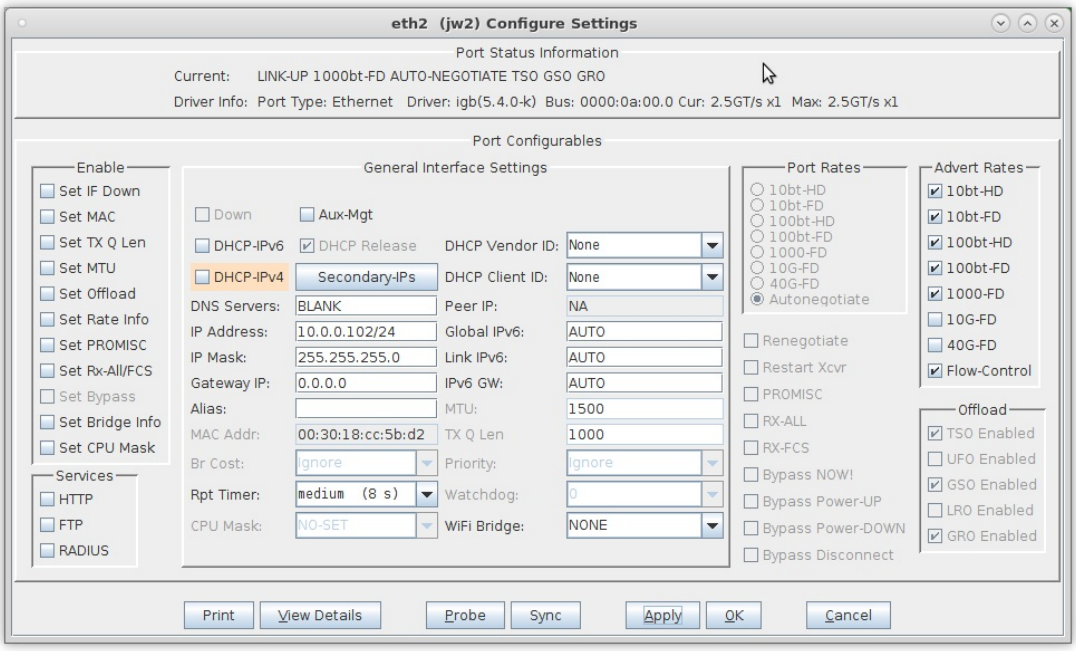
In this test scenario, LANforge Armageddon is set up to run at about 80,000 packets/second full-duplex to achieve near line-speed 1Gbps traffic generation.

Note: In order to use the LANforge Armageddon feature, your system must have the LANforge kernel patch applied and your system must be properly licensed. Please feel free to contact us at support@candelatech.com if you would like to obtain a demo license for the Armageddon feature.

1. Configure the physical interfaces.
   A. Go to the Port Manager and select ports eth2 and eth3

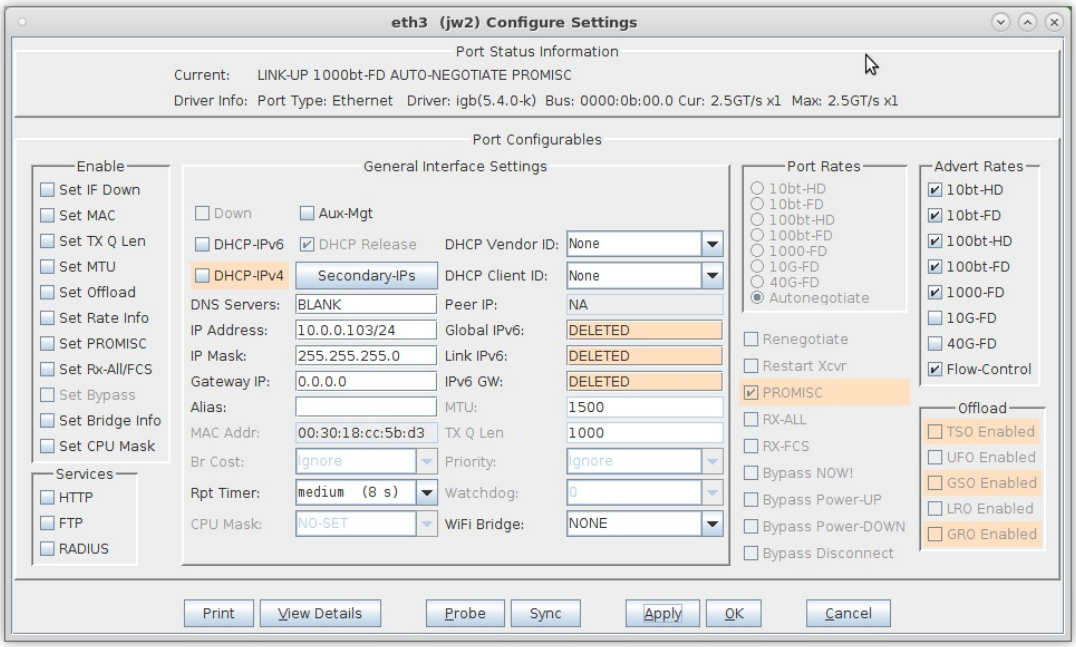


   B. Modify ports eth2 and eth3:



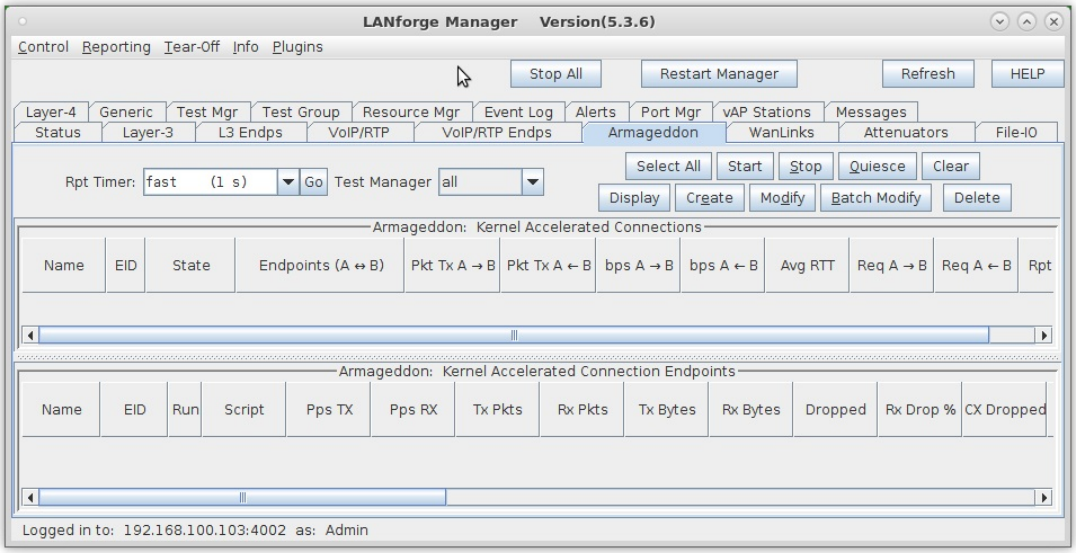
   A. In this example, eth2 and eth3 are physically connected with a patch cable

C. Configure each port with a valid IP address, then click OK

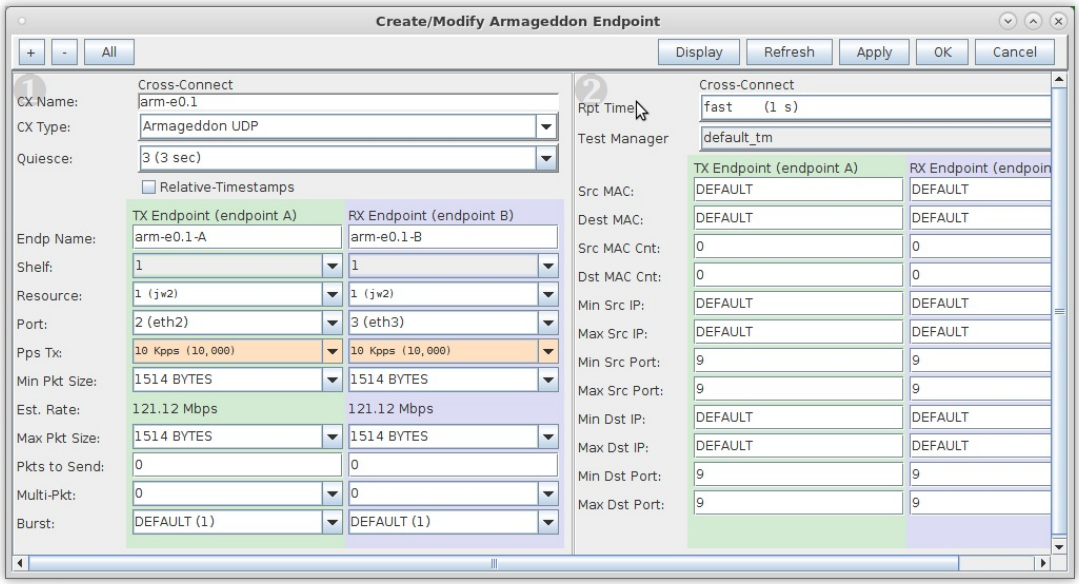


For more information see LANforge User's Guide: Ports (Interfaces)

2. Create the Armageddon cross-connect.
   A. On the **Armageddon** tab, click the **Create** button

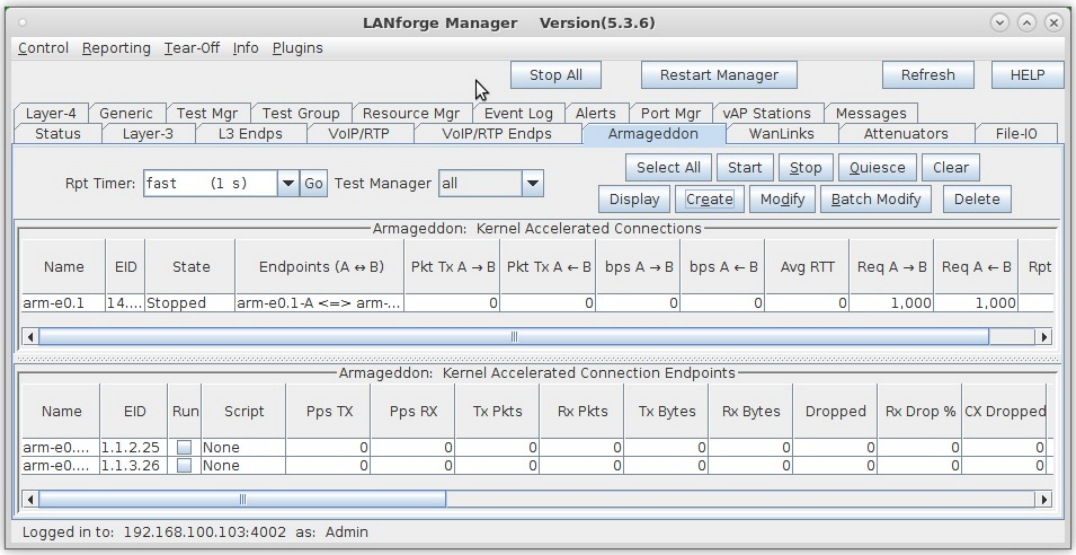


   B. Enter a CX Name, select ports eth2 and eth3, then enter the speed and packet size for both endpoints



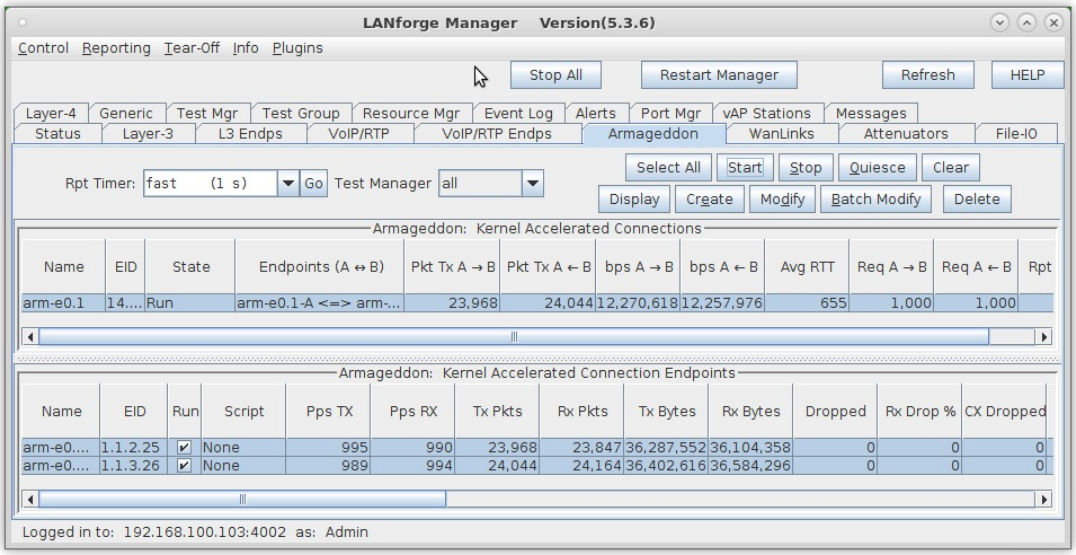
   A. For this example, 10000pps at 1514byte packet size should generate about 121Mbps
   B. Click **OK** when finished

C. Verify that the Armageddon connection is created

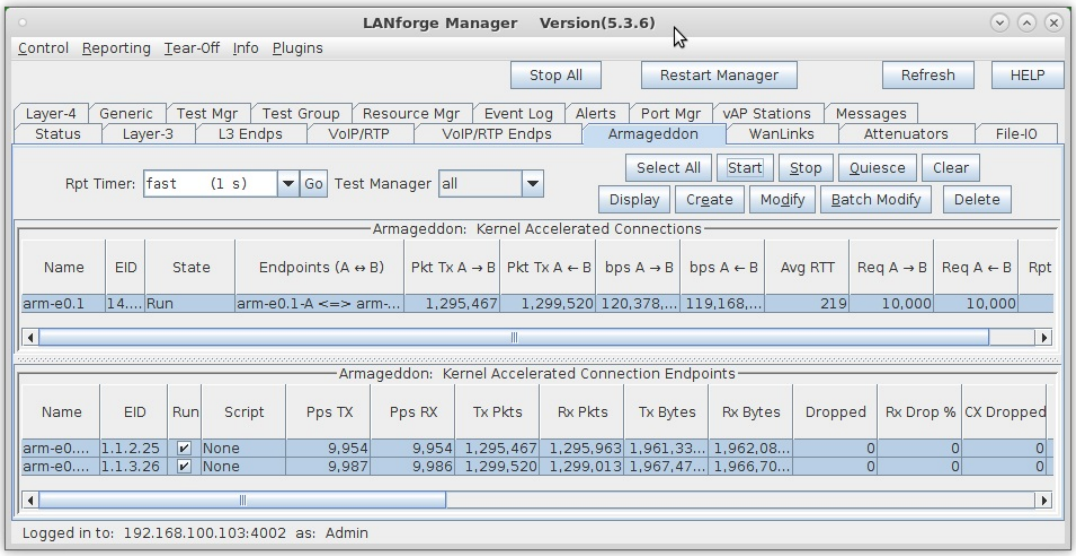


For more information see LANforge User's Guide: Armageddon (Accelerated UDP)

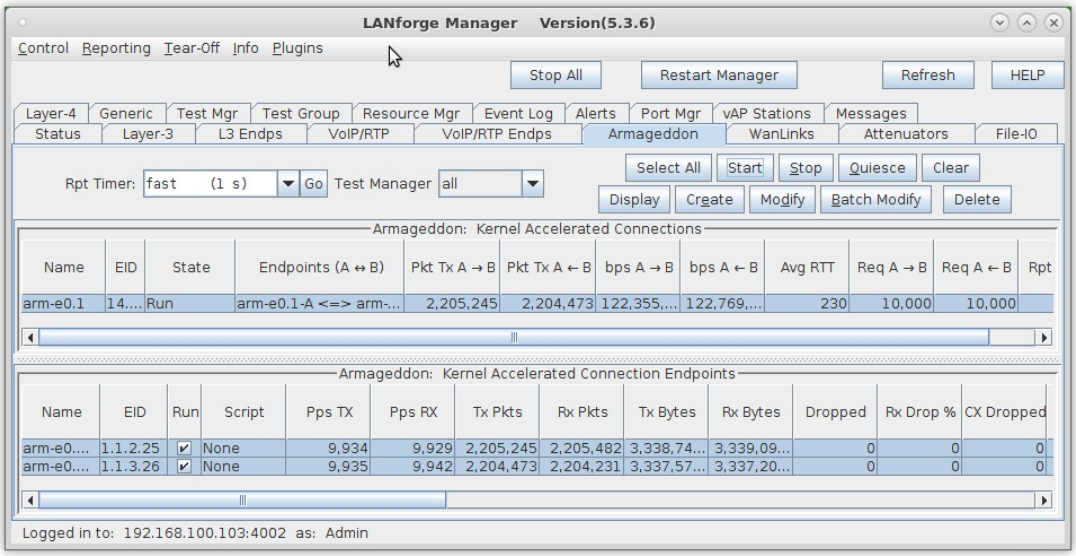
3. Run the Armageddon cross-connect and view results.
   A. Select the Armageddon connection then click Start



B. It will take a moment for the Armageddon traffic to stabilize. Depending on the hardware, LANforge will settle on an actual rate which may differ from the requested rate.



C. In this example, this system (1.8GHz Intel Atom D525, 6 onboard 10/100/1000 ports) is capable of reaching 81,000pps with 1514byte packets or, an equivalent speed of about 990-1005Mbps bi-directional.



A. **NOTE**: Delay for Armageddon connections is measured in microseconds (us) and in this example, the system experiences about 39us of delay when sending to itself.

For more information see LANforge User's Guide: Armageddon (Accelerated UDP)

Generating Armageddon Traffic Containing Random MAC Addresses

Goal: Set up and run traffic containing random MAC addresses using the LANforge Armageddon feature.

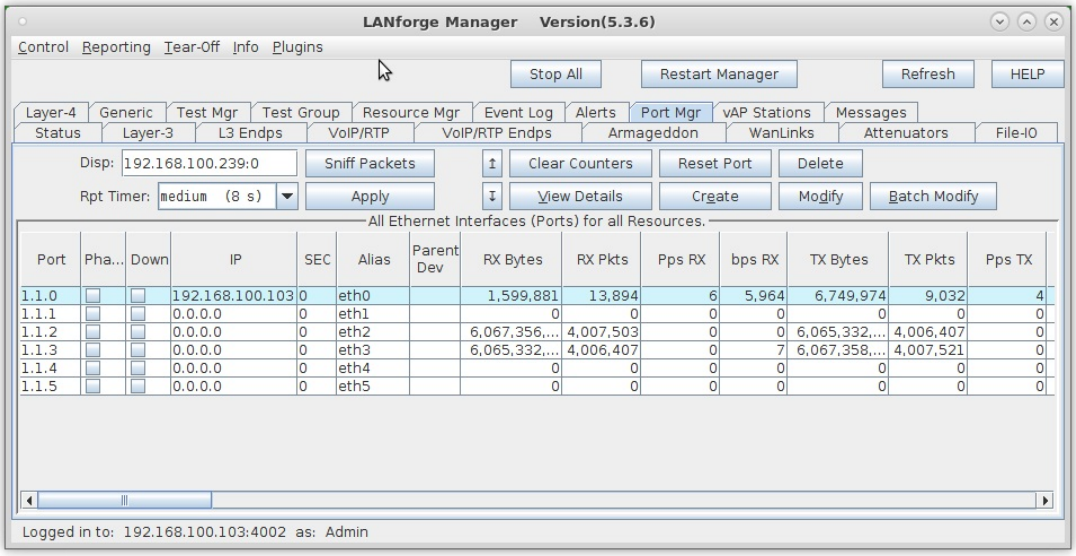
- **For more information, see the LANforge User's Guide: Armageddon (Accelerated UDP)**

In this test scenario, LANforge Armageddon is set up to run with random MAC addresses. This is useful when performance/stress testing network devices that may not be able to keep up with high-speed traffic containing rapidly changing MAC addresses.

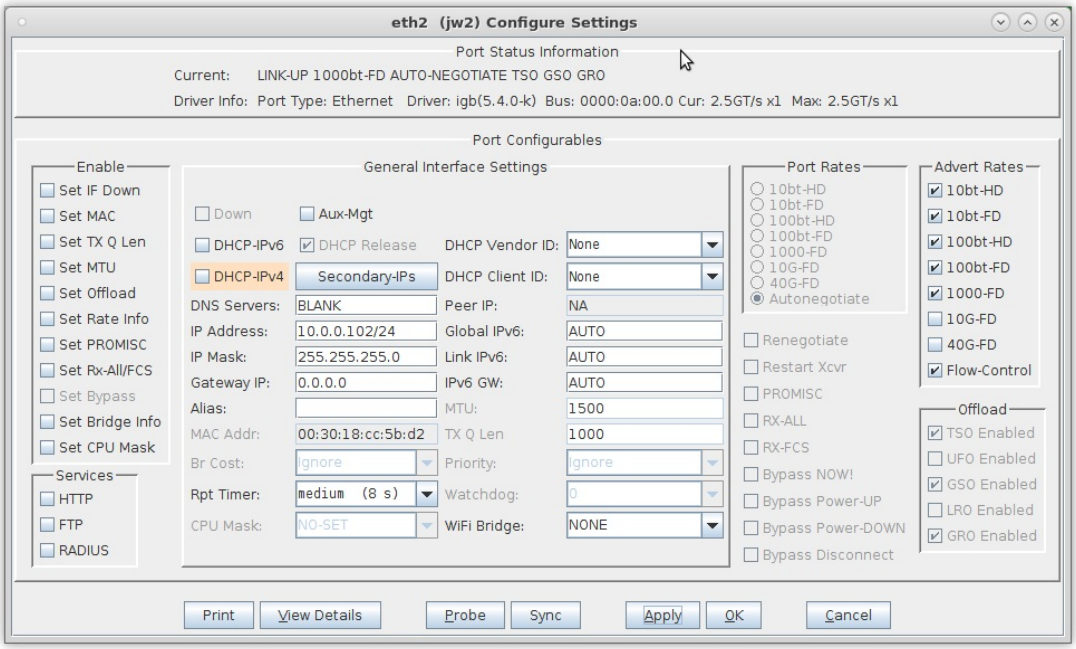
Note: In order to use the LANforge Armageddon feature, your system must have the LANforge kernel patch applied and your system must be properly licensed. Please feel free to contact us at

support@candelatech.com if you would like to obtain a demo license for the Armageddon feature.

1. Configure the physical interfaces.
   A. Go to the Port Manager and select ports eth2 and eth3

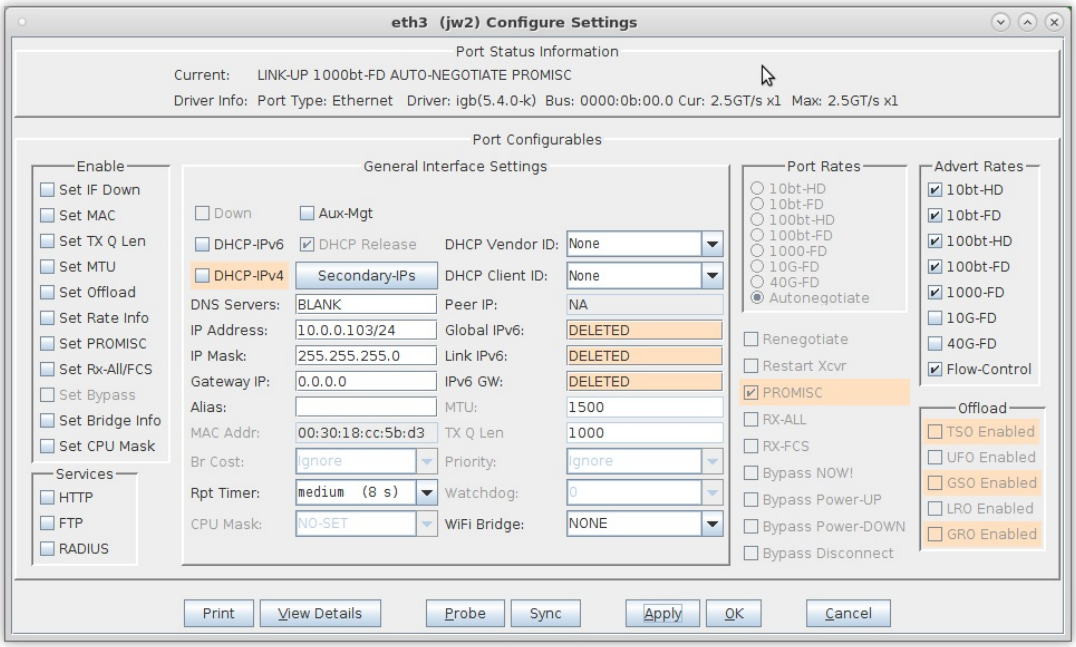


   B. Modify ports eth2 and eth3



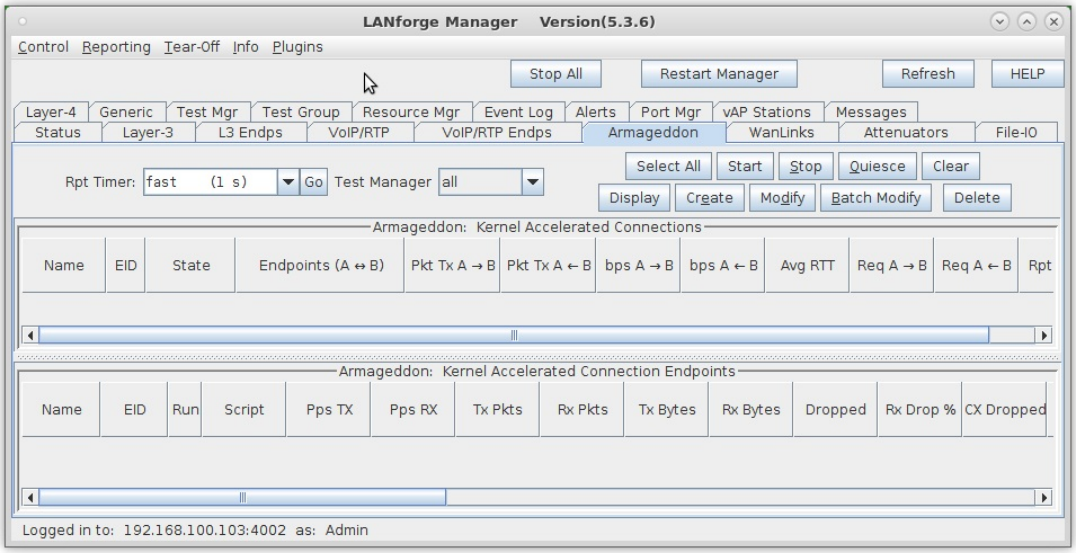
   A. In this example, eth2 and eth3 are connected to another LANforge system running a WanLink so that the Armageddon traffic can be sniffed on the other machine's interface
   B. **NOTE**: Be sure that both ports are in Promiscuous mode by selecting the **Set PROMISC** and **PROMISC** checkboxes

C. Configure each port with a valid IP address, then click **OK**

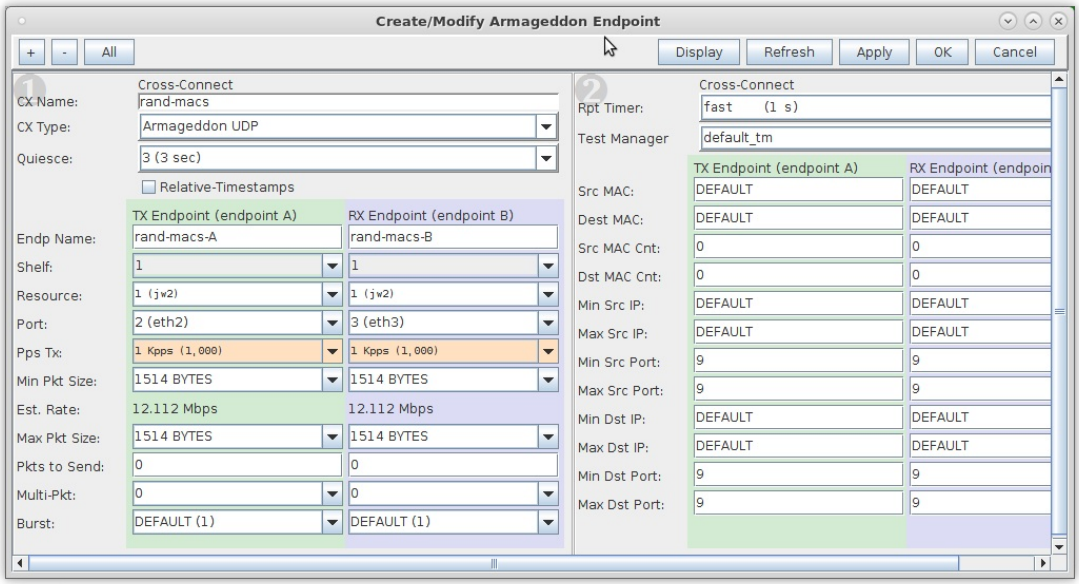


For more information see LANforge User's Guide: Ports (Interfaces)

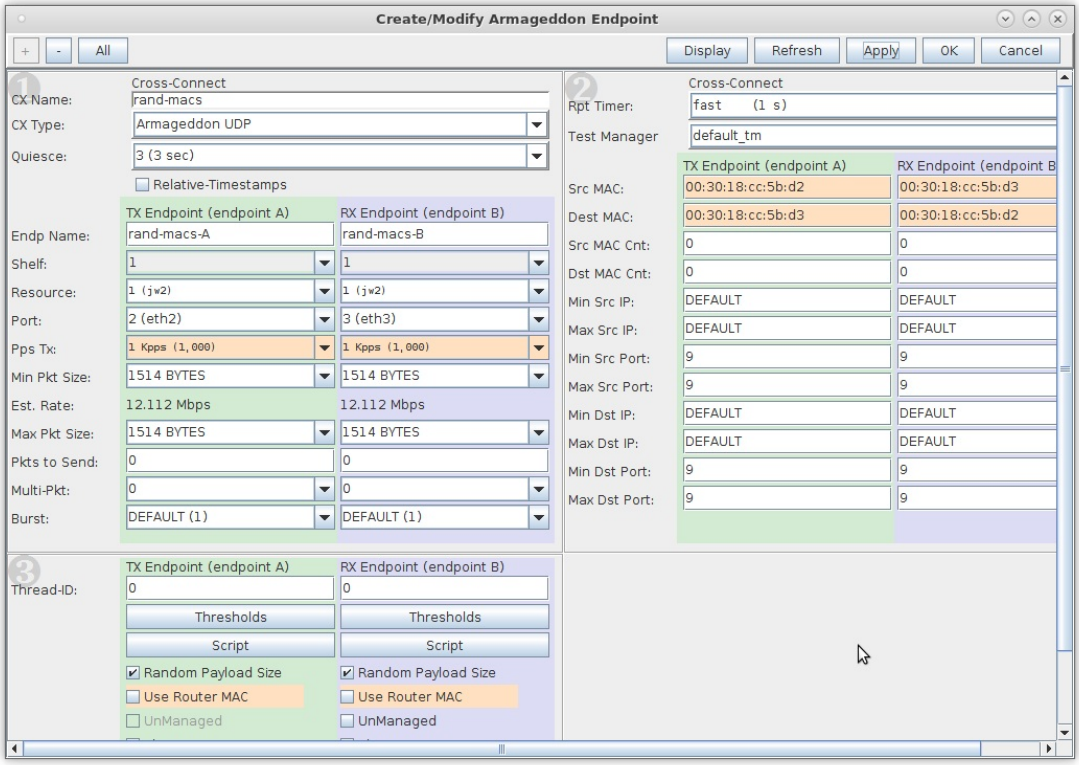
2. Create the Armageddon cross-connect.
   A. On the **Armageddon** tab, click **Create**



   B. Enter a CX Name, select ports eth2 and eth3, then enter the speed and packet size for both endpoints

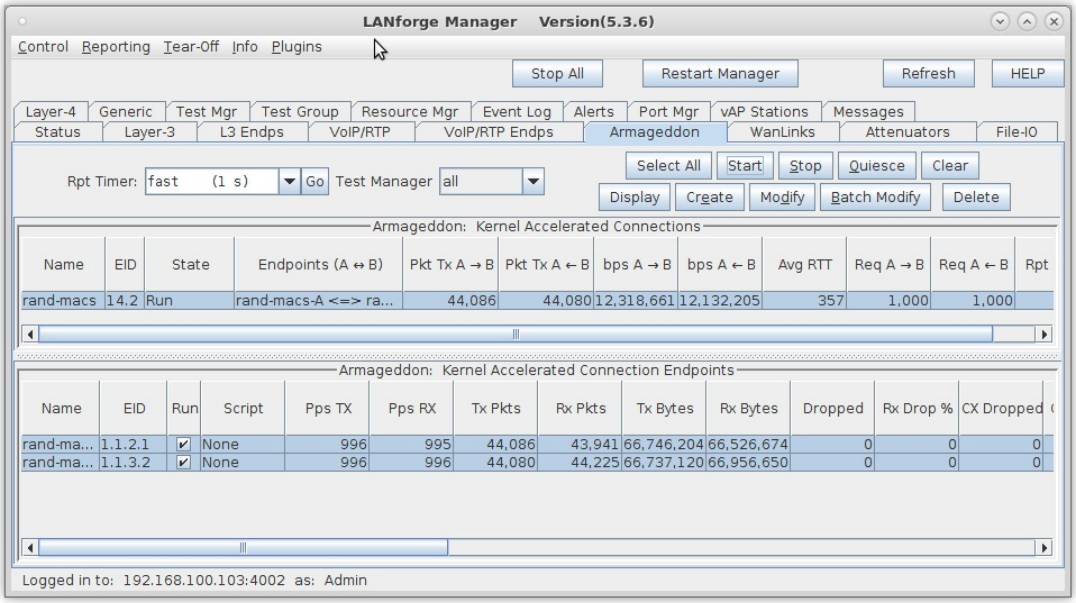


C. Enter values for the Source and Destination MAC addresses, specify a MAC count, and deselect Use Router MAC for both endpoints.

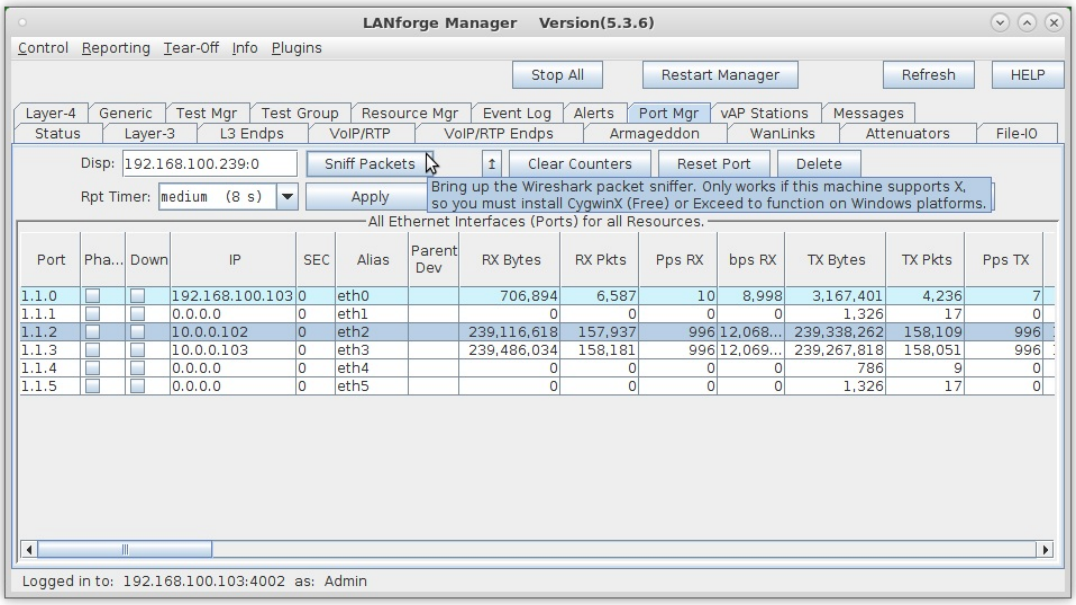


For more information see LANforge User's Guide: Armageddon (Accelerated UDP)

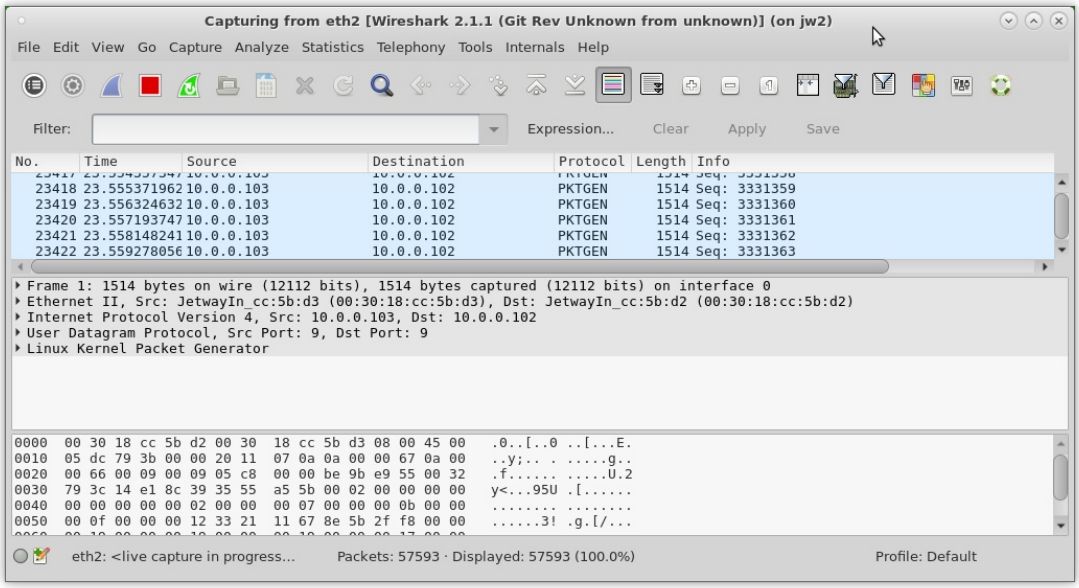
3. Run the Armageddon cross-connect and verify results with Wireshark.
   A. Select the Armageddon connection then click **Start**



B. On the **Port Mgr** tab of the other LANforge system, select one of the physical interfaces in the Armageddon connection

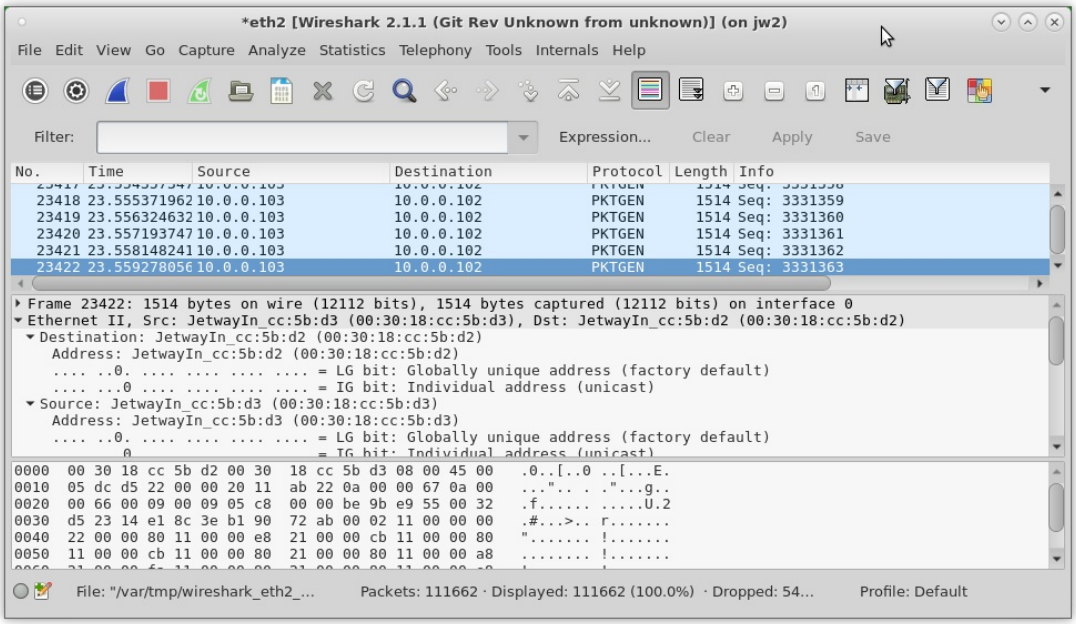


C. Click **Sniff Packets** to launch Wireshark and begin sniffing traffic.

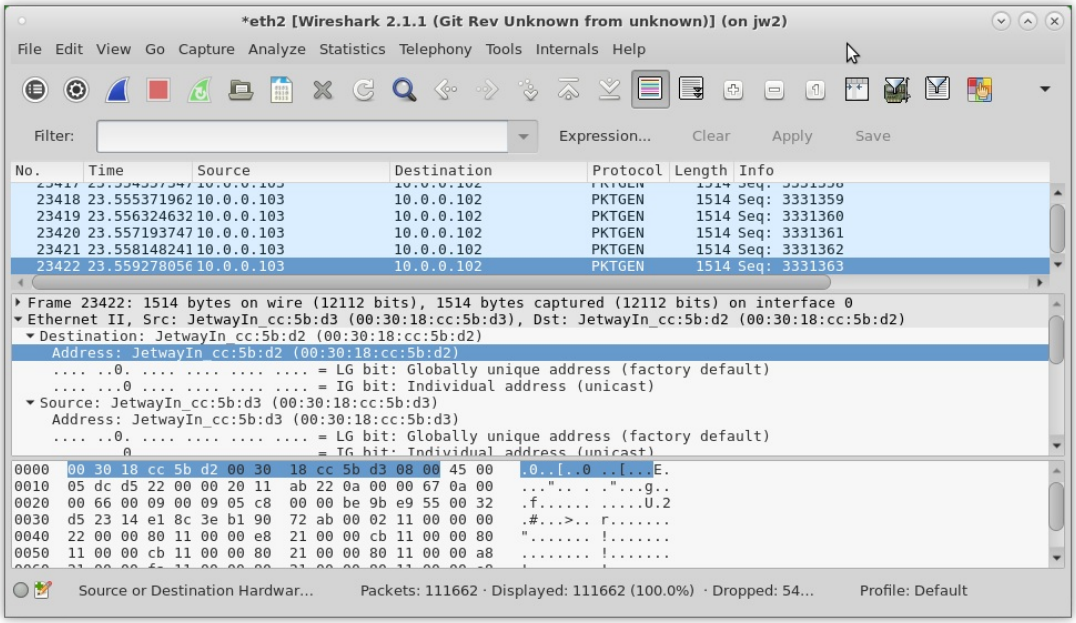


A. Stop the Wireshark capture after a few seconds via the stop icon or pull-down menu (**Capture**>**Stop**)

D. Select several packets and note their MAC addresses



E. Verify that the MAC addresses for each packet are different



For more information see LANforge User's Guide: Armageddon (Accelerated UDP)

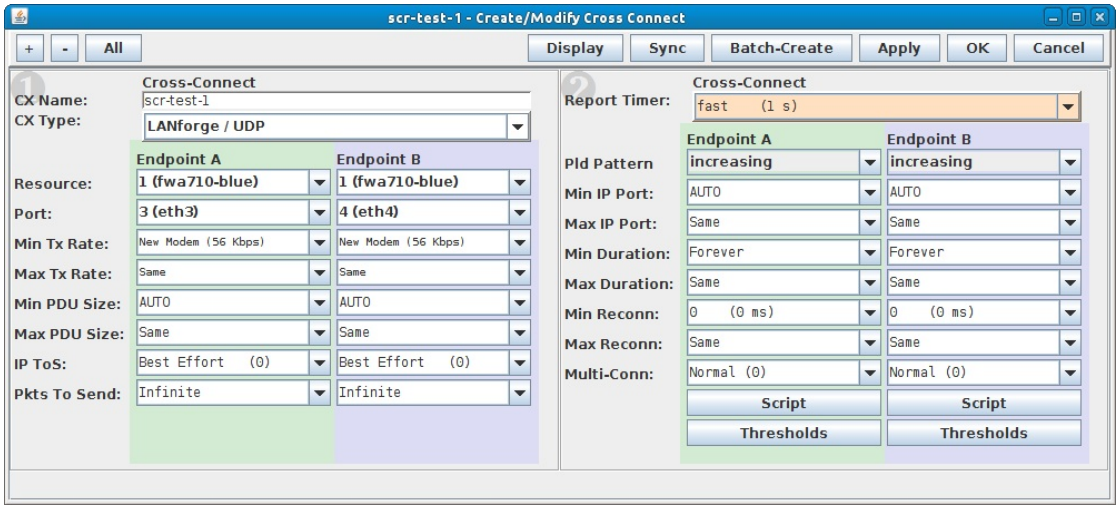
Scripted Layer-3 Test

Goal: Use RFC-2544 as a guide to create a Layer-3 connection that can run automatically through various payload sizes and rates for a specified duration.

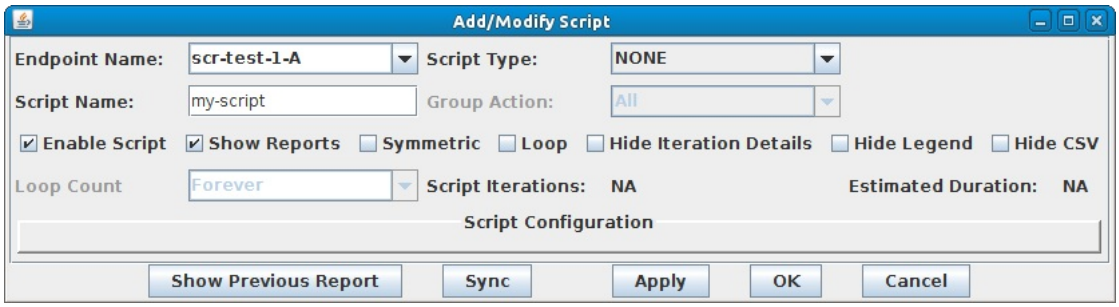
In this example, LANforge is used to set up a scripted connection that will iterate through a user-defined list of payload sizes and transmission rates. Each iteration will run for a user-defined duration with a user-defined pause between iterations. A summary text report is generated at the conclusion of all iterations.

1. Create a Layer-3 connection. For more information see Generating Traffic to a Switched Network

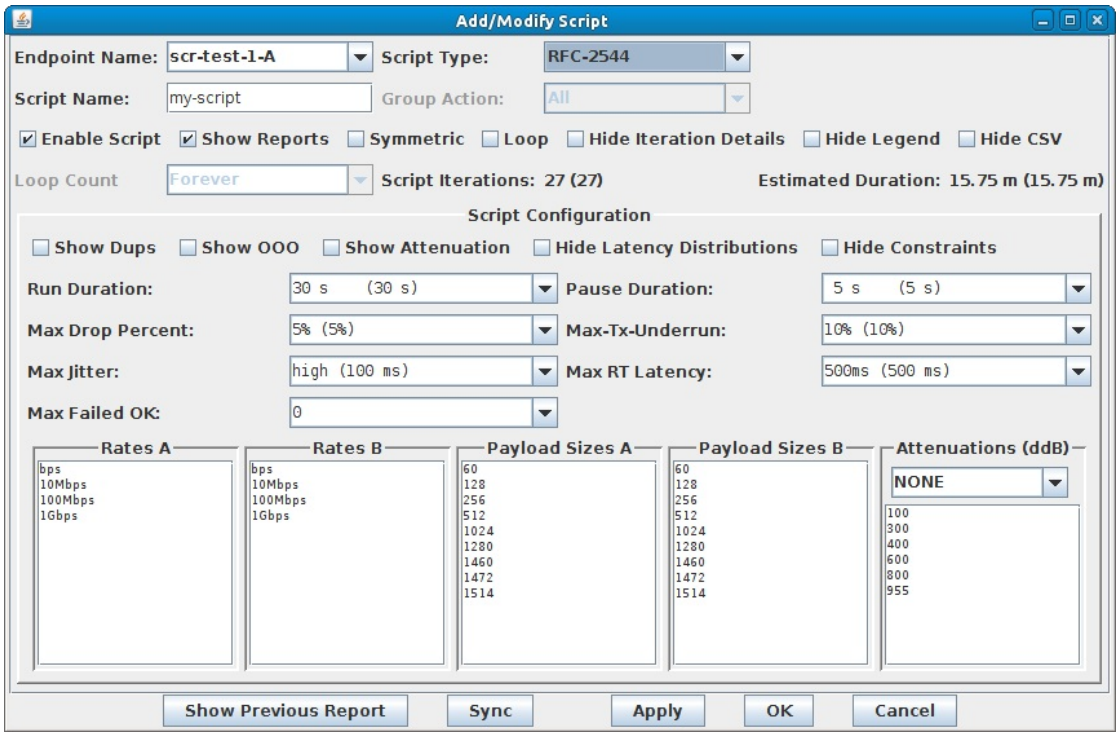
2. Modify the Layer-3 connection to add the script.
   A. Highlight the Layer-3 connection and select **Modify**.



   B. Select the **Script** button on Endpoint A.



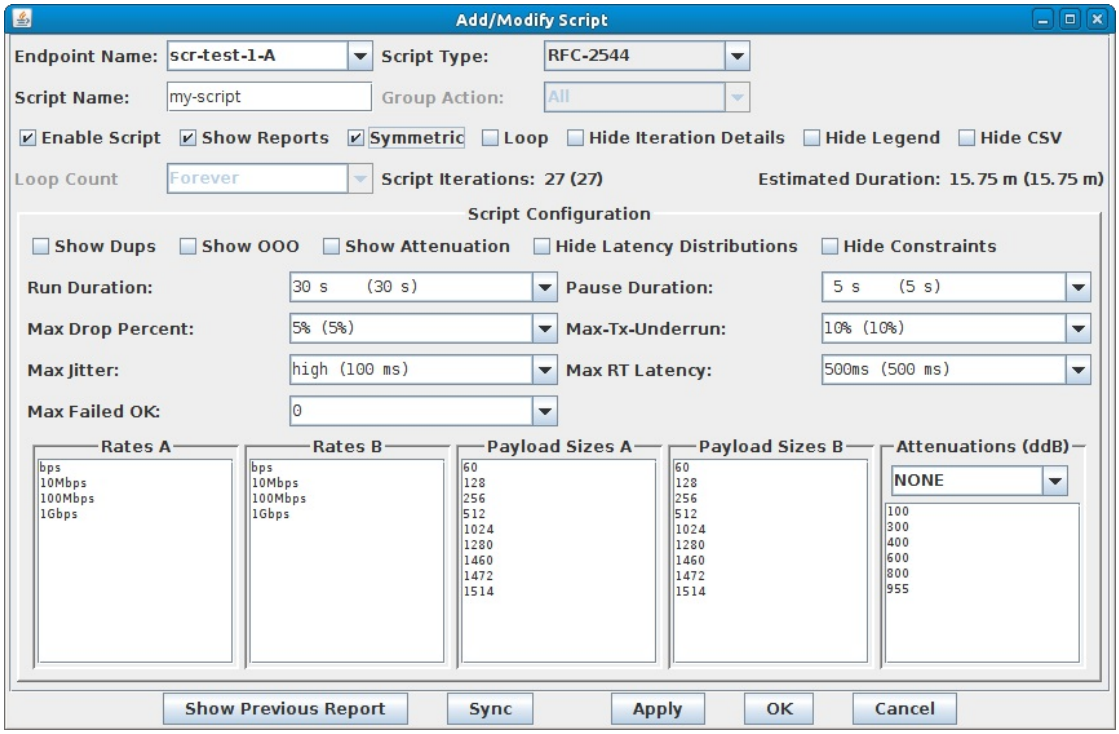
   C. Select the Script Type, RFC-2544.



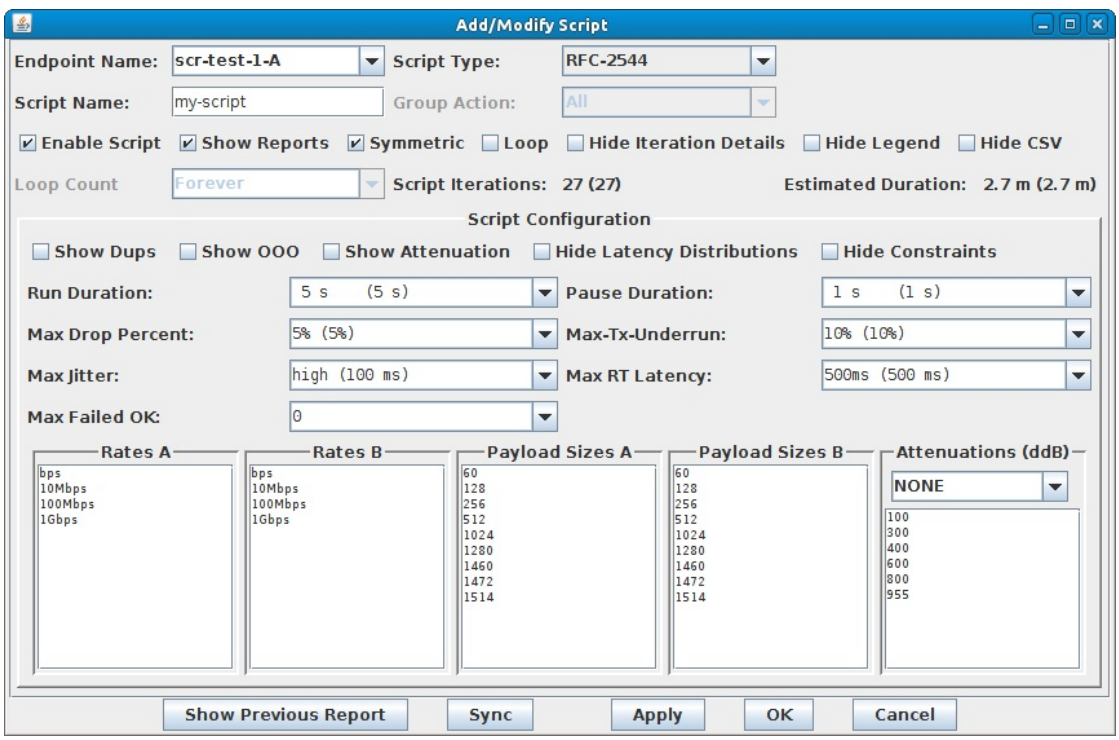
   A. **Note**: A default set of payload sizes are set up based on RFC-2544 but, can be changed by typing over the default values.

   B. **Note**: For Layer-3 UDP and TCP connections, 'payload size' refers to size of the payload being carried by the protocol and not the ethernet frame size.

3. Set up script options. For details refer to: LANforge User's Guide: Scripted Cross Connect
   A. Select **Symmetric** for the script to run both endpoints for a bi-directional traffic test.

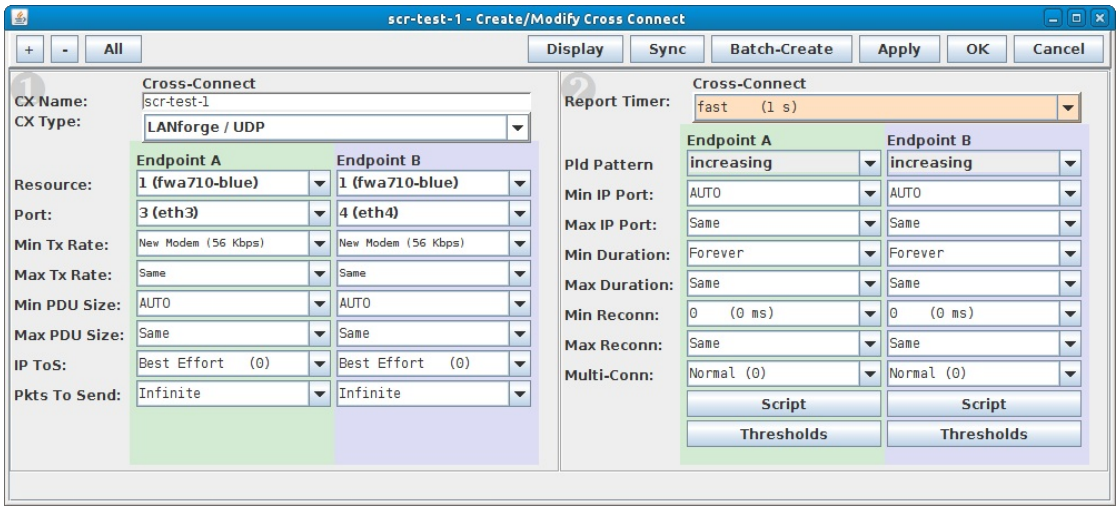


   B. Set the **Run** and **Pause Duration**.

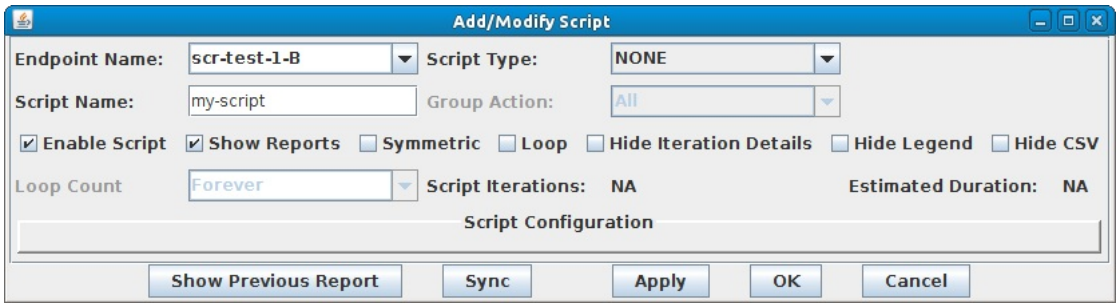


      A. Note the total number of **Script Iterations** and **Estimated Total Duration** to help determine how long it will take to run this script.

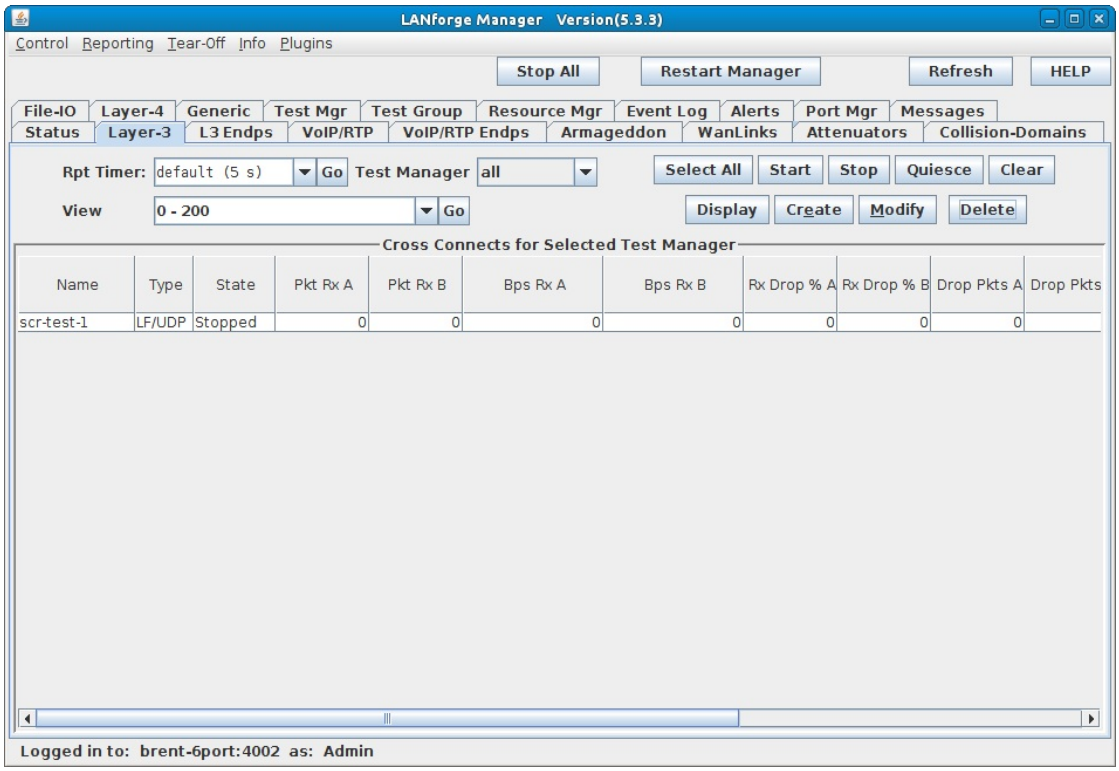
C. Select OK to close the Add/Modify Script window.



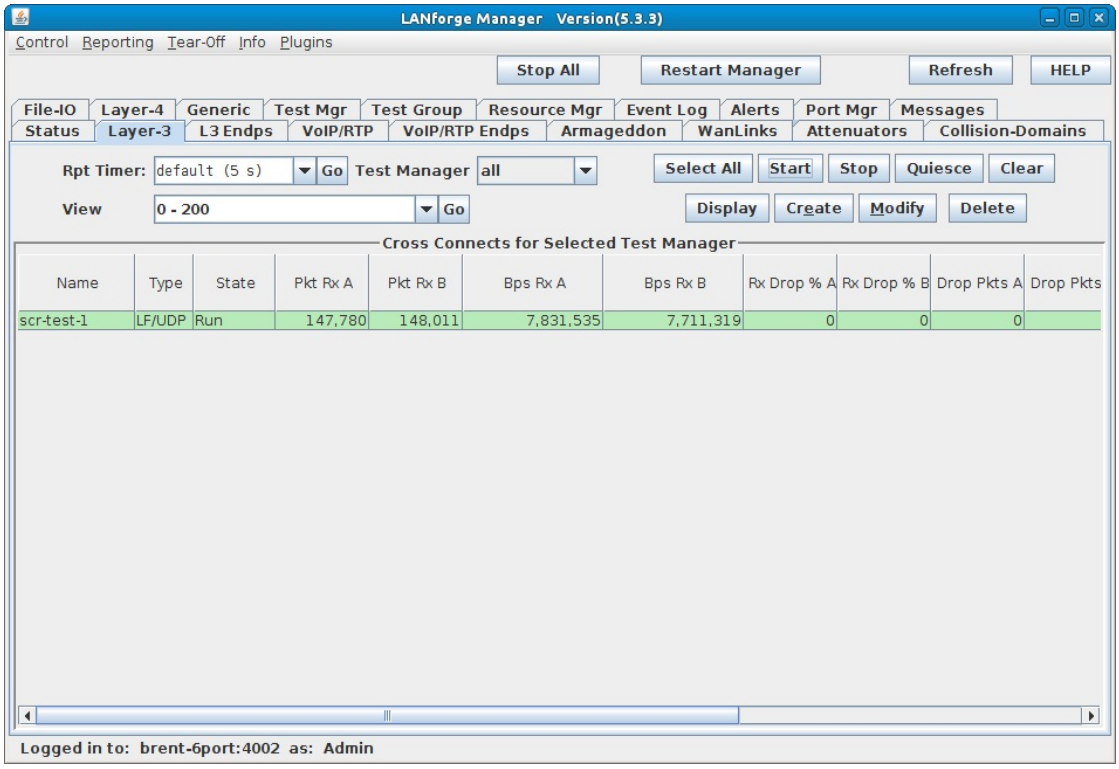
D. The Script Type for Endpoint B is set to NONE because Endpoint A is controlling both ends of the connection in this symmetric script example.



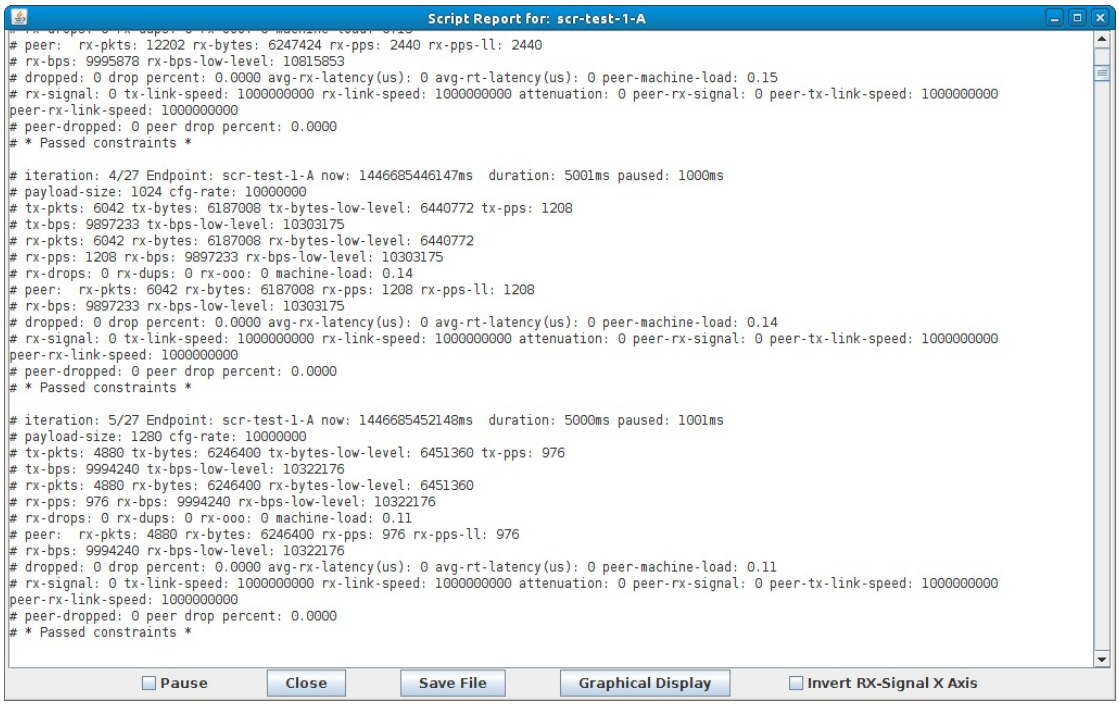
E. Select OK to close the Create/Modify Cross Connect window.



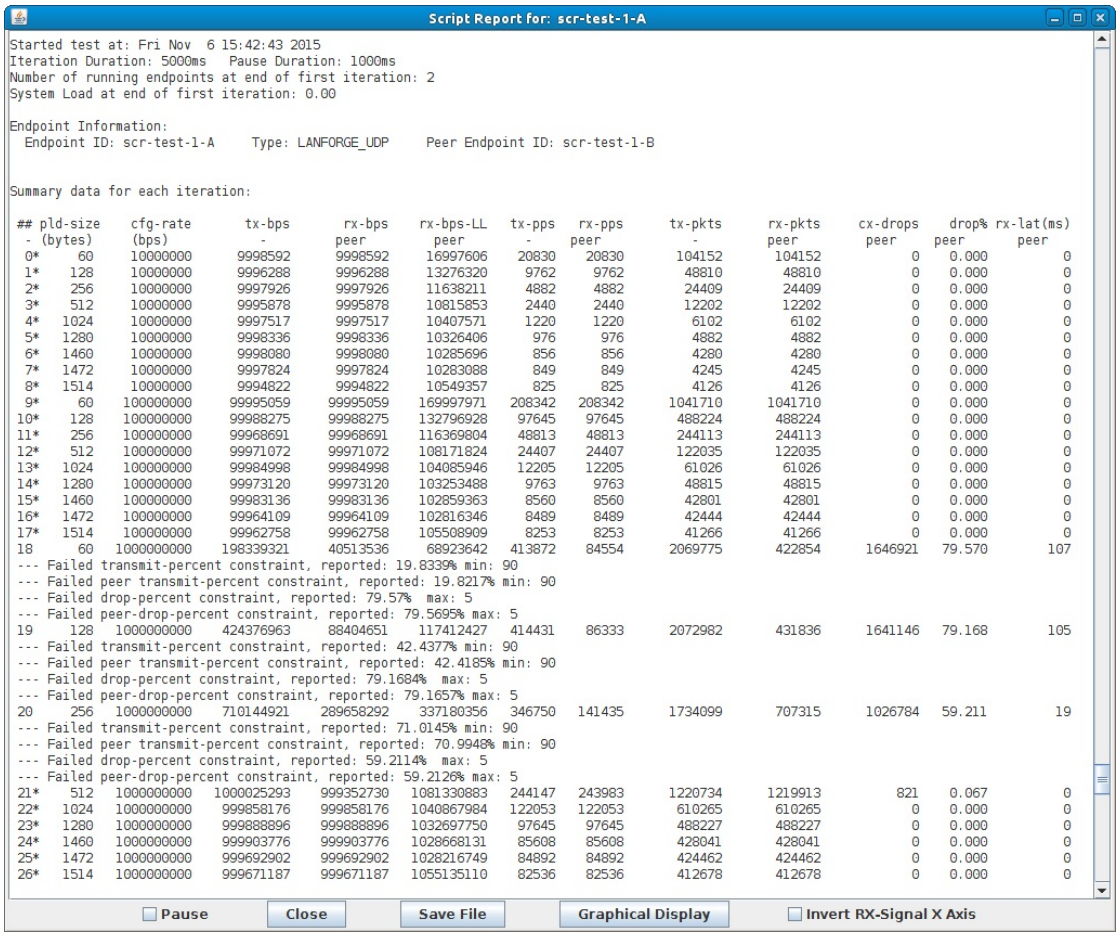
4. Start the Scripted Layer-3 Cross Connect.
   A. Highlight the Layer-3 connection and select Start.



   B. A script report window will pop up and show the details of each iteration of the scripted connection as it is run.



C. At the conclusion of the script, the report window will display a summary of the entire scripted connection results. Full Script Report for this example:



Script Report for: scr-test-1-A

```
Started test at: Fri Nov  6 15:42:43 2015
Iteration Duration: 5000ms   Pause Duration: 1000ms
Number of running endpoints at end of first iteration: 2
System Load at end of first iteration: 0.00

Endpoint Information:
  Endpoint ID: scr-test-1-A     Type: LANFORGE_UDP     Peer Endpoint ID: scr-test-1-B


Summary data for each iteration:

 ## pld-size    cfg-rate       tx-bps       rx-bps    rx-bps-LL   tx-pps   rx-pps      tx-pkts      rx-pkts     cx-drops    drop% rx-lat(ms)
  - (bytes)     (bps)            -          peer         peer        -     peer           -          peer         peer     peer       peer
  0*     60     10000000      9998592      9998592     16997606    20830    20830       104152       104152          0    0.000          0
  1*    128     10000000      9996288      9996288     13276320     9762     9762        48810        48810          0    0.000          0
  2*    256     10000000      9997926      9997926     11638211     4882     4882        24409        24409          0    0.000          0
  3*    512     10000000      9995878      9995878     10815853     2440     2440        12202        12202          0    0.000          0
  4*   1024     10000000      9997517      9997517     10407571     1220     1220         6102         6102          0    0.000          0
  5*   1280     10000000      9998336      9998336     10326406      976      976         4882         4882          0    0.000          0
  6*   1460     10000000      9998080      9998080     10285696      856      856         4280         4280          0    0.000          0
  7*   1472     10000000      9997824      9997824     10283088      849      849         4245         4245          0    0.000          0
  8*   1514     10000000      9994822      9994822     10549357      825      825         4126         4126          0    0.000          0
  9*     60    100000000     99995059     99995059    169997971   208342   208342      1041710      1041710          0    0.000          0
 10*    128    100000000     99988275     99988275    132796928    97645    97645       488224       488224          0    0.000          0
 11*    256    100000000     99968691     99968691    116369804    48813    48813       244113       244113          0    0.000          0
 12*    512    100000000     99971072     99971072    108171824    24407    24407       122035       122035          0    0.000          0
 13*   1024    100000000     99984998     99984998    104085946    12205    12205        61026        61026          0    0.000          0
 14*   1280    100000000     99973120     99973120    103253488     9763     9763        48815        48815          0    0.000          0
 15*   1460    100000000     99983136     99983136    102859363     8560     8560        42801        42801          0    0.000          0
 16*   1472    100000000     99964109     99964109    102816346     8489     8489        42444        42444          0    0.000          0
 17*   1514    100000000     99962758     99962758    105508909     8253     8253        41266        41266          0    0.000          0
 18      60   1000000000    198339321     40513536     68923642   413872    84554      2069775       422854    1646921   79.570        107
 --- Failed transmit-percent constraint, reported: 19.8339% min: 90
 --- Failed peer transmit-percent constraint, reported: 19.8217% min: 90
 --- Failed drop-percent constraint, reported: 79.57%  max: 5
 --- Failed peer-drop-percent constraint, reported: 79.5695% max: 5
 19     128   1000000000    424376963     88404651    117412427   414431    86333      2072982       431836    1641146   79.168        105
 --- Failed transmit-percent constraint, reported: 42.4377% min: 90
 --- Failed peer transmit-percent constraint, reported: 42.4185% min: 90
 --- Failed drop-percent constraint, reported: 79.1684%  max: 5
 --- Failed peer-drop-percent constraint, reported: 79.1657% max: 5
 20     256   1000000000    710144921    289658292    337180356   346750   141435      1734099       707315    1026784   59.211         19
 --- Failed transmit-percent constraint, reported: 71.0145% min: 90
 --- Failed peer transmit-percent constraint, reported: 70.9948% min: 90
 --- Failed drop-percent constraint, reported: 59.2114%  max: 5
 --- Failed peer-drop-percent constraint, reported: 59.2126% max: 5
 21*    512   1000000000   1000025293    999352730   1081330883   244147   243983      1220734      1219913       821    0.067          0
 22*   1024   1000000000    999858176    999858176   1040867984   122053   122053       610265       610265         0    0.000          0
 23*   1280   1000000000    999888896    999888896   1032697750    97645    97645       488227       488227         0    0.000          0
 24*   1460   1000000000    999903776    999903776   1028668131    85608    85608       428041       428041         0    0.000          0
 25*   1472   1000000000    999692902    999692902   1028216749    84892    84892       424462       424462         0    0.000          0
 26*   1514   1000000000    999671187    999671187   1055135110    82536    82536       412678       412678         0    0.000          0
```

Pause | Close | Save File | Graphical Display | Invert RX-Signal X Axis

A. per iteration details

B. raw CSV data for all iterations

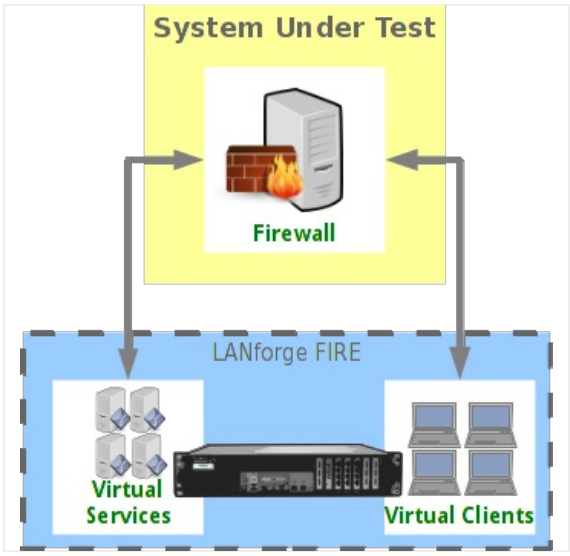
C. spreadsheet matrices for creating your own 3D graphs

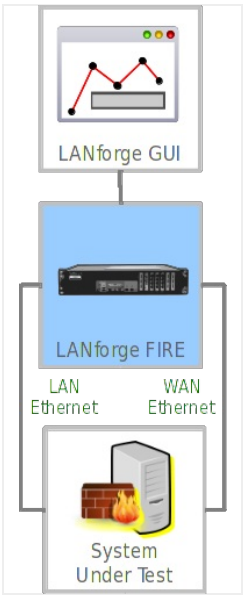
D. system information

Scripted Armageddon Test

Goal: Use RFC-2544 as a guide to create an Armageddon connection that can run automatically through various payload sizes and rates for a specified duration.

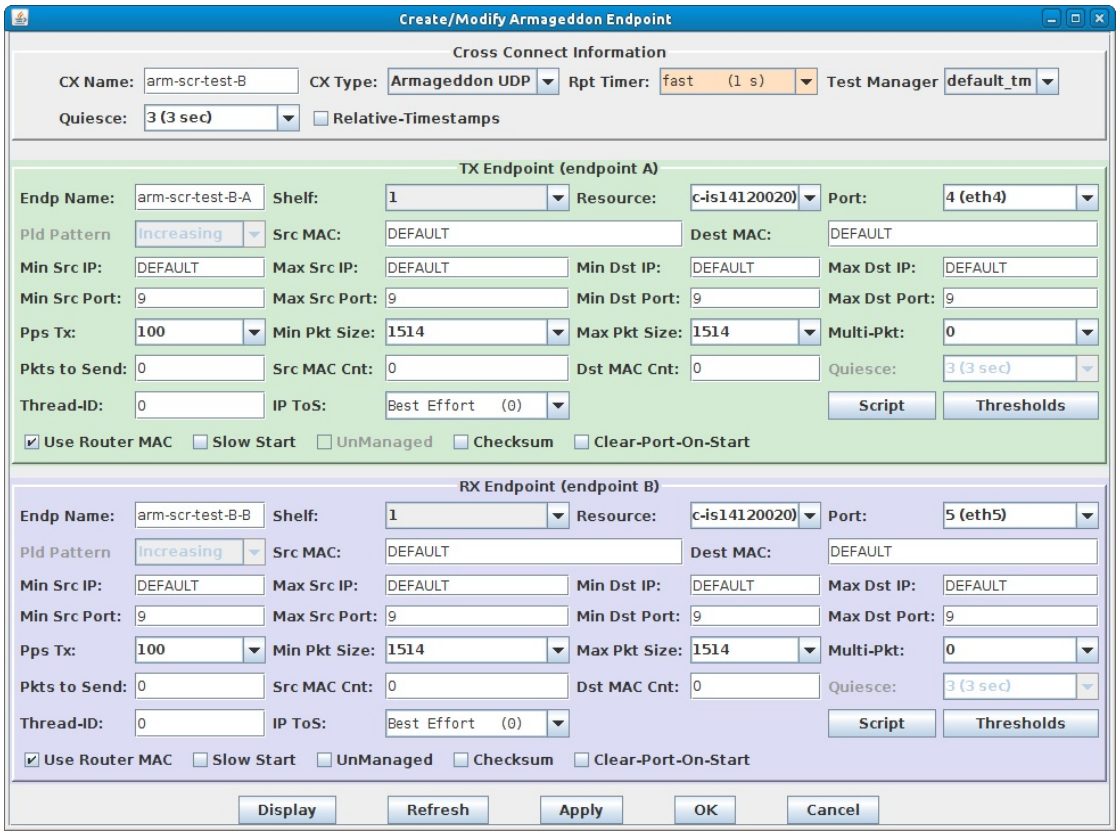
In this example, LANforge is used to set up a scripted connection that will iterate through a user-defined list of payload sizes and transmission rates. Each iteration will run for a user-defined duration with a user-defined pause between iterations. A summary text report is generated at the conclusion of all iterations.



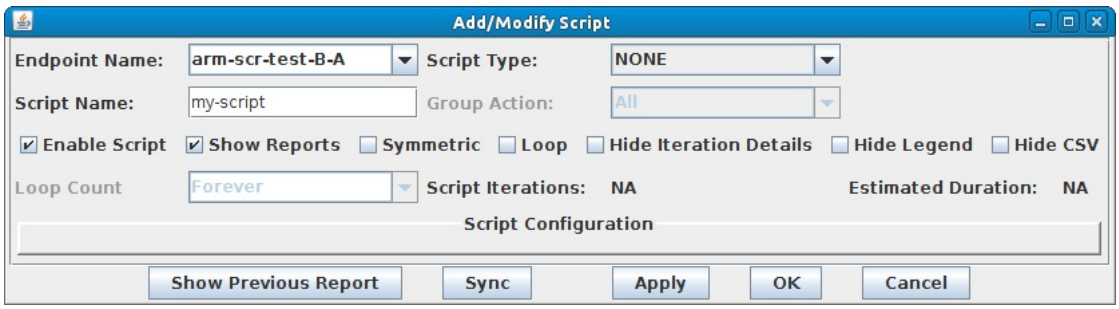


1. Create an Armageddon connection. For more information see Armageddon Testing (Accelerated UDP)

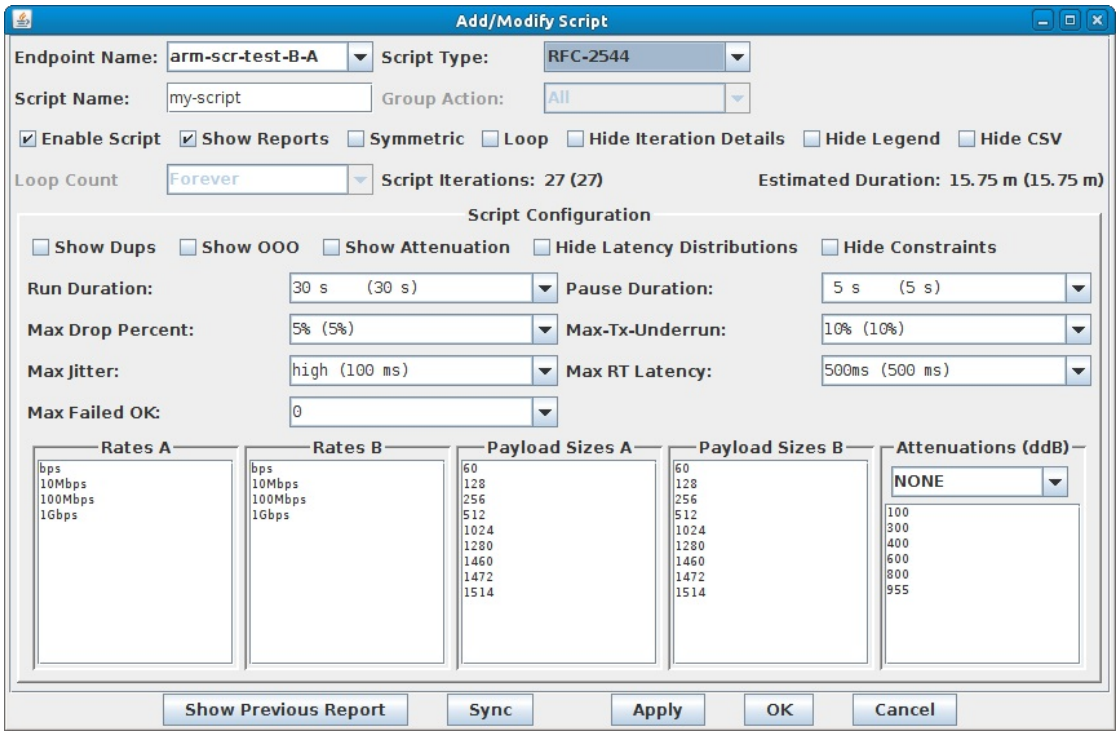
2. Modify the Armageddon connection to add the script.
   A. Highlight the Armageddon connection and select **Modify**.



   B. Select the **Script** button on Endpoint A.



C. Select the **Script Type** 'RFC-2544'.

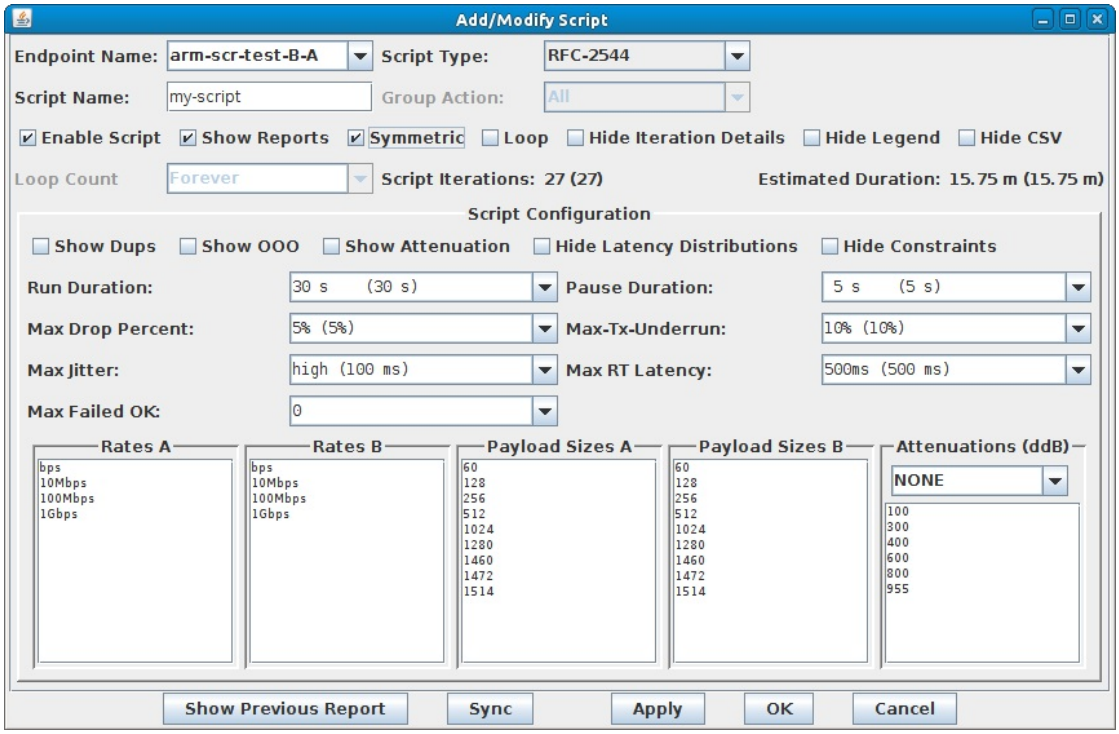


A. **Note**: A default set of payload sizes are set up based on RFC-2544 but, can be changed by typing over the default values.

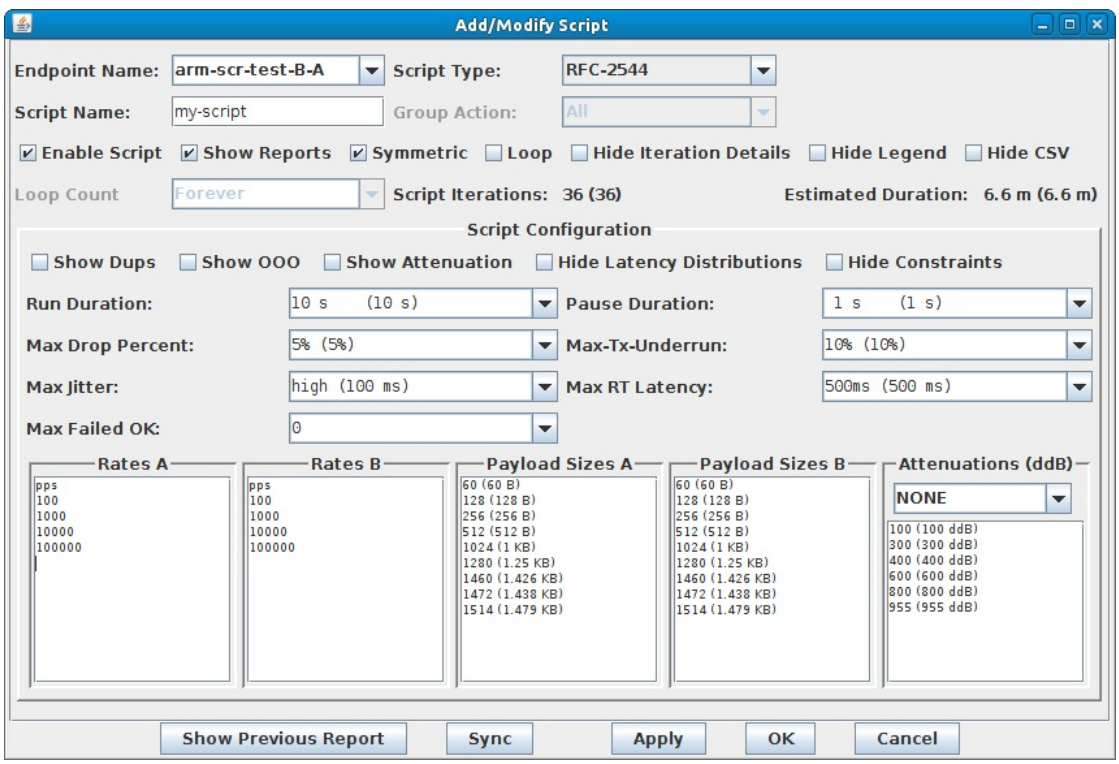
B. **Note**: For Armageddon UDP connections, 'payload size' refers to the ethernet frame size.

For more information see LANforge User's Guide: Scripted Armageddon Cross Connect

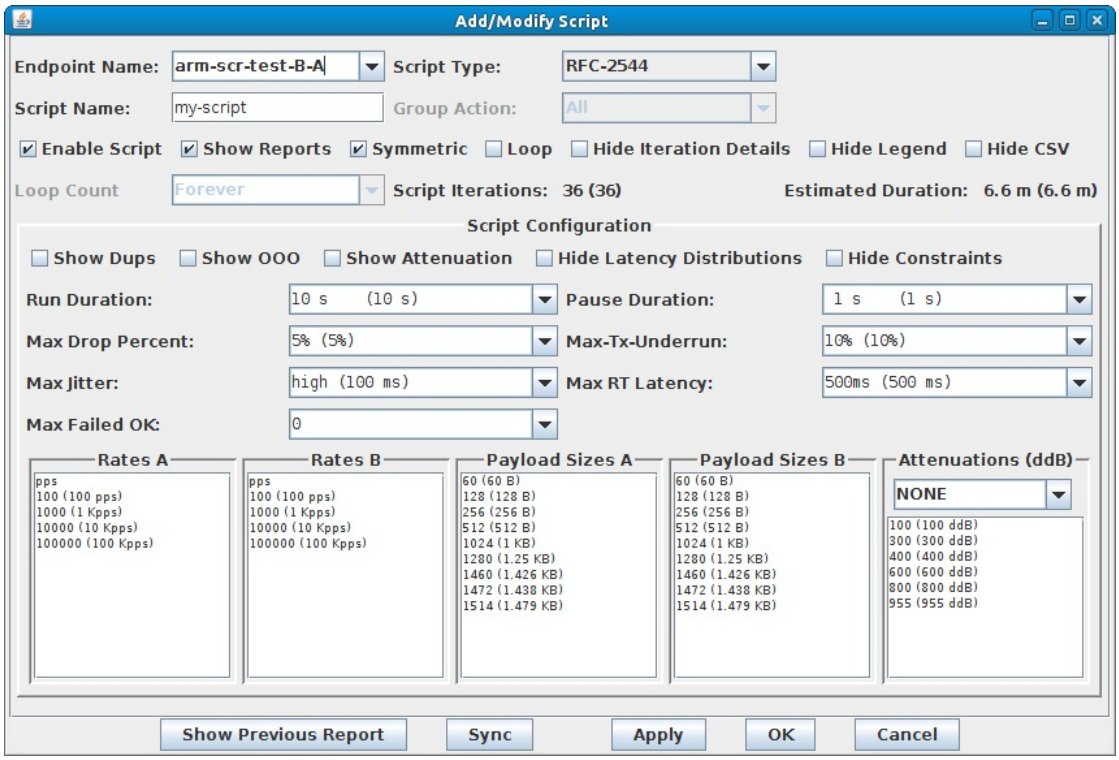
3. Set up script options.
   A. Select Symmetric for the script to run both endpoints for a bi-directional traffic test.



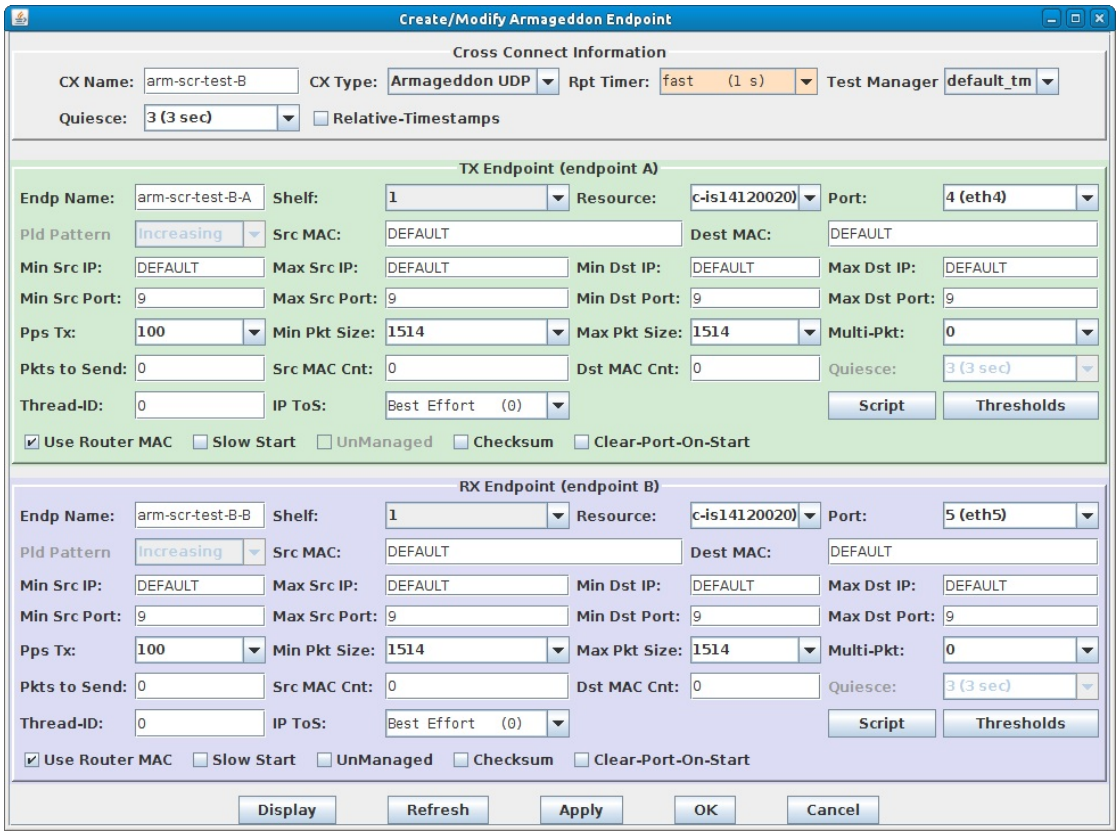
   B. Set the Run and Pause Duration, max failure thresholds, and modify Rates and Payload Sizes as needed.



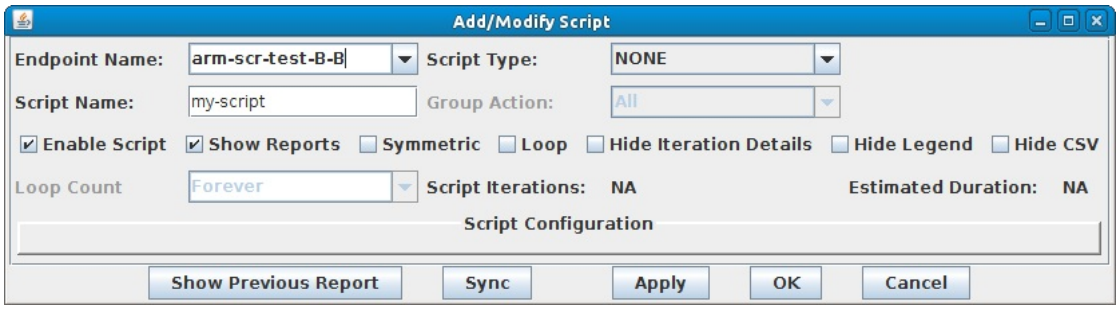
C. Note the total number of Script Iterations and Estimated Total Duration to help determine how long it will take to run this script.



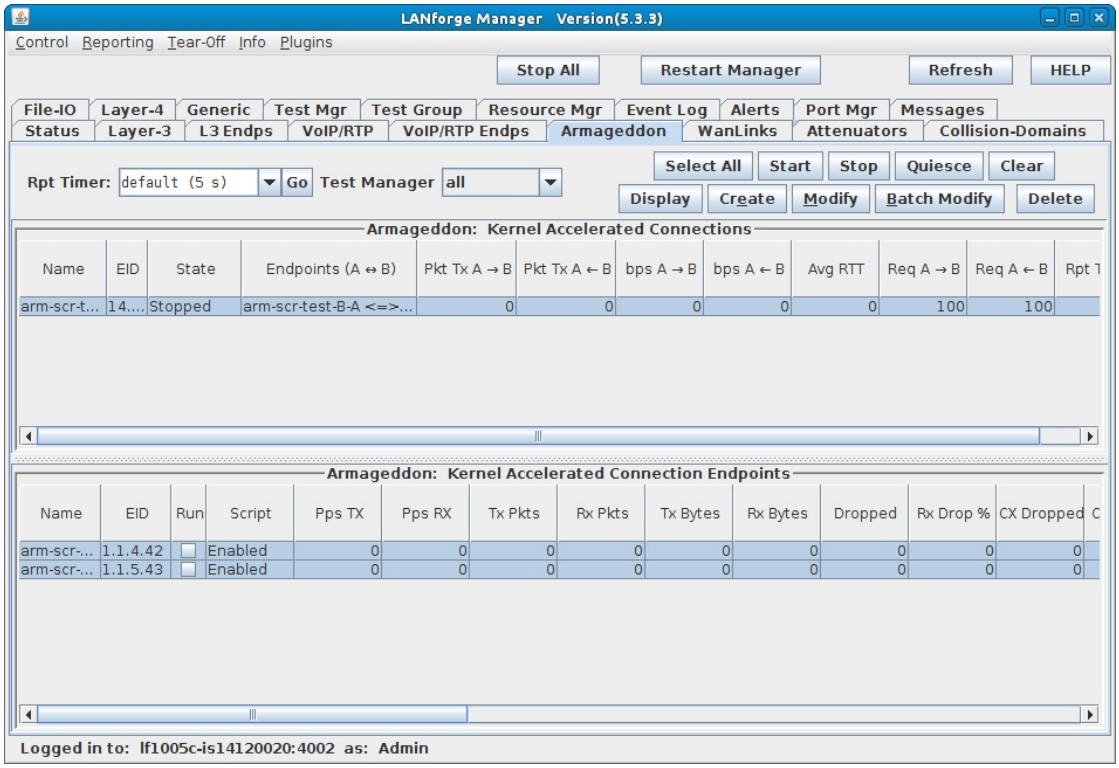
D. Select **OK** to close the **Add/Modify Script** window.



E. The Script Type for Endpoint B is set to NONE because Endpoint A is controlling both ends of the connection in this symmetric script example.

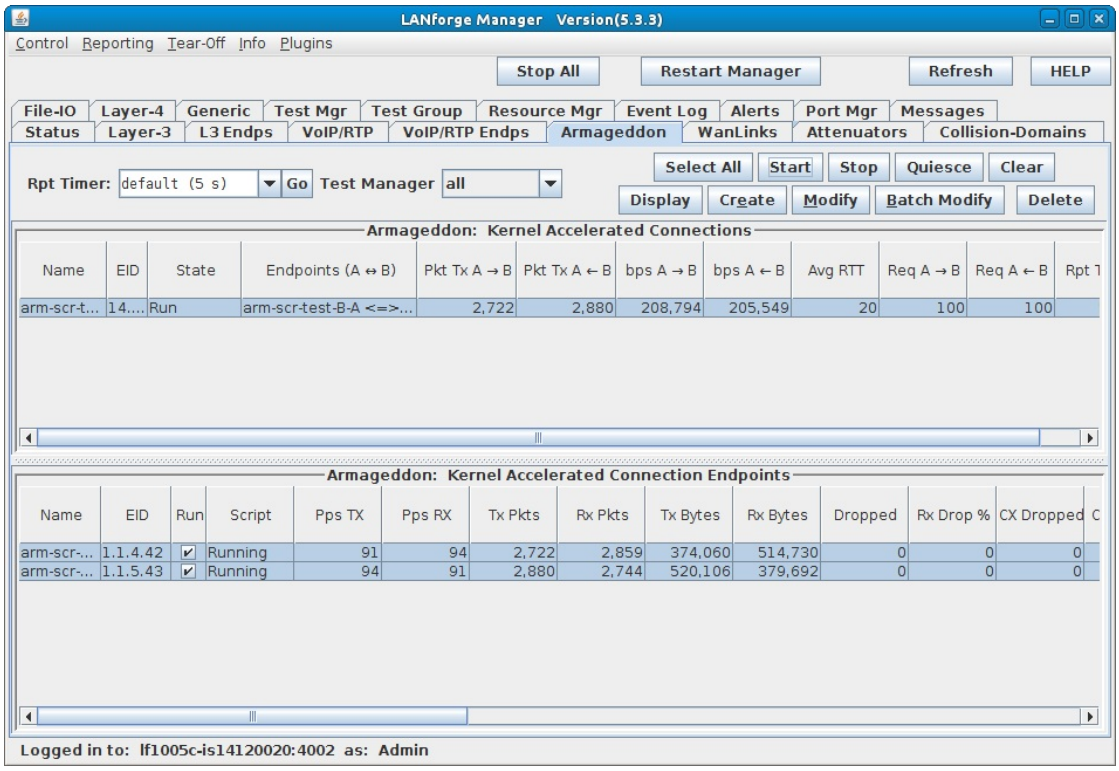


F. Select **OK** to close the **Create/Modify Cross Connect** window.

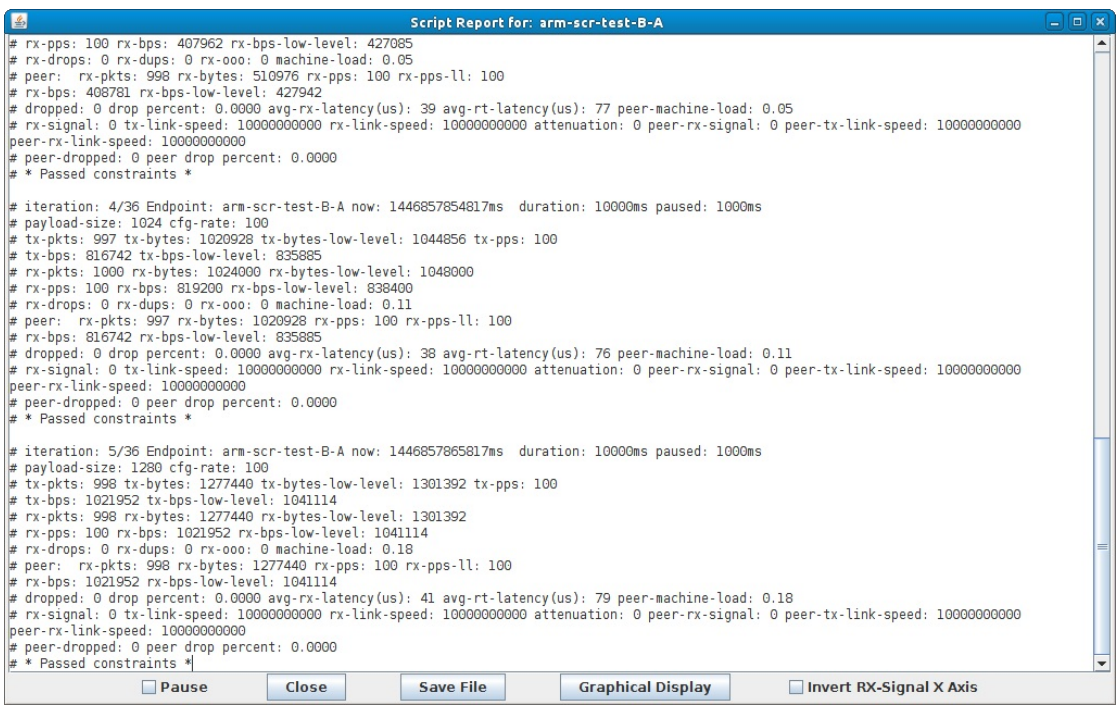


For more information see LANforge User's Guide: Scripted Armageddon Cross Connect

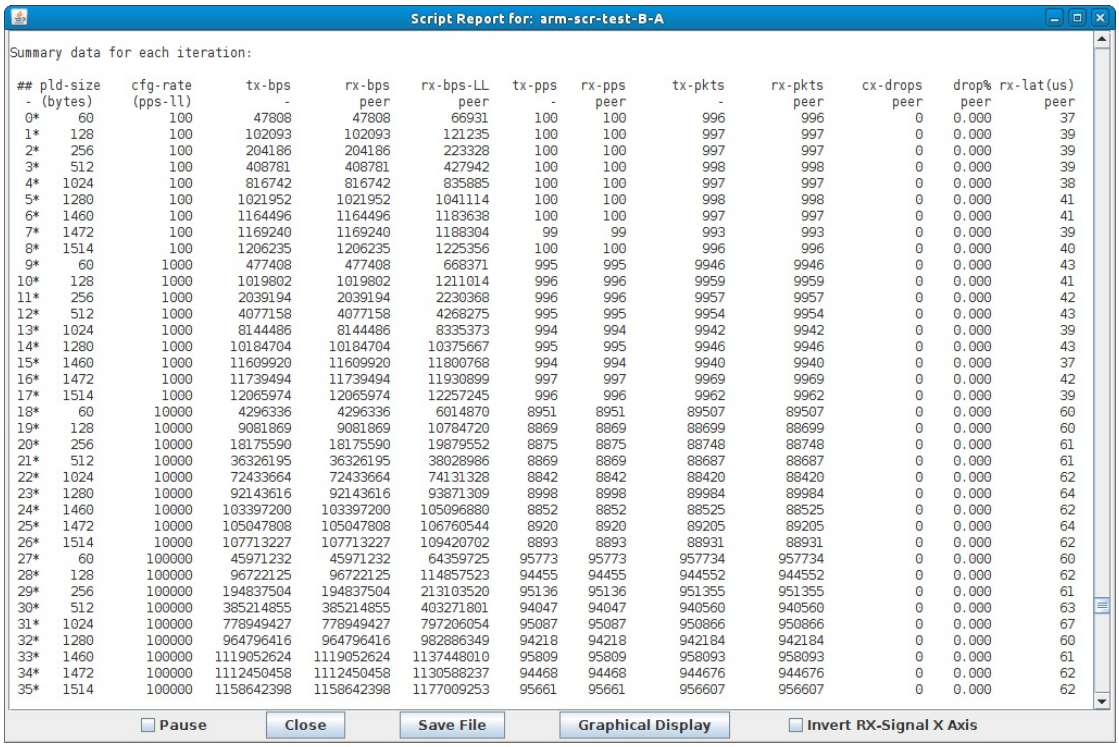
4. Start the Scripted Armageddon Cross Connect.
   A. Highlight the Armageddon connection and select **Start**.



   B. A script report window will pop up and show the details of each iteration of the scripted connection as it is run.



C. At the conclusion of the script, the report window will display a summary of the entire scripted connection results.



Summary data for each iteration:

##	pld-size - (bytes)	cfg-rate (pps-ll)	tx-bps -	rx-bps peer	rx-bps-LL peer	tx-pps -	rx-pps peer	tx-pkts -	rx-pkts peer	cx-drops peer	drop% peer	rx-lat(us) peer
0*	60	100	47808	47808	66931	100	100	996	996	0	0.000	37
1*	128	100	102093	102093	121235	100	100	997	997	0	0.000	39
2*	256	100	204186	204186	223328	100	100	997	997	0	0.000	39
3*	512	100	408781	408781	427942	100	100	998	998	0	0.000	39
4*	1024	100	816742	816742	835885	100	100	997	997	0	0.000	38
5*	1280	100	1021952	1021952	1041114	100	100	998	998	0	0.000	41
6*	1460	100	1164496	1164496	1183638	100	100	997	997	0	0.000	41
7*	1472	100	1169240	1169240	1188304	99	99	993	993	0	0.000	39
8*	1514	100	1206235	1206235	1225356	100	100	996	996	0	0.000	40
9*	60	1000	477408	477408	668371	995	995	9946	9946	0	0.000	43
10*	128	1000	1019802	1019802	1211014	996	996	9959	9959	0	0.000	41
11*	256	1000	2039194	2039194	2230368	996	996	9957	9957	0	0.000	42
12*	512	1000	4077158	4077158	4268275	995	995	9954	9954	0	0.000	43
13*	1024	1000	8144486	8144486	8335373	994	994	9942	9942	0	0.000	39
14*	1280	1000	10184704	10184704	10375667	995	995	9946	9946	0	0.000	43
15*	1460	1000	11609920	11609920	11800768	994	994	9940	9940	0	0.000	37
16*	1472	1000	11739494	11739494	11930899	997	997	9969	9969	0	0.000	42
17*	1514	1000	12065974	12065974	12257245	996	996	9962	9962	0	0.000	39
18*	60	10000	4296336	4296336	6014870	8951	8951	89507	89507	0	0.000	60
19*	128	10000	9081869	9081869	10784720	8869	8869	88699	88699	0	0.000	60
20*	256	10000	18175590	18175590	19879552	8875	8875	88748	88748	0	0.000	61
21*	512	10000	36326195	36326195	38028986	8869	8869	88687	88687	0	0.000	61
22*	1024	10000	72433664	72433664	74131328	8842	8842	88420	88420	0	0.000	62
23*	1280	10000	92143616	92143616	93871309	8998	8998	89984	89984	0	0.000	64
24*	1460	10000	103397200	103397200	105096880	8852	8852	88525	88525	0	0.000	62
25*	1472	10000	105047808	105047808	106760544	8920	8920	89205	89205	0	0.000	64
26*	1514	10000	107713227	107713227	109420702	8893	8893	88931	88931	0	0.000	62
27*	60	100000	45971232	45971232	64359725	95773	95773	957734	957734	0	0.000	60
28*	128	100000	96722125	96722125	114857523	94455	94455	944552	944552	0	0.000	62
29*	256	100000	194837504	194837504	213103520	95136	95136	951355	951355	0	0.000	61
30*	512	100000	385214855	385214855	403271801	94047	94047	940560	940560	0	0.000	63
31*	1024	100000	778949427	778949427	797206054	95087	95087	950866	950866	0	0.000	67
32*	1280	100000	964796416	964796416	982886349	94218	94218	942184	942184	0	0.000	60
33*	1460	100000	1119052624	1119052624	1137448010	95809	95809	958093	958093	0	0.000	61
34*	1472	100000	1112450458	1112450458	1130588237	94468	94468	944676	944676	0	0.000	62
35*	1514	100000	1158642398	1158642398	1177009253	95661	95661	956607	956607	0	0.000	62

A. per iteration details

B. raw CSV data for all iterations

C. spreadsheet matrices for creating your own 3D graphs

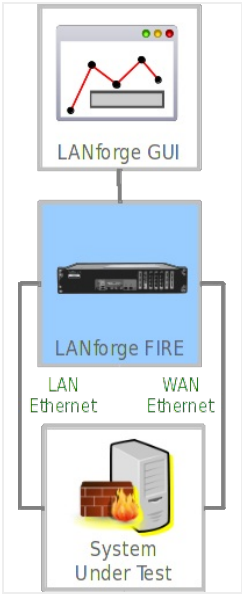
D. system information

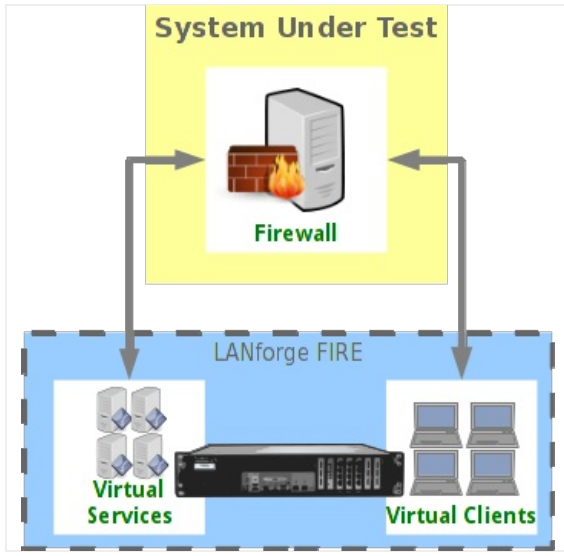
For more information see Full Script Report for this example.

5000 concurrent TCP streams and IP addresses

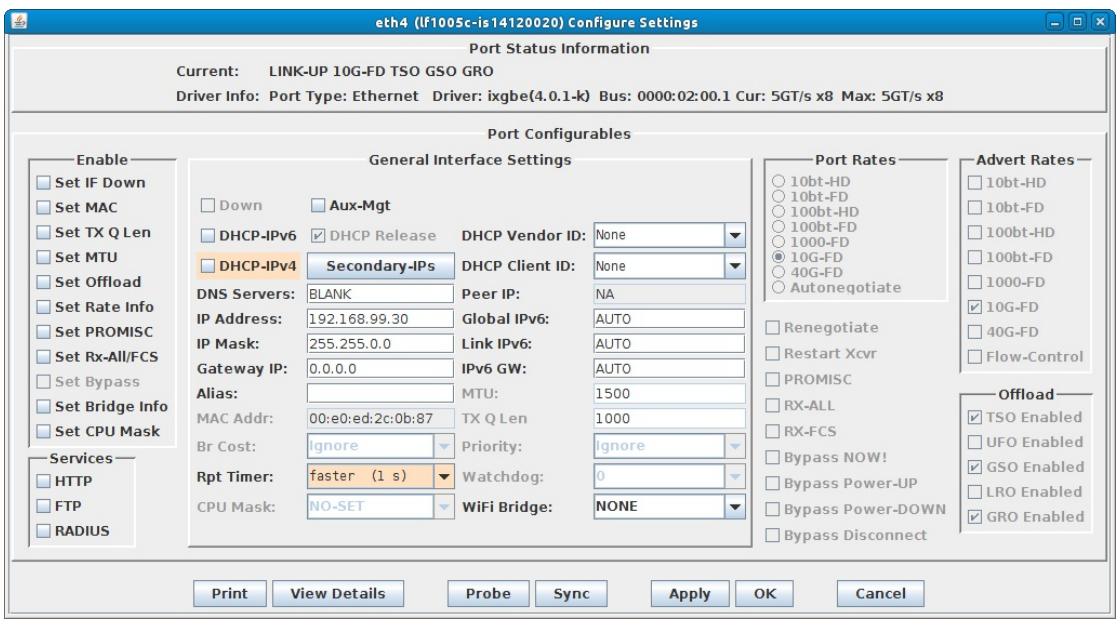
Goal: Quickly set up 5000 TCP connections from 5000 different IP addresses.

Requires LANforge 5.2.10 or later. Use the new feature in 5.2.10 that lets 'multi-conn' Layer-3 TCP connections use a range of secondary IPs to quickly create 5000 stateful TCP/IP connections on different IP addresses. This example uses two 10G ports on a system similar to the LANforge CT503-MIX systems but the procedure should work on all properly licensed higher-end systems. A standard LANforge license only supports 1000 concurrent connections, contact your sales representative for additional licenses. This cookbook assumes basic familiarity with LANforge.

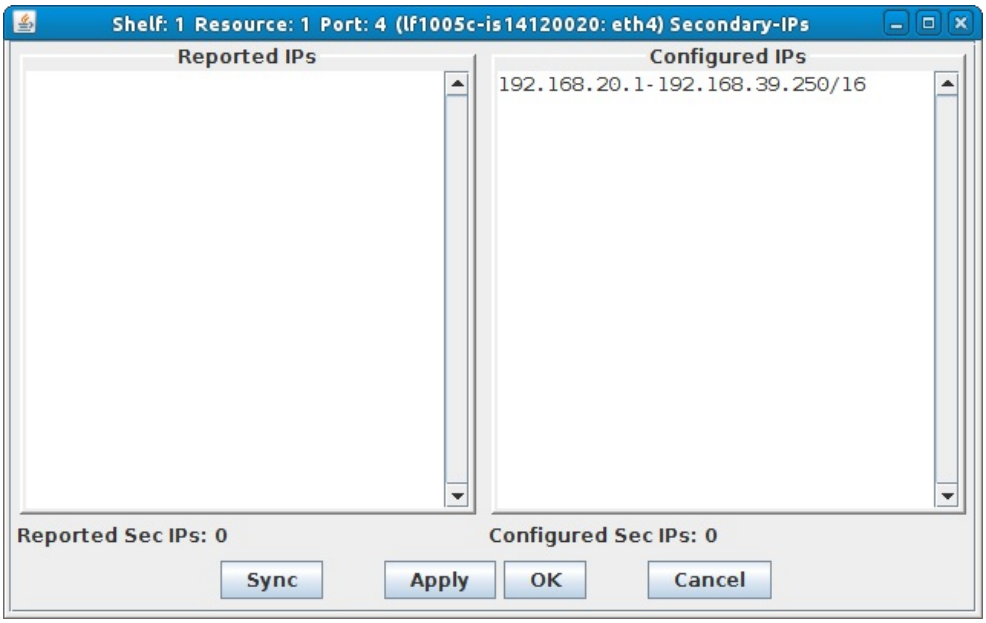




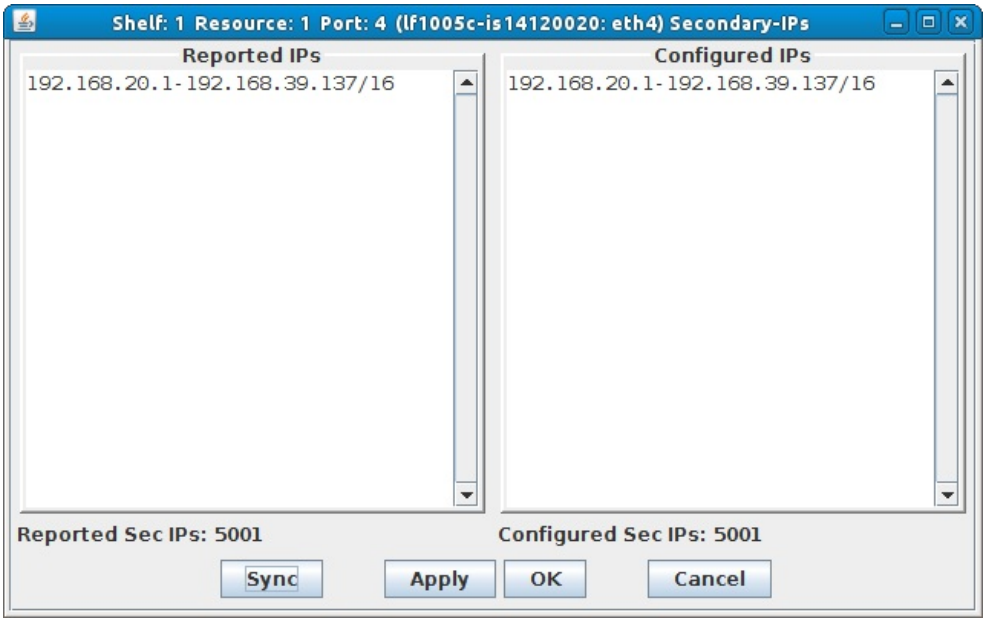
1. Create 5000 secondary IP addresses on an ethernet port. The IP address range should be on the same subnet as the primary IP address on this interface.
   A. Go to the Port Manager tab, select the client-side ethernet port and click Modify. Ensure subnet mask is 255.255.0.0 or similar so we have plenty of room to add the 5000 secondary IPs on the subnet. Apply if changes were made.



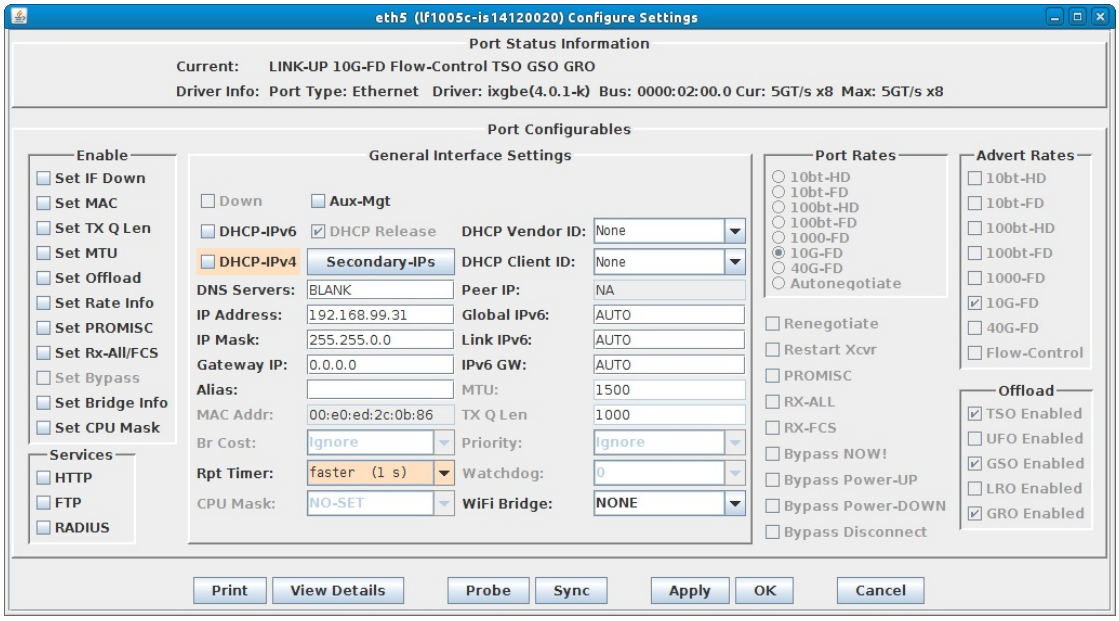
B. Click the **Secondary-IPS** button to bring up the Secondary IP address management window. Enter the IP address range similar to the image below and click **Apply**:



C. Click 'Sync' after a few minutes to make sure all of the new IP addresses were properly created.

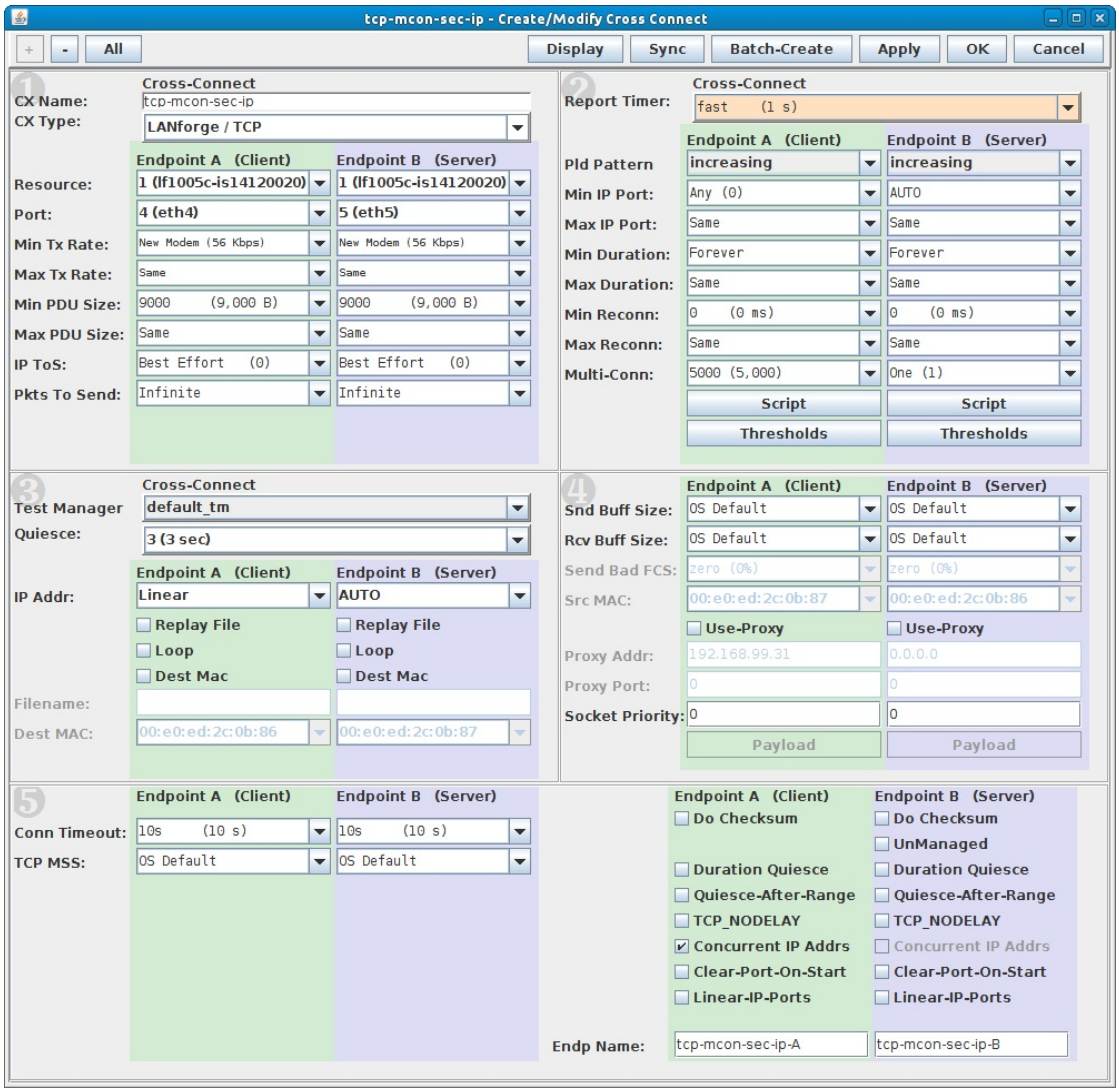


D. Make sure a second ethernet port on the LANforge is properly configured for the server-side of the network. In this case, we are using a network-emulator in bridge mode as device-under-test, so all IP addresses are on the same subnet.

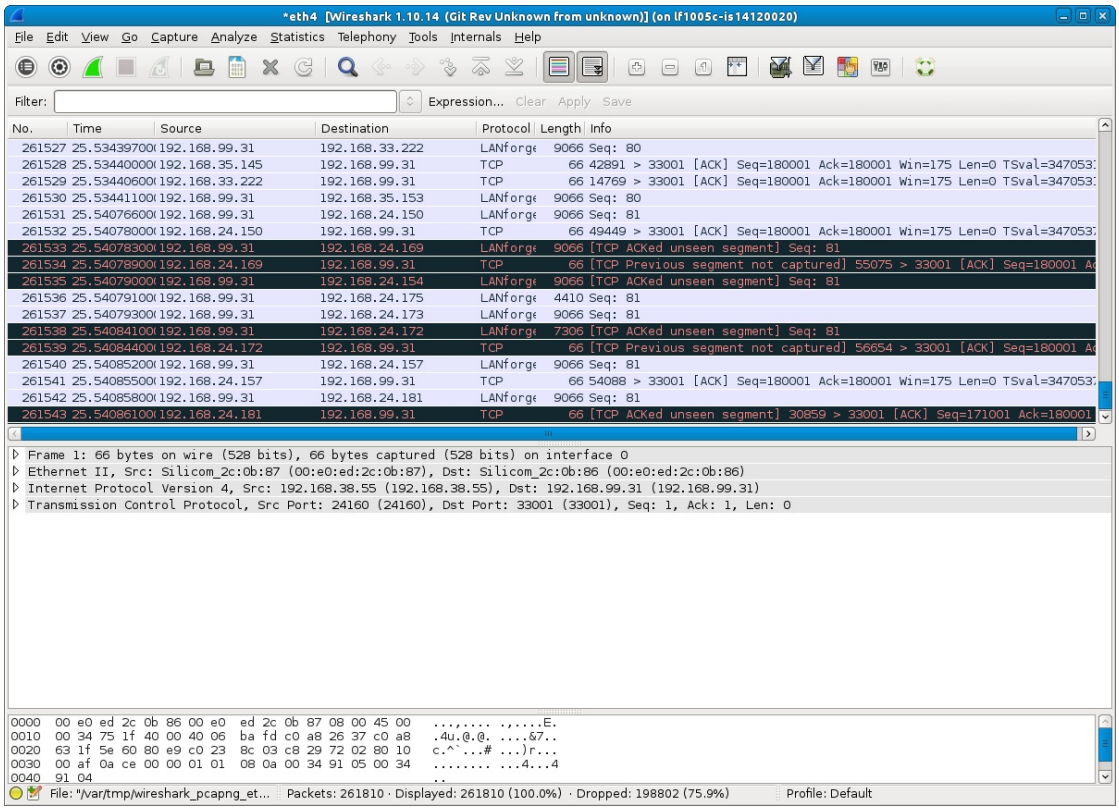


For more information see LANforge User's Guide: Ports (Interfaces)

2. Create a Layer-3 TCP connection to utilize these 5000 secondary IPs and start the traffic.
   A. Go to the **Layer-3** tab and click Create. Configure the **CX Name**, **CX Type**, **Ports**, and **PDU Size** in section 1. The port in the green section should be the one with the secondary IP Addresses. In section 2, configure **Min IP Port** in the green section, and **Multi-Conn** in both. In section 3, select 'Linear' for the **IP Addr** selection box in the green section. In section 5, select the **Concurrent IP Addrs** checkbox in the green section.

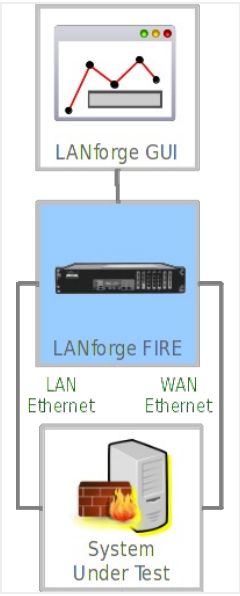


B. Start the connection, and then go to the **Port Mgr** tab, select one of the ethernet ports, and sniff the traffic (or use third-party tools) to verify the IP address range is as expected:



Multiple Thousands of TCP Connections

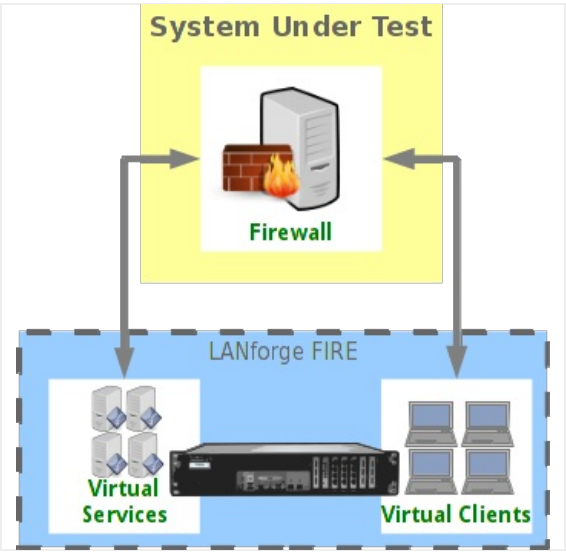
Goal: Set up and run traffic on multiple thousands of stateful TCP connections.



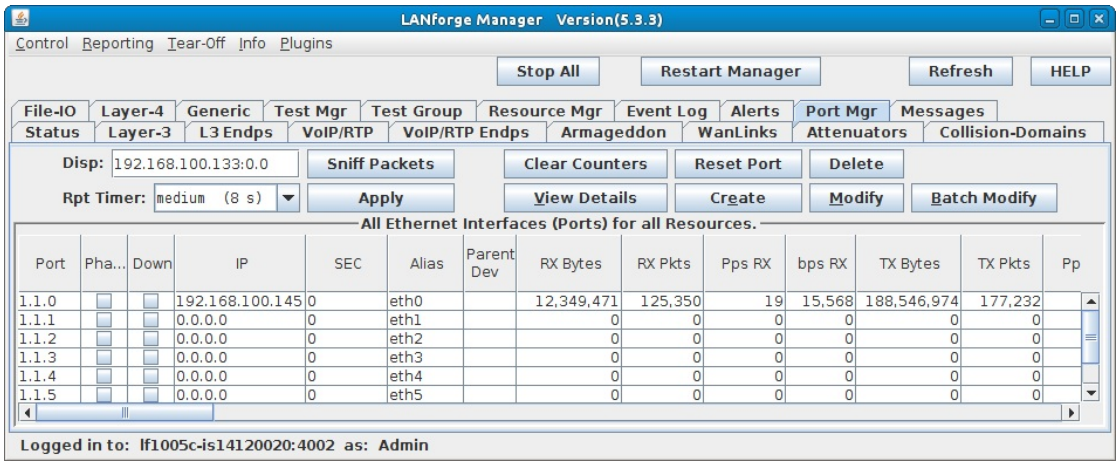
As of LANforge release version 5.1.4, LANforge-FIRE has the capability to create multiple thousands of TCP connections using only two ports on one Layer-3 connection that can all run simultaneously and in a stateful manner. This feature can be used to test connection state aware devices such as firewalls, stateful packet inspection systems and intrusion detection/prevention systems.

The upper limit of the total number of concurrent connections is determined by the processor speed and total memory available on the system running the LANforge software. In addition, the packet rate and payload size of each connection will also affect performance.

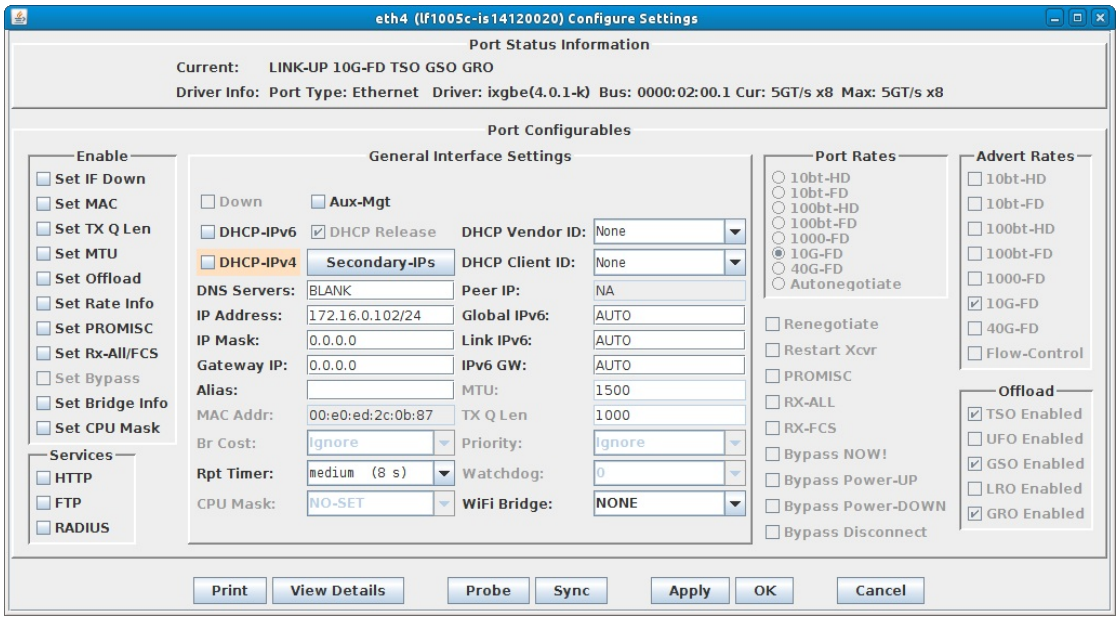
NOTE: If you are attempting to run this test scenario, you will need a LANforge license key that enables the correct number of multi-connections. Please contact us at support@candelatech.com for assistance.



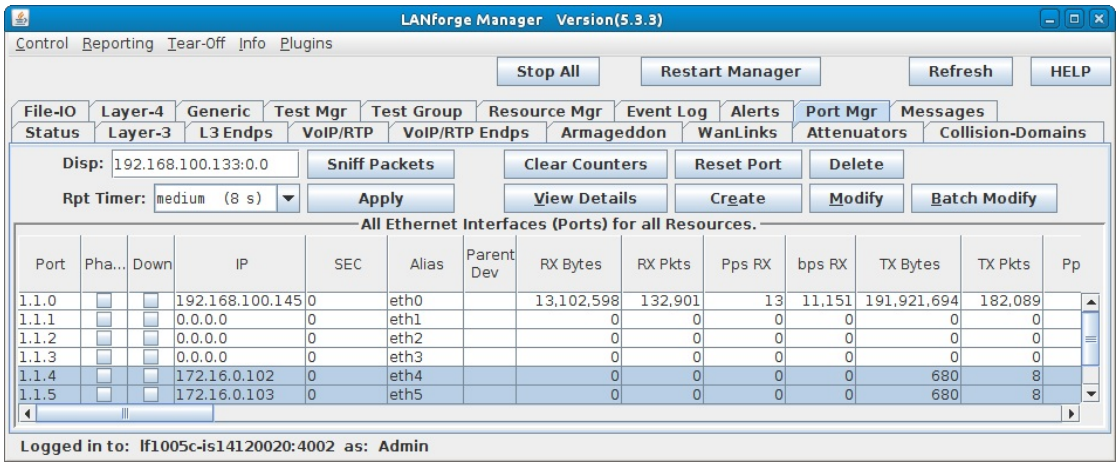
1. For this example, two physical ports on a single LANforge system are connected together.
2. Set up the LANforge ports so that they have valid IP addresses and IP masks.
   A. Using the LANforge GUI, go to the Port Manager tab.



B. Assign IP addresses and masks to the two ports.

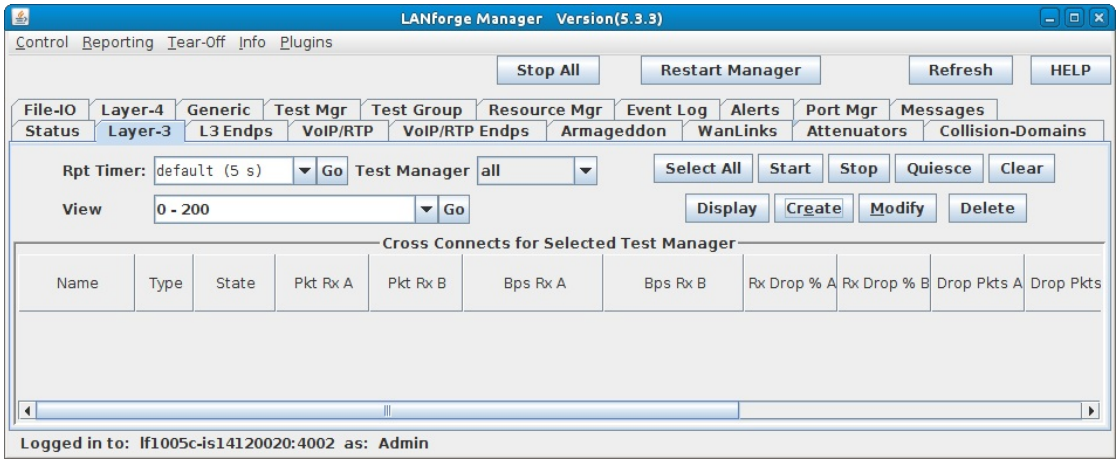


C. Verify the port configuration.

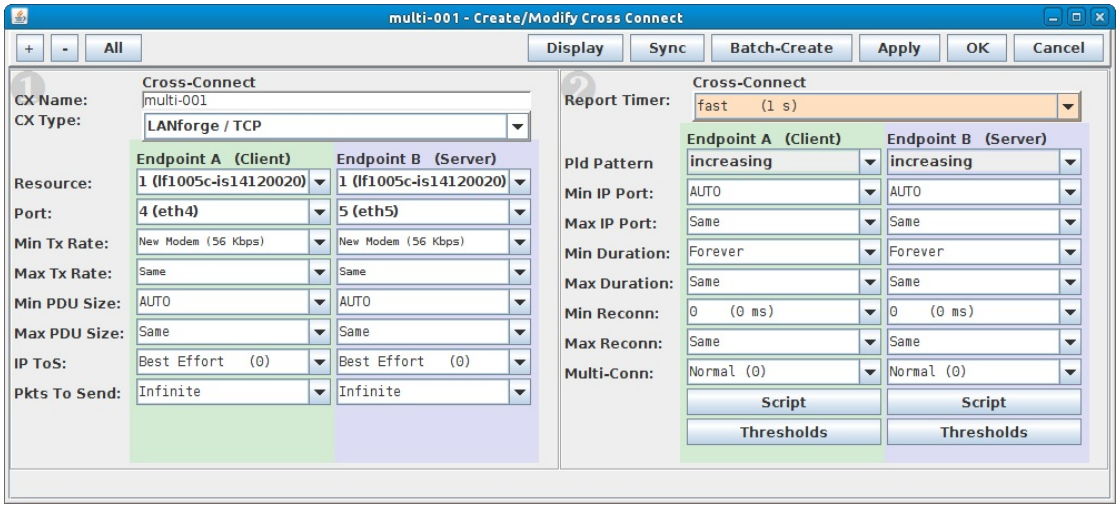


For more information see LANforge User's Guide: Ports (Interfaces)

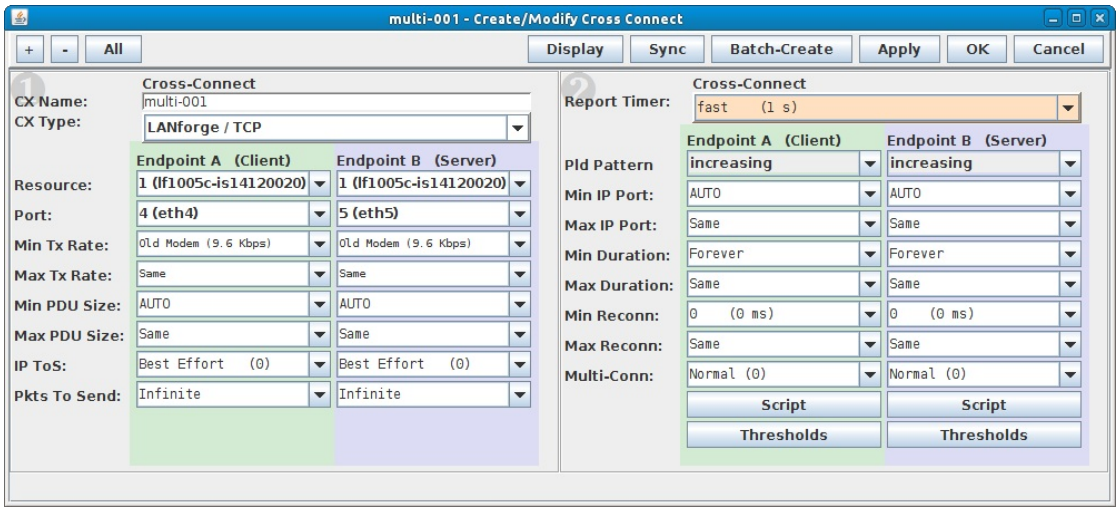
3. Set up a Layer-3 Multi-Conn TCP connection.
   A. Go to the **Layer-3** tab, and select the **Create** button.



   B. For Endpoint A, type in a name in the **CX Name** field, choose a **CX Type** of 'LANforge/TCP', choose the correct ports and set the **Report Timer** to 'fast (1 s)'.

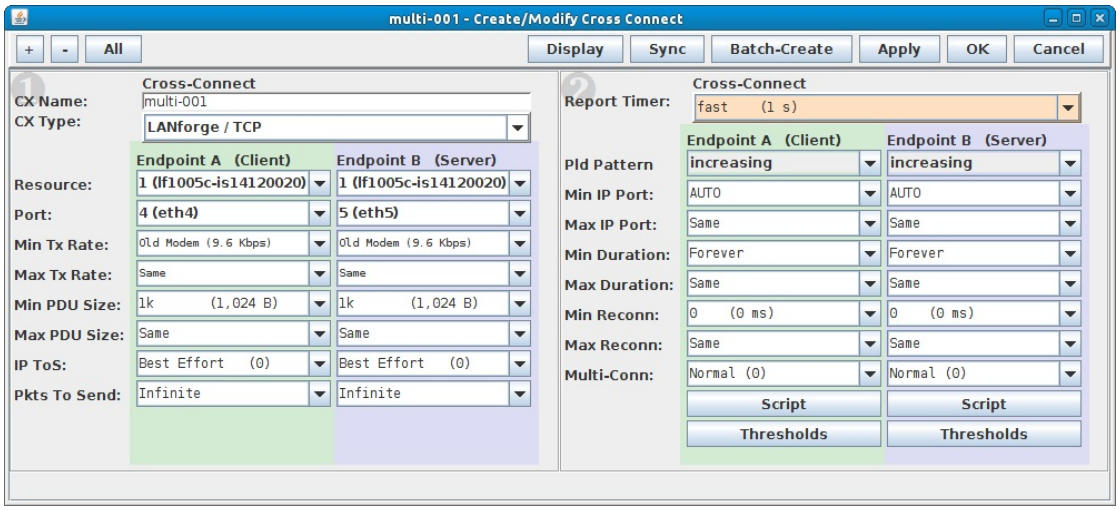


   C. Set the **Min Tx Rate** on both Endpoints A and B to '9.6Kbps'. Leave the Max Tx Rate on both set to 'Same'.



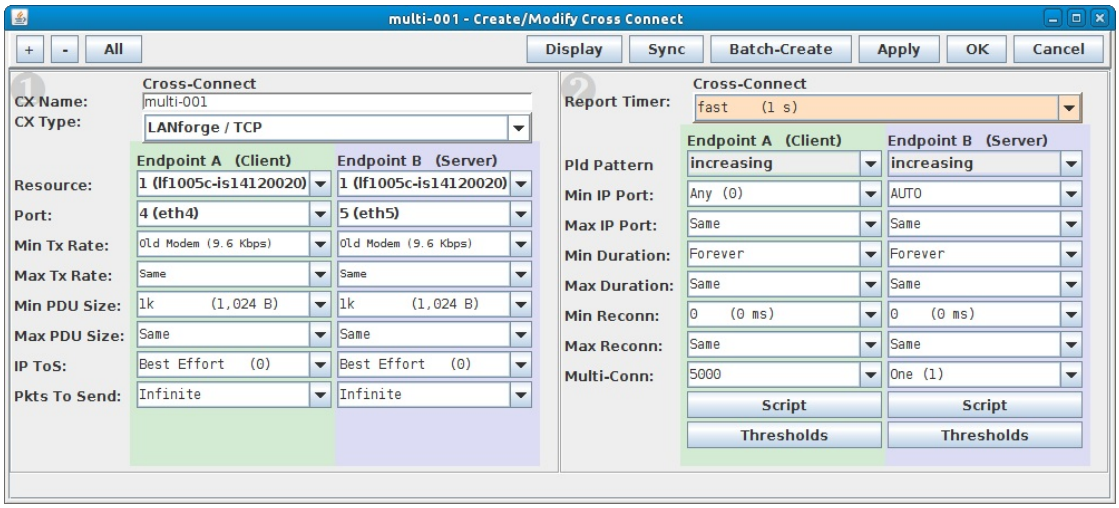
      A. **NOTE**: Higher speeds and/or bursty transmit rates will require more memory to process per connection.

D. Set the **Min PDU Size** on both Endpoints A and B to '1024B'. Leave the Max PDU Size on both set to 'Same'.



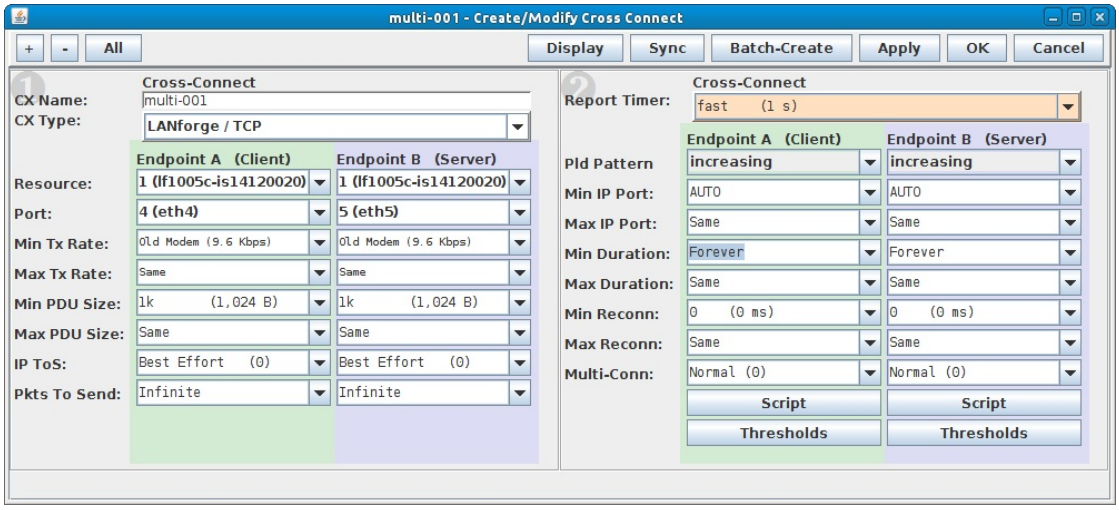
A. **NOTE**: Larger packet sizes will use more memory, smaller packet sizes will require more CPU processing time.

E. Set **Min IP Port** to **0 (zero)** on Endpoint A. Leave the Min IP Port on Endpoint B set to AUTO.

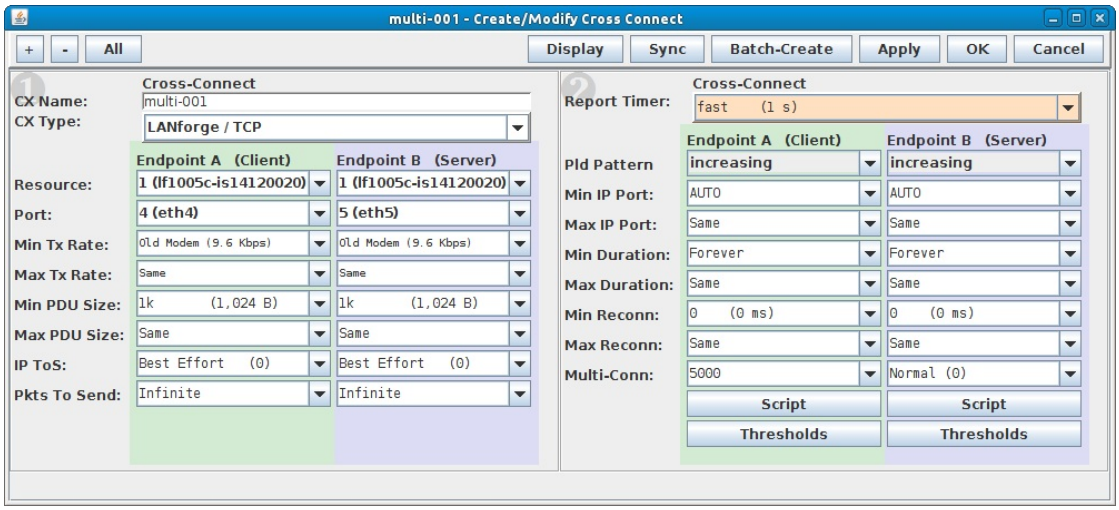


A. Leaving Min IP Port on AUTO would make connections very slow to reconnect due to IP port re-use issues. Multi-conn may not work at all with a fixed IP port.

F. Leave the **Min Duration** set to Forever and **Max Duration** set to Same for both endpoints, unless you want each of the multiple TCP connections to teardown and restart after a specified duration.

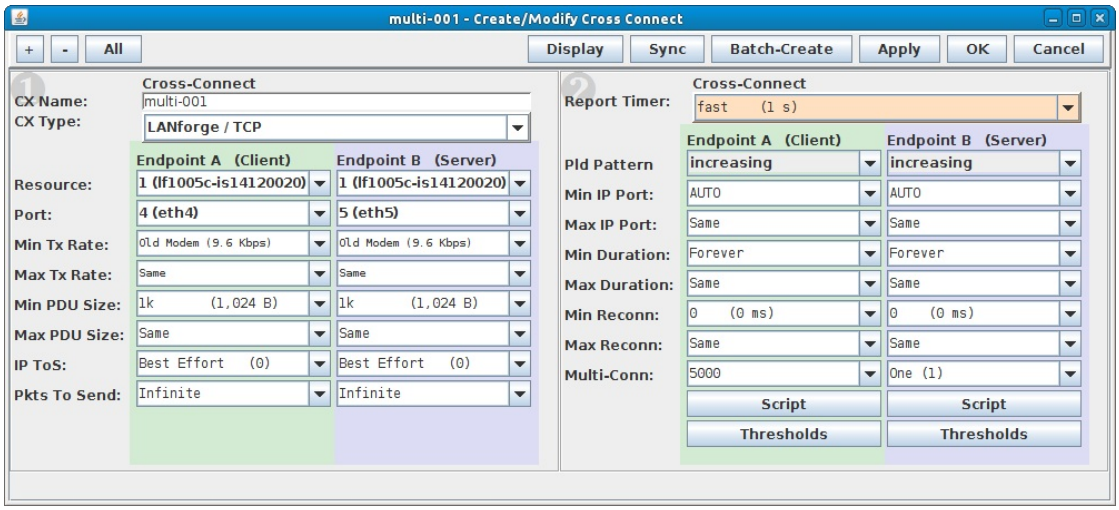


G. For Endpoint A, set the **Multi-Conn** to the desired number of multiple TCP connections.



A. You can type in any value that your license allows here or choose one from the drop down box. The recommended approach is to create a new Multi-Conn connection for every 5000 connections.

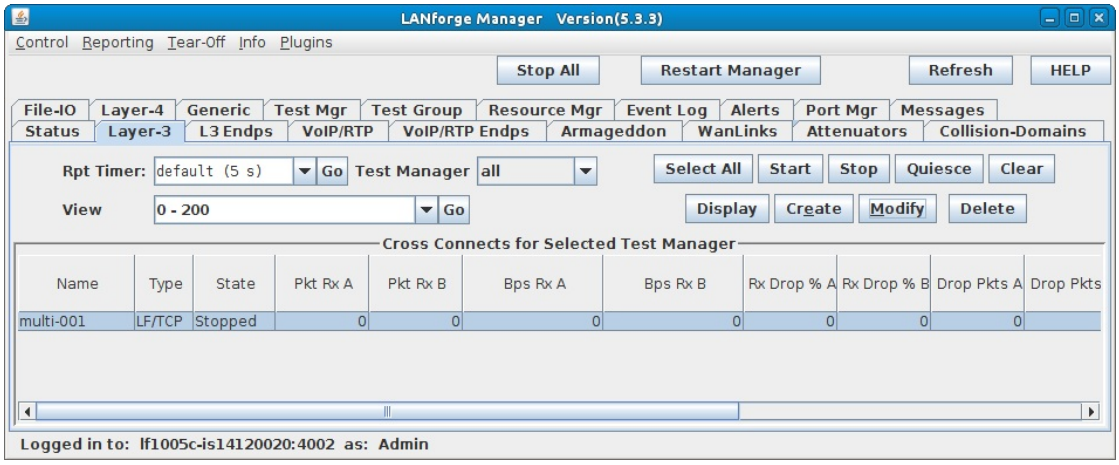
H. For Endpoint B, set the **Multi-Conn** to one, then select **OK** to create the connection.



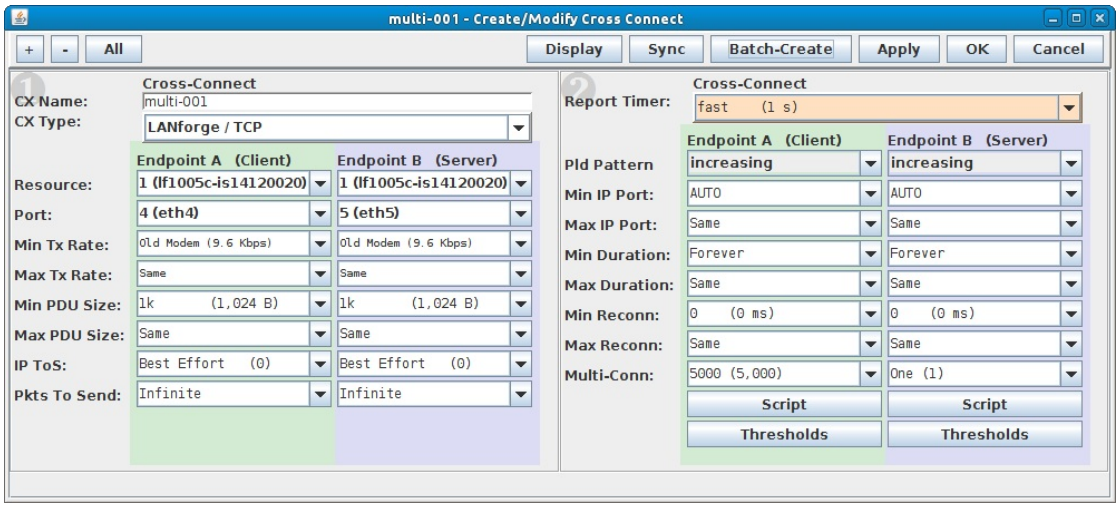
A. Endpoint B is the 'server' side of the TCP connection, therefore it is only necessary to have one Multi-Conn set up on Endpoint B to reply to all of the Endpoint A TCP SYN packets when the 5000 connections are being established.

For more information see LANforge User's Guide: Creating & Modifying Cross Connects

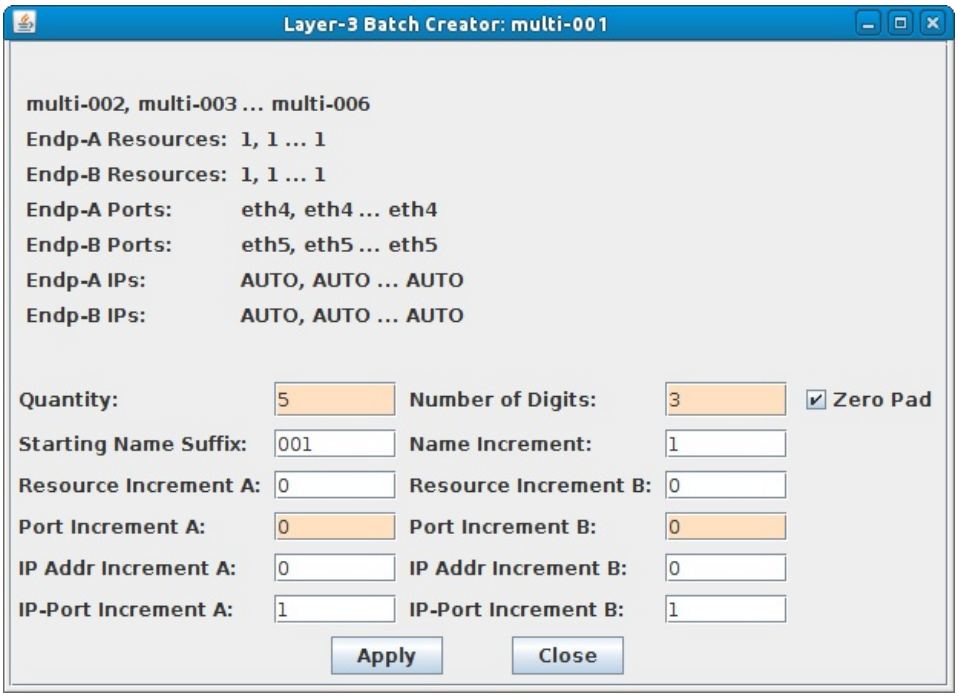
4. Create 5 more Layer-3 Multi-Conn TCP connections.
   A. On the **Layer-3** tab, highlight the first connection created in the previous step and select **Modify**.



B. Select the **Batch-Create** button.

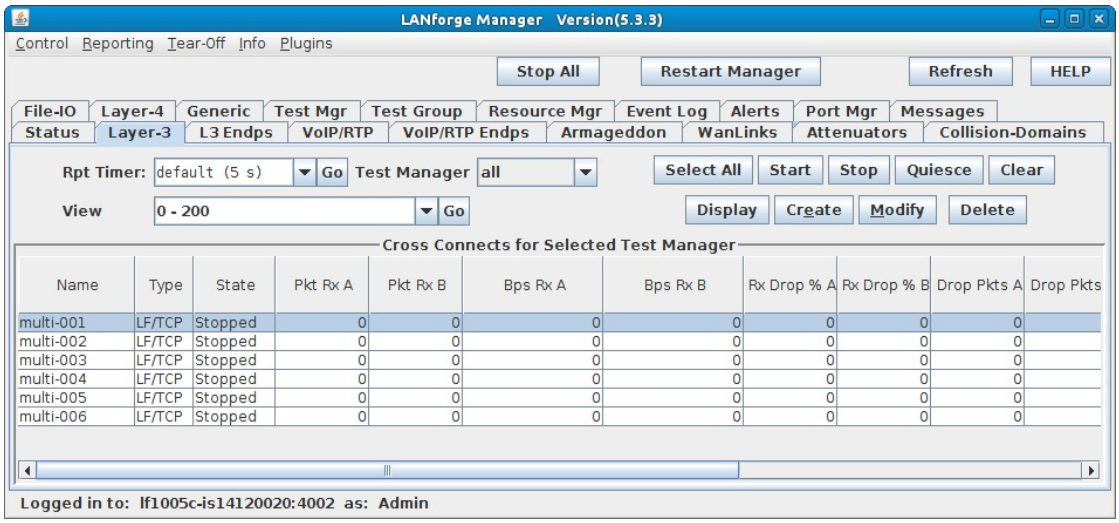


C. Set the **Quantity** to 5, and set **Port Increment A** and **Port Increment B** to **0 (zero)**



D. Select the **Apply** button.

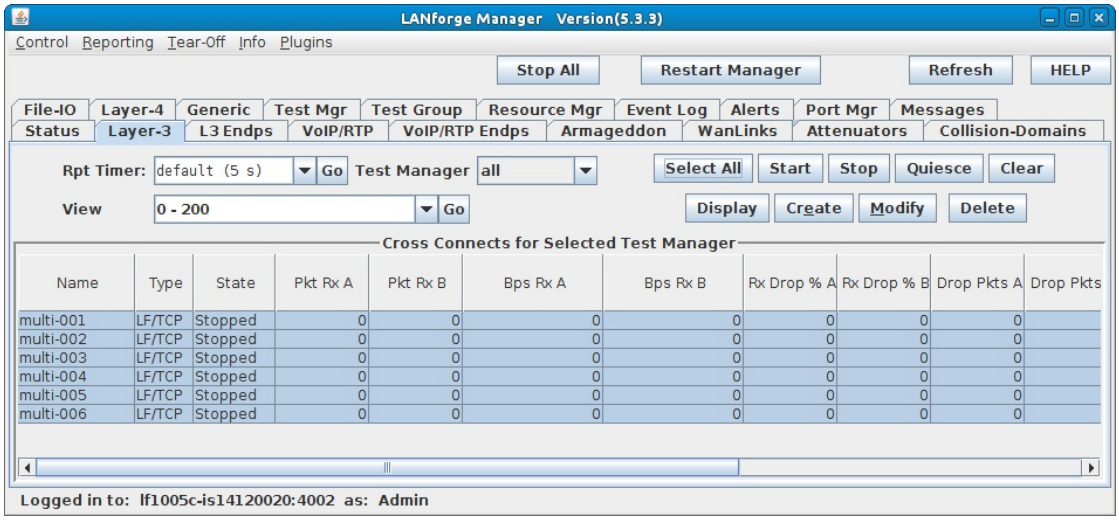
E. Verify that six Multi-Conn TCP connections were created.



Name	Type	State	Pkt Rx A	Pkt Rx B	Bps Rx A	Bps Rx B	Rx Drop % A	Rx Drop % B	Drop Pkts A	Drop Pkts
multi-001	LF/TCP	Stopped	0	0	0	0	0	0	0	
multi-002	LF/TCP	Stopped	0	0	0	0	0	0	0	
multi-003	LF/TCP	Stopped	0	0	0	0	0	0	0	
multi-004	LF/TCP	Stopped	0	0	0	0	0	0	0	
multi-005	LF/TCP	Stopped	0	0	0	0	0	0	0	
multi-006	LF/TCP	Stopped	0	0	0	0	0	0	0	

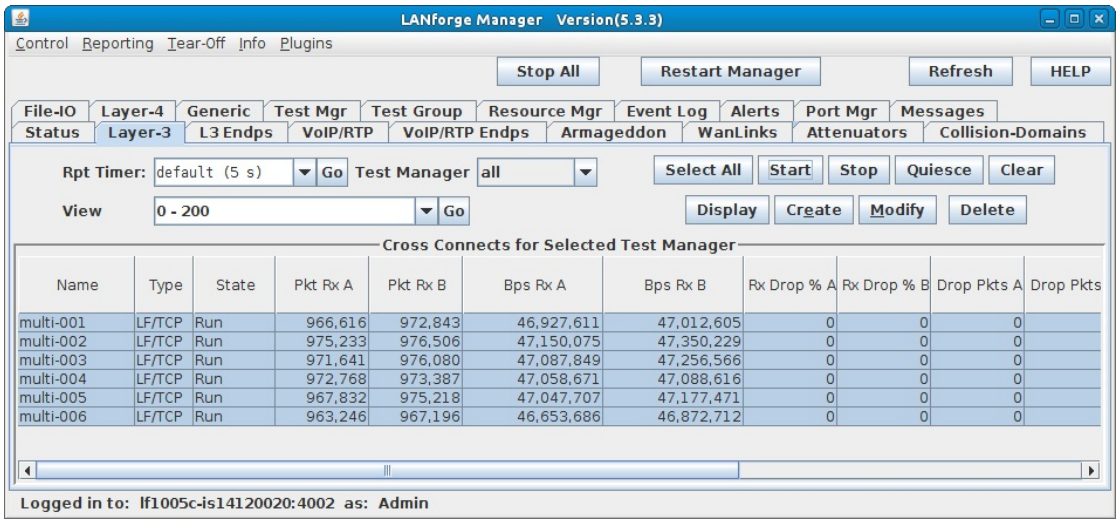
For more information see LANforge User's Guide: Layer-3 Endpoints (FIRE)

5. Run traffic and verify results.
   A. Highlight all six connections.



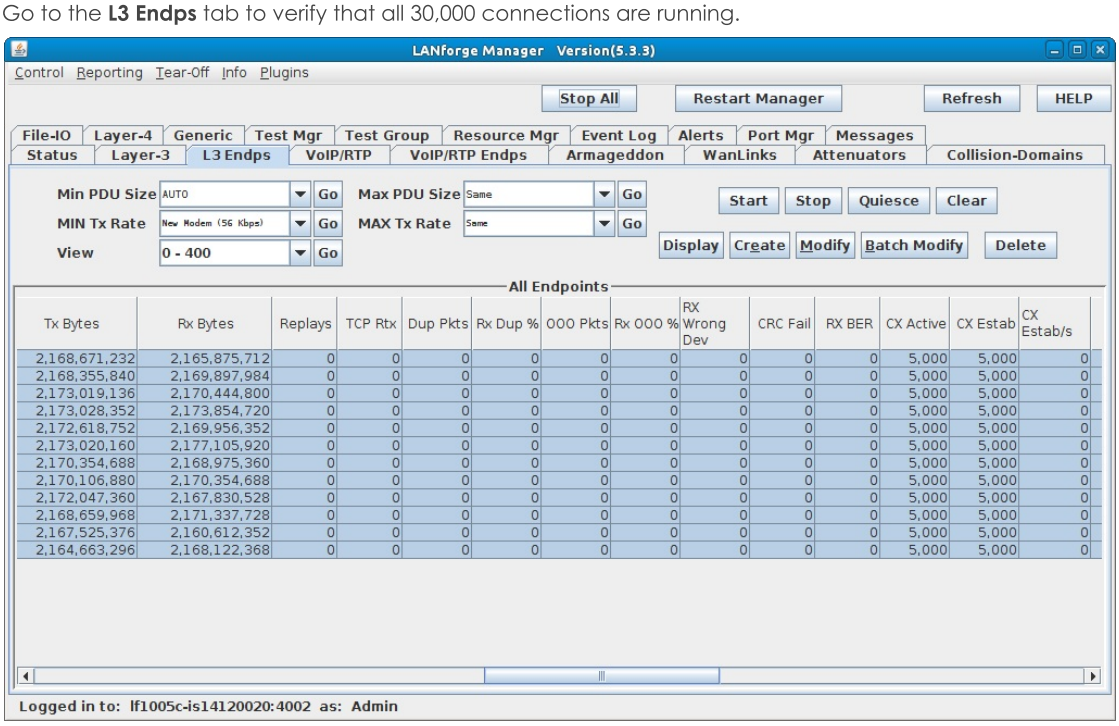
Name	Type	State	Pkt Rx A	Pkt Rx B	Bps Rx A	Bps Rx B	Rx Drop % A	Rx Drop % B	Drop Pkts A	Drop Pkts
multi-001	LF/TCP	Stopped	0	0	0	0	0	0	0	
multi-002	LF/TCP	Stopped	0	0	0	0	0	0	0	
multi-003	LF/TCP	Stopped	0	0	0	0	0	0	0	
multi-004	LF/TCP	Stopped	0	0	0	0	0	0	0	
multi-005	LF/TCP	Stopped	0	0	0	0	0	0	0	
multi-006	LF/TCP	Stopped	0	0	0	0	0	0	0	

   B. Select the **Start** button



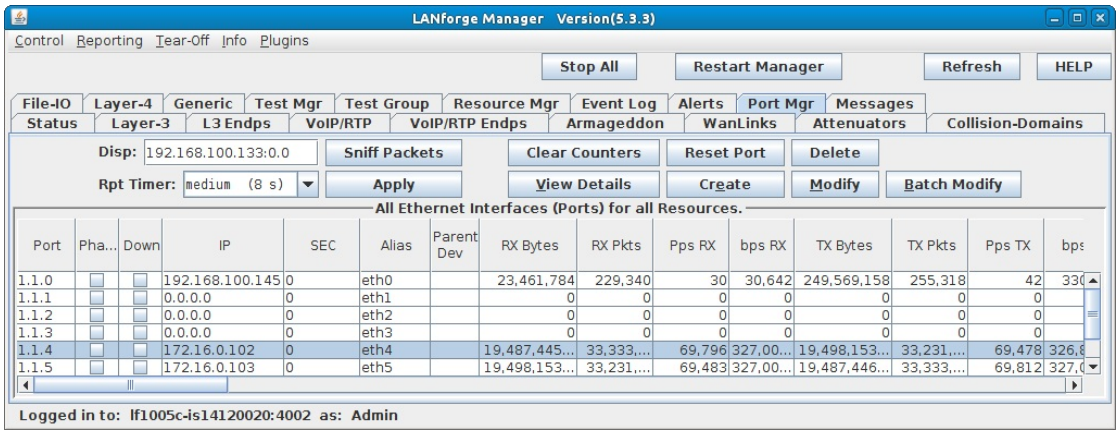
Name	Type	State	Pkt Rx A	Pkt Rx B	Bps Rx A	Bps Rx B	Rx Drop % A	Rx Drop % B	Drop Pkts A	Drop Pkts
multi-001	LF/TCP	Run	966,616	972,843	46,927,611	47,012,605	0	0	0	
multi-002	LF/TCP	Run	975,233	976,506	47,150,075	47,350,229	0	0	0	
multi-003	LF/TCP	Run	971,641	976,080	47,087,849	47,256,566	0	0	0	
multi-004	LF/TCP	Run	972,768	973,387	47,058,671	47,088,616	0	0	0	
multi-005	LF/TCP	Run	967,832	975,218	47,047,707	47,177,471	0	0	0	
multi-006	LF/TCP	Run	963,246	967,196	46,653,686	46,872,712	0	0	0	

   C. Go to the **L3 Endps** tab to verify that all 30,000 connections are running.



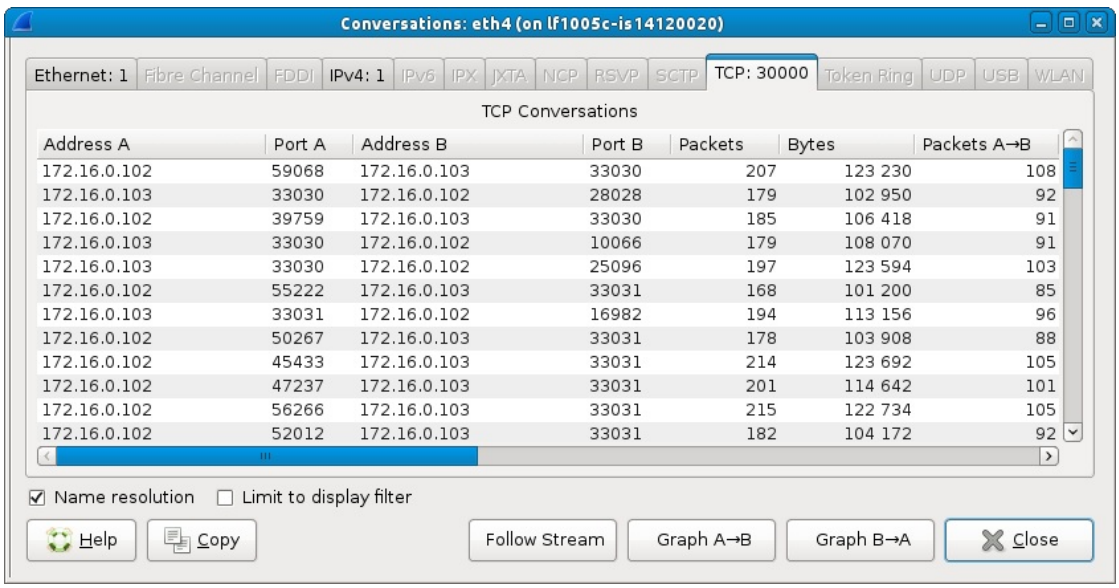
Tx Bytes	Rx Bytes	Replays	TCP Rtx	Dup Pkts	Rx Dup %	000 Pkts	Rx 000 %	RX Wrong Dev	CRC Fail	RX BER	CX Active	CX Estab	CX Estab/s
2,168,671,232	2,165,875,712	0	0	0	0	0	0	0	0	0	5,000	5,000	0
2,168,355,840	2,169,897,984	0	0	0	0	0	0	0	0	0	5,000	5,000	0
2,173,019,136	2,170,444,800	0	0	0	0	0	0	0	0	0	5,000	5,000	0
2,173,028,352	2,173,854,720	0	0	0	0	0	0	0	0	0	5,000	5,000	0
2,172,618,752	2,169,956,352	0	0	0	0	0	0	0	0	0	5,000	5,000	0
2,173,020,160	2,177,105,920	0	0	0	0	0	0	0	0	0	5,000	5,000	0
2,170,354,688	2,168,975,360	0	0	0	0	0	0	0	0	0	5,000	5,000	0
2,170,106,880	2,170,354,688	0	0	0	0	0	0	0	0	0	5,000	5,000	0
2,172,047,360	2,167,830,528	0	0	0	0	0	0	0	0	0	5,000	5,000	0
2,168,659,968	2,171,337,728	0	0	0	0	0	0	0	0	0	5,000	5,000	0
2,167,525,376	2,160,612,352	0	0	0	0	0	0	0	0	0	5,000	5,000	0
2,164,663,296	2,168,122,368	0	0	0	0	0	0	0	0	0	5,000	5,000	0

D. Or, go to the **Port Mgr** tab, and sniff one of the ports while the **Multi-Conn TCP** connections are running.



A. **NOTE**: You will need a lot of memory to run a Wireshark capture on 30,000 connections. It is probably best to stop all connections, set up the Wireshark capture, start all connections and let them run for 30 seconds to 1 minute before stopping them all, then stop the Wireshark capture and wait for it to process all those packets!

E. After the Wireshark capture is complete, go to **Statistics - Conversations** to allow Wireshark to analyze the conversations and show that all 30,000 connections were captured. This will take some time to complete depending on the size of the capture.



For more information see LANforge User's Guide: Layer-3 Endpoints (FIRE)

Multiple Thousands of UDP Connections

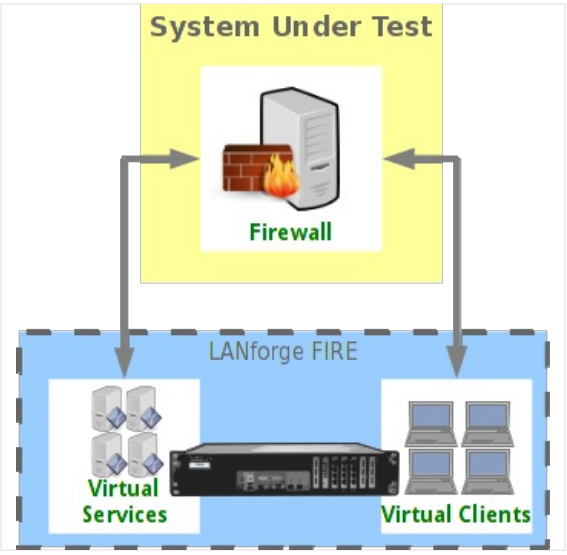
Goal: Set up and run traffic on multiple thousands of UDP connections.

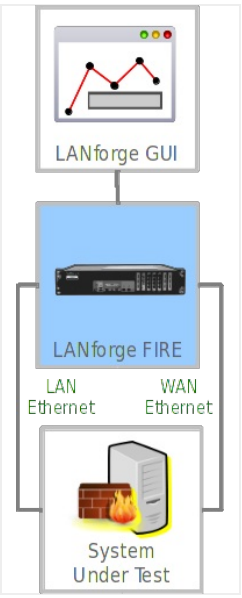
LANforge-FIRE has the capability to create multiple thousands of UDP connections using only two ports. This example creates 1000 Layer-3 connections between two sets of 1000 MAC-VLANS that can all run simultaneously. This feature can be used to test stateless connection services such as DNS, DHCP, and file streaming systems.

The upper limit of the total number of concurrent connections is determined by the processor speed and total memory available on the system running the LANforge server software. In addition, the packet rate and payload size of each connection will also affect performance. Larger packet sizes take less CPU time to process.

NOTE: If you are attempting to run this test scenario, you will need a LANforge license key that enables the correct number of ports. Please contact us at

support@candelatech.com for assistance.





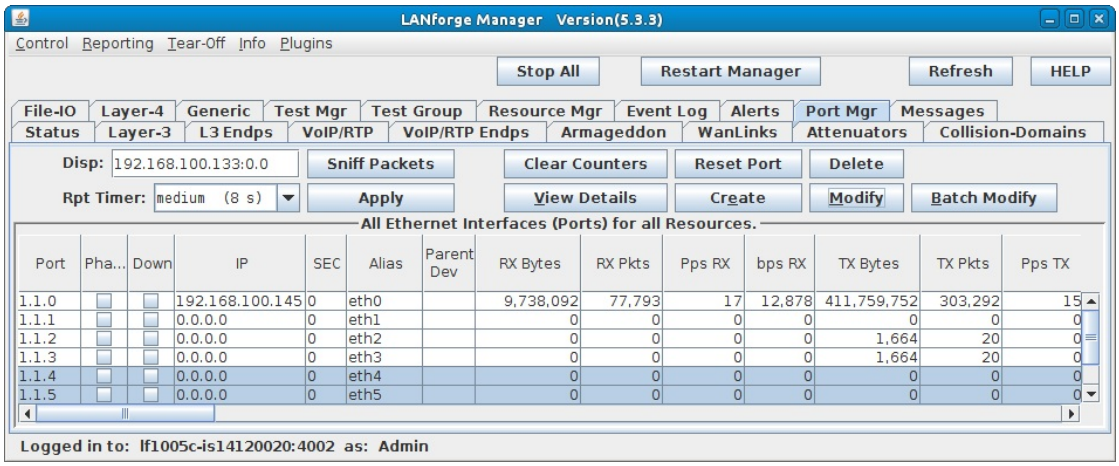
1. For this example, we are using a CT503-MIX 1Gx4 with multiple 10 Gigabit fiber NICs. We have cabled two physical ports on this single LANforge system together. Our plan of action will be to create 1000 MAC-VLANs on each port as endpoints for 1000 Layer-3 UDP cross connections.

   A. **Connection Suggestion:** If you are using the LANforge GUI Client remotely connected to your LANforge Server (like using a VPN), this process will appear rather sluggish. This suggishness is caused by the volume of reporting traffic flowing to the GUI. We suggest you connect to the LANforge Server desktop using Remote Desktop or VNC.

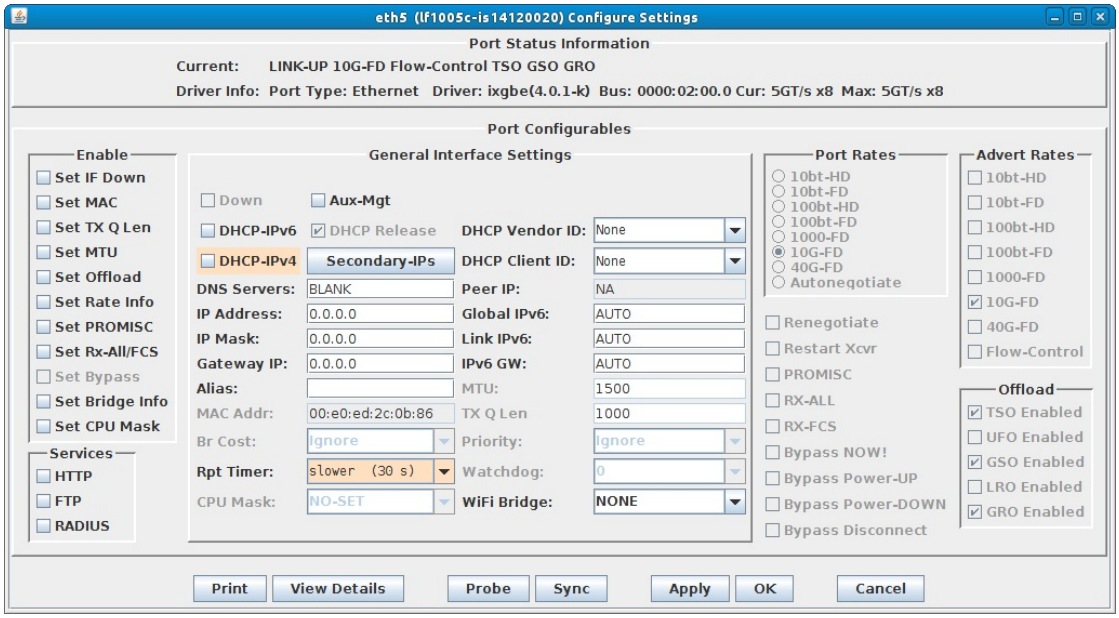
   B. **Reporting Suggestion:** When creating 2000 MAC-VLANs and 1000 Layer-3 connections, we are actually creating `2000 + (3 x 1000) = 5000` reportable entities that could generate up to `5000 x 4kB = 20,000kB` of traffic per reporting interval. We intentionally set our reporting intervals low (10 or more seconds) to keep the GUI client responsive.

2. Set up two physical ports so that they have 0.0.0.0 IP addresses and IP masks. We will choose `eth4` and `eth5` for this example. We will base our MAC-VLANs off these two physical ports. Set the reporting level for these two ports to a low frequency: `30 sec`.

A. In the LANforge GUI, go to the Port Manager tab.



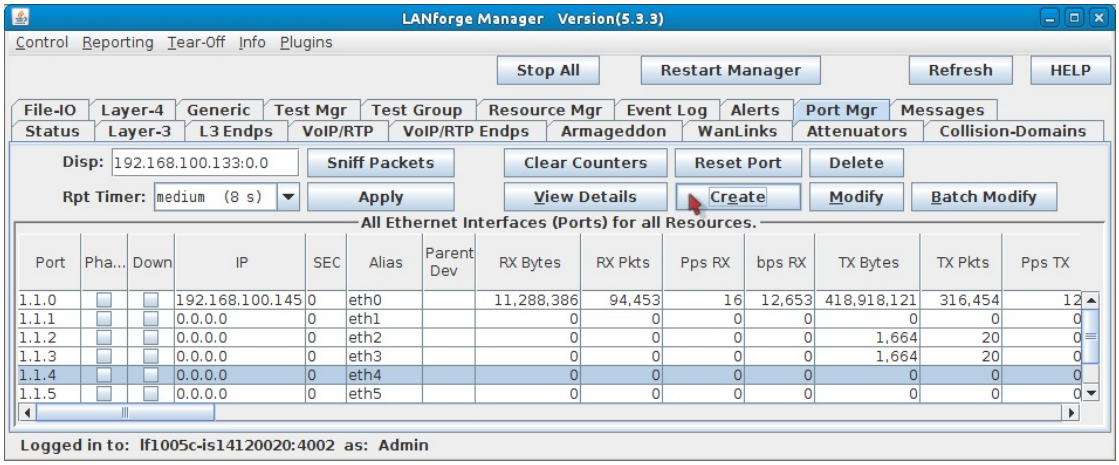
B. Assign IP addresses and masks to the two ports.



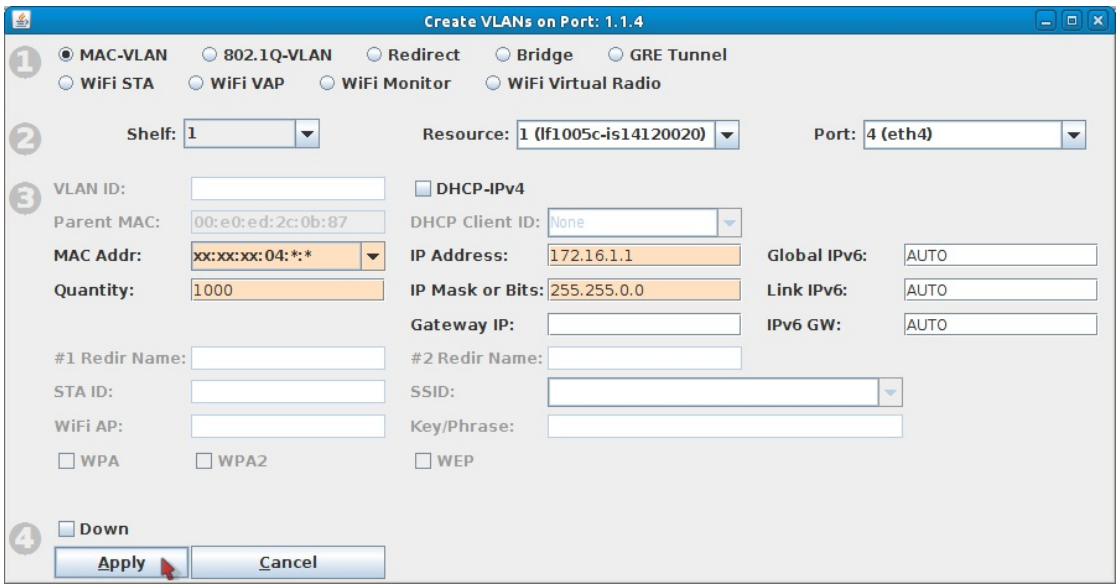
C. Verify the port configuration.

For more information see LANforge User's Guide: Ports (Interfaces)

3. We will use the Port Batch Create tool to create 1000 MAC-VLANs on each of the two ports.These MAC-VLAN ports should report more frequently than the parent ports: set them to report every 15 seconds. We also want to create MAC addresses that are distinct for these two groups: the first three octets will match the parent port, but we will set their fourth octets to 04 and 05 to remind us of their parent ports. The following two octets will be random (specified with `'*'`).
   A. In the **Ports** tab, highlight port `eth4` and click the **Create** button.

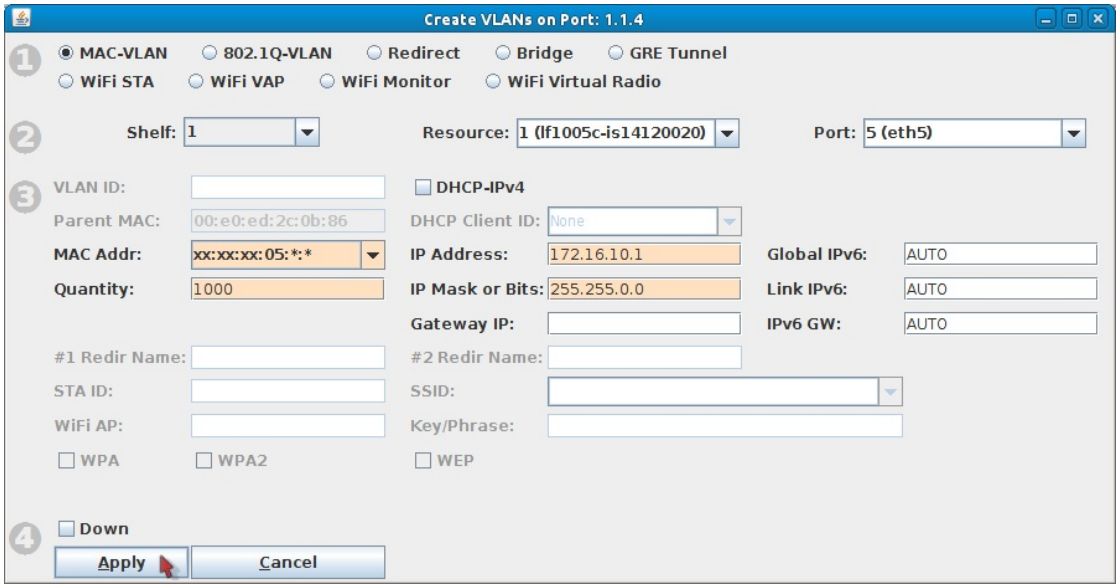


   B. Follow the these steps to create 1000 MAC-VLANs on the port.



      A. Create a **MAC Address** pattern similar to the parent MAC address pattern: `00:e0:ed:30:*:*`
      B. Set **Quantity** to `1000`
      C. Set an initial **IP Address**: `172.16.1.1`
      D. Set the **IP Mask**: `255.255.0.0`
      E. Click **Apply**, this takes a second to start.
      F. You do not need to close the **Create** window.

C. Repeat the previous step with these changes to create the next 1000 MAC-VLANs on port `eth5`.



A. Create a **MAC Address** pattern similar to the parent MAC address pattern: `00:e0:ed:05:*:*`

B. Set **Quantity** to `1000`

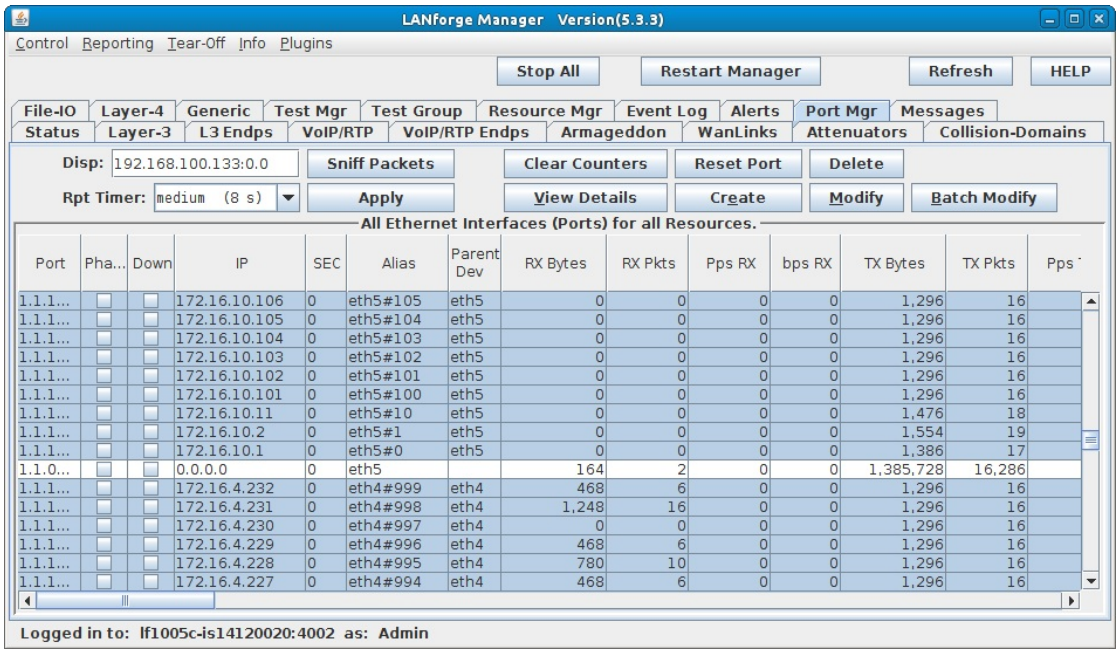
C. Set an initial **IP Address**: `172.16.10.1`

D. Set the **IP Mask**: `255.255.0.0`

E. Click **Apply**, this takes a second to start.

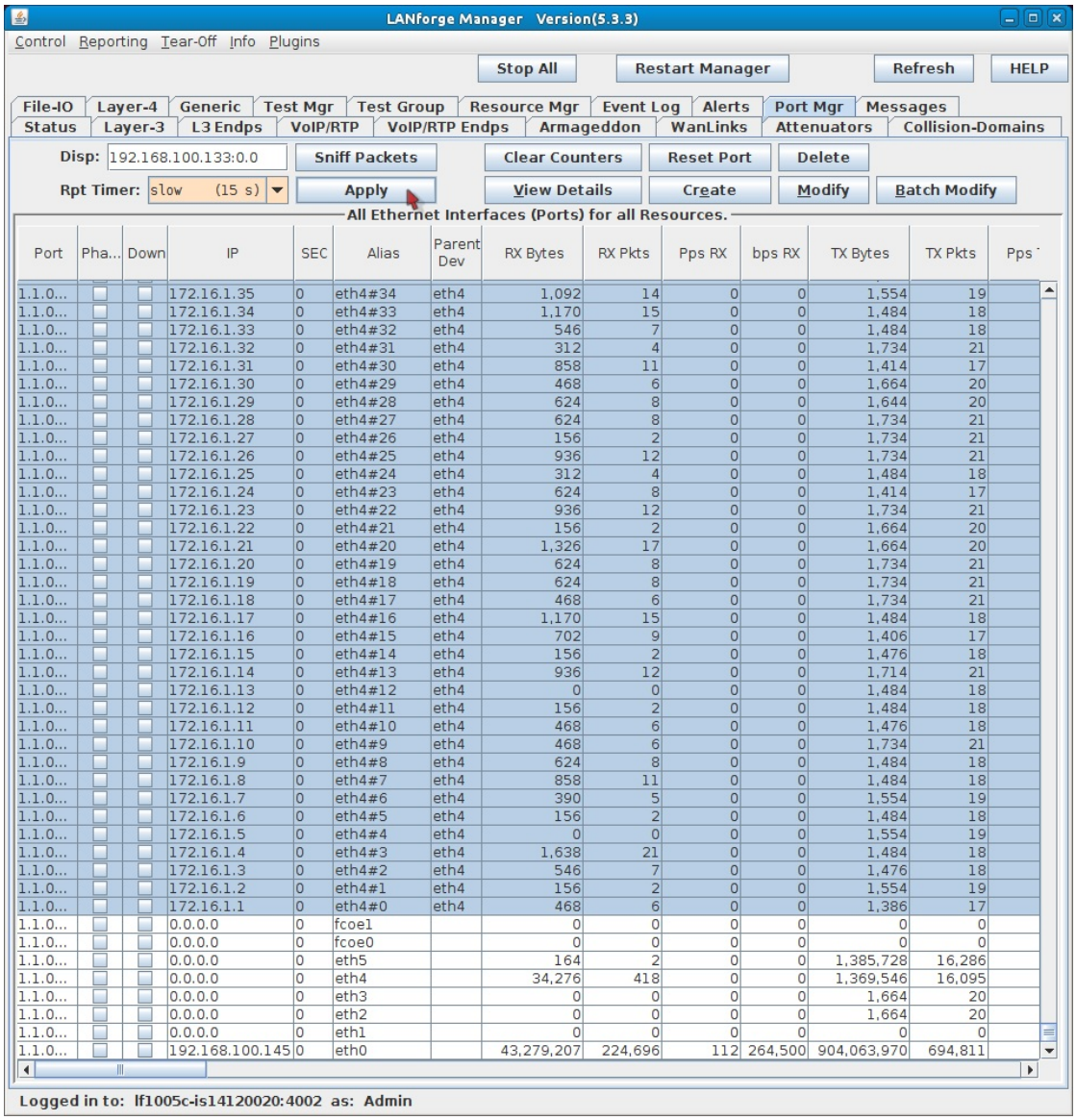
F. When the progress bar switches to Ready you can click **Cancel** to close the Create window.

D. Check the **Ports** tab to watch the IP addresses get assigned to the MAC-VLANs. This will take 5 or more minutes.



Port	Pha...	Down	IP	SEC	Alias	Parent Dev	RX Bytes	RX Pkts	Pps RX	bps RX	TX Bytes	TX Pkts	Pps
1.1.1...			172.16.10.106	0	eth5#105	eth5	0	0	0	0	1,296	16	
1.1.1...			172.16.10.105	0	eth5#104	eth5	0	0	0	0	1,296	16	
1.1.1...			172.16.10.104	0	eth5#103	eth5	0	0	0	0	1,296	16	
1.1.1...			172.16.10.103	0	eth5#102	eth5	0	0	0	0	1,296	16	
1.1.1...			172.16.10.102	0	eth5#101	eth5	0	0	0	0	1,296	16	
1.1.1...			172.16.10.101	0	eth5#100	eth5	0	0	0	0	1,296	16	
1.1.1...			172.16.10.11	0	eth5#10	eth5	0	0	0	0	1,476	18	
1.1.1...			172.16.10.2	0	eth5#1	eth5	0	0	0	0	1,554	19	
1.1.1...			172.16.10.1	0	eth5#0	eth5	0	0	0	0	1,386	17	
1.1.0...			0.0.0.0	0	eth5		164	2	0	0	1,385,728	16,286	
1.1.1...			172.16.4.232	0	eth4#999	eth4	468	6	0	0	1,296	16	
1.1.1...			172.16.4.231	0	eth4#998	eth4	1,248	16	0	0	1,296	16	
1.1.1...			172.16.4.230	0	eth4#997	eth4	0	0	0	0	1,296	16	
1.1.1...			172.16.4.229	0	eth4#996	eth4	468	6	0	0	1,296	16	
1.1.1...			172.16.4.228	0	eth4#995	eth4	780	10	0	0	1,296	16	
1.1.1...			172.16.4.227	0	eth4#994	eth4	468	6	0	0	1,296	16	

E. Adjust the Report Timer.



Port	Pha...	Down	IP	SEC	Alias	Parent Dev	RX Bytes	RX Pkts	Pps RX	bps RX	TX Bytes	TX Pkts	Pps
1.1.0...			172.16.1.35	0	eth4#34	eth4	1,092	14	0	0	1,554	19	
1.1.0...			172.16.1.34	0	eth4#33	eth4	1,170	15	0	0	1,484	18	
1.1.0...			172.16.1.33	0	eth4#32	eth4	546	7	0	0	1,484	18	
1.1.0...			172.16.1.32	0	eth4#31	eth4	312	4	0	0	1,734	21	
1.1.0...			172.16.1.31	0	eth4#30	eth4	858	11	0	0	1,414	17	
1.1.0...			172.16.1.30	0	eth4#29	eth4	468	6	0	0	1,664	20	
1.1.0...			172.16.1.29	0	eth4#28	eth4	624	8	0	0	1,644	20	
1.1.0...			172.16.1.28	0	eth4#27	eth4	624	8	0	0	1,734	21	
1.1.0...			172.16.1.27	0	eth4#26	eth4	156	2	0	0	1,734	21	
1.1.0...			172.16.1.26	0	eth4#25	eth4	936	12	0	0	1,734	21	
1.1.0...			172.16.1.25	0	eth4#24	eth4	312	4	0	0	1,484	18	
1.1.0...			172.16.1.24	0	eth4#23	eth4	624	8	0	0	1,414	17	
1.1.0...			172.16.1.23	0	eth4#22	eth4	936	12	0	0	1,734	21	
1.1.0...			172.16.1.22	0	eth4#21	eth4	156	2	0	0	1,664	20	
1.1.0...			172.16.1.21	0	eth4#20	eth4	1,326	17	0	0	1,664	20	
1.1.0...			172.16.1.20	0	eth4#19	eth4	624	8	0	0	1,734	21	
1.1.0...			172.16.1.19	0	eth4#18	eth4	624	8	0	0	1,734	21	
1.1.0...			172.16.1.18	0	eth4#17	eth4	468	6	0	0	1,734	21	
1.1.0...			172.16.1.17	0	eth4#16	eth4	1,170	15	0	0	1,484	18	
1.1.0...			172.16.1.16	0	eth4#15	eth4	702	9	0	0	1,406	17	
1.1.0...			172.16.1.15	0	eth4#14	eth4	156	2	0	0	1,476	18	
1.1.0...			172.16.1.14	0	eth4#13	eth4	936	12	0	0	1,714	21	
1.1.0...			172.16.1.13	0	eth4#12	eth4	0	0	0	0	1,484	18	
1.1.0...			172.16.1.12	0	eth4#11	eth4	156	2	0	0	1,484	18	
1.1.0...			172.16.1.11	0	eth4#10	eth4	468	6	0	0	1,476	18	
1.1.0...			172.16.1.10	0	eth4#9	eth4	468	6	0	0	1,734	21	
1.1.0...			172.16.1.9	0	eth4#8	eth4	624	8	0	0	1,484	18	
1.1.0...			172.16.1.8	0	eth4#7	eth4	858	11	0	0	1,484	18	
1.1.0...			172.16.1.7	0	eth4#6	eth4	390	5	0	0	1,554	19	
1.1.0...			172.16.1.6	0	eth4#5	eth4	156	2	0	0	1,484	18	
1.1.0...			172.16.1.5	0	eth4#4	eth4	0	0	0	0	1,554	19	
1.1.0...			172.16.1.4	0	eth4#3	eth4	1,638	21	0	0	1,484	18	
1.1.0...			172.16.1.3	0	eth4#2	eth4	546	7	0	0	1,476	18	
1.1.0...			172.16.1.2	0	eth4#1	eth4	156	2	0	0	1,554	19	
1.1.0...			172.16.1.1	0	eth4#0	eth4	468	6	0	0	1,386	17	
1.1.0...			0.0.0.0	0	fcoe1		0	0	0	0	0	0	
1.1.0...			0.0.0.0	0	fcoe0		0	0	0	0	0	0	
1.1.0...			0.0.0.0	0	eth5		164	2	0	0	1,385,728	16,286	
1.1.0...			0.0.0.0	0	eth4		34,276	418	0	0	1,369,546	16,095	
1.1.0...			0.0.0.0	0	eth3		0	0	0	0	1,664	20	
1.1.0...			0.0.0.0	0	eth2		0	0	0	0	1,664	20	
1.1.0...			0.0.0.0	0	eth1		0	0	0	0	0	0	
1.1.0...			192.168.100.145	0	eth0		43,279,207	224,696	112	264,500	904,063,970	694,811	

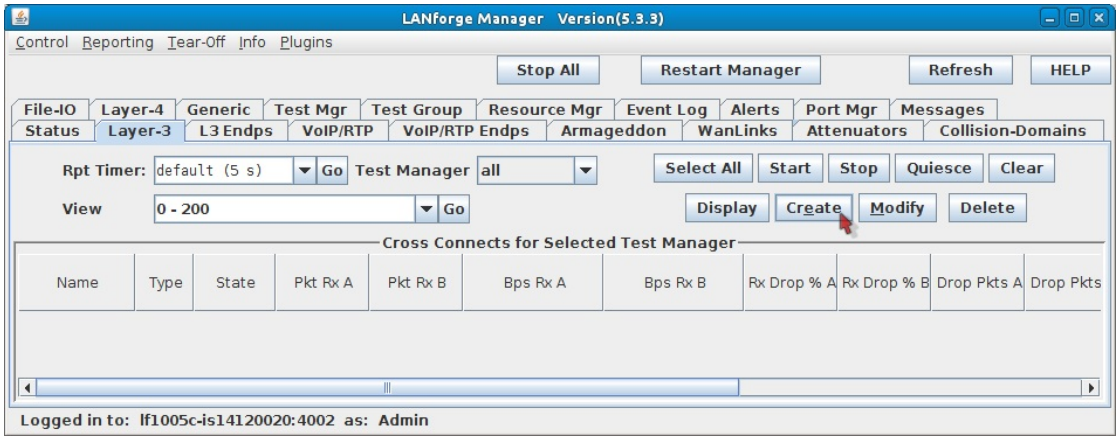
A. Click the Column header labeled **Port** to sort all the newly created MAC-VLANs to the top.

B. Select all the MAC-VLANs.

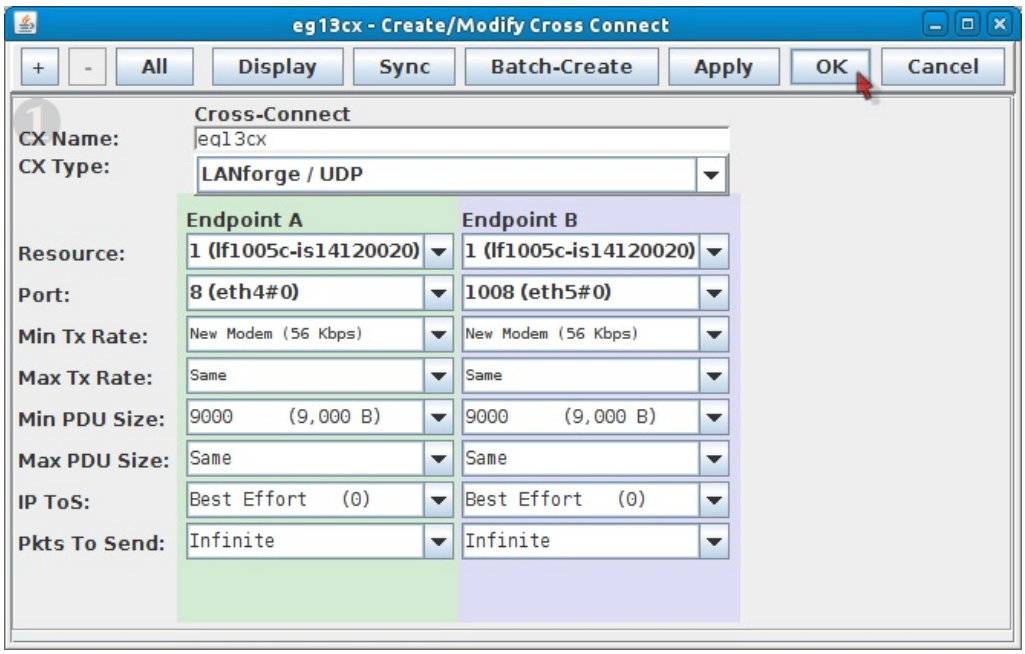
C. Change the **Rpt Timer** setting to `slow (15 s)`

D. Click **Apply**.

4. Create Cross Connects between the MAC-VLANs. We can create all these cross connections in two steps. First we will create the first cross-connect. Next, we will batch-create the remaining 999 cross connections using the first as a template.
   A. Switch to the **Layer-3** tab and click the **Create** button to see the **Create/Modify Cross Connect** window.

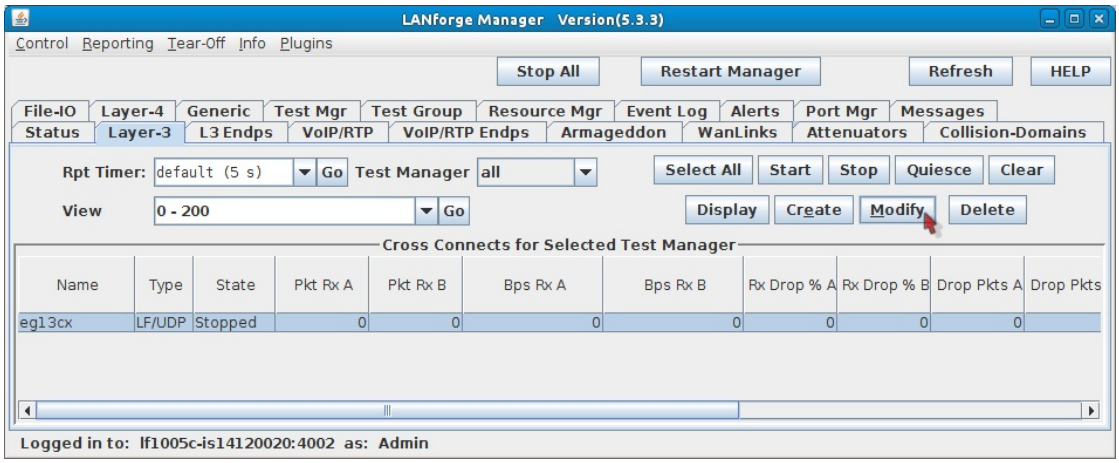


   B. Set up the cross connection between the first two MAC-VLANs.



      A. Connection name: `eg13cx`
      B. **Port, TX Endpoint:** `eth4#0`
      C. **Port, RX Endpoint:** `eth5#0`
      D. **Min PDU Size:** `9,000 B`
      E. Click the **OK** button to save.

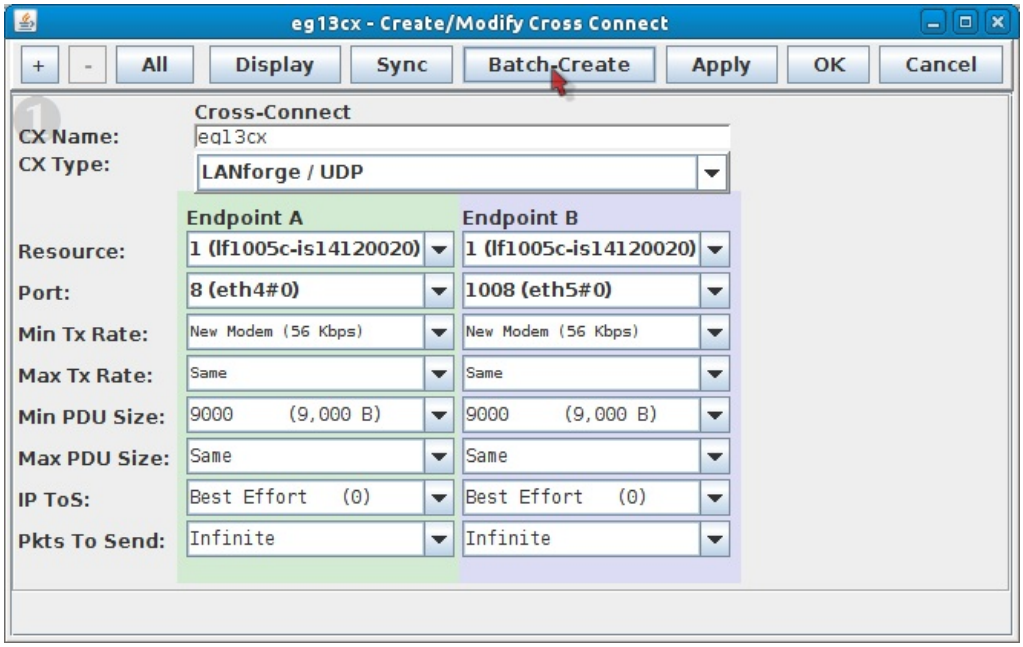
C. In the **Layer-3** tab...



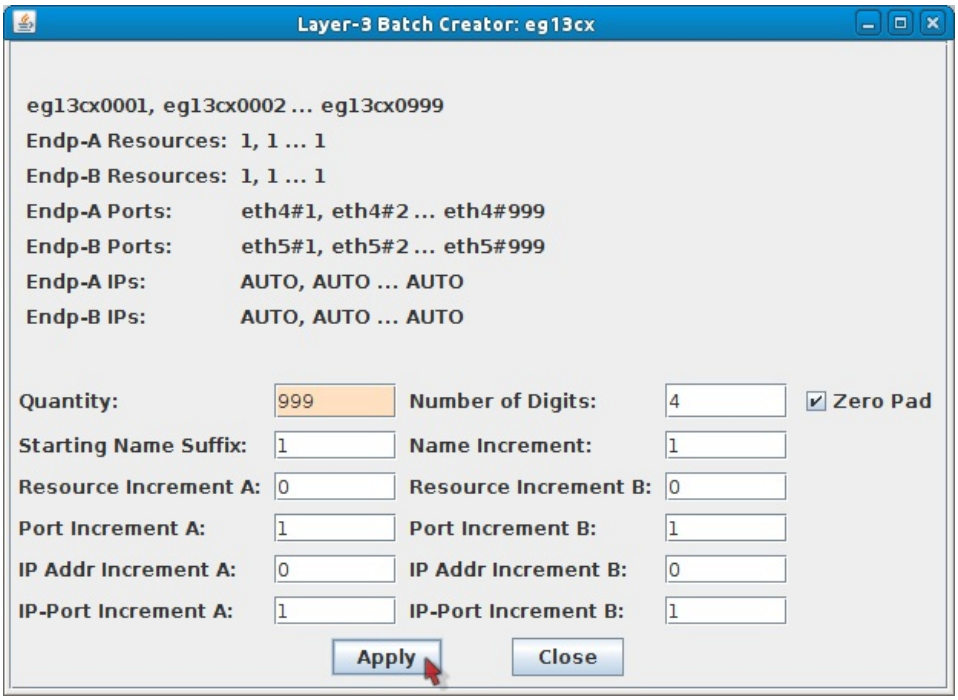
A. Select the cross-connect `eg13cx`

B. Click **Modify** in the **Layer-3** tab.

D. Click **Batch Create** in the **Create/Modify Cross Connect** window.



E. We can create cross connects in one batch. Using the **Layer-3 Batch Creator** window:

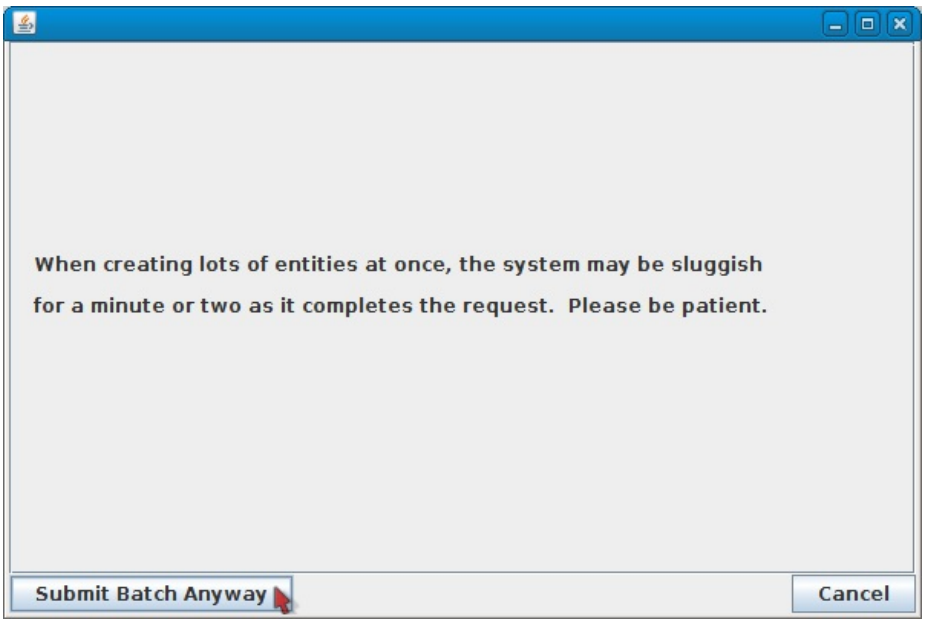


A. Specify for **Quantity**: `999`. This brings us to 1000 cross connects.

B. **Starting Name Suffix**: `1`

C. Click the **Apply** button to create the cross connects.

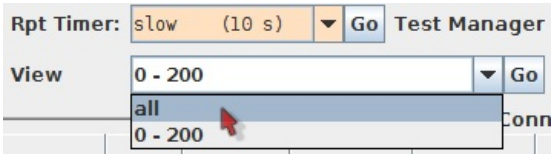
F. Expect the system to work for a few minutes while it creates the cross connects.



A. When the process completes, the Batch Warning will disappear.

B. Click the **Cancel** button on the **Create/Modify Cross Connects** window.

G. Update Report Timer for all cross connects.



A. The **Layer-3** tab now shows the first 200 Layer-3 cross connects.

B. Use the **View** dropdown to display all the cross connects: select `All` and press **Go**.

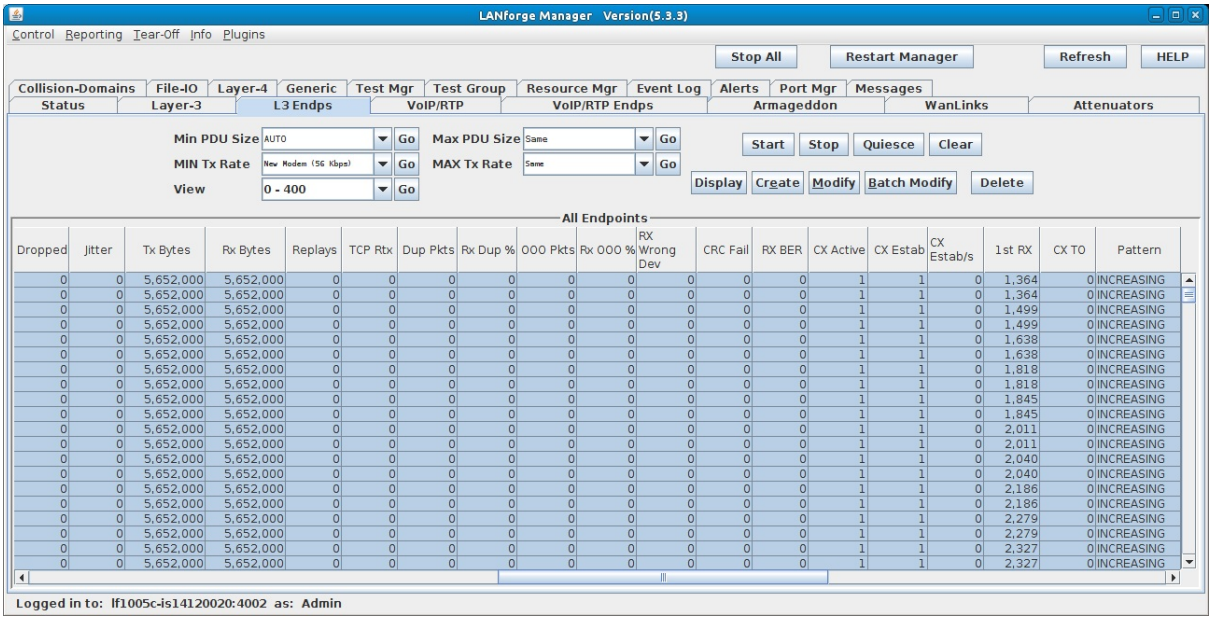
C. Click the **Select All** button to select all cross connects.

D. Select `slow (10s)` in the **Rpt Timer** dropdown.

E. Click the Report Timer **Go** button to apply. This will help keep the GUI responsive.

5. Start your cross connects.
    A. In the **Layer-3** tab, click the **Select All** button.
    B. Then click the **Start** button.

6. Monitoring your cross connects.



Dropped	Jitter	Tx Bytes	Rx Bytes	Replays	TCP Rtx	Dup Pkts	Rx Dup %	OOO Pkts	Rx OOO %	RX Wrong Dev	CRC Fail	RX BER	CX Active	CX Estab	CX Estab/s	1st RX	CX TO	Pattern
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	1,364	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	1,364	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	1,499	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	1,499	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	1,638	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	1,638	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	1,818	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	1,818	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	1,845	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	1,845	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	2,011	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	2,011	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	2,040	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	2,040	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	2,186	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	2,186	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	2,279	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	2,279	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	2,327	0	INCREASING
0	0	5,652,000	5,652,000	0	0	0	0	0	0	0	0	0	1	1	0	2,327	0	INCREASING

A. As you scroll to the left in the **L3 Endpoints** table, you can monitor the connection qualities.

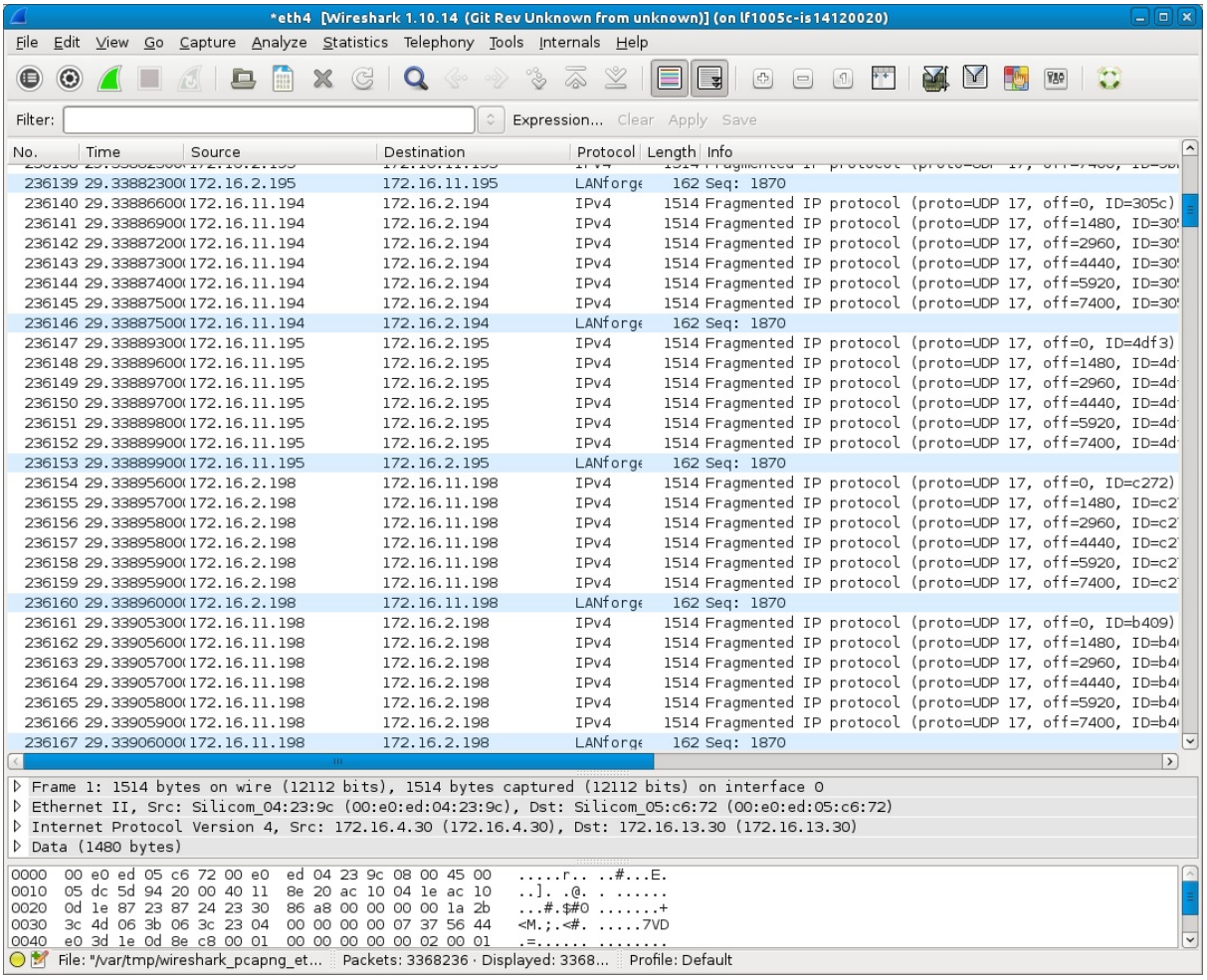
B. `Tx Bytes` and `Rx Bytes` columns show traffic amount.

Tx Bytes	Rx Bytes
5,652,000	5,652,000
5,652,000	5,652,000
5,652,000	5,652,000
5,652,000	5,652,000
5,652,000	5,652,000
5,652,000	5,652,000
5,652,000	5,652,000
5,652,000	5,652,000
5,652,000	5,652,000

C. The `Pattern` column shows traffic trend.

Pattern
INCREASING
INCREASING
INCREASING
INCREASING
INCREASING

7. Sniffing your cross connects with Wireshark from the parent ports `eth4` or `eth5`.



   A. To sniff the parent ports that the MAC-VLANs are associated with, you want to be connected to the LANforge server desktop using either Remote Desktop or VNC.

   B. Operating this many cross connects can be taxing on many machines. To minimize the sluggishness of Wireshark, start Wireshark when none of the Cross-Connects are active, and use the LANforge client to start and stop the traffic.

   C. In the **Layer-3** tab, click **Select All** and click **Stop**.

   D. In the **Port Mgr** tab, select interface `eth4` and click the **Sniff Packets** button.

   E. Wireshark will open and be capturing packets.

   F. In the **Layer-3** tab, click the **Start** button, and let your test run for one or two minutes.

   G. Then click the **Stop** or **Quiesce** buttons to end the test.

   H. Now you can save the Wireshark capture.

Making FIRE Chili

Goal: Make some tasty chili.

In this test scenario, LANforge-FIRE is not used to generate traffic. Instead, the test engineer should realize that all work and no play makes Homer something something.

Please note that this chili recipe is just as quick and dirty as the rest of the examples in this cookbook, but ever more tasty. Feel free to improvise...

1. Get yourself in a kitchen with some groceries.

2. Prep some ingredients.
   A. Chop up some onions. One should be good enough.

   B. Chop up some garlic. Just a few cloves.

3. Cook it all up

A. Brown some ground beef.

B. Heat up tomato sauce, kidney beans. 16oz of each is good.

C. Add garlic, onions, beef and spices such as cayenne, cumin, chili powder, salt and pepper.

D. Don't forget the beer!

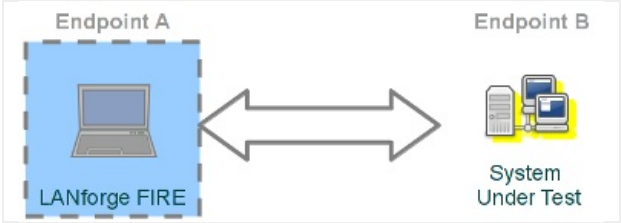
A. Note: The beer is for you, not the chili...unless of course you are experimenting!

E. Cornbread is also nice to have.

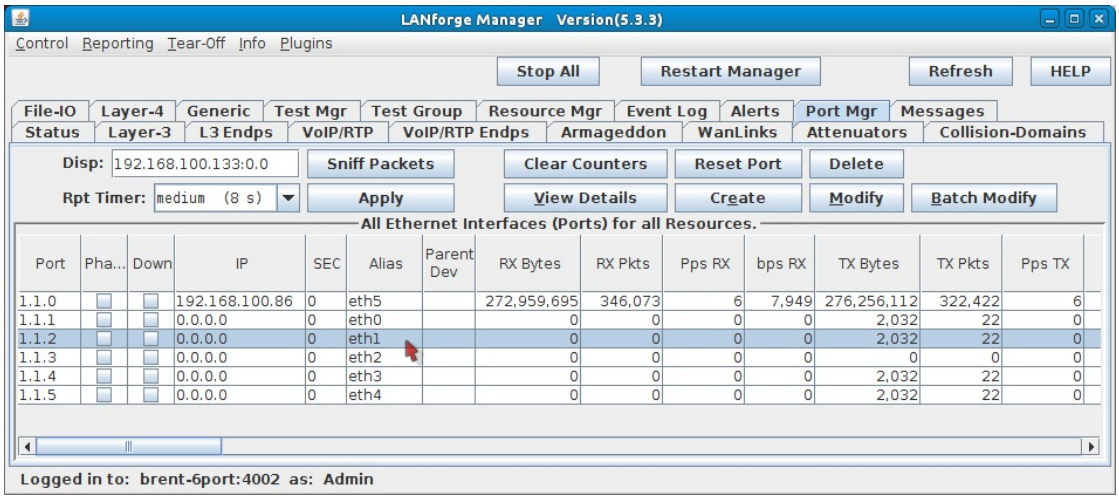
4. Enjoy!
   A. Serve with additional chopped onions, corn chips and cheddar cheese.

Goal: Generate one-sided traffic to a network device with a fixed destination IP address.

This scenario is useful for testing switches, firewalls and data loggers that have to handle highly varied or very fast UDP packet streams with a fixed destination. A one-sided traffic stream is used to send packets to a network device under test when round-trip reporting is not required.

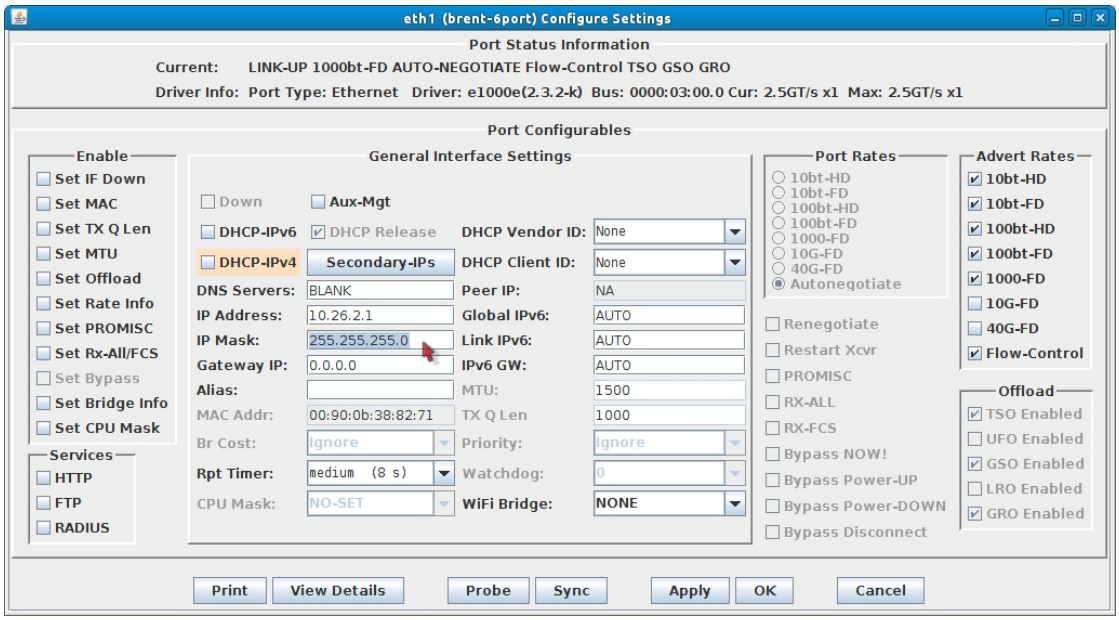


1. Configure an ethernet port.
   A. On the **Port Mgr** tab, select a port within the table and click the **Modify** button.



   A. This example will use port `eth1`.

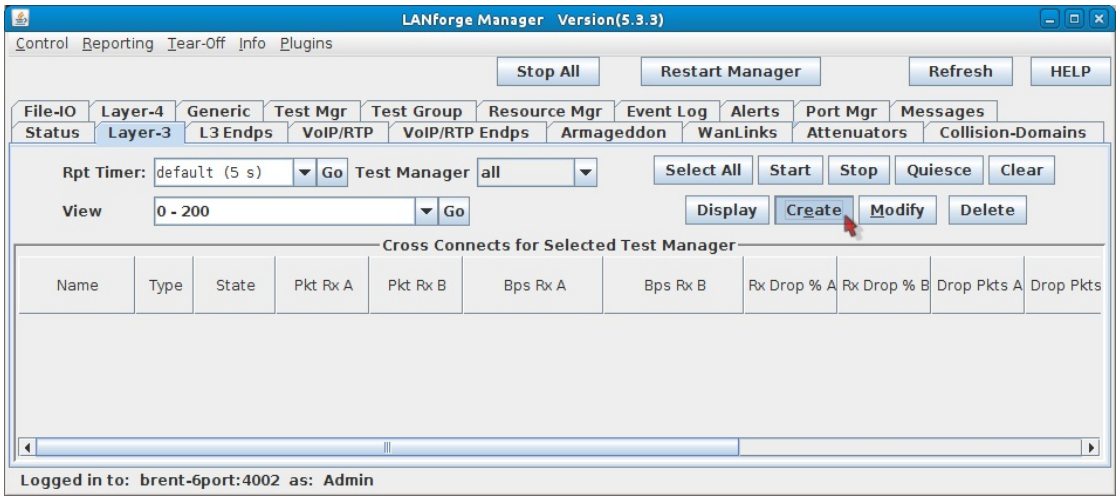
B. Assign an IP and Mask as necessary.



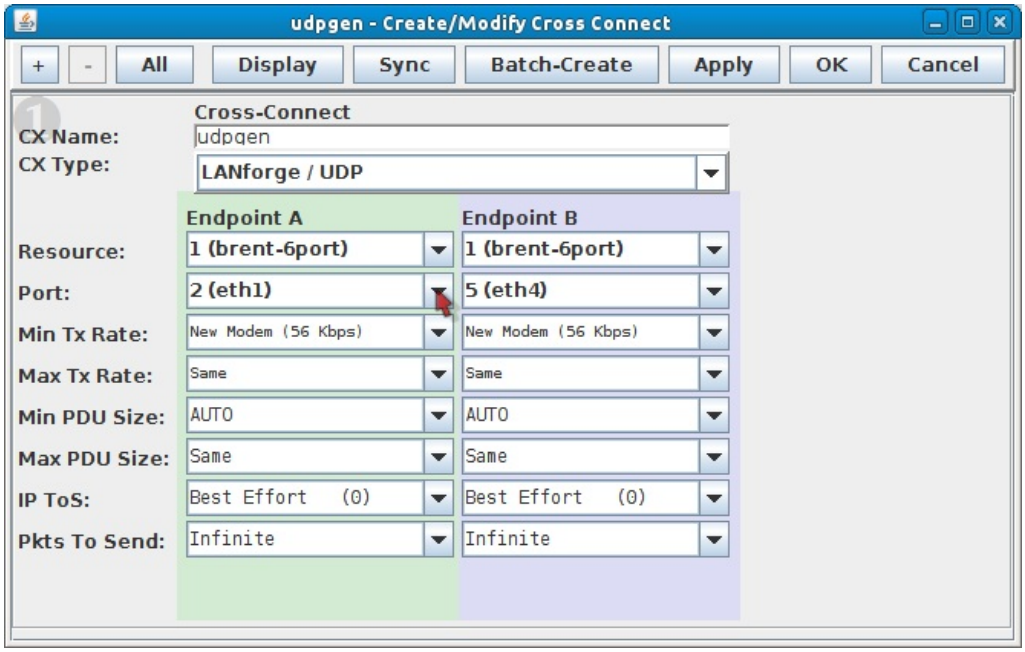
C. Click **OK**.

2. Configure the Layer-3 connection.

A. On the **Layer-3** tab, click **Create**.

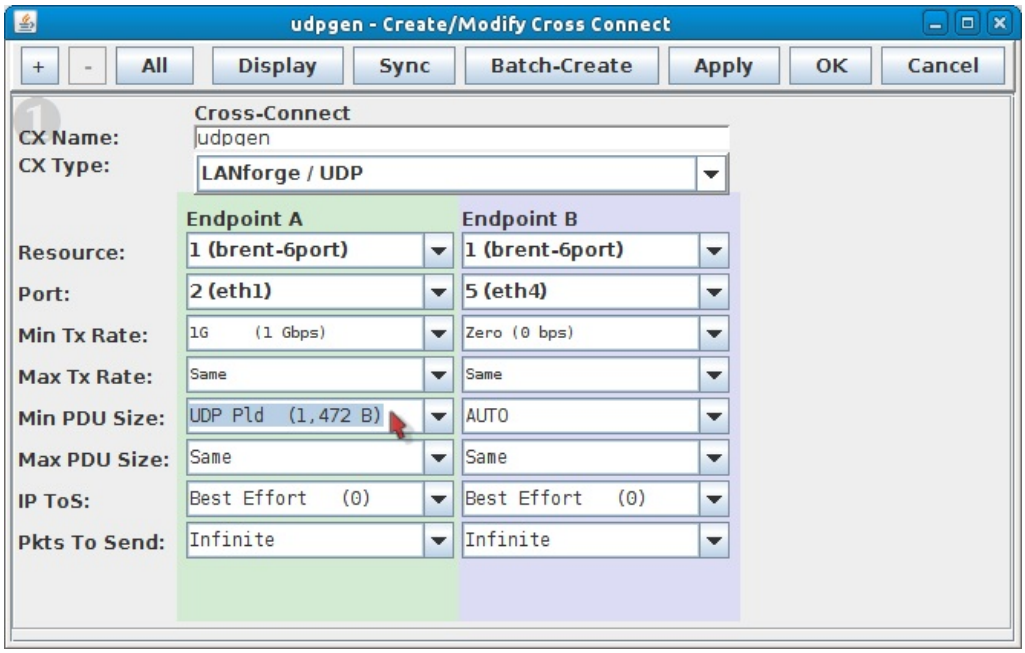


B. Assign port `eth1` to **Endpoint-A**.



A. You will not need to assign **Endpoint-B** because that will become unmanaged.

C. Configure the attributes in section 1:

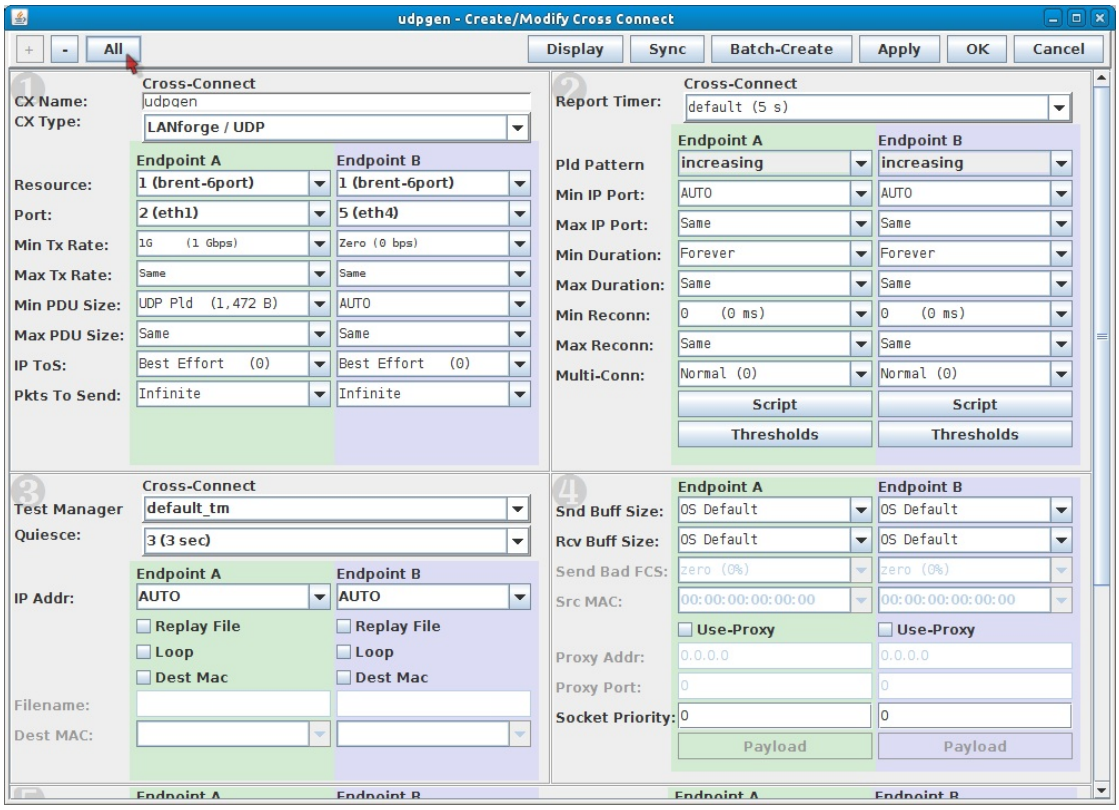


A. Endpoint-A Min Tx Rate: `1Gbps`

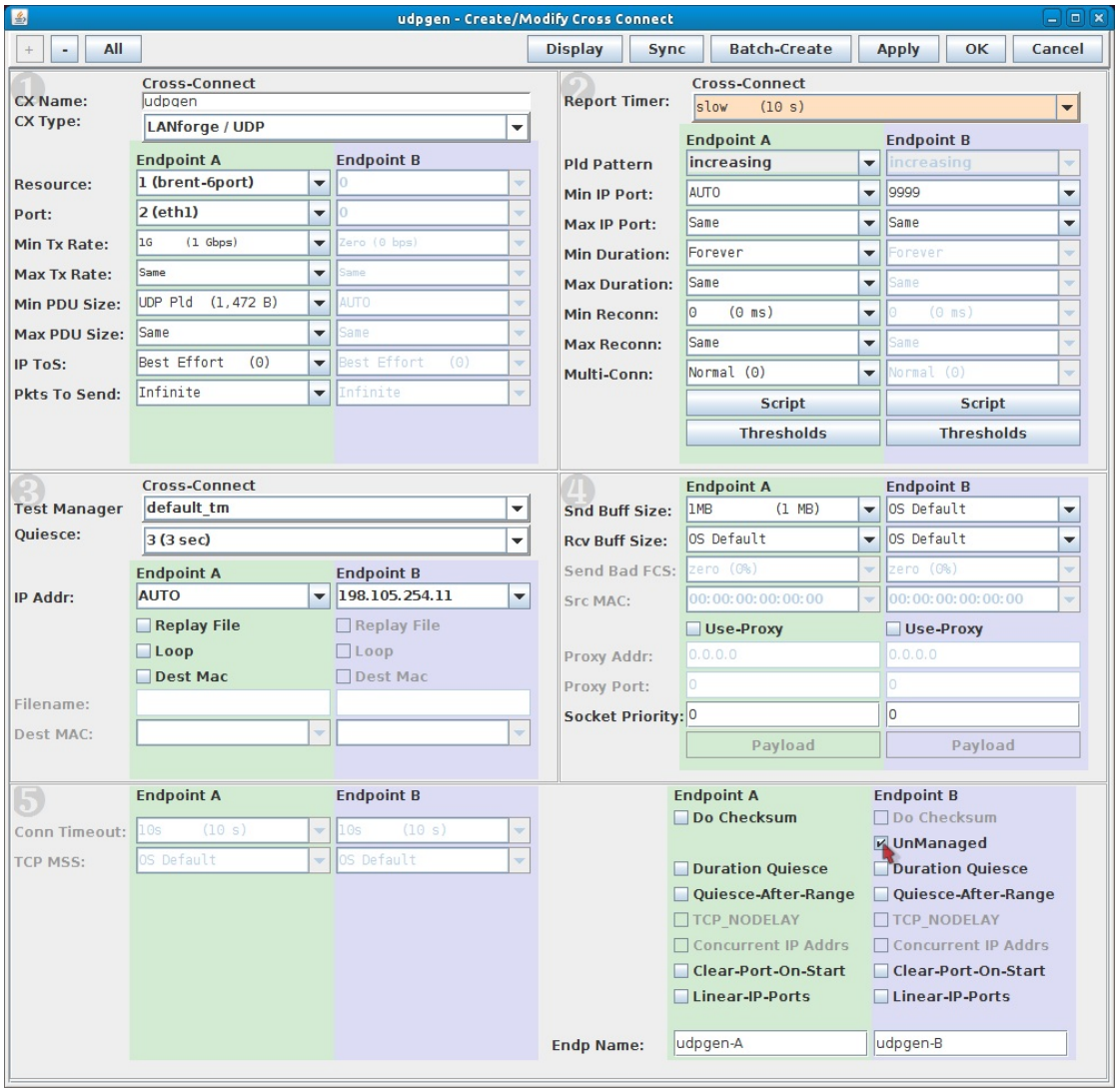
B. Endpoint-**B** Min Tx Rate: `Zero (0 bps)`

C. Endpoint-A Min PDU Size: `UDP Pld (1,472 B)`

D. Use the **All** button at the top to expand to the last detail level.



E. Configure the Layer-3 connection to the system under test (Endpoint-B) by following these steps:



A. In section 5, on the right side, Endpoint-B (blue), select **UnManaged**. This will gray-out most of the Endpoint-B options.

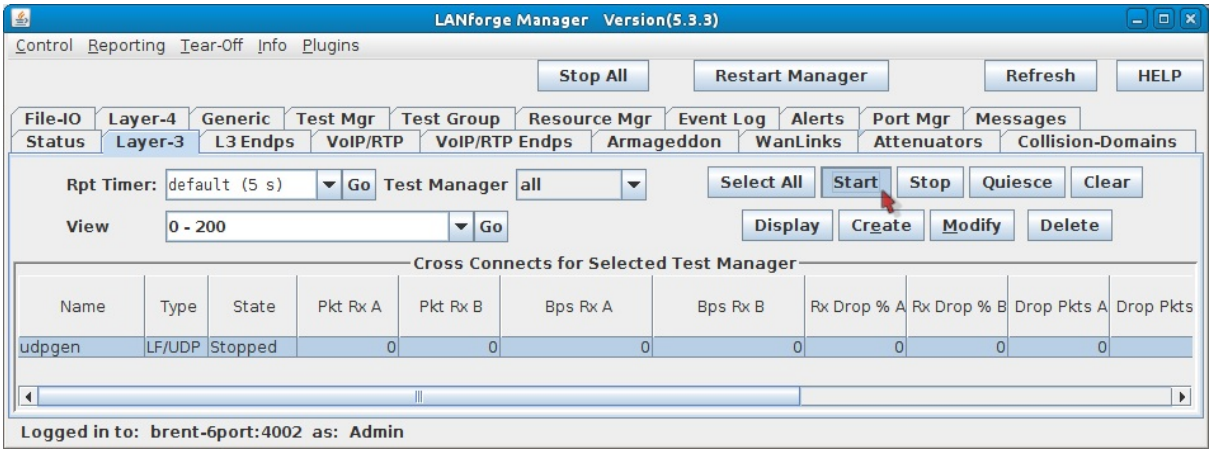
B. In section 2, set the Report Timer to `slow (10 s)`. Also, set the Endpoint-B Min IP port: `9999`. If you have a service under test this port should match, if not, this setting still needs to be present to generate valid traffic.

C. In section 3, set the Endpoint-B IP to the system under test IP address. Our example shows `198.105.254.11`.

D. In section 4, set the send buffer size (Snd Buff Size) to `1MB`.

E. Click **OK** at the top to commit the changes.

3. Start generating traffic.



A. In the **Layer-3** tab, select the connection `udpgen`.

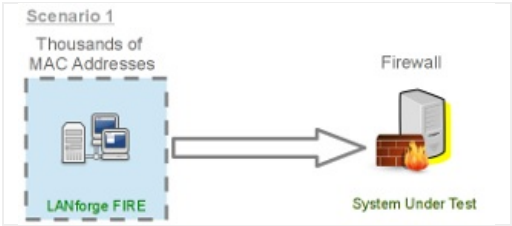
B. Click **Start**.

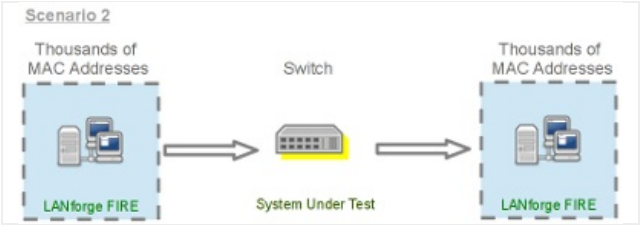
Armageddon UDP Traffic Generation with Random MAC Addresses.

Goal: Generate network traffic to a network device with one or many destination MAC addresses.

These scenarios are useful for testing switches and firewalls that have to handle UDP traffic from thousands of source MAC addresses and one or many destination MAC addresses. This cookbook covers two scenarios:

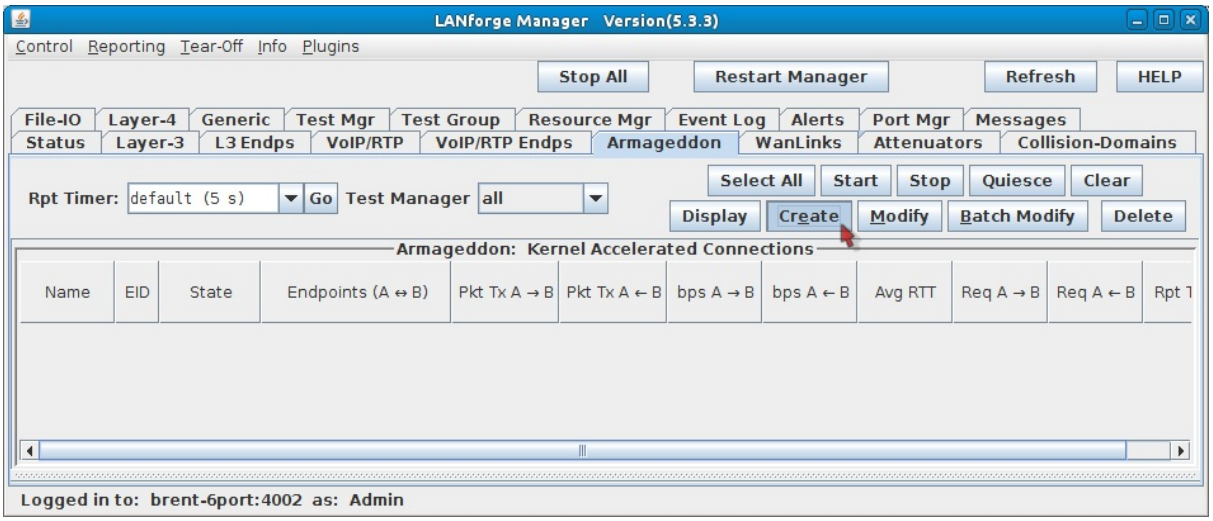
1. A single destination MAC address. (This would exercise a firewall or router.)
2. Thousands of destination MAC addresses. (This would exercise a switch by overflowing the device CAM table.)



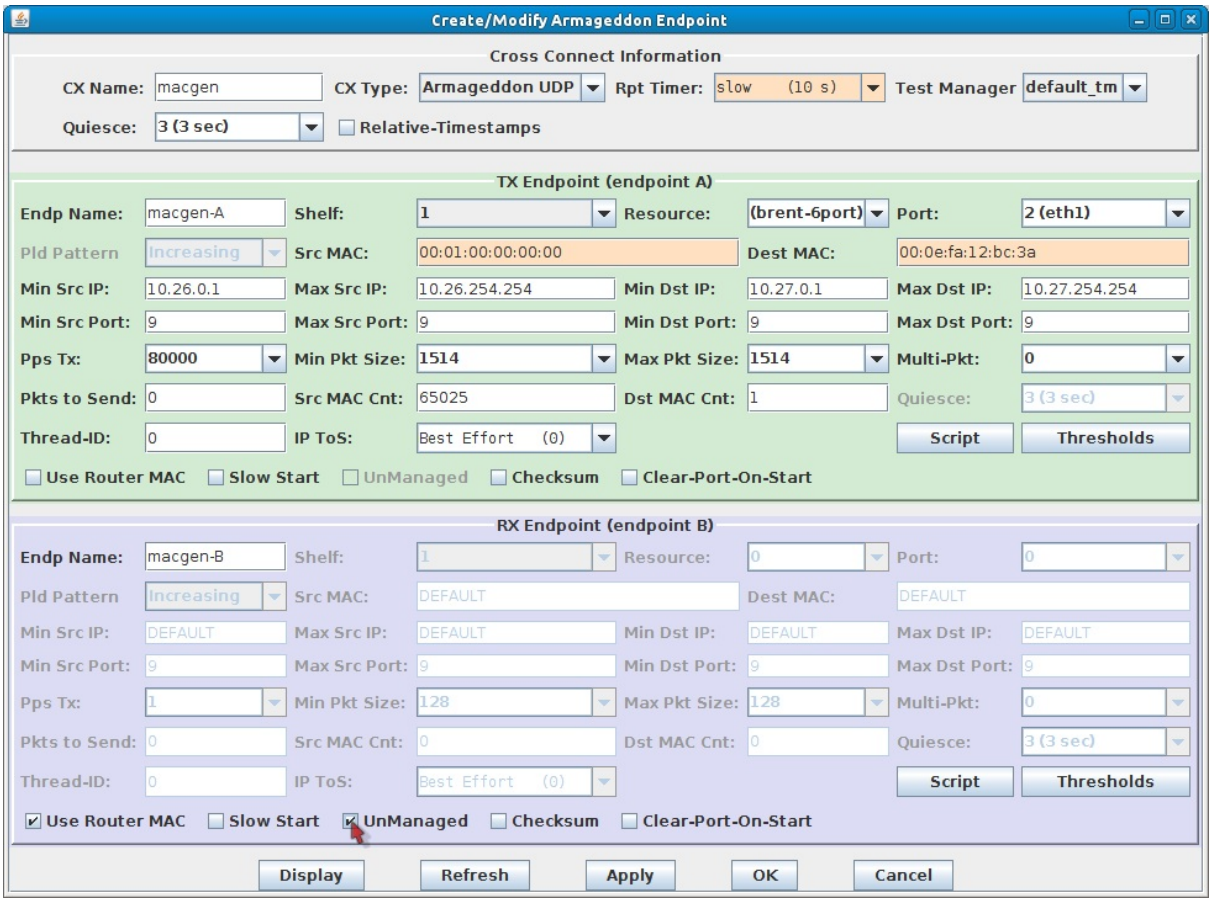


A one-sided traffic stream is used to send packets to a network device under test when round-trip reporting is not required.

1. On the **Armageddon** tab, click **Create**.



2. **Firewall Scenario:** Configure the Armageddon connection with the following values:



A. In the TX Endpoint (green box):

   A. The connection name for this example is `macgen`.

   B. Src MAC: `00:01:00:00:00:00`. This is the starting point for emulated remote MAC addresses.

   C. Dest MAC: `00:0e:fa:12:bc:3a`. Use the destination MAC address of your system under test.

   D. Min Src IP: `10.26.0.1`.

   E. Max Src IP: `10.26.254.254`. This emulates about 65,000 remote hosts.

   F. Min Dst IP: `10.27.0.1`.

   G. Max Dst IP: `10.27.254.254`. This emulates about 65,000 destination addresses.

   H. Pps Tx: `80,000`. This is a 1Gbps packet rate.

   I. Min Pkt Size: `1514`.

   J. Max Pkt Size: `1514`.

   K. Src MAC Cnt: `65,025`. This emulates 65,025 remote devices.

   L. Dst MAC Cnt: `1`.

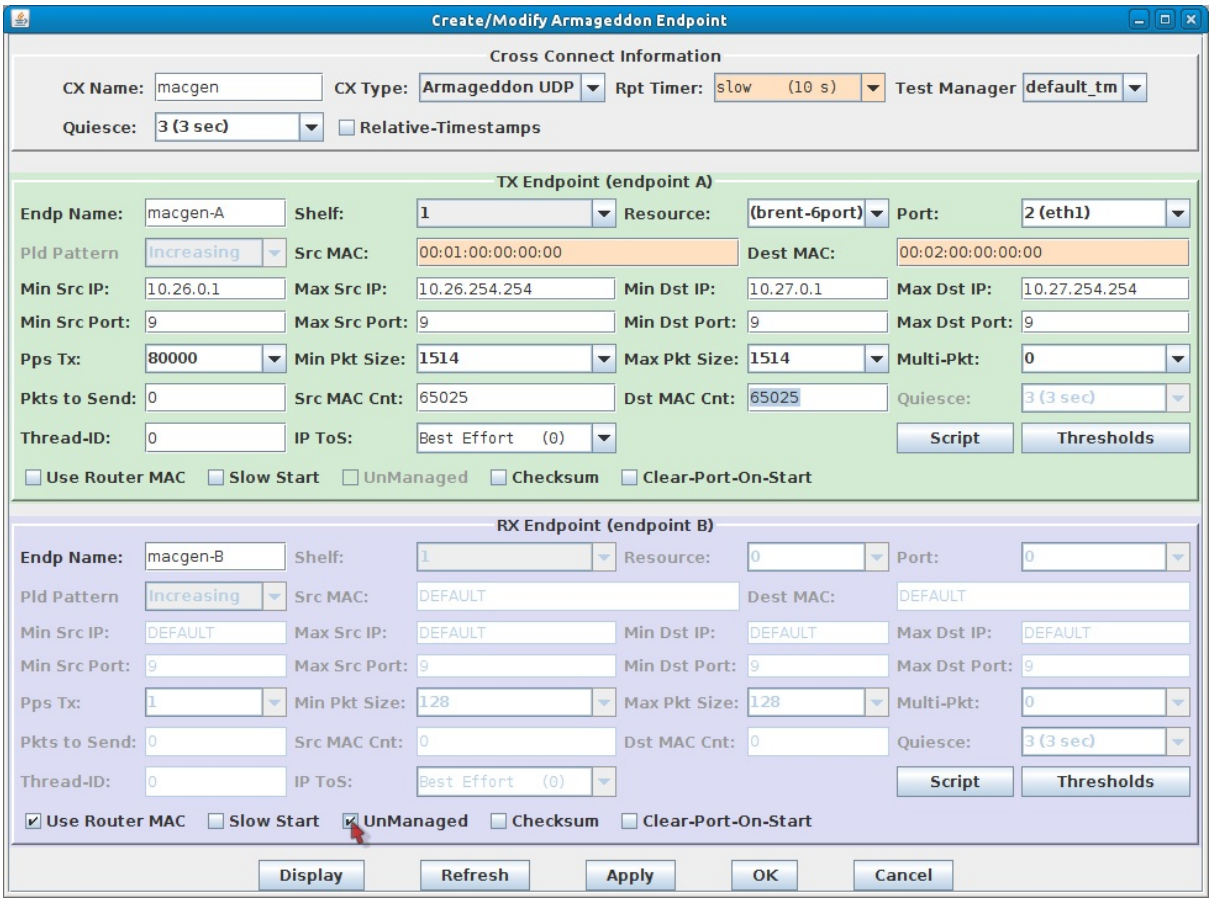
   M. **Deselect** `Use Router MAC`.

B. In the RX Endpoint (blue box):

   A. **Select** `UnManaged`. This allows us to fire and forget the packets.

C. Click **OK** to commit the configuration.

3. **Switch Scenario:** Configure the Armageddon connection with the following values:



A. In the TX Endpoint (green box):

   A. The connection name for this example is `macgen`.

   B. Src MAC: `00:01:00:00:00:00`. This is the starting point for emulated source MAC addresses.

   C. Dest MAC: `00:02:00:00:00:00`. This is the starting point for emulated destination MAC addresses. This address is 4.2 billion addresses after the starting source MAC address.

   D. Min Src IP: `10.26.0.1`.

   E. Max Src IP: `10.26.254.254`. This emulates about 65,000 remote hosts.

   F. Min Dst IP: `10.27.0.1`.

   G. Max Dst IP: `10.27.254.254`. This emulates about 65,000 destination addresses.

   H. Pps Tx: `80,000`. This is a 1Gbps packet rate.

   I. Min Pkt Size: `1514`.

   J. Max Pkt Size: `1514`.

   K. Src MAC Cnt: `65,025`. This emulates 65,025 source devices.

   L. Dst MAC Cnt: `65,025`. This emulates 65,025 destination devices.

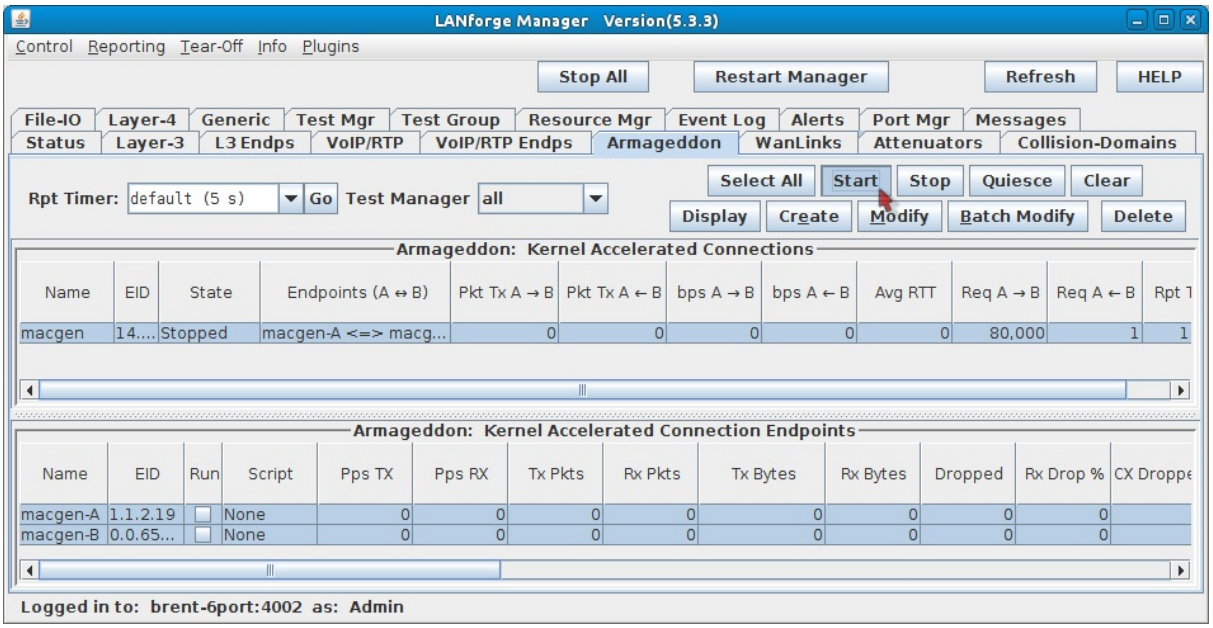
   M. **Deselect** `Use Router MAC`.

B. In the RX Endpoint (blue box):

   A. **Select** `UnManaged`. This allows us to fire and forget the packets.

C. Click **OK** to commit the configuration.

4. Start traffic generation:

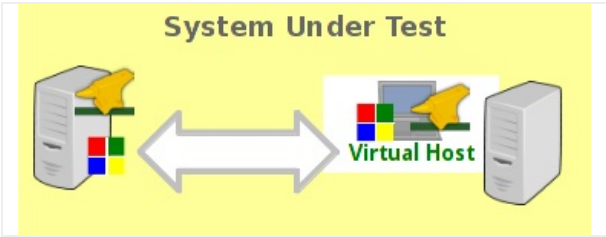


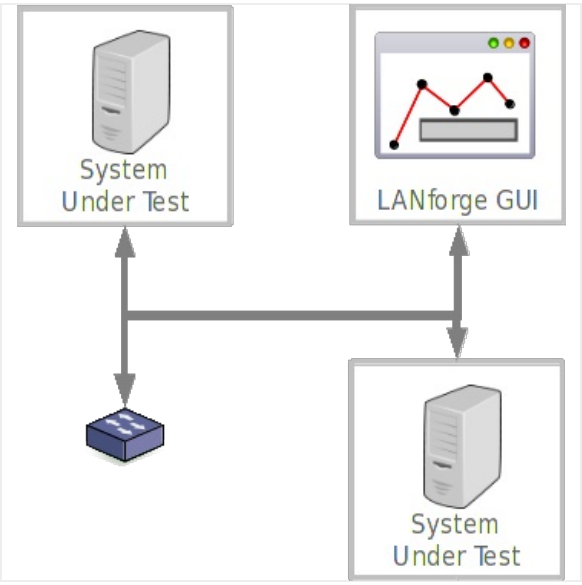
A. In the **Armageddon** tab, click the **Start** button. Traffic will begin.

Layer 3 Testing on Windows

Goal: Test layer-3 connection throughput between two Windows hosts.

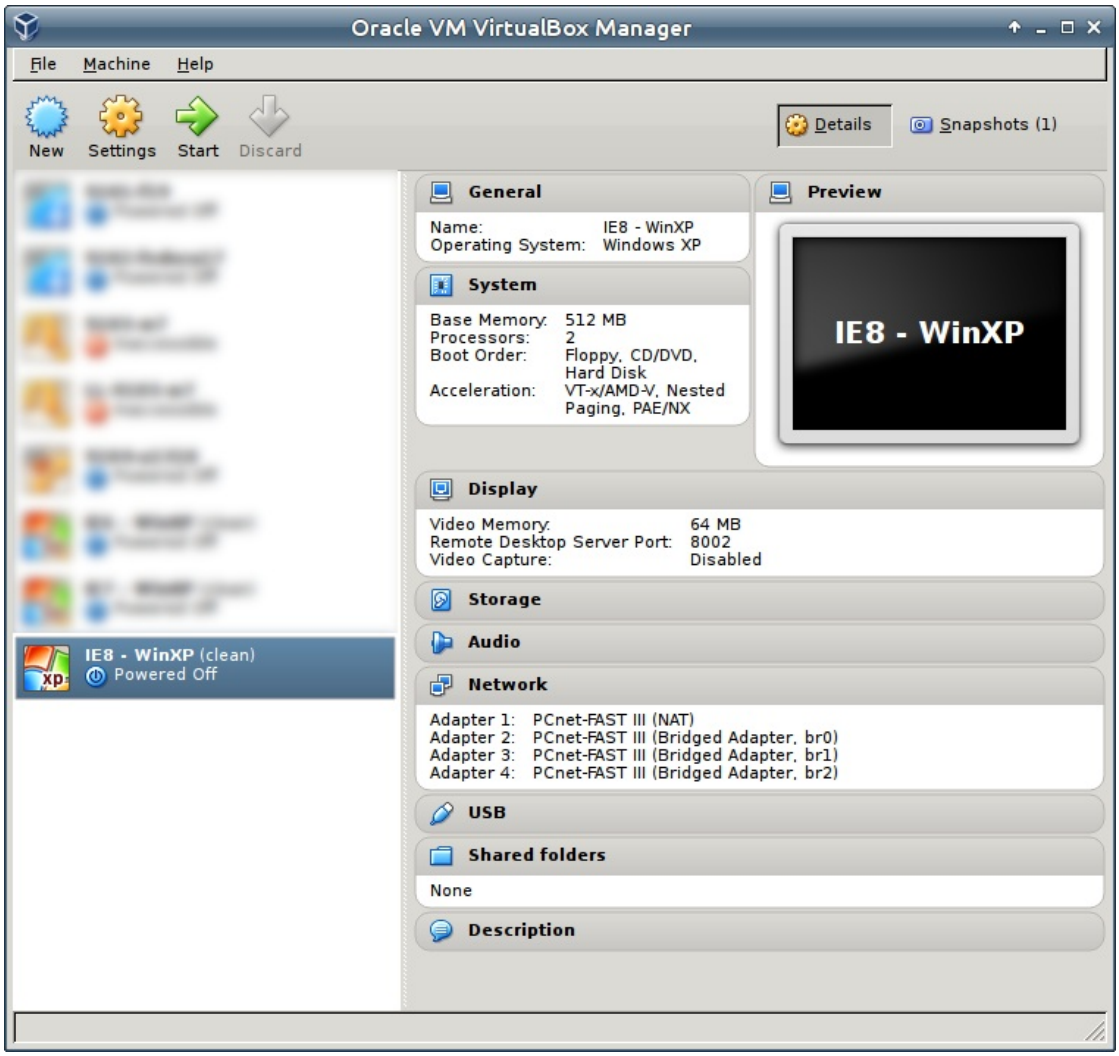
This scenario involves installation on one physical Windows 7 workstation with three network ports and a Windows XP virtual host with four network ports running within VirtualBox. We will create a pair of layer three connections and see test results.





1. Configure Virtual Guest Windows machine
   A. **Note:** LANforge installations on Windows are not able to change the IP of ports. Please do this before running LANforge.

B. Set up four network ports on the virtual guest. This example shows three ports connected to the host machine over bridge interfaces.



C. Set the IPs for the various ports using the Windows Network Settings control panel.

   A. **Local Area Connection**: `DHCP` This is the default VirtualBox NAT port. We'll leave this one as a backup port for safety sake.

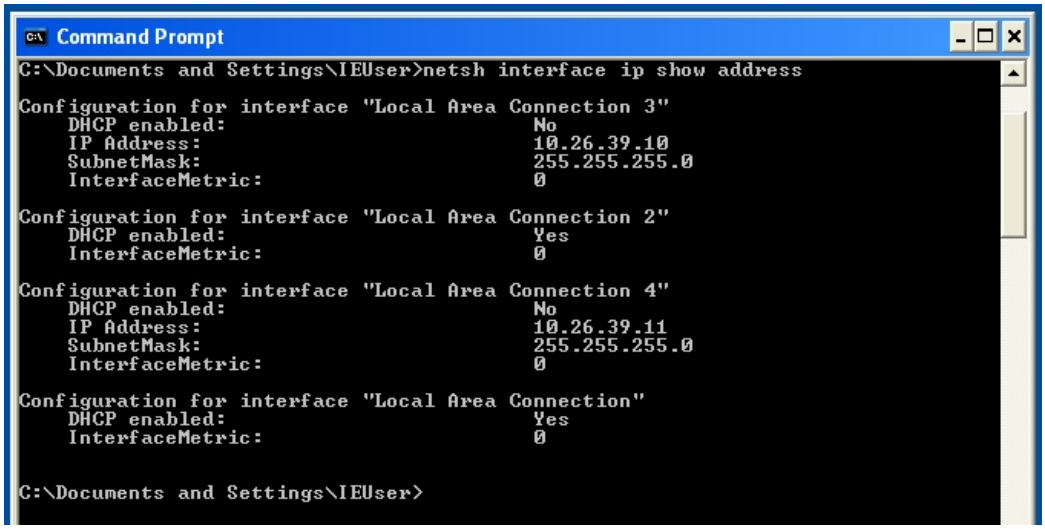
   B. **Local Area Connection 2**: `192.168.1.201`, we will use this for normal LAN and LANforge communications

   C. **Local Area Connection 3**: `10.26.39.1`, for LANforge use.

   D. **Local Area Connection 4**: `10.26.39.10`, for LANforge use.

   For more information see Windows IP Address

D. Review the IPs for the various ports using `netsh`.



E. Set up network ports on the Windows workstation. The configuration includes one physical port on the motherboard and a four-port Intel PCIe card.

   A. **Local Area Connection**: `192.168.100.39`, we will use this for normal LAN and LANforge communications

   B. **Local Area Connection 2**: `10.26.39.2`, for LANforge use.

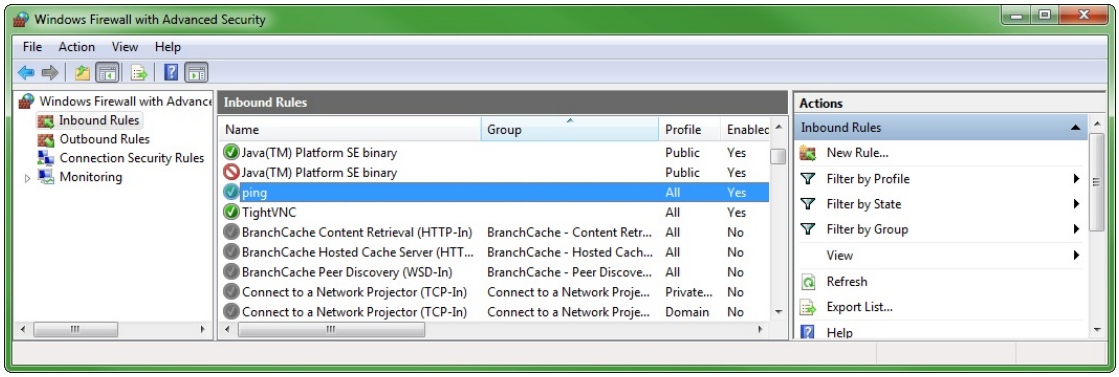
   C. **Local Area Connection 3**: `10.26.39.3`, for LANforge use.

   D. **Local Area Connection 4**: `10.26.39.4`, for LANforge use.

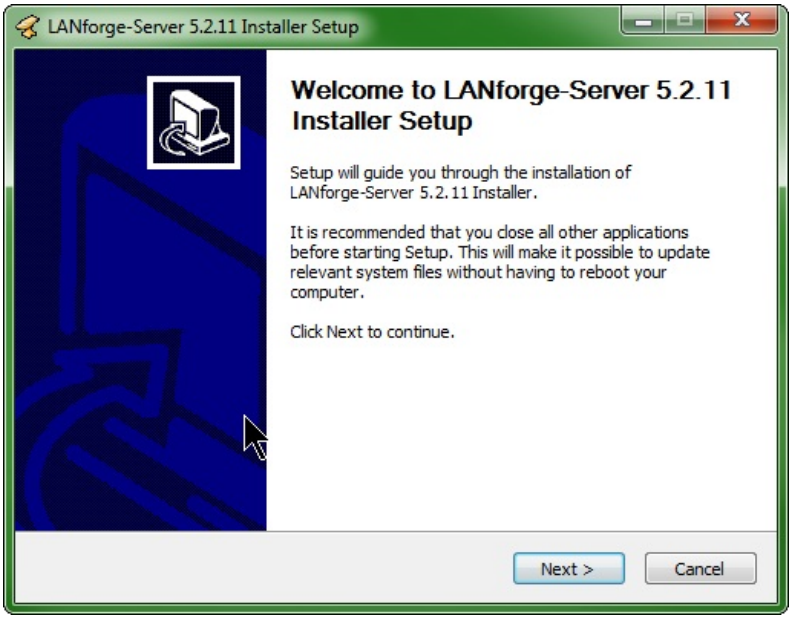
F. Make sure your LANforge client GUI can ping the IP of the virtual guest management port.

G. Make sure your LANforge Manager can ping the IPs of the virtual guest management ports.

H. If necessary, configure the Windows firewall to allow ICMP packets. Or disable the Windows firewall.

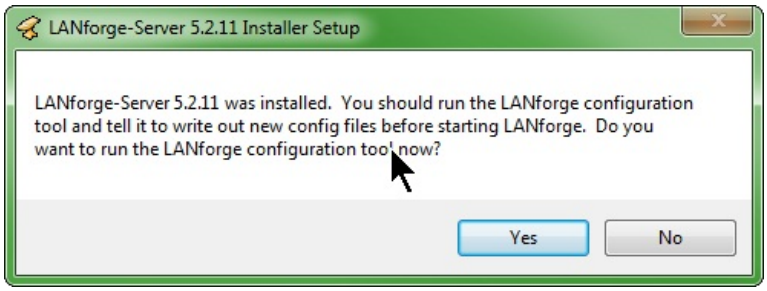


2. Install LANforge Server on the Windows workstation



A. Make sure you install WinPCAP if it is not already installed.

B. At the end of the LANforge Server install, launch the LANforge Configuration Utility

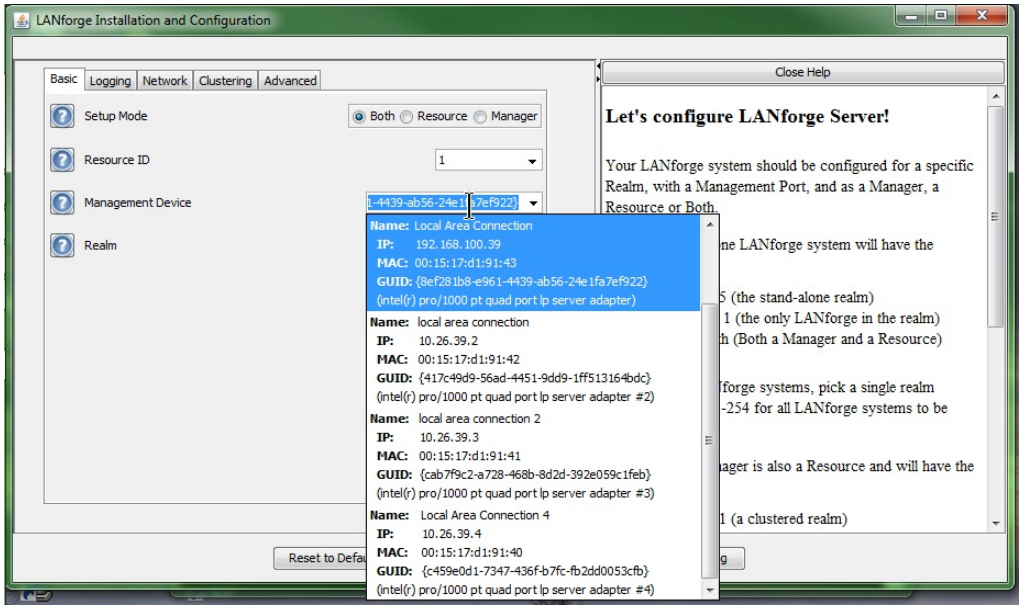


A. Set the **Realm** to `1`

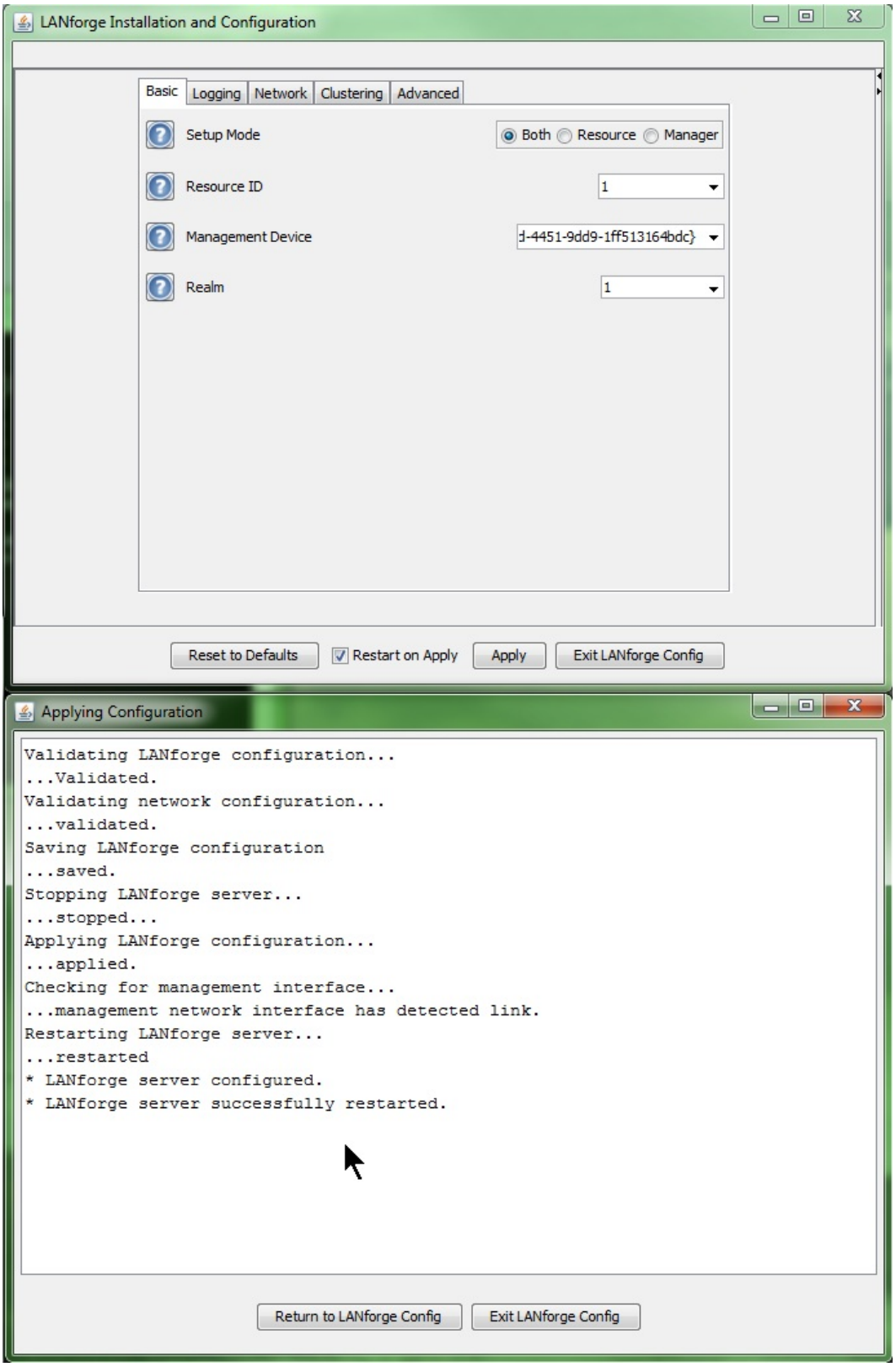
B. Set the **Resource** to `1`

C. Set the **Mode** to `Both`

D. Set the Management port to the local LAN addressed port.



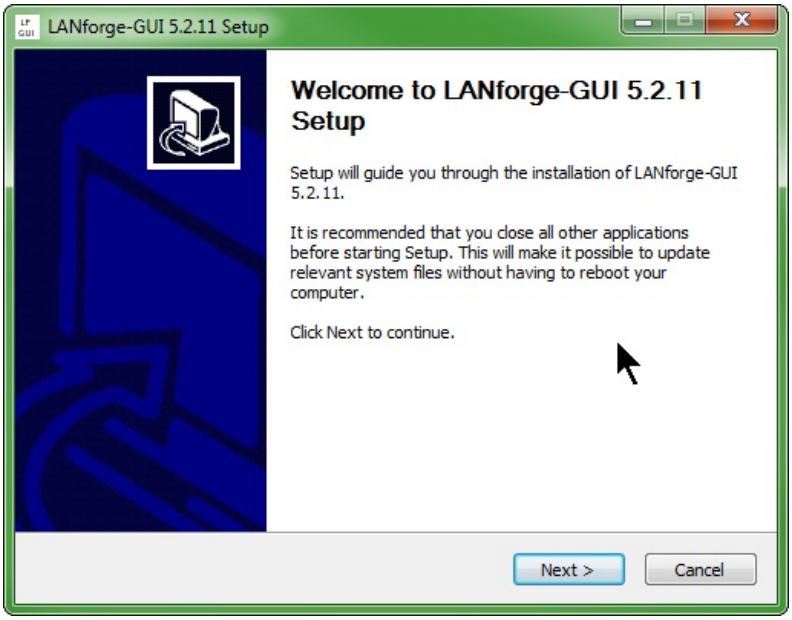
C. Apply the configuration and LANforge Server will start in Manager mode.



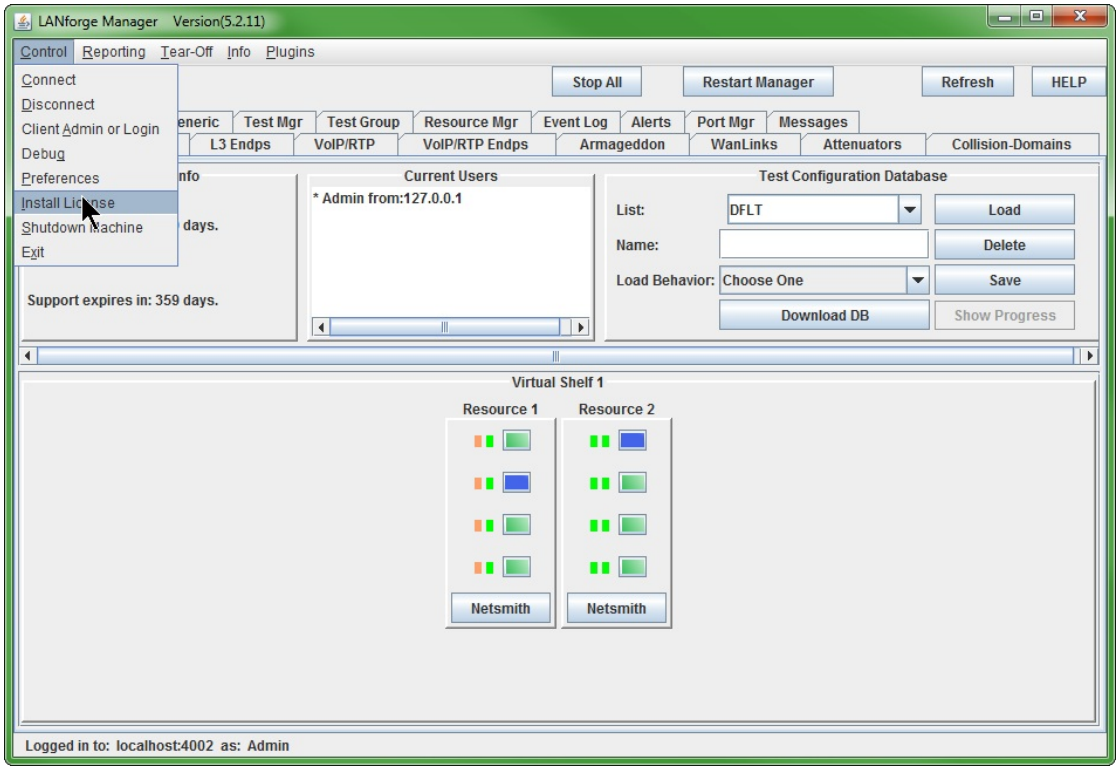
D. Click the **Exit LANforge Config** button. Leave the LANforge Server running (you may minimize the DOS windows.)

E. Desktop icons allow you to start and configure LANforge Server later.

3. Install the LANforge GUI on the Windows workstation.



A. Don't forget to install the licences.



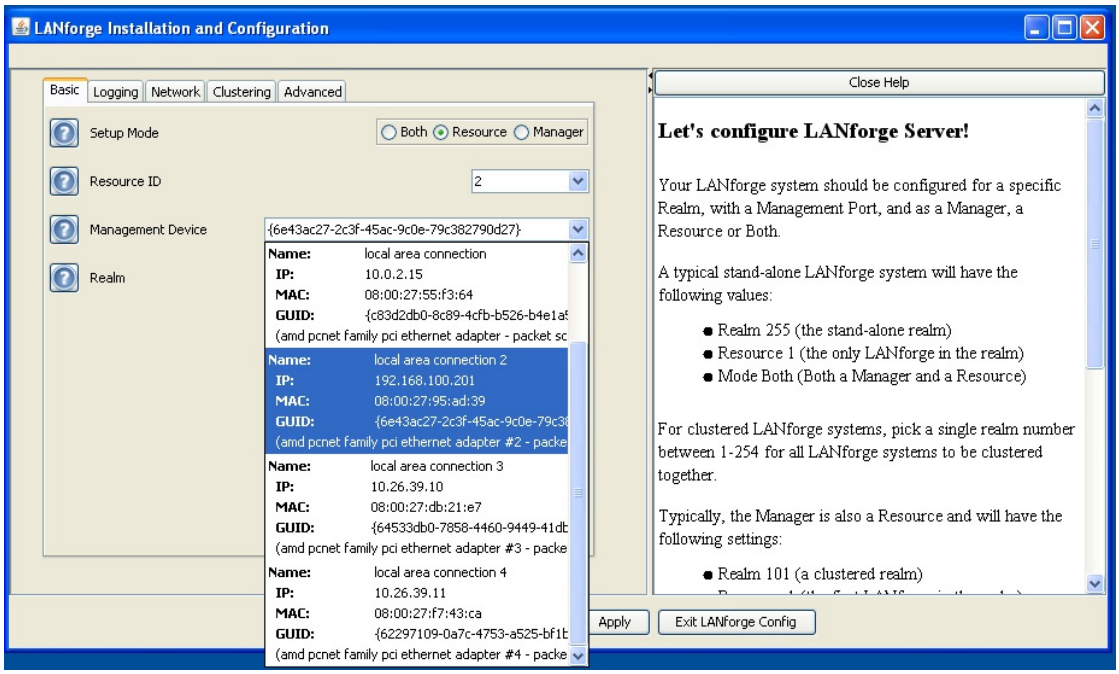
4. Install LANforge Server on the virtual guest

A. Make sure you install WinPCAP if it is not already installed.

B. At the end of the system install, launch the LANforge Configuration Utility

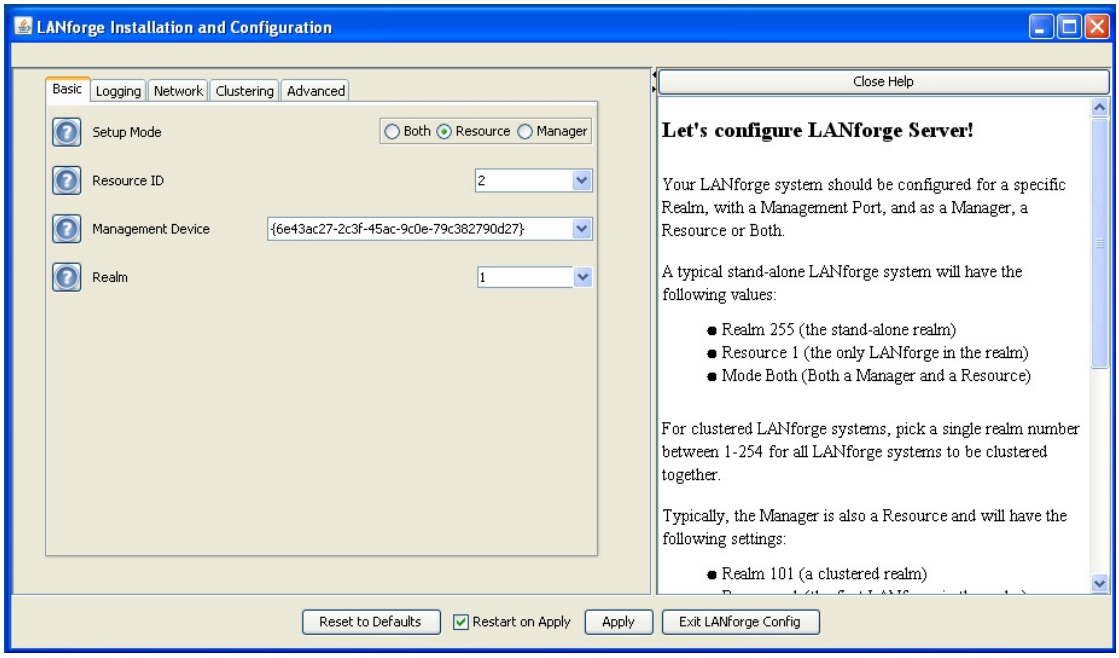
C. Set the **Mode** to `Resource`

D. Set the Management port to the local LAN addressed port.

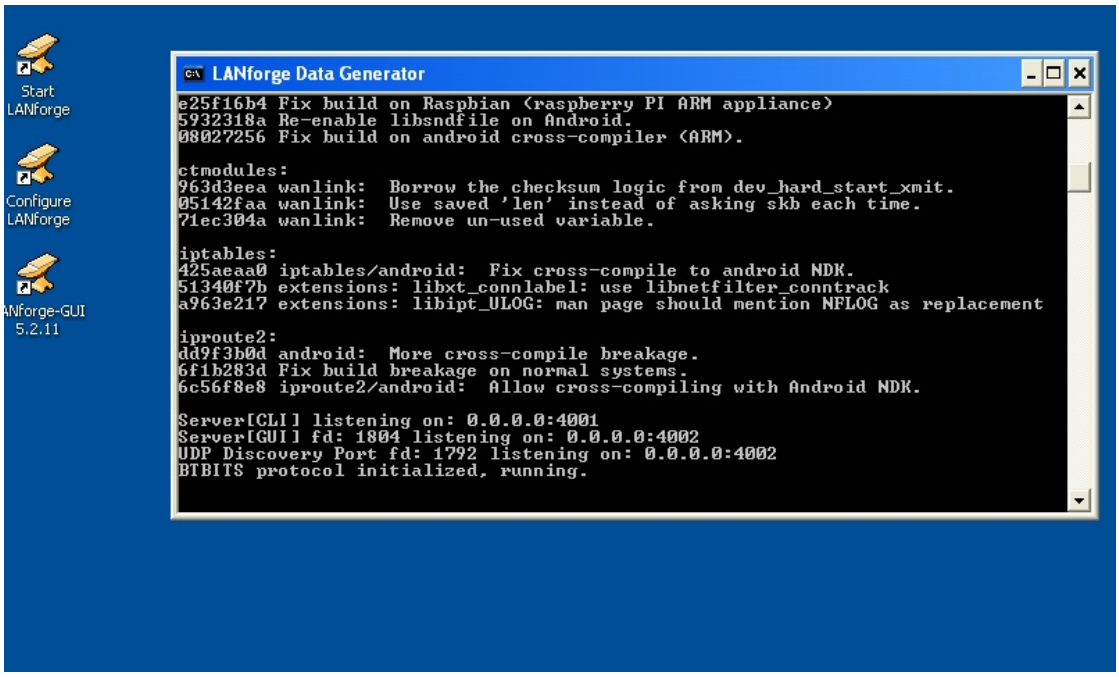


E. Set the **Realm** to `1`

F. Set the **Resource** to `2`

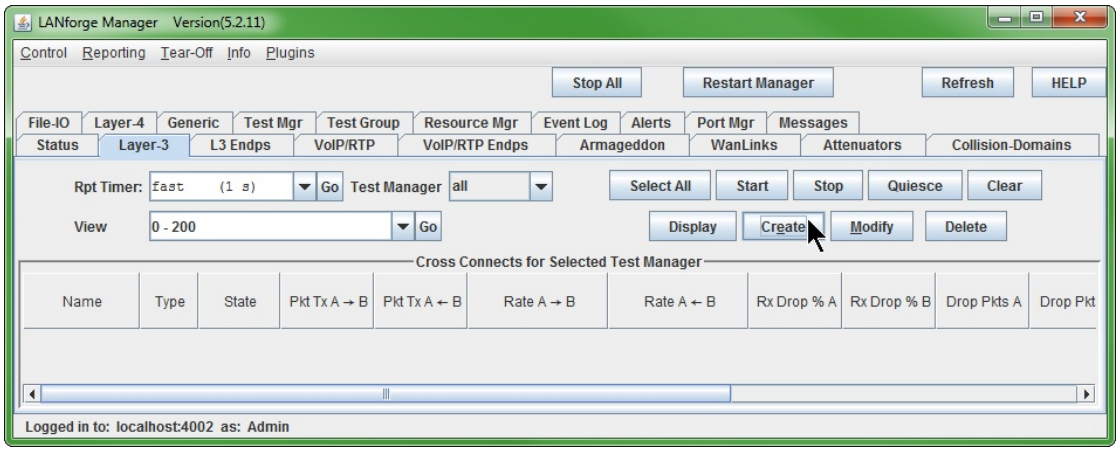


G. Apply the configuration and LANforge Server will start in resource mode.

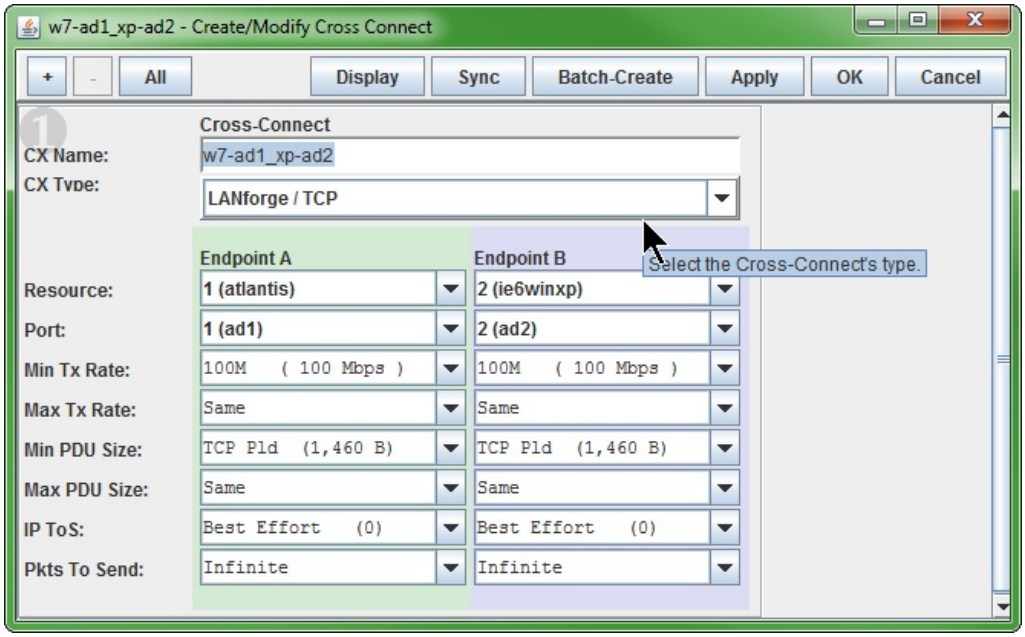


5. Create Layer 3 connections

A. In the **Layer-3** tab, click **Create**



B. In the `Level-1` box, create the first cross connect:



A. Name the **Cross Connect** to `w7-ad1_xp-ad2`

B. The **CX Type** is `LANforge / TCP`

C. Set the **Endpoint A Resource** to `Windows 7`

D. The **Endpoint A Port** to the first 10.x addressed port, `ad1`

E. Set the **Endpoint A Min PDU Size** to `TCP (1460 B)`

F. Set the **Endpoint A Min Tx Rate** to `100Mbit`. This is a limit of the XP guest.

G. **Endpoint B Resource:** `XP Guest`

H. **Endpoint B Port:** `ad2`

I. **Endpoint B Min Tx Rate:** `100Mbit`

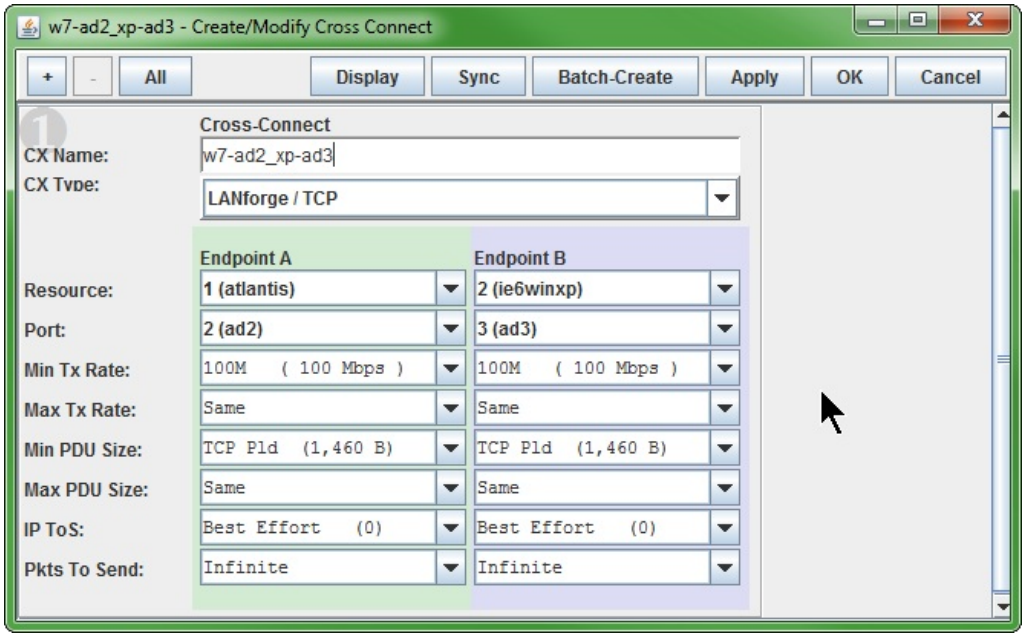
J. Set the **Endpoint B Min PDU Size** to `TCP (1460 B)`

K. Set the **Endpoint B Min Tx Rate** to `100Mbit`. This is a limit of the XP guest.

L. Click **OK** to commit those settings

C. For the second cross connect, click **Create** in the **Layer-3** tab.

D. In the `Level-1` box, create the second cross connect:



A. Name the **Cross Connect** to `w7-ad2_xp-ad3`

B. The **CX Type** is `LANforge / TCP`

C. Set the **Endpoint A Resource** to `Windows 7`

D. The **Endpoint A Port** will be the next free 10.x port, `ad2`

E. Set the **Endpoint A Min PDU Size** to `TCP (1460 B)`

F. Set the **Endpoint A Min Tx Rate** to `100Mbit`. This is a limit of the XP guest.

G. The opposite end **Endpoint B Resource** is the `XP Guest`

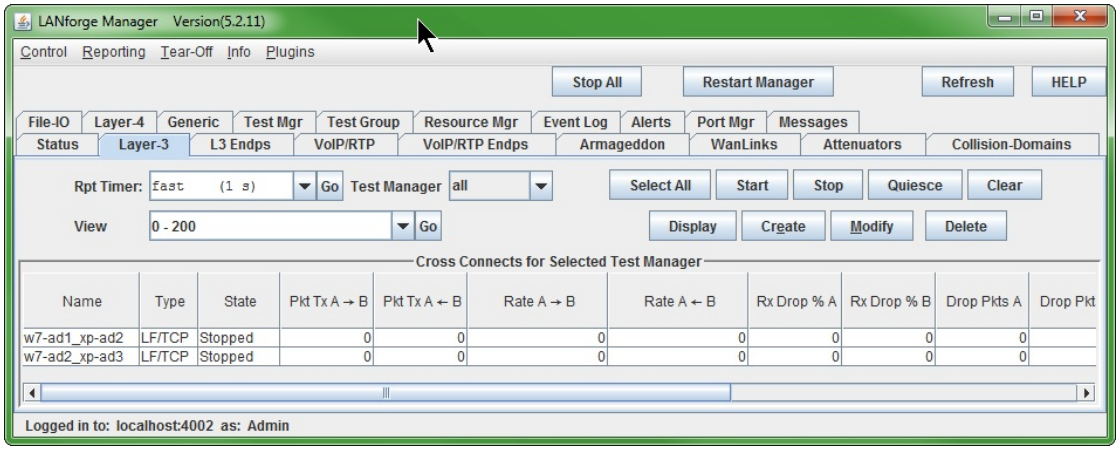
H. Set the **Endpoint B Port** to `ad3`

I. Set the **Endpoint B Min PDU Size** to `TCP (1460 B)`

J. Set the **Endpoint B Min Tx Rate** to `100Mbit`. This is a limit of the XP guest.

K. Click **OK** to commit those settings

E. In the **Layer-3** tab, you will now see your two cross-connects:

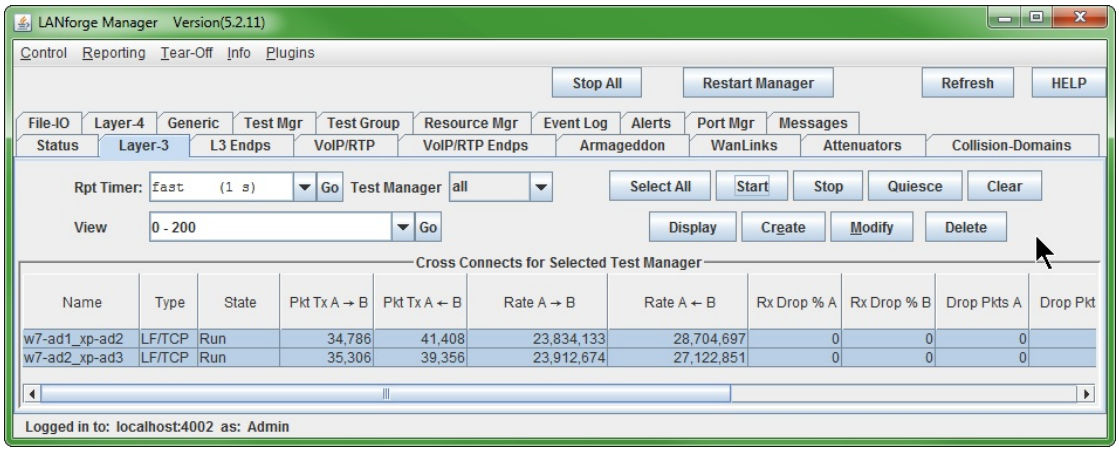


6. Start Layer 3 connections

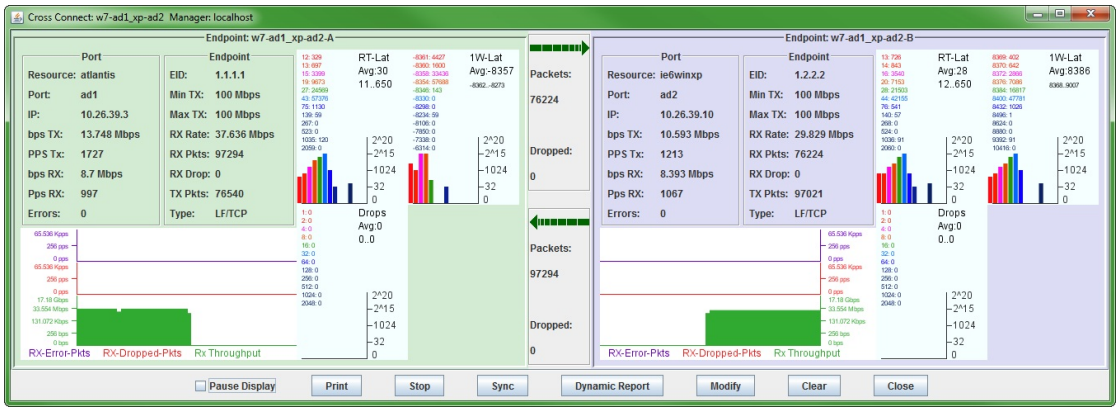
A. Highlight the connections

B. Click **Start**

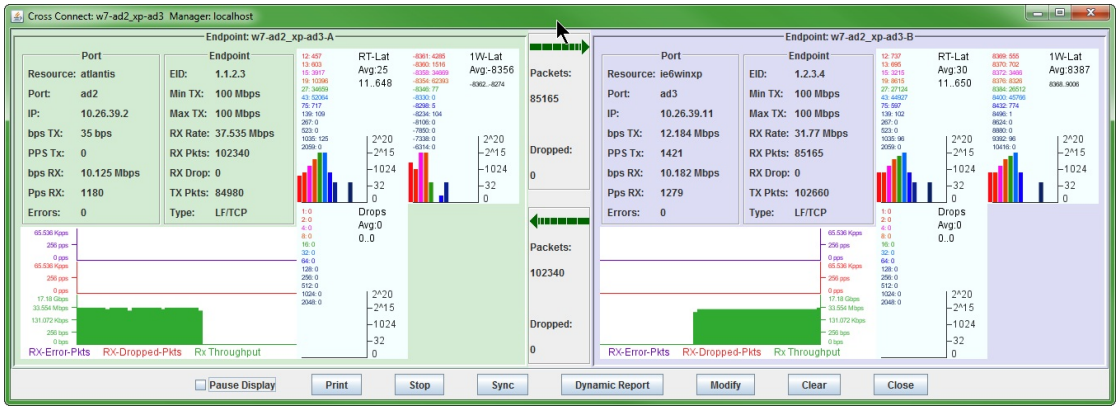
C. You will see data transferring along the cross connects.



D. Highlight both connections and click **Display** to see the connection statistics for each. Here is the first:



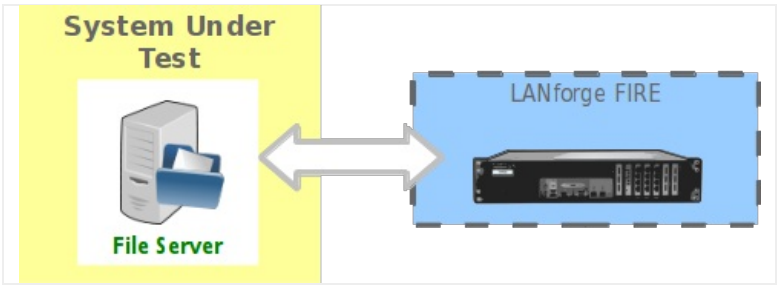
E. Here is the second:

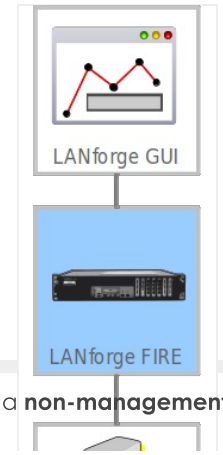


LANforge File-IO with CIFS and NFS

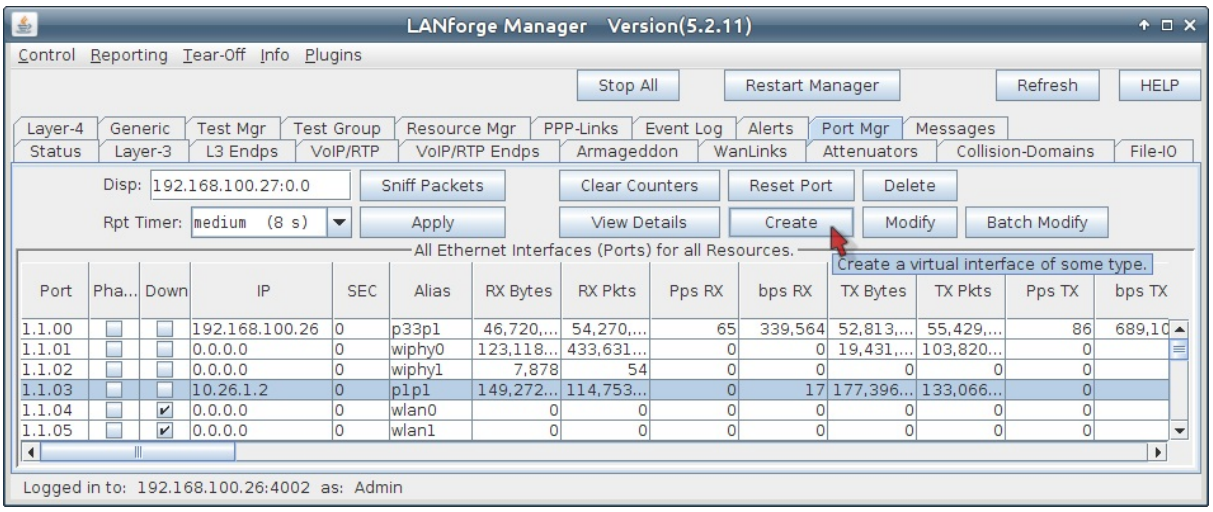
Goal: Create a series of MAC-VLAN based clients to emulate CIFS and NFS traffic.

This cookbook connects a LANforge system to a file server with CIFS and NFS shares available. The file server in this example will be `10.26.1.3`. It will be sharing `smb://10.26.1.3/fileio` and `10.26.1.3:/home/fileio`. We will create ten readers and ten writers for each file sharing protocol. This demonstrates using the **Batch Create** tool in the **FileIO** tab.

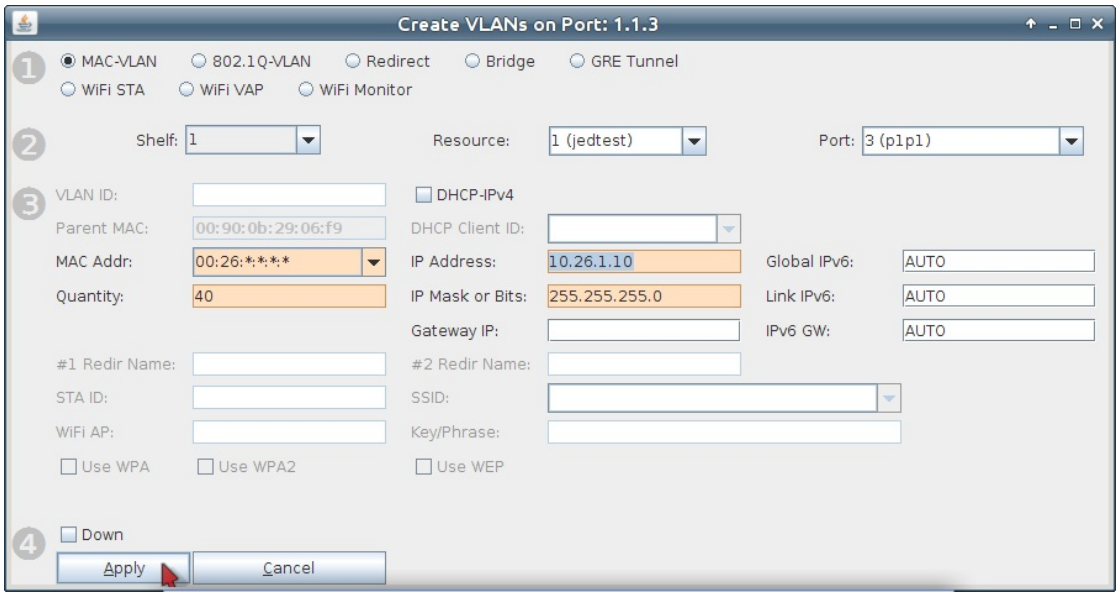




1. Create initial MAC VLANS for our emulated endpoints. In the **Port Mgr** tab, highlight a a **non-management** port on your LANforge FIRE system and click **Create.**

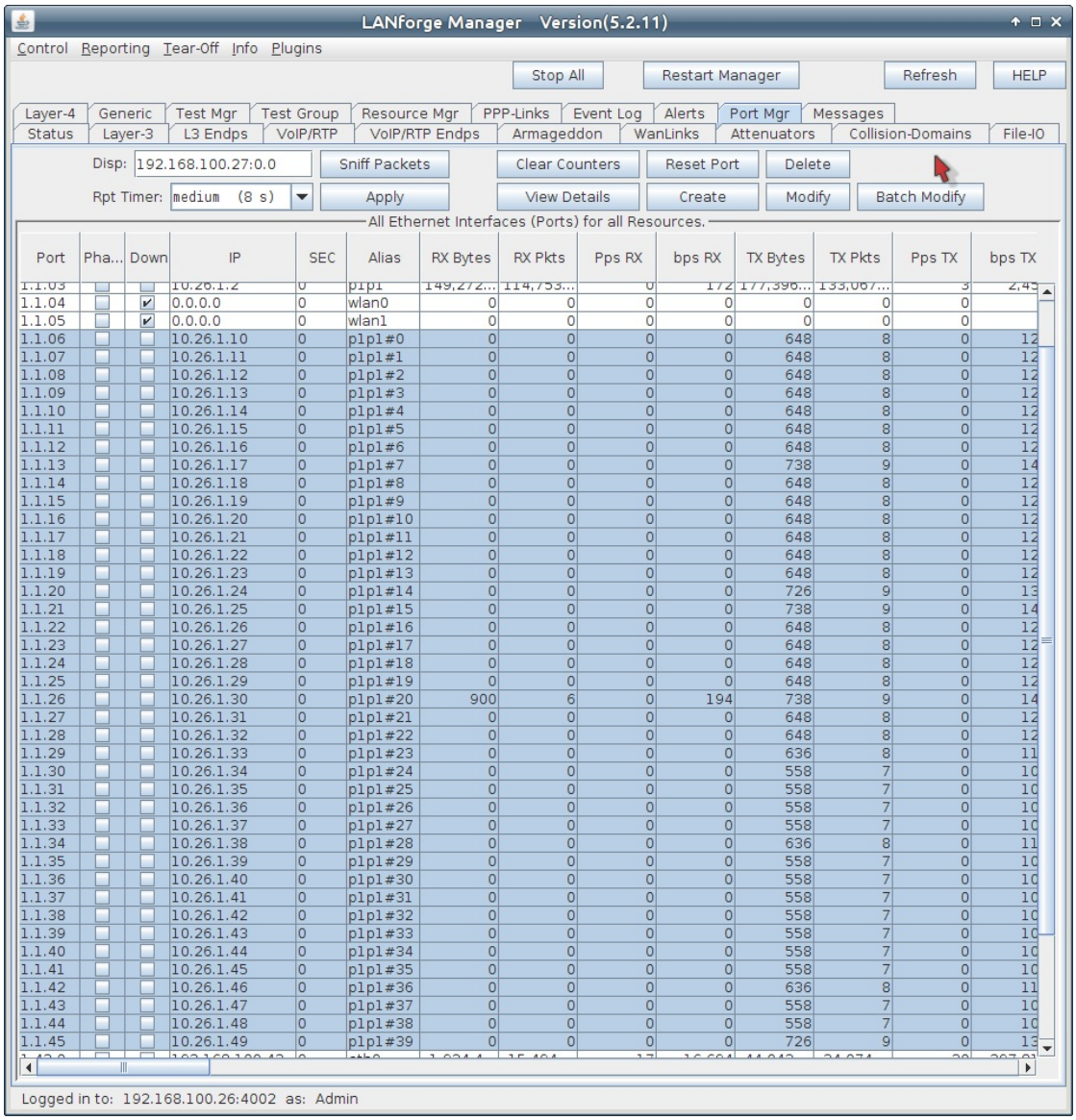


   A. Create 40 MAC VLANS with IPs starting at `10.26.1.10`.



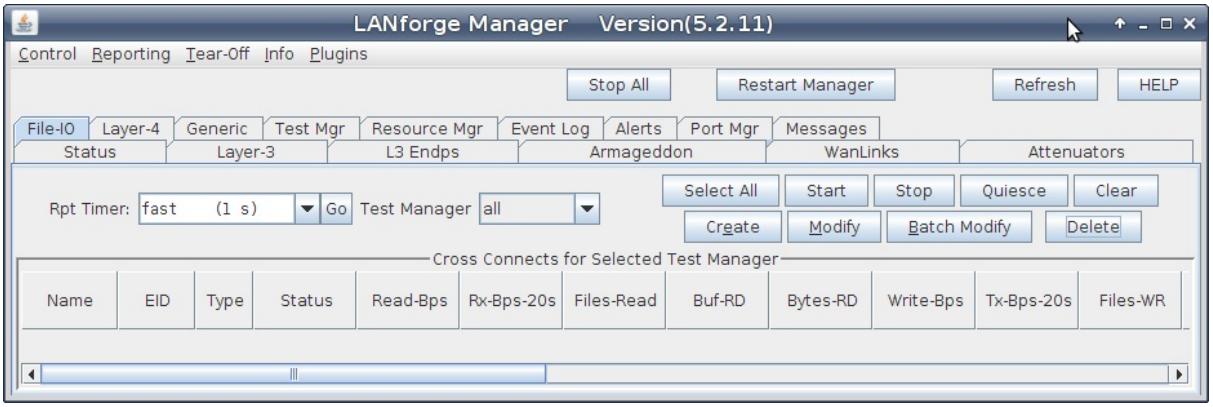
   A. Select **MAC-VLAN**
   B. Choose either `*:*:*:*:*:*` (random) or select a starting MAC address, like `00:26:*:*:*:*`
   C. Enter `10.26.1.10` and `255.255.255.0`for the IP and netmask.
   D. Set **Quantity** to `40.`
   E. Click **Apply** to create the MAC VLANs.
   F. Click **Cancel** to close the **Create** window.

B. See the 40 MAC VLANs in the **Ports** tab.

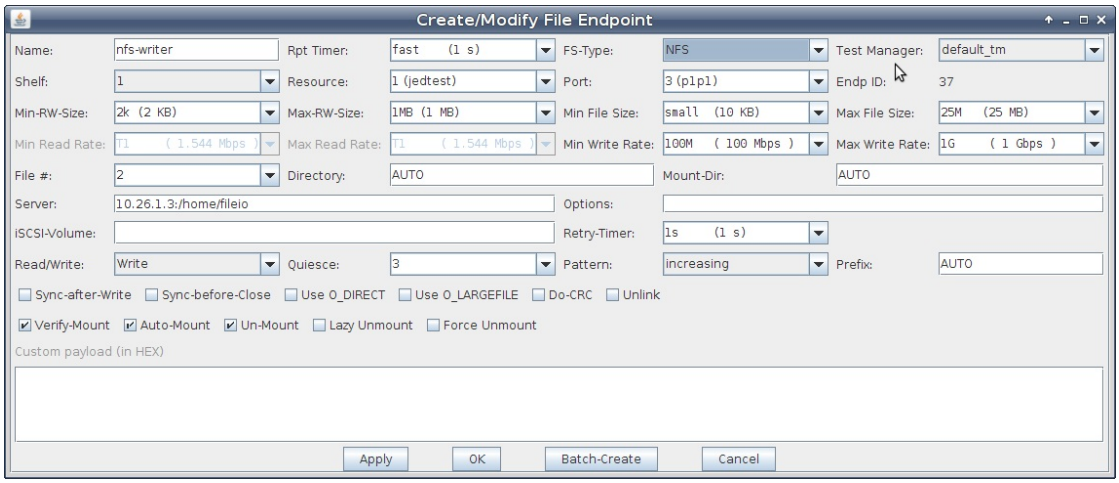


For more information see GUI Users Guide

2. Create your first FileIO NFS Endpoint. In the **FileIO** tab, click **Create.**



A. Use the following settings to create a NFS reader writer endpoint.



A. Enter `nfs-writer` for the name.

B. Select `NFS` for **FS-Type**

C. Use the first MAC-VLAN `p1p1#0` for **Port**

D. Select `2KB` for the **Min-RW Size**

E. Choose `1MB` for the **Max-RW Size**

F. For **Min File Size** choose `10KB`

G. Then for **Max File Size** choose `25MB`

H. The **Min Write Rate** is the minimum inbound line rate, start at `100Mbps`

I. Then set the **Max Write Rate** at `1Gbps`

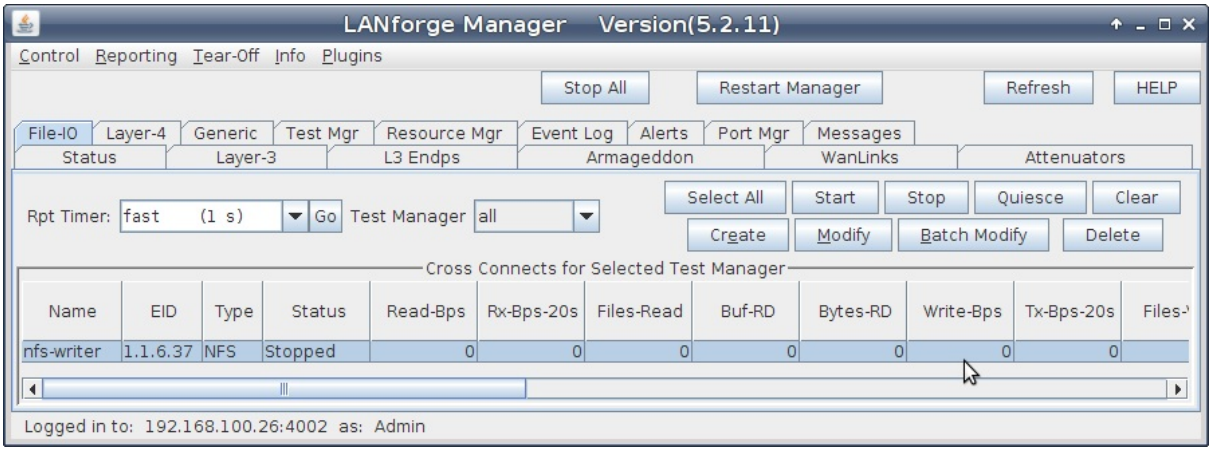
J. Set the number of files written per connection: set **File #** to `2`

K. Now we specify the NFS server: set **Server** to `10.26.1.3:/home/fileio`

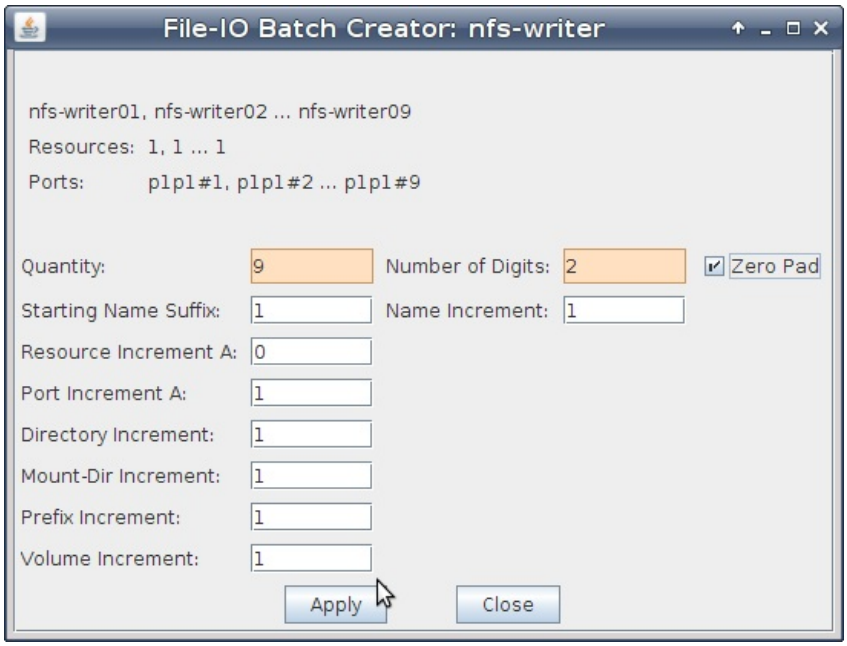
L. You can leave **Directory**, and **Mount-Dir** at `AUTO`

M. Click **OK** to commit the settings.

3. In the **File-IO** tab, select the endpoint you just created and click **Modify**



A. The **File-IO Details** dialog appears. Click the **Batch Create** button at the bottom of the screen.



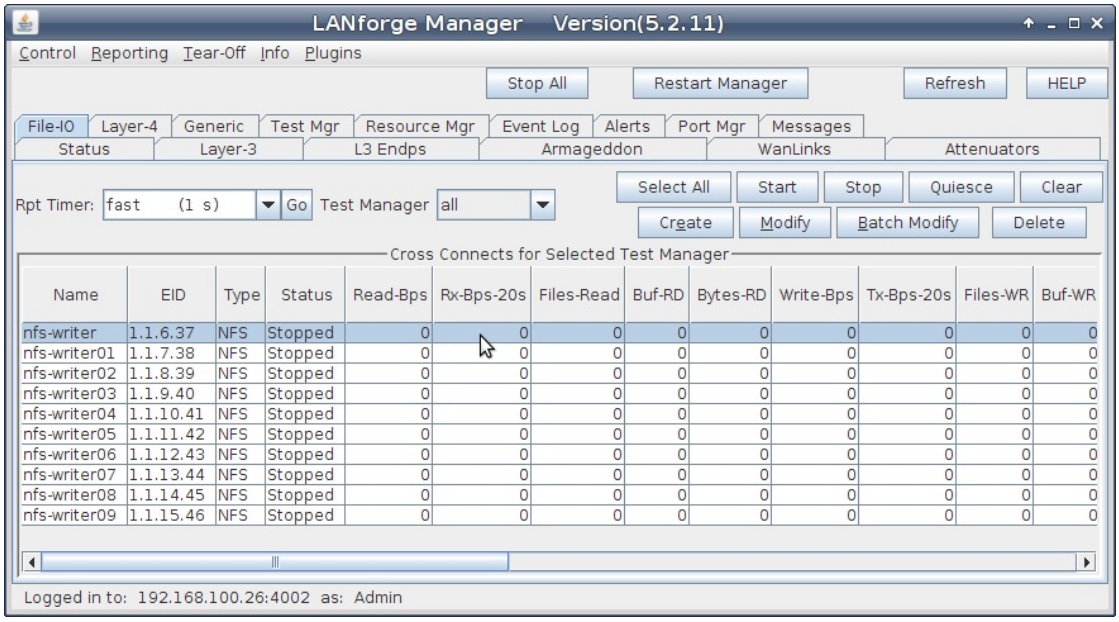
B. Enter these values into the Batch Create dialog:

   A. **Quantity** should be 9

   B. **Number of Digits** should be 2

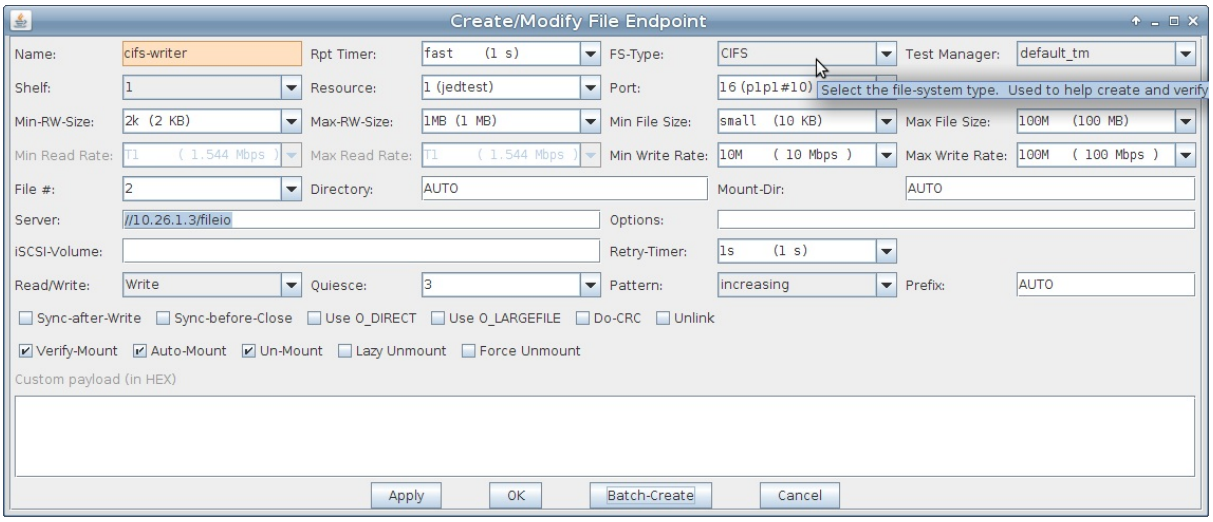
   C. Click **Apply**

C. Close the Batch Create window. You will see the new endpoints.



Name	EID	Type	Status	Read-Bps	Rx-Bps-20s	Files-Read	Buf-RD	Bytes-RD	Write-Bps	Tx-Bps-20s	Files-WR	Buf-WR
nfs-writer	1.1.6.37	NFS	Stopped	0	0	0	0	0	0	0	0	0
nfs-writer01	1.1.7.38	NFS	Stopped	0	0	0	0	0	0	0	0	0
nfs-writer02	1.1.8.39	NFS	Stopped	0	0	0	0	0	0	0	0	0
nfs-writer03	1.1.9.40	NFS	Stopped	0	0	0	0	0	0	0	0	0
nfs-writer04	1.1.10.41	NFS	Stopped	0	0	0	0	0	0	0	0	0
nfs-writer05	1.1.11.42	NFS	Stopped	0	0	0	0	0	0	0	0	0
nfs-writer06	1.1.12.43	NFS	Stopped	0	0	0	0	0	0	0	0	0
nfs-writer07	1.1.13.44	NFS	Stopped	0	0	0	0	0	0	0	0	0
nfs-writer08	1.1.14.45	NFS	Stopped	0	0	0	0	0	0	0	0	0
nfs-writer09	1.1.15.46	NFS	Stopped	0	0	0	0	0	0	0	0	0

4. Create initial endpoint for CIFS writer.



A. Name the endoint `cifs-writer`

B. Set **FS-Type** to `CIFS`

C. Then set the **Port** to the next open MAC VLAN: `p1p1#10`

D. Set the **Min-RW Size** and **Max-RW Size** to `2k` and `1M`

E. For the **Min File Size** and **Max File Size** enter `10KB` and `100MB`

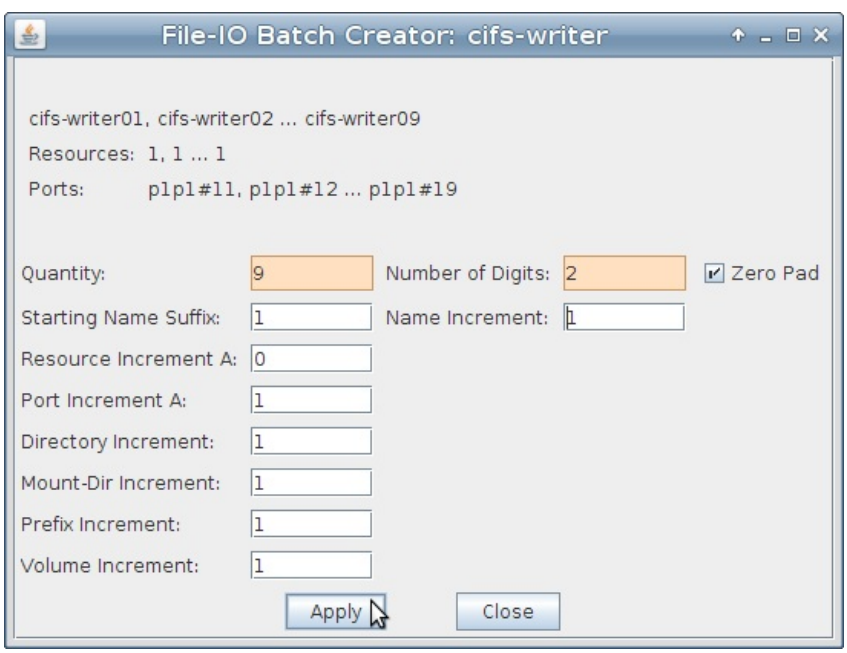
F. The **Min Write Rate** and the **Max Write Rate** should be `10 Mbps` and `100 Mbps`

G. Enter `//10.26.1.3/fileio` for **Server**

H. Supply the credentials for the CIFS mount point in **Options**. For this example, our username and password are 'lanforge' and 'lanforge'. Write them as options to the mount command: `user=lanforge,passwd=lanforge`

I. Click **Apply**

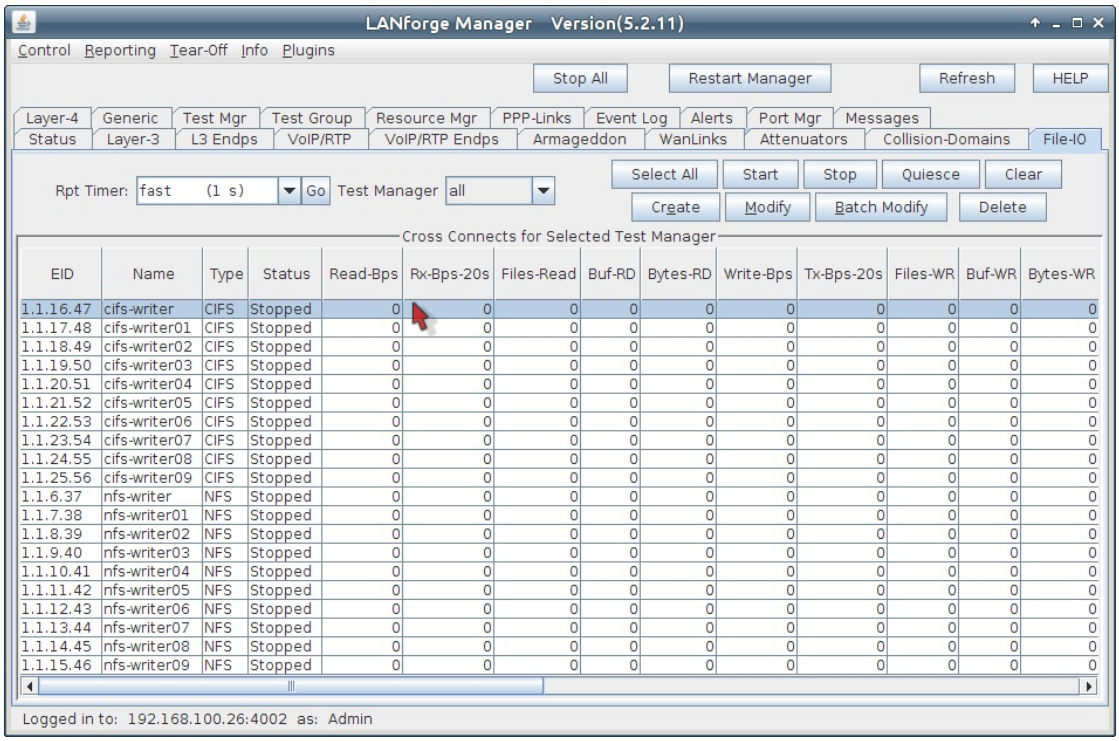
5. Use Batch Create to create nine more CIFS endpoints. You do not actually need to close the **Create/Modify** window. Click on **Batch Create** directly.
   A. In the **Batch Create** window, Enter:



   A. **Quantity** is `9`
   B. **Number of Digits** is `2`
   C. Then click **Apply**

   B. In **Create/Modify** click **Cancel**

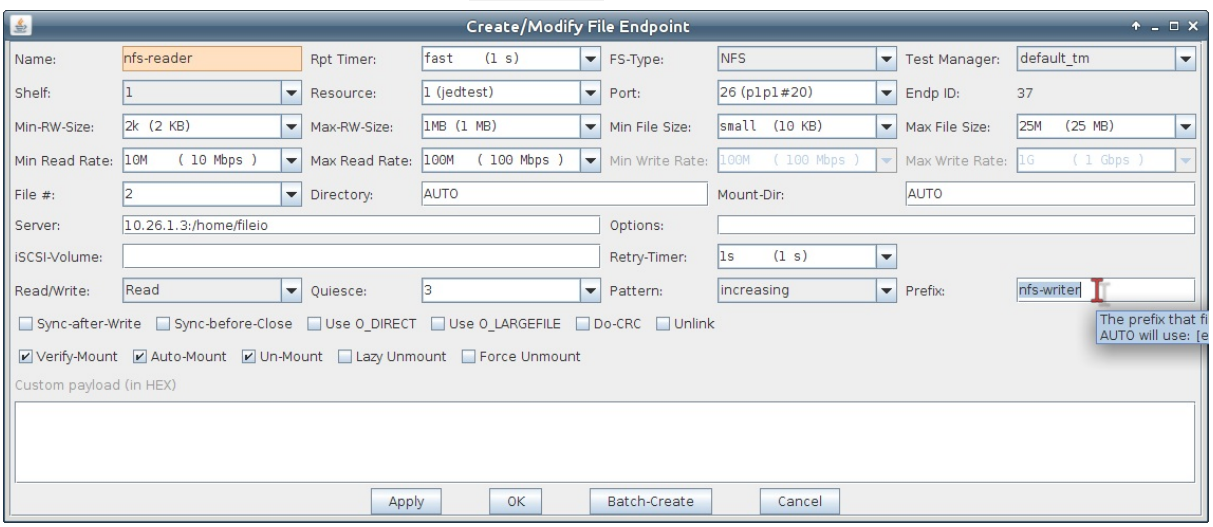
C. In the **File-IO** tab, you will see ten more endpoints.



EID	Name	Type	Status	Read-Bps	Rx-Bps-20s	Files-Read	Buf-RD	Bytes-RD	Write-Bps	Tx-Bps-20s	Files-WR	Buf-WR	Bytes-WR
1.1.16.47	cifs-writer	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.17.48	cifs-writer01	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.18.49	cifs-writer02	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.19.50	cifs-writer03	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.20.51	cifs-writer04	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.21.52	cifs-writer05	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.22.53	cifs-writer06	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.23.54	cifs-writer07	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.24.55	cifs-writer08	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.25.56	cifs-writer09	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.6.37	nfs-writer	NFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.7.38	nfs-writer01	NFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.8.39	nfs-writer02	NFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.9.40	nfs-writer03	NFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.10.41	nfs-writer04	NFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.11.42	nfs-writer05	NFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.12.43	nfs-writer06	NFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.13.44	nfs-writer07	NFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.14.45	nfs-writer08	NFS	Stopped	0	0	0	0	0	0	0	0	0	0
1.1.15.46	nfs-writer09	NFS	Stopped	0	0	0	0	0	0	0	0	0	0

6. We will proceed to creating the same number of **NFS reader endpoints** by using a writer as a template for a reader:

7. Open the Modify window of the endpoint `nfs-writer`



A. Change the **Name** to `nfs-reader`

B. Set the **Port** to the next unused MAC VLAN, `p1p1#20`

C. Change **Read/Write** to `Read`

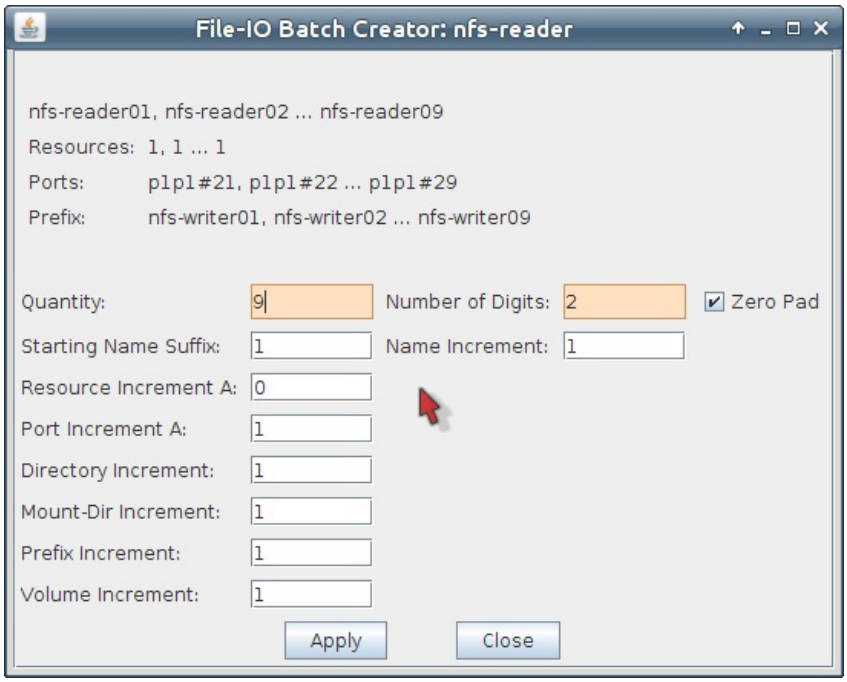
D. Set the **Min Read Rate** and **Max Read Rate** to `10 Mbps` and `100 Mbps`

E. To match this reader to a writer, set the **Prefix** field to `nfs-writer`

F. Click **Apply**

8. Before closing the Modify windows we can use Batch Create to create nine more NFS endpoints:

9. Click on the **Batch Create** window of the endpoint `nfs-reader`



A. Change the **Quantity** to `9`

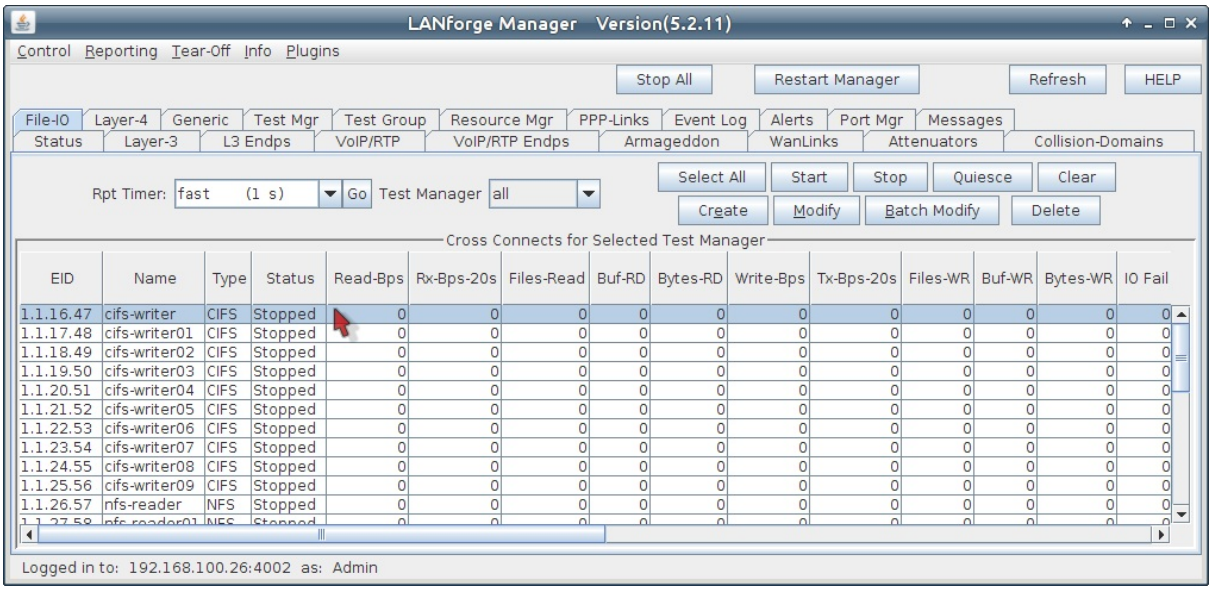
B. Set the **Number of Digits** to `2`

C. Click **Apply**

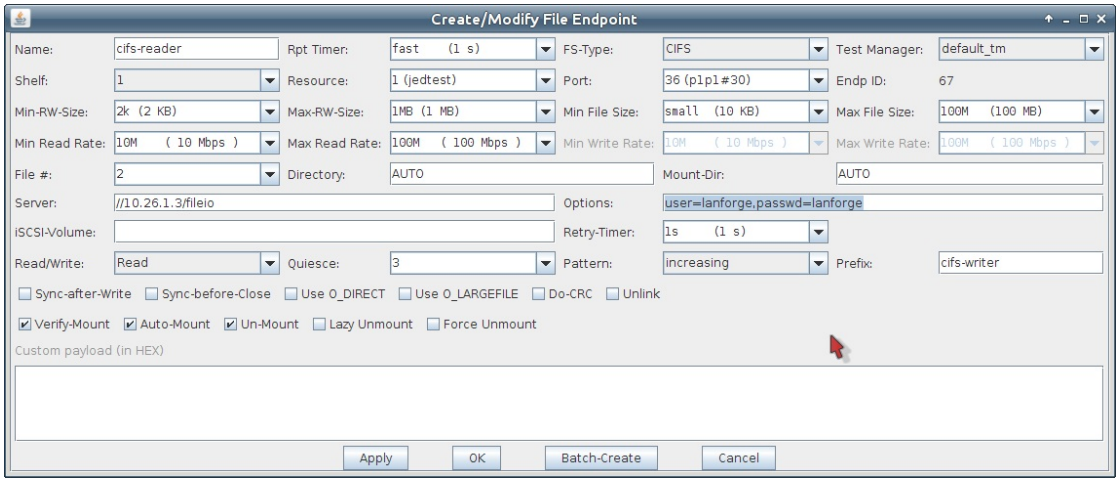
D. Click **Close**

E. In the **Modify** window, click **Cancel**

10. And we now create the same number of **CIFS reader endpoints**. Start by using `cifs-writer` as a template:



A. Open the Modify window of the endpoint `cifs-writer`



A. Change the **Name** to `cifs-reader`

B. Set the **Port** to the next unused MAC VLAN, `p1p1#30`

C. Change **Read/Write** to `Read`

D. Set the **Min Read Rate** and **Max Read Rate** to `10 Mbps` and `100 Mbps`

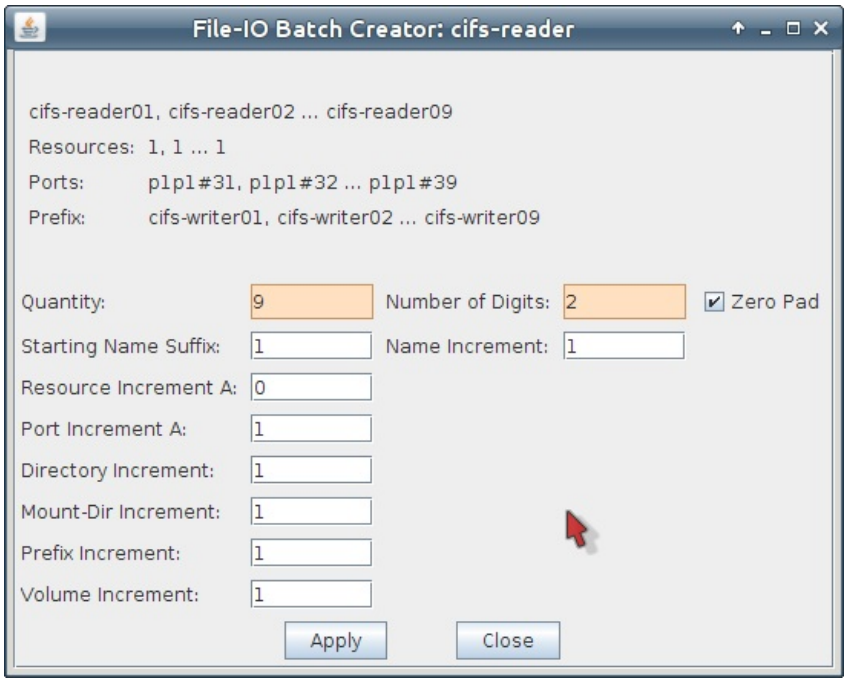
E. To match this reader to a writer, set the **Prefix** field to `cifs-writer`

F. Supply the credentials for the CIFS mount point in **Options**. For this example, our username and password are 'lanforge' and 'lanforge'. Write them as options to the mount command: `user=lanforge,passwd=lanforge`

G. Click **Apply**

B. Before closing the Modify windows we can use Batch Create to create nine more CIFS reader endpoints:

C. Click on the **Batch Create** window of the endpoint `cifs-reader`



A. Change the **Quantity** to `9`

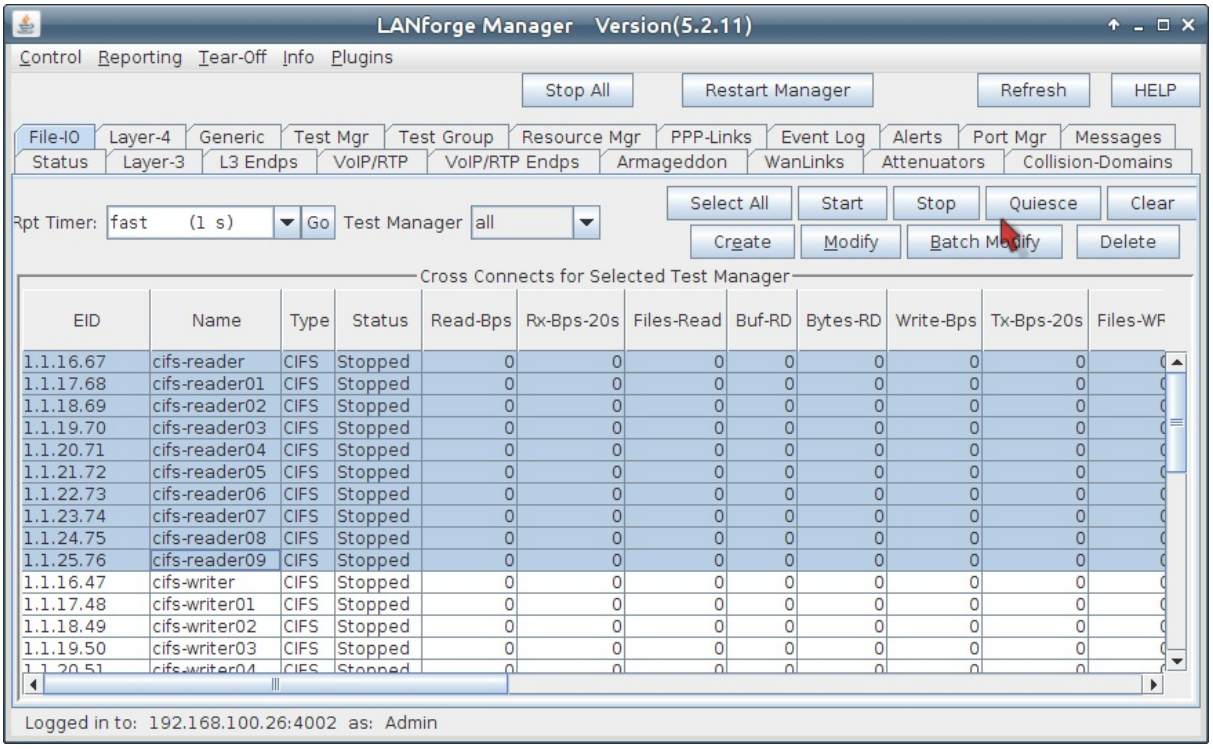
B. Set the **Number of Digits** to `2`

C. Click **Apply**

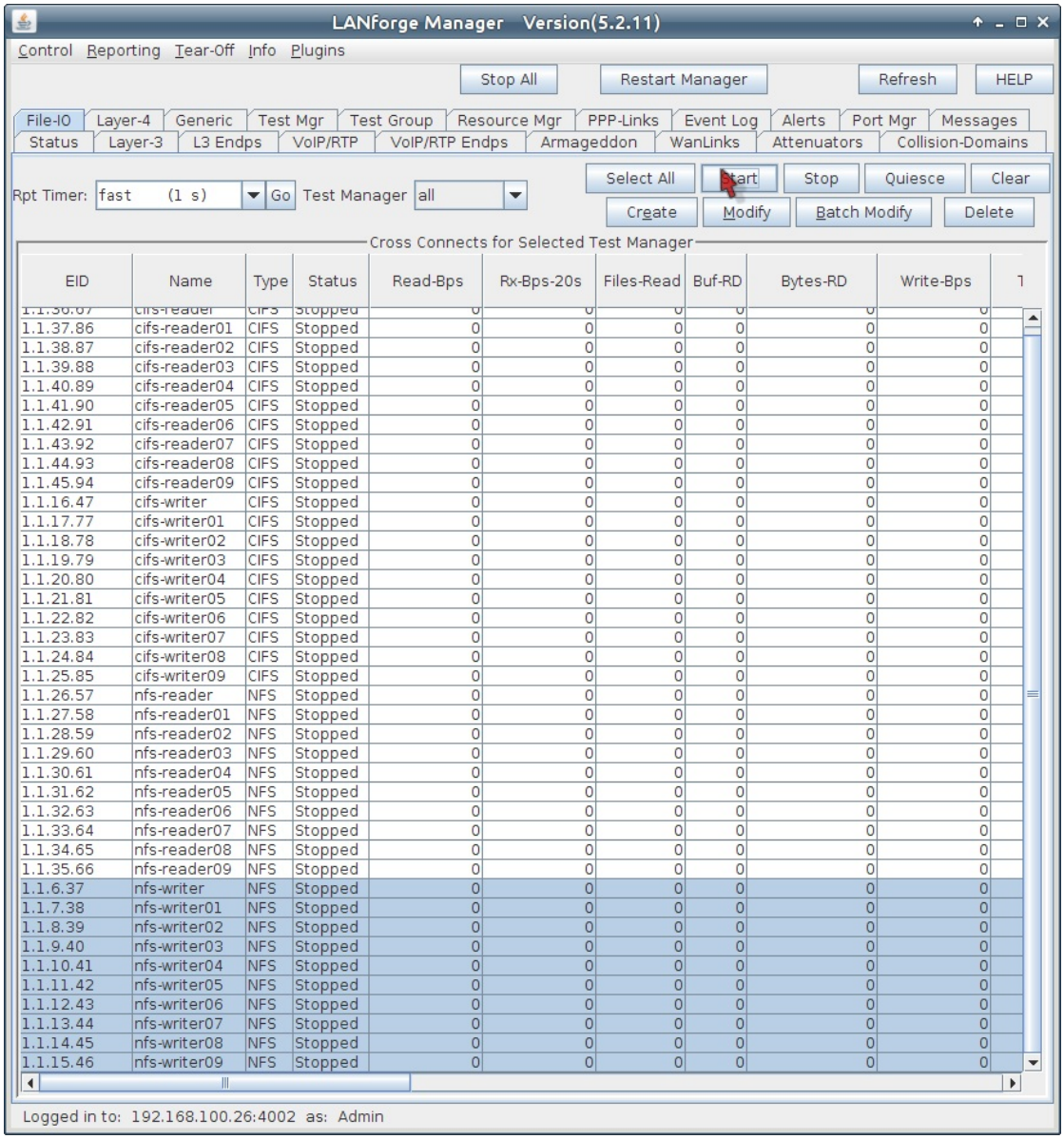
D. Click **Close**

E. In the **Modify** window, click **Cancel**

11. In the **File-IO** tab, we see our newly created reader endpoints.



12. Start the reader/writer testing.
    A. In the **File-IO** tab, begin by starting the NFS writers



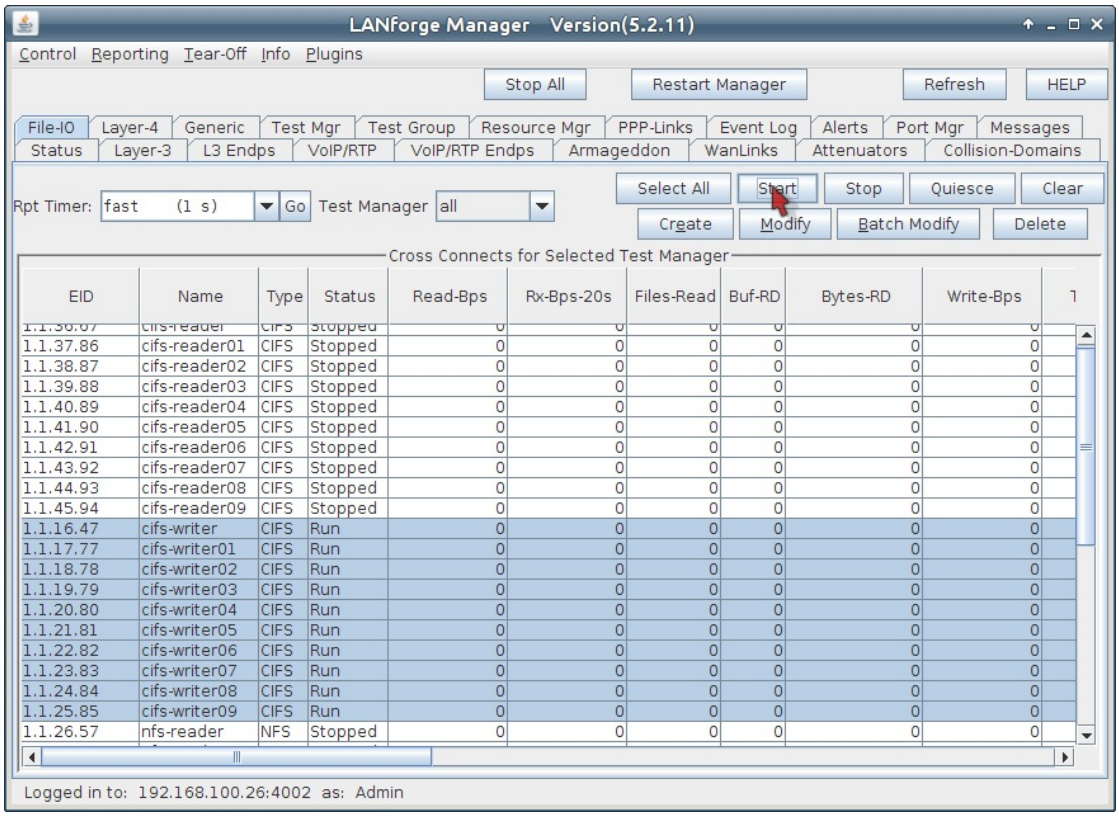
EID	Name	Type	Status	Read-Bps	Rx-Bps-20s	Files-Read	Buf-RD	Bytes-RD	Write-Bps
1.1.36.67	cifs-reader	CIFS	Stopped	0	0	0	0	0	0
1.1.37.86	cifs-reader01	CIFS	Stopped	0	0	0	0	0	0
1.1.38.87	cifs-reader02	CIFS	Stopped	0	0	0	0	0	0
1.1.39.88	cifs-reader03	CIFS	Stopped	0	0	0	0	0	0
1.1.40.89	cifs-reader04	CIFS	Stopped	0	0	0	0	0	0
1.1.41.90	cifs-reader05	CIFS	Stopped	0	0	0	0	0	0
1.1.42.91	cifs-reader06	CIFS	Stopped	0	0	0	0	0	0
1.1.43.92	cifs-reader07	CIFS	Stopped	0	0	0	0	0	0
1.1.44.93	cifs-reader08	CIFS	Stopped	0	0	0	0	0	0
1.1.45.94	cifs-reader09	CIFS	Stopped	0	0	0	0	0	0
1.1.16.47	cifs-writer	CIFS	Stopped	0	0	0	0	0	0
1.1.17.77	cifs-writer01	CIFS	Stopped	0	0	0	0	0	0
1.1.18.78	cifs-writer02	CIFS	Stopped	0	0	0	0	0	0
1.1.19.79	cifs-writer03	CIFS	Stopped	0	0	0	0	0	0
1.1.20.80	cifs-writer04	CIFS	Stopped	0	0	0	0	0	0
1.1.21.81	cifs-writer05	CIFS	Stopped	0	0	0	0	0	0
1.1.22.82	cifs-writer06	CIFS	Stopped	0	0	0	0	0	0
1.1.23.83	cifs-writer07	CIFS	Stopped	0	0	0	0	0	0
1.1.24.84	cifs-writer08	CIFS	Stopped	0	0	0	0	0	0
1.1.25.85	cifs-writer09	CIFS	Stopped	0	0	0	0	0	0
1.1.26.57	nfs-reader	NFS	Stopped	0	0	0	0	0	0
1.1.27.58	nfs-reader01	NFS	Stopped	0	0	0	0	0	0
1.1.28.59	nfs-reader02	NFS	Stopped	0	0	0	0	0	0
1.1.29.60	nfs-reader03	NFS	Stopped	0	0	0	0	0	0
1.1.30.61	nfs-reader04	NFS	Stopped	0	0	0	0	0	0
1.1.31.62	nfs-reader05	NFS	Stopped	0	0	0	0	0	0
1.1.32.63	nfs-reader06	NFS	Stopped	0	0	0	0	0	0
1.1.33.64	nfs-reader07	NFS	Stopped	0	0	0	0	0	0
1.1.34.65	nfs-reader08	NFS	Stopped	0	0	0	0	0	0
1.1.35.66	nfs-reader09	NFS	Stopped	0	0	0	0	0	0
1.1.6.37	nfs-writer	NFS	Stopped	0	0	0	0	0	0
1.1.7.38	nfs-writer01	NFS	Stopped	0	0	0	0	0	0
1.1.8.39	nfs-writer02	NFS	Stopped	0	0	0	0	0	0
1.1.9.40	nfs-writer03	NFS	Stopped	0	0	0	0	0	0
1.1.10.41	nfs-writer04	NFS	Stopped	0	0	0	0	0	0
1.1.11.42	nfs-writer05	NFS	Stopped	0	0	0	0	0	0
1.1.12.43	nfs-writer06	NFS	Stopped	0	0	0	0	0	0
1.1.13.44	nfs-writer07	NFS	Stopped	0	0	0	0	0	0
1.1.14.45	nfs-writer08	NFS	Stopped	0	0	0	0	0	0
1.1.15.46	nfs-writer09	NFS	Stopped	0	0	0	0	0	0

A. Click on the **Name** column header to sort the rows of readers and writers

B. You can use `control-left-click` and drag to select the group of writers

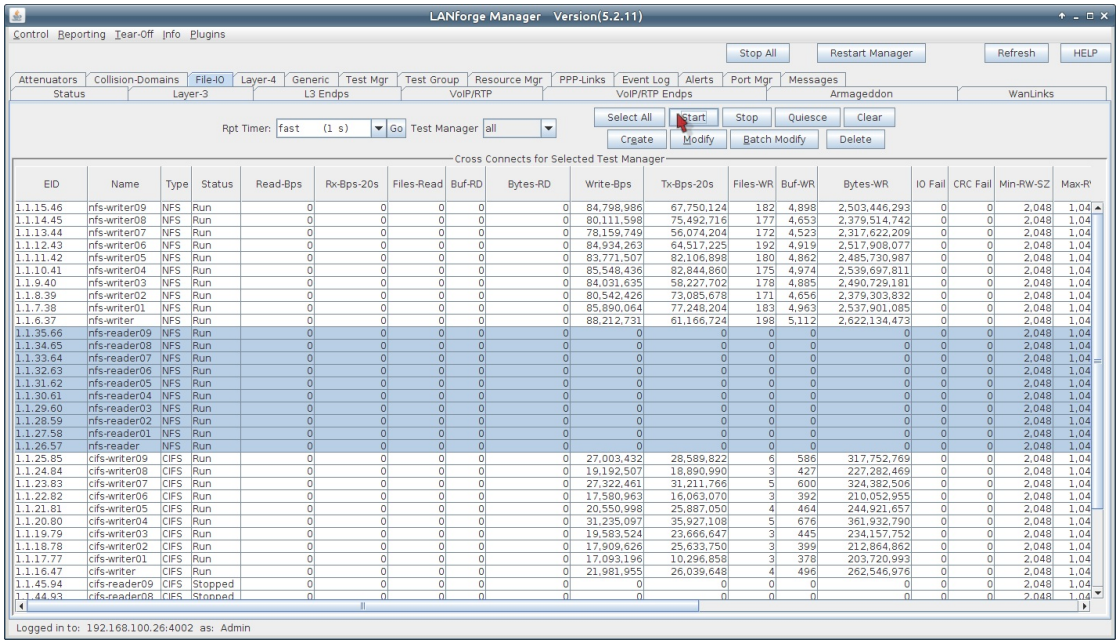
C. Click the **Start** button at the top.

B. Next, start the CIFS writers



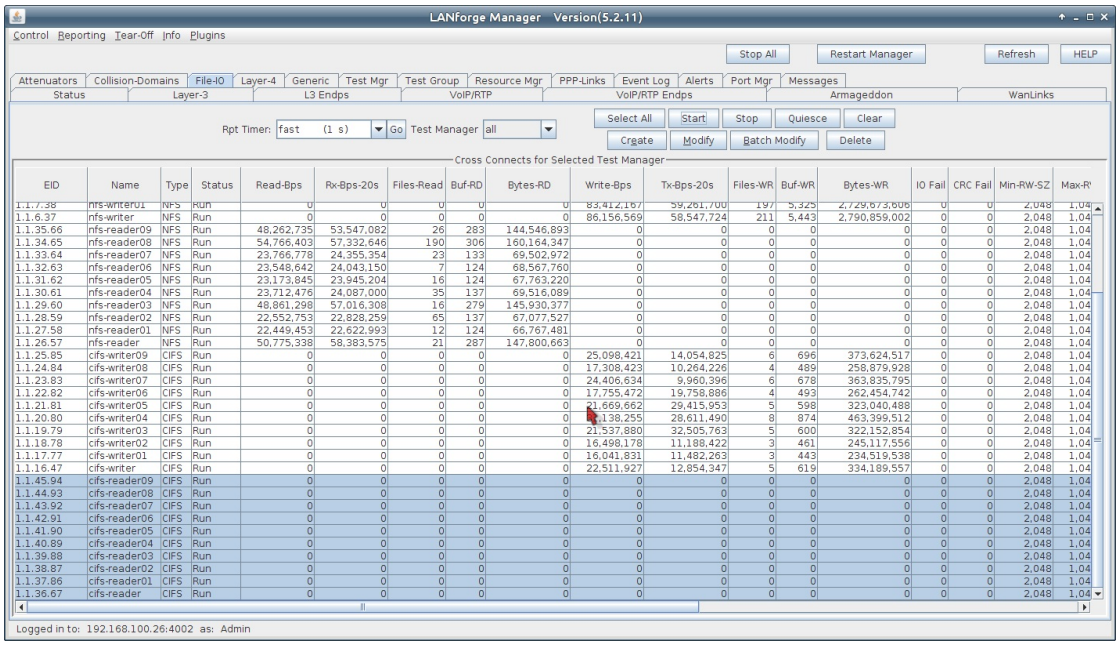
EID	Name	Type	Status	Read-Bps	Rx-Bps-20s	Files-Read	Buf-RD	Bytes-RD	Write-Bps
1.1.36.67	cifs-reader	CIFS	Stopped	0	0	0	0	0	0
1.1.37.86	cifs-reader01	CIFS	Stopped	0	0	0	0	0	0
1.1.38.87	cifs-reader02	CIFS	Stopped	0	0	0	0	0	0
1.1.39.88	cifs-reader03	CIFS	Stopped	0	0	0	0	0	0
1.1.40.89	cifs-reader04	CIFS	Stopped	0	0	0	0	0	0
1.1.41.90	cifs-reader05	CIFS	Stopped	0	0	0	0	0	0
1.1.42.91	cifs-reader06	CIFS	Stopped	0	0	0	0	0	0
1.1.43.92	cifs-reader07	CIFS	Stopped	0	0	0	0	0	0
1.1.44.93	cifs-reader08	CIFS	Stopped	0	0	0	0	0	0
1.1.45.94	cifs-reader09	CIFS	Stopped	0	0	0	0	0	0
1.1.16.47	cifs-writer	CIFS	Run	0	0	0	0	0	0
1.1.17.77	cifs-writer01	CIFS	Run	0	0	0	0	0	0
1.1.18.78	cifs-writer02	CIFS	Run	0	0	0	0	0	0
1.1.19.79	cifs-writer03	CIFS	Run	0	0	0	0	0	0
1.1.20.80	cifs-writer04	CIFS	Run	0	0	0	0	0	0
1.1.21.81	cifs-writer05	CIFS	Run	0	0	0	0	0	0
1.1.22.82	cifs-writer06	CIFS	Run	0	0	0	0	0	0
1.1.23.83	cifs-writer07	CIFS	Run	0	0	0	0	0	0
1.1.24.84	cifs-writer08	CIFS	Run	0	0	0	0	0	0
1.1.25.85	cifs-writer09	CIFS	Run	0	0	0	0	0	0
1.1.26.57	nfs-reader	NFS	Stopped	0	0	0	0	0	0

C. Then the NFS readers



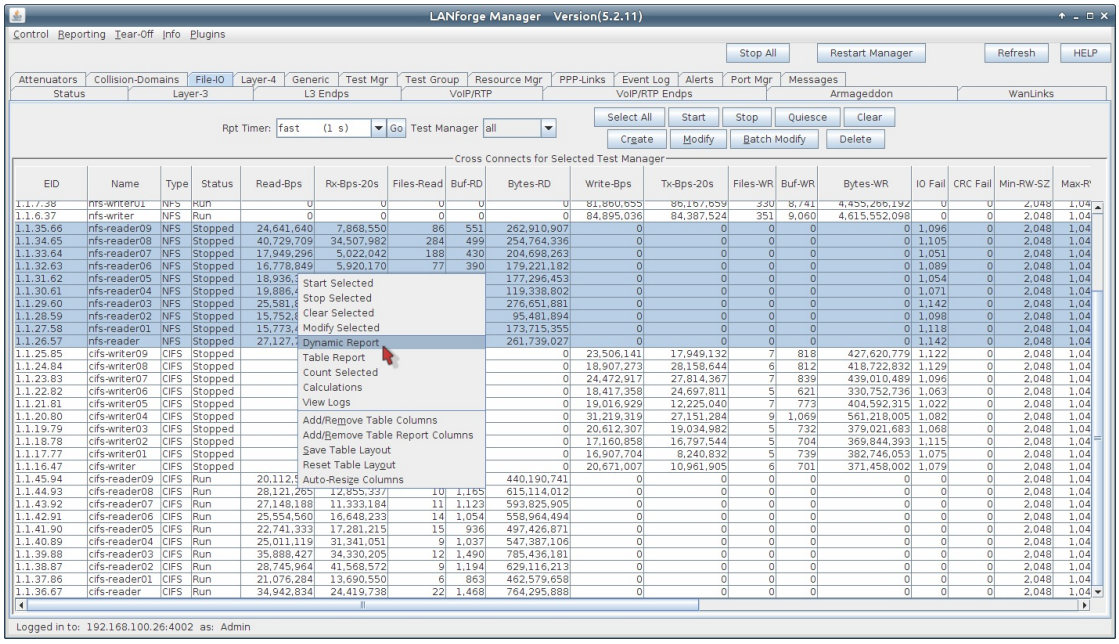
EID	Name	Type	Status	Read-Bps	Rx-Bps-20s	Files-Read	Buf-RD	Bytes-RD	Write-Bps	Tx-Bps-20s	Files-WR	Buf-WR	Bytes-WR	IO Fail	CRC Fail	Min-RW-SZ	Max-R'
1.1.15.46	nfs-writer09	NFS	Run	0	0	0	0	0	84,798,986	67,750,124	182	4,898	2,503,446,293	0	0	2,048	1,04
1.1.14.45	nfs-writer08	NFS	Run	0	0	0	0	0	80,111,598	75,492,716	177	4,653	2,379,514,742	0	0	2,048	1,04
1.1.13.44	nfs-writer07	NFS	Run	0	0	0	0	0	78,159,749	56,074,204	172	4,523	2,317,622,209	0	0	2,048	1,04
1.1.12.43	nfs-writer06	NFS	Run	0	0	0	0	0	84,934,263	64,517,225	192	4,919	2,517,908,077	0	0	2,048	1,04
1.1.11.42	nfs-writer05	NFS	Run	0	0	0	0	0	83,771,507	82,106,898	180	4,862	2,485,730,987	0	0	2,048	1,04
1.1.10.41	nfs-writer04	NFS	Run	0	0	0	0	0	85,548,436	82,844,860	175	4,974	2,539,697,811	0	0	2,048	1,04
1.1.9.40	nfs-writer03	NFS	Run	0	0	0	0	0	84,031,635	58,227,702	178	4,885	2,490,729,181	0	0	2,048	1,04
1.1.8.39	nfs-writer02	NFS	Run	0	0	0	0	0	80,542,426	73,085,678	171	4,656	2,379,303,832	0	0	2,048	1,04
1.1.7.38	nfs-writer01	NFS	Run	0	0	0	0	0	85,890,064	77,248,204	183	4,963	2,537,901,085	0	0	2,048	1,04
1.1.6.37	nfs-writer	NFS	Run	0	0	0	0	0	88,212,731	61,166,724	198	5,112	2,622,134,473	0	0	2,048	1,04
1.1.35.66	nfs-reader09	NFS	Run	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04
1.1.34.65	nfs-reader08	NFS	Run	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04
1.1.33.64	nfs-reader07	NFS	Run	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04
1.1.32.63	nfs-reader06	NFS	Run	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04
1.1.31.62	nfs-reader05	NFS	Run	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04
1.1.30.61	nfs-reader04	NFS	Run	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04
1.1.29.60	nfs-reader03	NFS	Run	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04
1.1.28.59	nfs-reader02	NFS	Run	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04
1.1.27.58	nfs-reader01	NFS	Run	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04
1.1.26.57	nfs-reader	NFS	Run	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04
1.1.25.85	cifs-writer09	CIFS	Run	0	0	0	0	0	27,003,432	28,589,822	6	586	317,752,769	0	0	2,048	1,04
1.1.24.84	cifs-writer08	CIFS	Run	0	0	0	0	0	19,192,507	18,890,990	3	427	227,282,469	0	0	2,048	1,04
1.1.23.83	cifs-writer07	CIFS	Run	0	0	0	0	0	27,322,461	31,211,766	5	600	324,382,506	0	0	2,048	1,04
1.1.22.82	cifs-writer06	CIFS	Run	0	0	0	0	0	17,580,963	16,063,070	3	392	210,052,955	0	0	2,048	1,04
1.1.21.81	cifs-writer05	CIFS	Run	0	0	0	0	0	20,550,998	25,887,050	4	464	244,921,657	0	0	2,048	1,04
1.1.20.80	cifs-writer04	CIFS	Run	0	0	0	0	0	31,235,097	35,927,108	5	676	361,932,790	0	0	2,048	1,04
1.1.19.79	cifs-writer03	CIFS	Run	0	0	0	0	0	19,583,524	23,666,647	3	445	234,157,752	0	0	2,048	1,04
1.1.18.78	cifs-writer02	CIFS	Run	0	0	0	0	0	17,909,626	25,633,750	3	399	212,864,862	0	0	2,048	1,04
1.1.17.77	cifs-writer01	CIFS	Run	0	0	0	0	0	17,093,196	10,296,858	3	378	203,720,993	0	0	2,048	1,04
1.1.16.47	cifs-writer	CIFS	Run	0	0	0	0	0	21,981,955	26,039,648	4	496	262,546,976	0	0	2,048	1,04
1.1.45.94	cifs-reader09	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04
1.1.44.93	cifs-reader08	CIFS	Stopped	0	0	0	0	0	0	0	0	0	0	0	0	2,048	1,04

D. Finally, start the CIFS readers

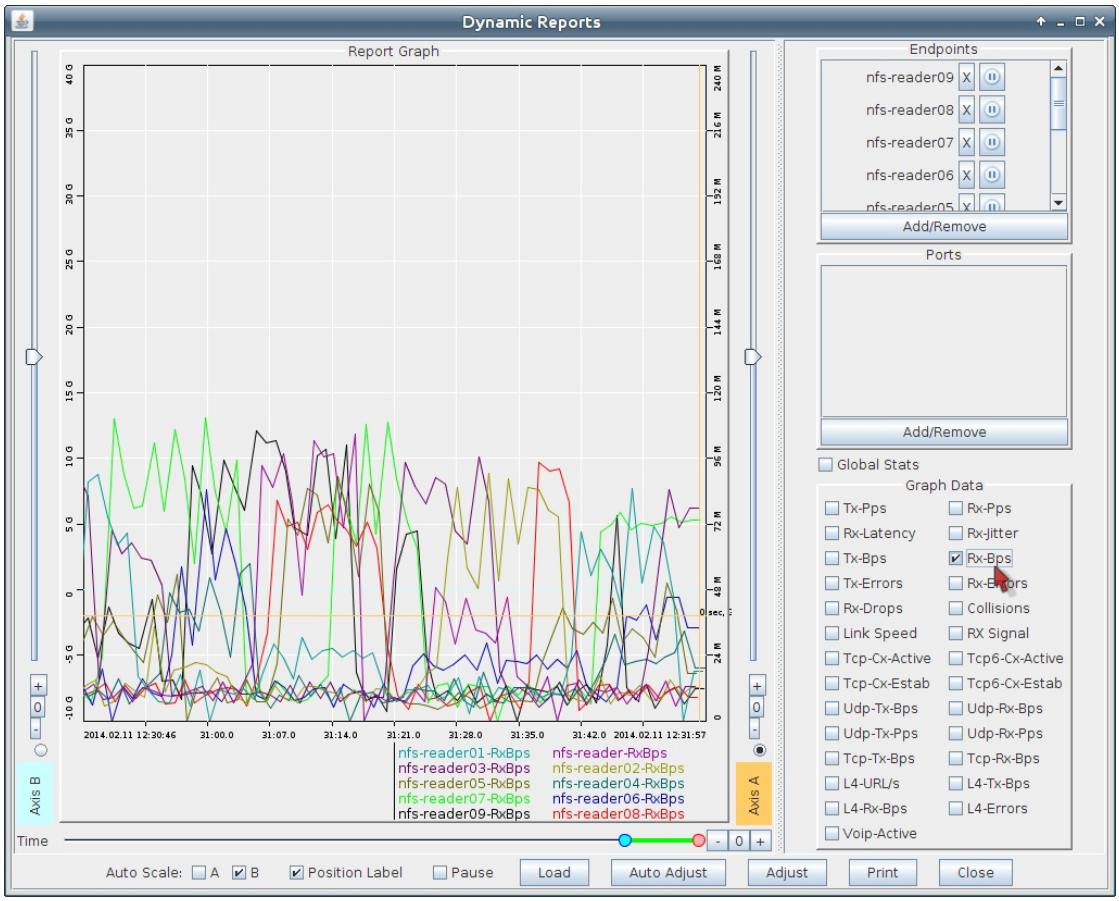


13. We can inspect the traffic in the Dynamic Report window and the File-IO tab:

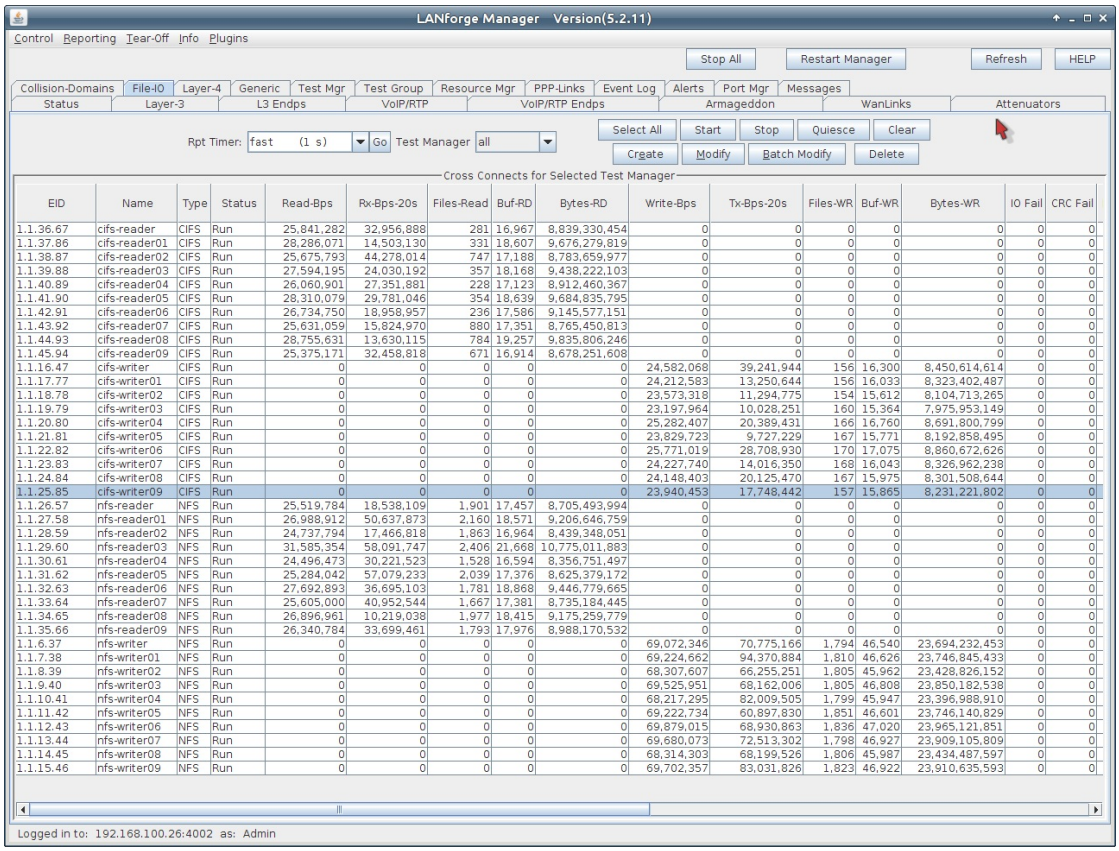
A. Select a group of endpoints and right click on them, select **Dynamic Report**



B. In the **Dynamic Report** window, select the checkbox **Rx-Bps**



C. In the **File-IO** window, we can watch for Files Written and IO Errors:



EID	Name	Type	Status	Read-Bps	Rx-Bps-20s	Files-Read	Buf-RD	Bytes-RD	Write-Bps	Tx-Bps-20s	Files-WR	Buf-WR	Bytes-WR	IO Fail	CRC Fail
1.1.36.67	cifs-reader	CIFS	Run	25,841,282	32,956,888	281	16,967	8,839,330,454	0	0	0	0	0	0	0
1.1.37.86	cifs-reader01	CIFS	Run	28,286,071	14,503,130	331	18,607	9,676,279,819	0	0	0	0	0	0	0
1.1.38.87	cifs-reader02	CIFS	Run	25,675,793	44,278,014	747	17,188	8,783,659,977	0	0	0	0	0	0	0
1.1.39.88	cifs-reader03	CIFS	Run	27,594,195	24,030,192	357	18,168	9,438,222,103	0	0	0	0	0	0	0
1.1.40.89	cifs-reader04	CIFS	Run	26,060,901	27,351,881	228	17,123	8,912,460,367	0	0	0	0	0	0	0
1.1.41.90	cifs-reader05	CIFS	Run	28,310,079	29,781,046	354	18,639	9,684,835,795	0	0	0	0	0	0	0
1.1.42.91	cifs-reader06	CIFS	Run	26,734,750	18,958,957	236	17,586	9,145,577,151	0	0	0	0	0	0	0
1.1.43.92	cifs-reader07	CIFS	Run	25,631,059	15,824,970	880	17,351	8,765,450,813	0	0	0	0	0	0	0
1.1.44.93	cifs-reader08	CIFS	Run	28,755,631	13,630,115	784	19,257	9,835,806,246	0	0	0	0	0	0	0
1.1.45.94	cifs-reader09	CIFS	Run	25,375,171	32,458,818	671	16,914	8,678,251,608	0	0	0	0	0	0	0
1.1.16.47	cifs-writer	CIFS	Run	0	0	0	0	0	24,582,068	39,241,944	156	16,300	8,450,614,614	0	0
1.1.17.77	cifs-writer01	CIFS	Run	0	0	0	0	0	24,212,583	13,250,644	156	16,033	8,323,402,487	0	0
1.1.18.78	cifs-writer02	CIFS	Run	0	0	0	0	0	23,573,318	11,294,775	154	15,612	8,104,713,265	0	0
1.1.19.79	cifs-writer03	CIFS	Run	0	0	0	0	0	23,197,964	10,028,251	160	15,364	7,975,953,149	0	0
1.1.20.80	cifs-writer04	CIFS	Run	0	0	0	0	0	25,282,407	20,389,431	166	16,760	8,691,800,799	0	0
1.1.21.81	cifs-writer05	CIFS	Run	0	0	0	0	0	23,829,723	9,727,229	167	15,771	8,192,858,495	0	0
1.1.22.82	cifs-writer06	CIFS	Run	0	0	0	0	0	25,771,019	28,708,930	170	17,075	8,860,672,626	0	0
1.1.23.83	cifs-writer07	CIFS	Run	0	0	0	0	0	24,227,740	14,016,350	168	16,043	8,326,962,238	0	0
1.1.24.84	cifs-writer08	CIFS	Run	0	0	0	0	0	24,148,403	20,125,470	167	15,975	8,301,508,644	0	0
1.1.25.85	cifs-writer09	CIFS	Run	0	0	0	0	0	23,940,453	17,748,442	157	15,865	8,231,221,802	0	0
1.1.26.57	nfs-reader	NFS	Run	25,519,784	18,538,109	1,901	17,457	8,705,493,994	0	0	0	0	0	0	0
1.1.27.58	nfs-reader01	NFS	Run	26,988,912	50,637,873	2,160	18,571	9,206,646,759	0	0	0	0	0	0	0
1.1.28.59	nfs-reader02	NFS	Run	24,737,794	17,466,818	1,863	16,964	8,439,348,051	0	0	0	0	0	0	0
1.1.29.60	nfs-reader03	NFS	Run	31,585,354	58,091,747	2,406	21,668	10,775,011,883	0	0	0	0	0	0	0
1.1.30.61	nfs-reader04	NFS	Run	24,496,473	30,221,523	1,528	16,594	8,356,751,497	0	0	0	0	0	0	0
1.1.31.62	nfs-reader05	NFS	Run	25,284,042	57,079,233	2,039	17,376	8,625,379,172	0	0	0	0	0	0	0
1.1.32.63	nfs-reader06	NFS	Run	27,692,893	36,695,103	1,781	18,868	9,446,779,665	0	0	0	0	0	0	0
1.1.33.64	nfs-reader07	NFS	Run	25,605,000	40,952,544	1,667	17,381	8,735,184,445	0	0	0	0	0	0	0
1.1.34.65	nfs-reader08	NFS	Run	26,896,961	10,219,038	1,977	18,415	9,175,259,779	0	0	0	0	0	0	0
1.1.35.66	nfs-reader09	NFS	Run	26,340,784	33,699,461	1,793	17,976	8,988,170,532	0	0	0	0	0	0	0
1.1.6.37	nfs-writer	NFS	Run	0	0	0	0	0	69,072,346	70,775,166	1,794	46,540	23,694,232,453	0	0
1.1.7.38	nfs-writer01	NFS	Run	0	0	0	0	0	69,224,662	94,370,884	1,810	46,626	23,746,845,433	0	0
1.1.8.39	nfs-writer02	NFS	Run	0	0	0	0	0	68,307,607	66,255,251	1,805	45,962	23,428,826,152	0	0
1.1.9.40	nfs-writer03	NFS	Run	0	0	0	0	0	69,525,951	68,162,006	1,805	46,808	23,850,182,538	0	0
1.1.10.41	nfs-writer04	NFS	Run	0	0	0	0	0	68,217,295	82,009,505	1,799	45,947	23,396,988,910	0	0
1.1.11.42	nfs-writer05	NFS	Run	0	0	0	0	0	69,222,734	60,897,830	1,851	46,601	23,746,140,829	0	0
1.1.12.43	nfs-writer06	NFS	Run	0	0	0	0	0	69,879,015	68,930,863	1,836	47,020	23,965,121,851	0	0
1.1.13.44	nfs-writer07	NFS	Run	0	0	0	0	0	69,680,073	72,513,302	1,798	46,927	23,909,105,809	0	0
1.1.14.45	nfs-writer08	NFS	Run	0	0	0	0	0	68,314,303	68,199,526	1,806	45,987	23,434,487,597	0	0
1.1.15.46	nfs-writer09	NFS	Run	0	0	0	0	0	69,702,357	83,031,826	1,823	46,922	23,910,635,593	0	0

14.

Advanced File-IO Tunings

System tuning is desirable when determining machine performance. The following topics can apply to NFS or CIFS clients:

15. Tuning network card ring-buffers
    A. Network cards often have tunable memory buffers that can be modified using `ethtool`
    B. Read the available capacities using `ethtool -g eth1`
    C. Adjust the tx and rx parameters using `ethtool -G eth1 tx 4096 rx 4096`
    D. This feature depends on the card driver and system state. Not all cards are tunable.
    E. The efficacy of this technique is difficult to measure, but is some manufacturer's websites show this step being done in their white papers.

16. Matching buffer sizes to negotiated transfer size
    A. NFS mount parameters can be inspected using either of two commands:
    B. `mount | grep nfs` or `grep nfs /proc/mounts`
    C. Notice the rsize and wsize parameters might not match your request values in the Modify Properties box for the mount point.
    D. Setting your Min/Max Read and Min/Max Write sizes to these negotiated values will maximize efficiency by reducing memcpy activity in the kernel.

17. Tuning TCP window sizes
    A. While you can modify Layer 3 TCP window size and buffers, you cannot directly change those settings using the LANforge GUI for the system. This Discussion on NFS4 performance is worth a read.
    B. The commands below are suggested to increase your buffer sizes to 8-16 megabytes:
    ```
    sysctl net.core.rmem_default=8388608
    sysctl net.core.rmem_max=16777216
    sysctl net.core.wmem_default=8388608
    sysctl net.core.wmem_max=16777216
    ```

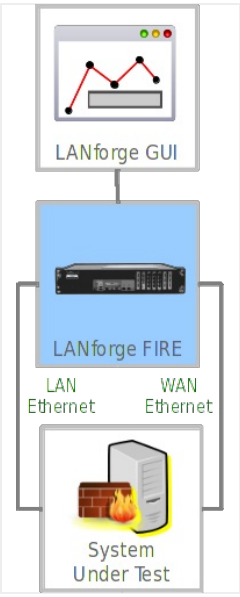
Providing HTTP Service on a Port

Goal: Configure and activate a http server bound to a specific port in LANforge.

This is useful if you want to provide an application layer target to interact with. This cookbook assumes you can access a shell prompt on the LANforge computer: the command-line instructions can be done at the system console, in a terminal over a remote desktop connection, or over `ssh`. The nginx service is only available on the Linux version of LANforge server.

Note: There are two web servers installed in a LANforge computer. The default web server is a stock version of Apache HTTPD that responds to all port 80 requests on all interfaces. For testing we recommend running Candela Technologies' version of Nginx on specific ports as covered in this cookbook. In contrast, Apache cannot bind to a network-device therefore we do not suggest running it.

Nginx can also be configured to listen to IPv6 traffic. Jump to the end of the cookbook for that technique.



1. Stop and disable LANforge system Apache instance.

A. Log into the LANforge computer as user lanforge

```
jreynolds@jed-shuttle:~ - Terminal

jreynolds@jed-shuttle ~
 > ssh lanforge@192.168.100.40
```

B. Become super-user using the command `su -` (or `sudo -s`)

```
root@jed-f20:/home/lanforge - Terminal
[lanforge@jed-f20 ~]$ sudo -s
[root@jed-f20 lanforge]#
```

C. Stop the Apache service with `systemctl stop httpd`

```
root@jed-f20:/home/lanforge - Terminal
[root@jed-f20 lanforge]# systemctl stop httpd
[root@jed-f20 lanforge]#
```

D. If you want this setting to persist after a reboot, disable the service: `systemctl disable httpd`

```
root@jed-f20:/home/lanforge - Terminal
[root@jed-f20 lanforge]# systemctl stop httpd
[root@jed-f20 lanforge]# systemctl disable httpd
rm '/etc/systemd/system/multi-user.target.wants/httpd.service'
[root@jed-f20 lanforge]#
```

E. Set this as system default with: `systemctl daemon-reload`

```
root@jed-f20:/home/lanforge - Terminal
[root@jed-f20 lanforge]# systemctl stop httpd
[root@jed-f20 lanforge]# systemctl disable httpd
rm '/etc/systemd/system/multi-user.target.wants/httpd.service'
[root@jed-f20 lanforge]# systemctl daemon-reload
[root@jed-f20 lanforge]#
```

2. (Optional) There are other options for running Apache if you want to have both web servers available. You would not need stop and disable Apache, just restart it. You can change Apache to:

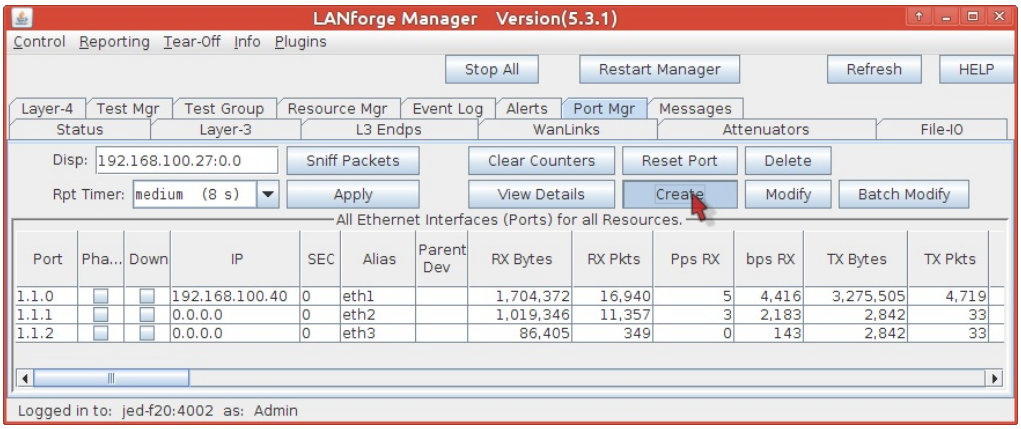
A. ...listen to a different port (like 81). Edit **/etc/httpd/conf/httpd.conf** and change the `Listen` option.

B. ...bind to a specific IP address, which is a good option if you configure the LANforge computer to have a fixed IP address on the management port. You would edit **httpd.conf** and change `Listen` to that specific IP address and port 80. Example: `Listen 192.168.1.40:80`

3. The demonstration test we'll create is making two ports, one for making requests and the other one for serving the protocol. Let's create two ports:

A. Create a redirect device:

A. In the Ports tab, click the **Create** button

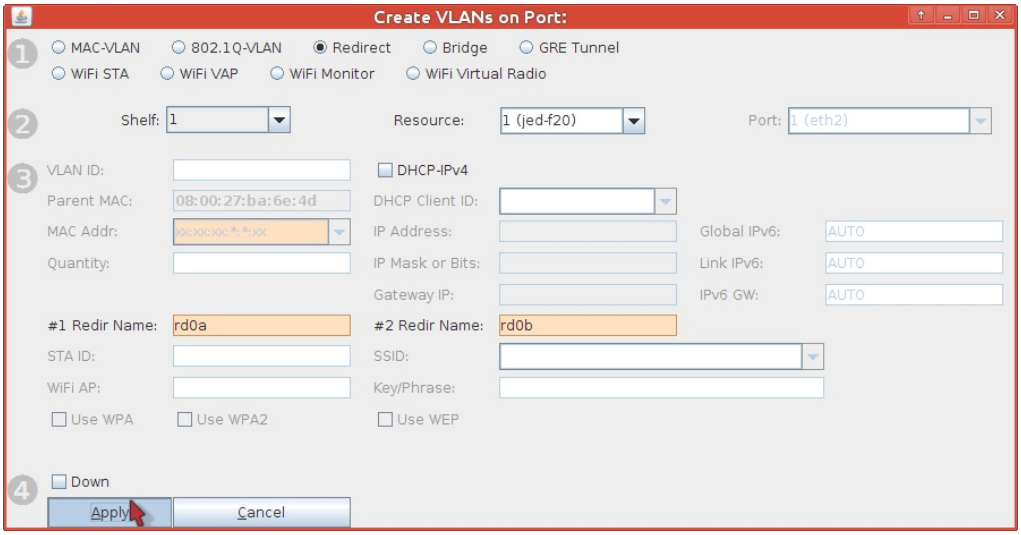


B. Select **Redirect**

C. Enter `rd0a` for #1 Redir name,

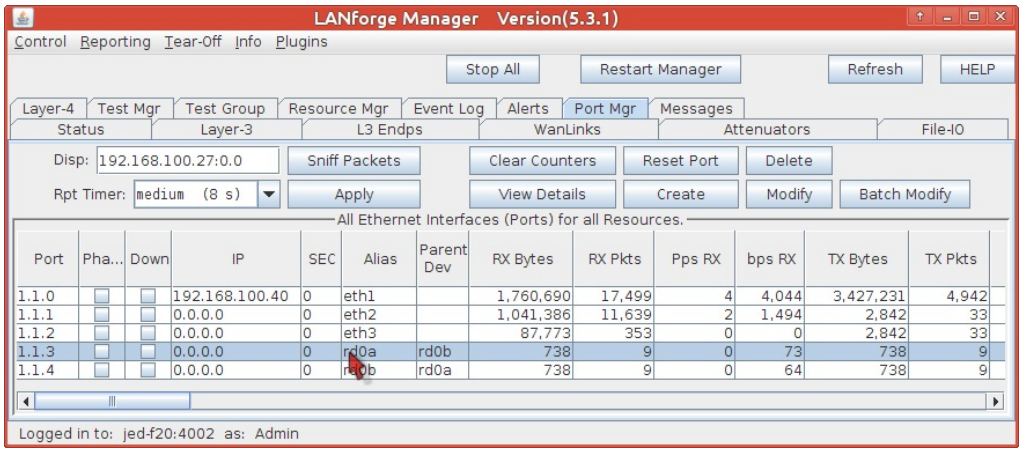
D. and `rd0b` for #2 Redir name.

E. Click **Apply** and then **Cancel** to close the window.



B. Configure **rd0a** as the service port:

A. In the Ports tab, double-click the row for port **rd0a**



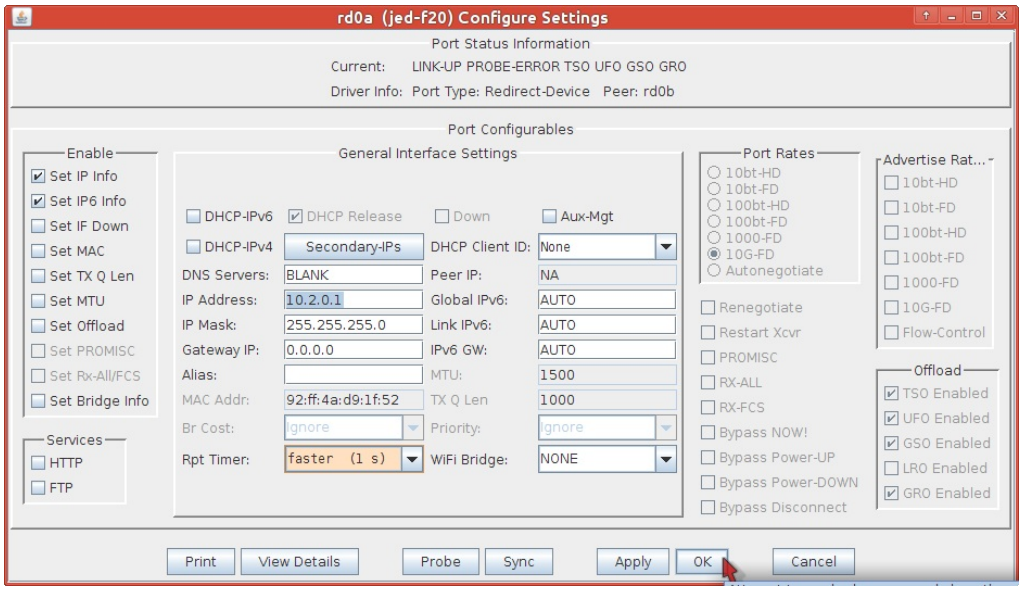
B. The Configure Settings window will appear

C. Set an IP of `10.2.0.1`,

D. a netmask of `255.255.255.0`

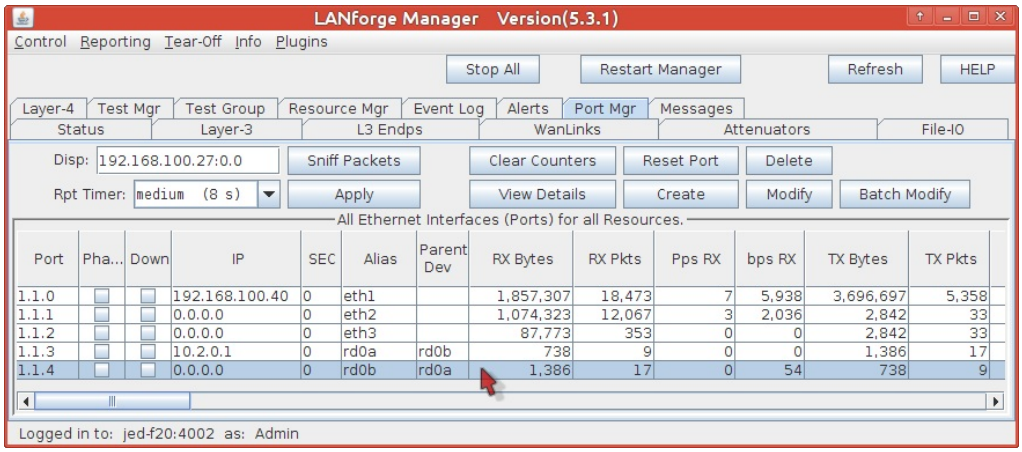
E. and the report time to faster (1 s).

F. Click **OK**



C. Configure **rd0b** as a client port

A. In the Ports tab, double-click the row for port **rd0b**



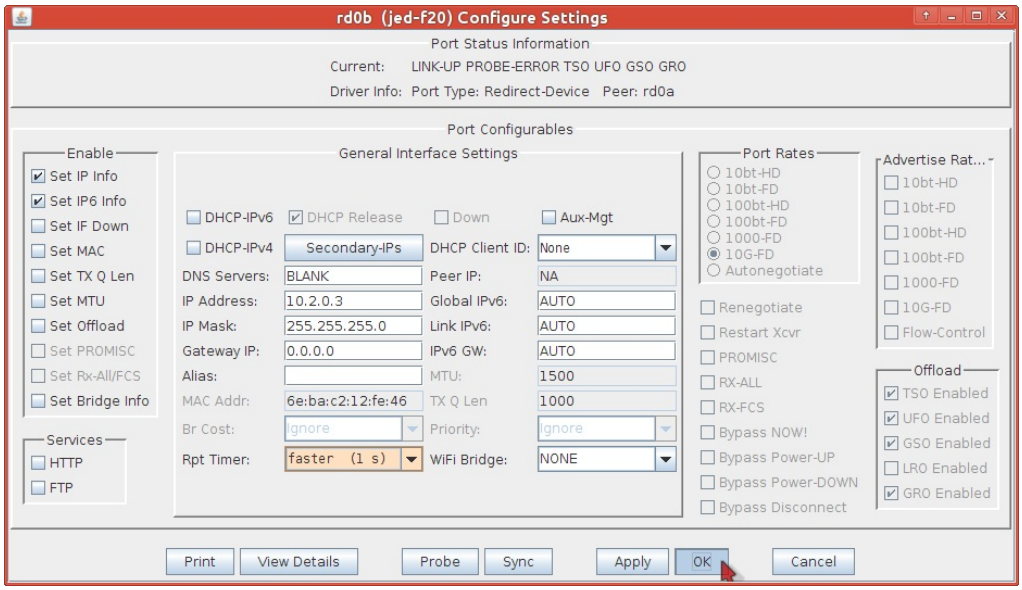
B. The Configure Settings window will appear

C. Set an IP of `10.2.0.3`,

D. a netmask of `255.255.255.0`

E. and the report time to faster (1 s).

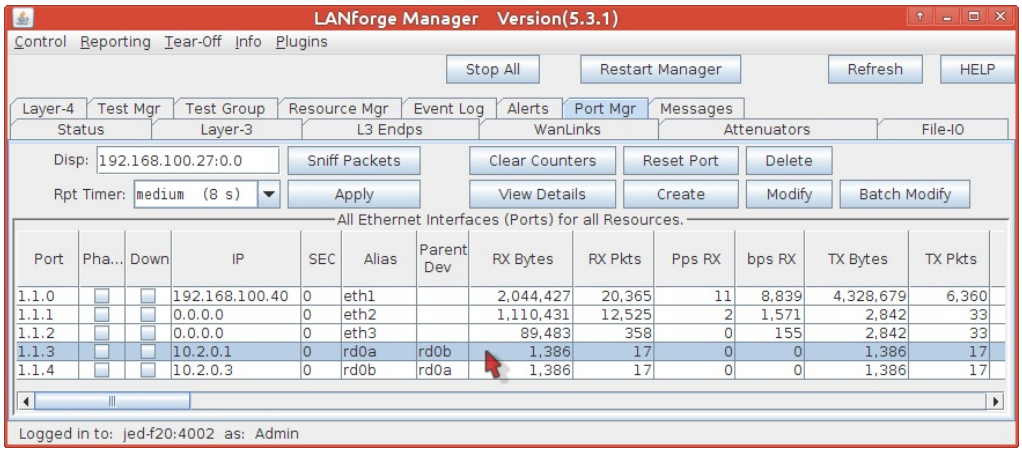
F. Click **OK**



4. Enable HTTP service on your upstream port using the following steps.

A. Enable the nginx service on port **rd0a**:

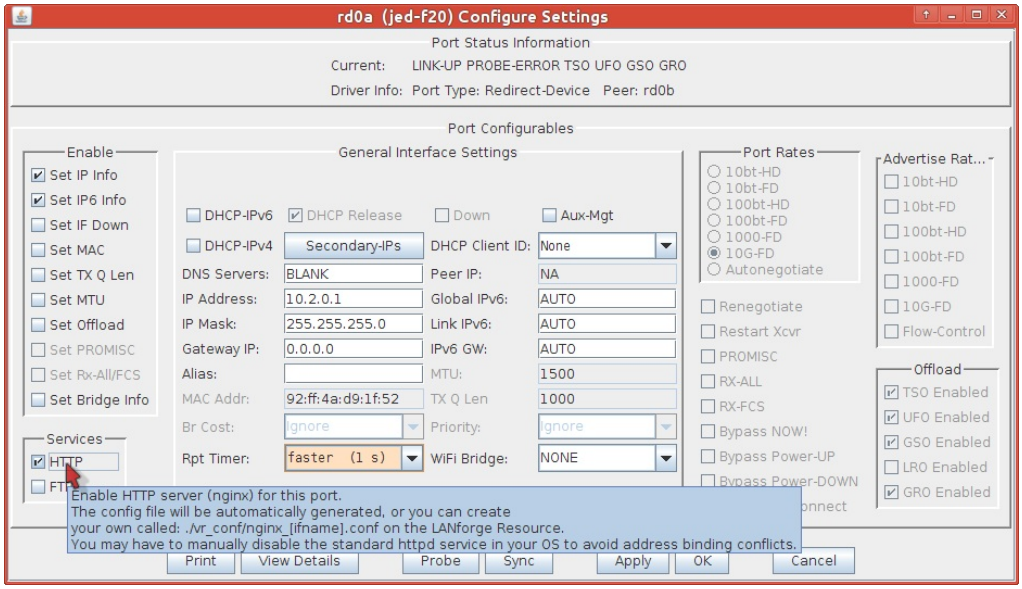
A. In the Ports tab, double-click the row for port **rd0a**



B. The Configure Settings window will appear

C. In the lower left column of the window, enable **HTTP** option.

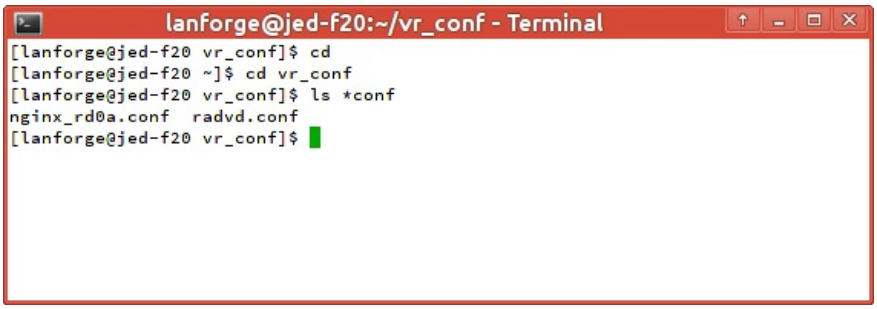
D. Click **OK** and then close the window.



B. (Optional) Modify the nginx config file for port **rd0a**. This example assumes you are logged into the LANforge system console, or have connected to the desktop using a remote-desktop client like vncviewer. If you are familiar with editing from the command-line, you will likely know how to do this via ssh.

A. From the desktop Accessories menu, select Terminal Emulator,

B. Change to the LANforge nginx directory: `cd /home/lanforge/vr_conf`



C. Edit the file `nginx_rd0a.conf`. If you do not see the file, you might have connected to the wrong LANforge resource, or you might not have clicked OK/Apply when enabling the Configure Settings window for port rd0a.

```
lanforge@jed-f20:~/vr_conf - Terminal
[lanforge@jed-f20 vr_conf]$ nano nginx_rd0a.conf
```

D. You will see that the listen directive is already set to `10.2.0.1:80 bind_dev=rd0a;`

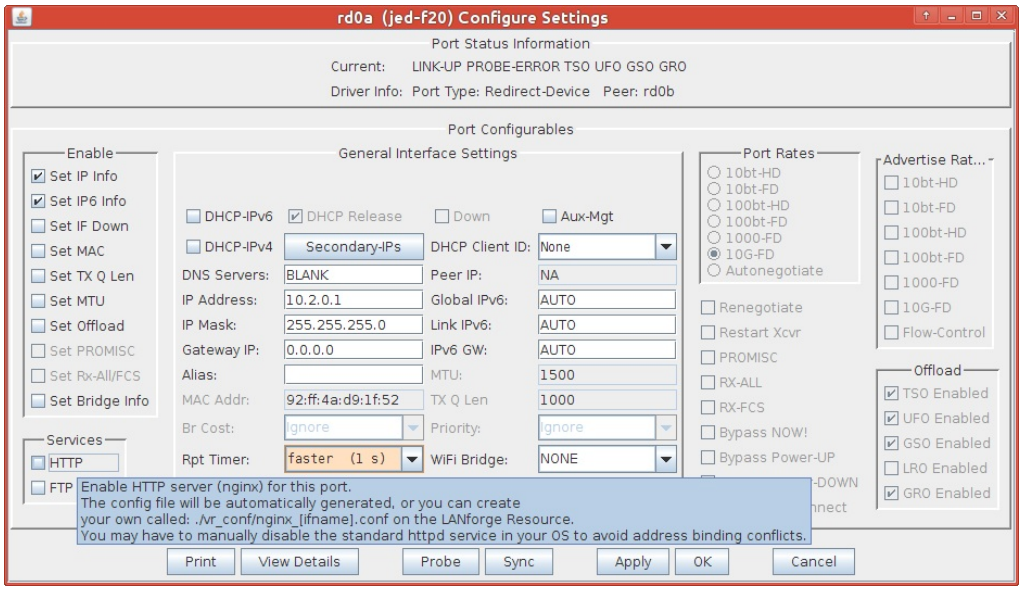
```
lanforge@jed-f20:~/vr_conf - Terminal
  GNU nano 2.3.2            File: nginx_rd0a.conf

# Autogenerated by LANforge, will be over-written by LANforge.
# Remove the first line '# Autogenerated by ...' and edit the file as
# desired for a custom config file.

worker_processes        1;
error_log logs/rd0a_error.log;
pid             /home/lanforge/vr_conf/nginx_rd0a.pid;
events {
    worker_connections  1024;
}

http {
    include       /usr/local/lanforge/nginx/conf/mime.types;
    default_type  application/octet-stream;
    access_log  logs/rd0a_access.log;
    sendfile            on;
    keepalive_timeout  65;

    server {
        listen       10.2.0.1:80 bind_dev=rd0a;
        server_name  localhost;
        access_log  logs/rd0a_host.access.log;

        location / {
            root   html;

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

E. If you want to change settings, delete the first line of the config file as part of your changes. This will signal LANforge not to overwrite the file.

```
lanforge@jed-f20:~/vr_conf - Terminal
  GNU nano 2.3.2            File: nginx_rd0a.conf

# Autogenerated by LANforge, will be over-written by LANforge.
# Remove the first line '# Autogenerated by ...' and edit the file as
# desired for a custom config file.

worker_processes        1;
error_log logs/rd0a_error.log;
pid             /home/lanforge/vr_conf/nginx_rd0a.pid;
events {
    worker_connections  1024;
}

http {
    include       /usr/local/lanforge/nginx/conf/mime.types;
    default_type  application/octet-stream;
    access_log  logs/rd0a_access.log;
    sendfile            on;
    keepalive_timeout  65;

    server {
        listen       10.2.0.1:80 bind_dev=rd0a;
        server_name  localhost;
        access_log  logs/rd0a_host.access.log;

        location / {
            root   html;

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

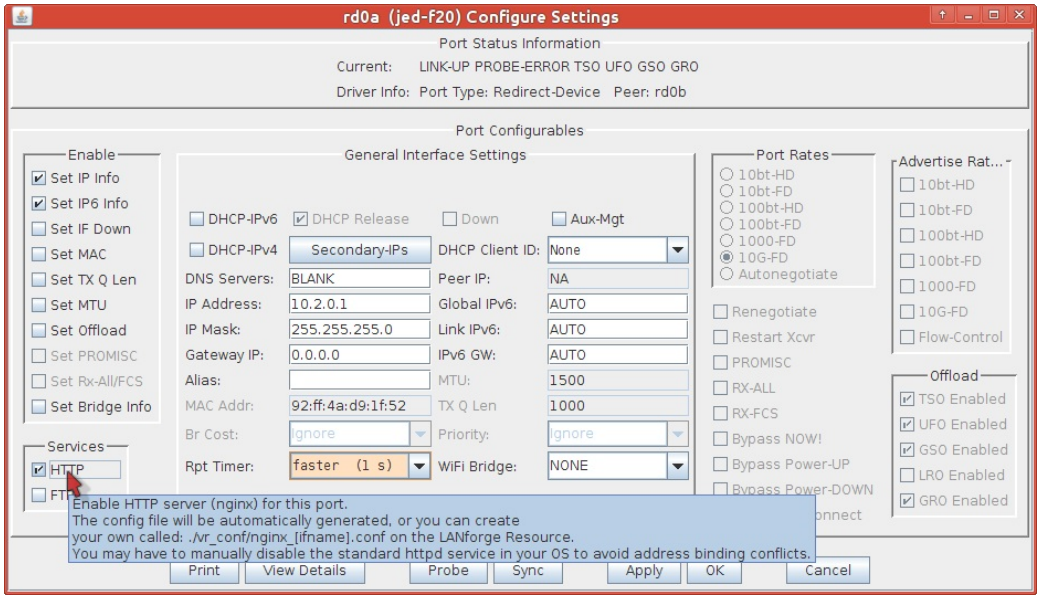
C. To apply the changes to the nginx service on this port:

A. Disable the HTTPD service in the rd0a Configure Settings window,



B. Click the **Apply** button,

C. Enable HTTPD service,



D. Click the **Apply** button,

E. Click the **Cancel** button to close the window if you are done.

D. A quick way to add a file into the nginx document root folder is to symlink the system dictionary there. You will need to be in a terminal or unix shell of the resource running nginx (192.168.100.40 in this example)

A. Change the ownership of the document root directory to user **lanforge**:

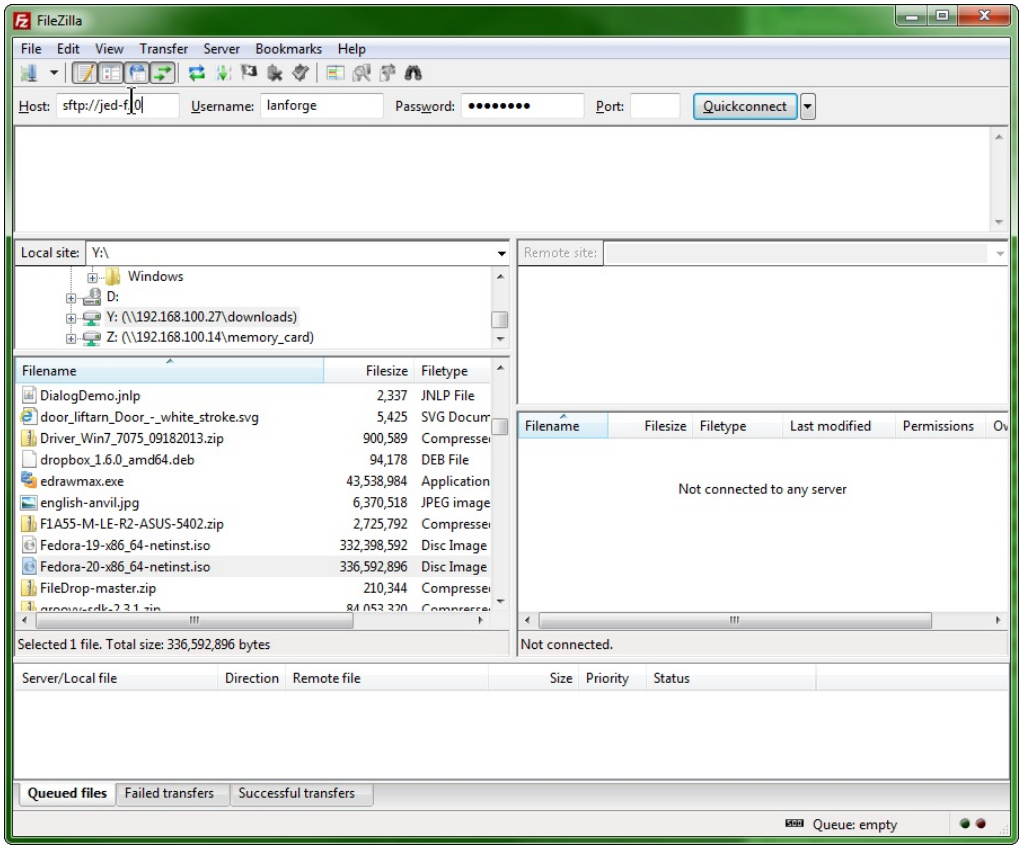
B. `sudo chown lanforge:lanforge /usr/local/lanforge/nginx/html`

C. `cd /usr/local/lanforge/nginx/html`

D. `ln -s /usr/share/dict/linux.words .`

E. (Optional) To load a larger payload (such as a DVD file), you can use Filezilla (or scp) to copy the file over. You will want to have changed the nginx document root directory owner to lanforge (as above):

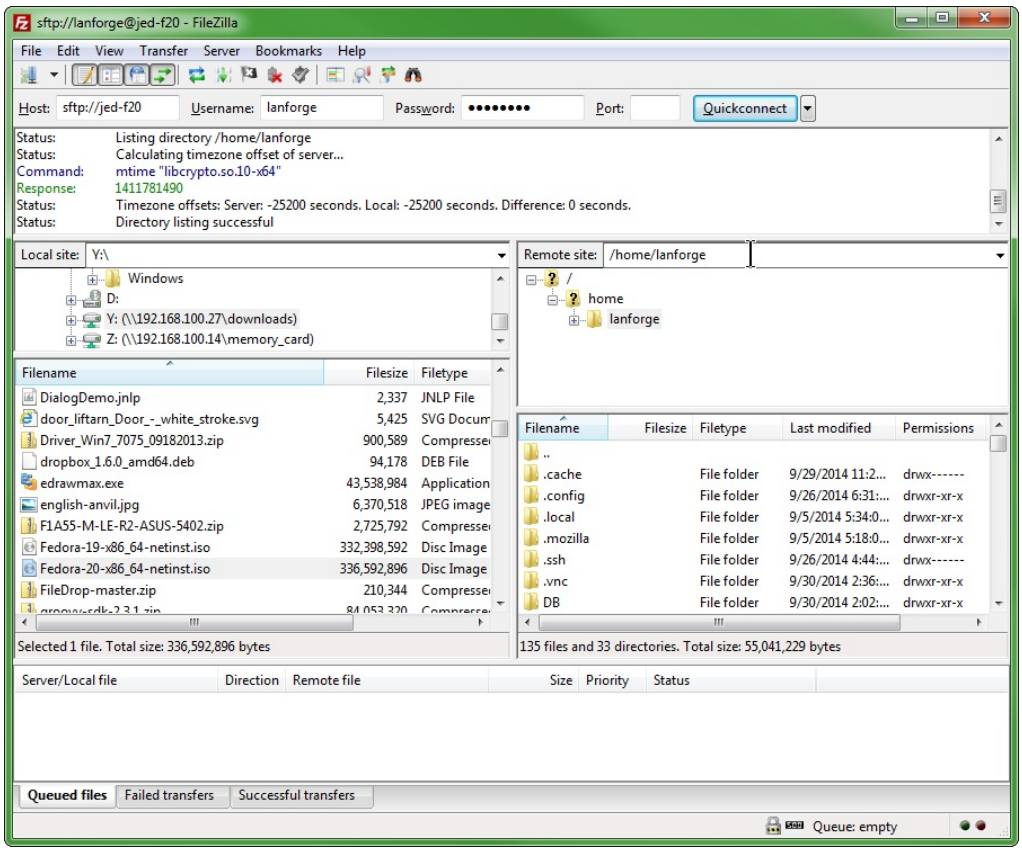
A. From your desktop, using Filezilla



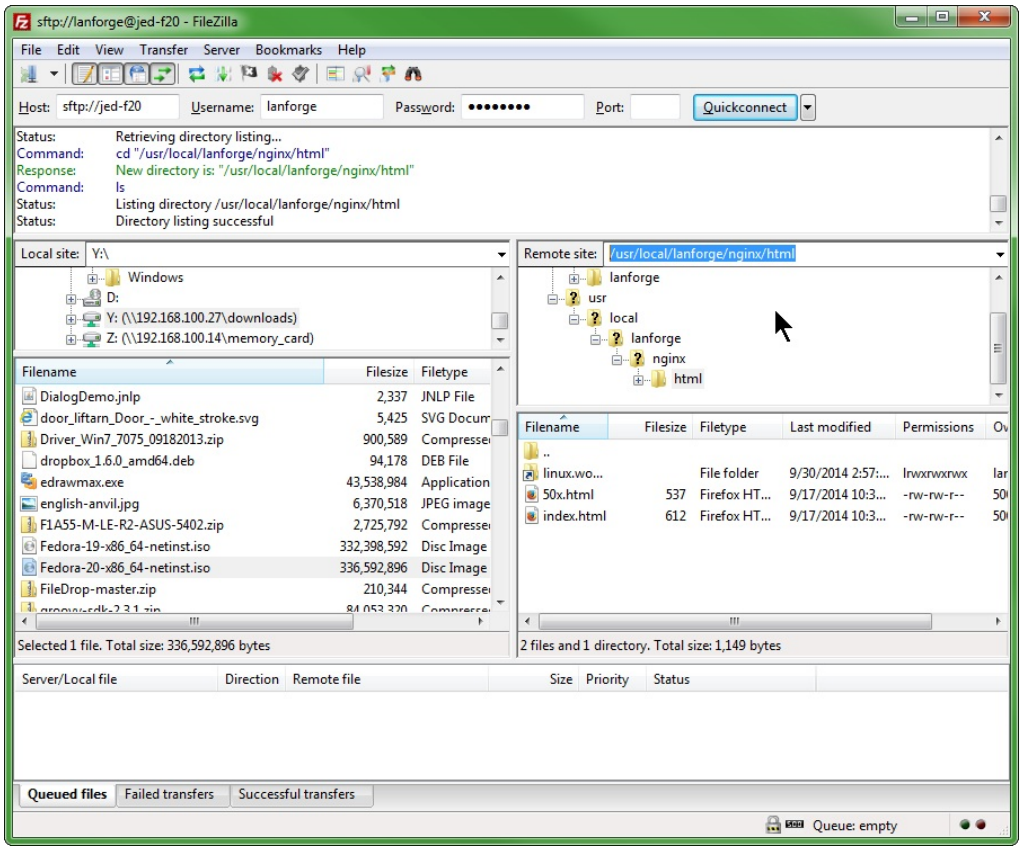
I. quick connect to sftp://192.168.100.40 (also named jed-f20 in this example)

II. using username `lanforge` and password `lanforge`

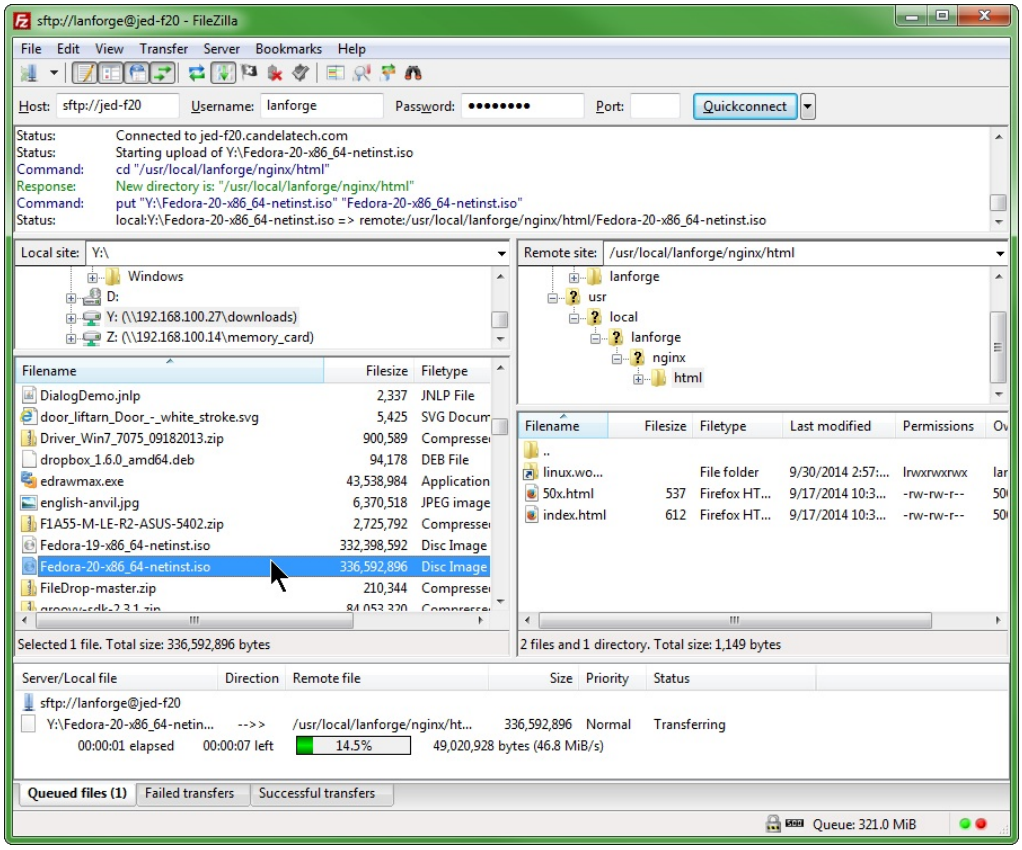
B. In the left column, choose your local directory (Y:\downloads)



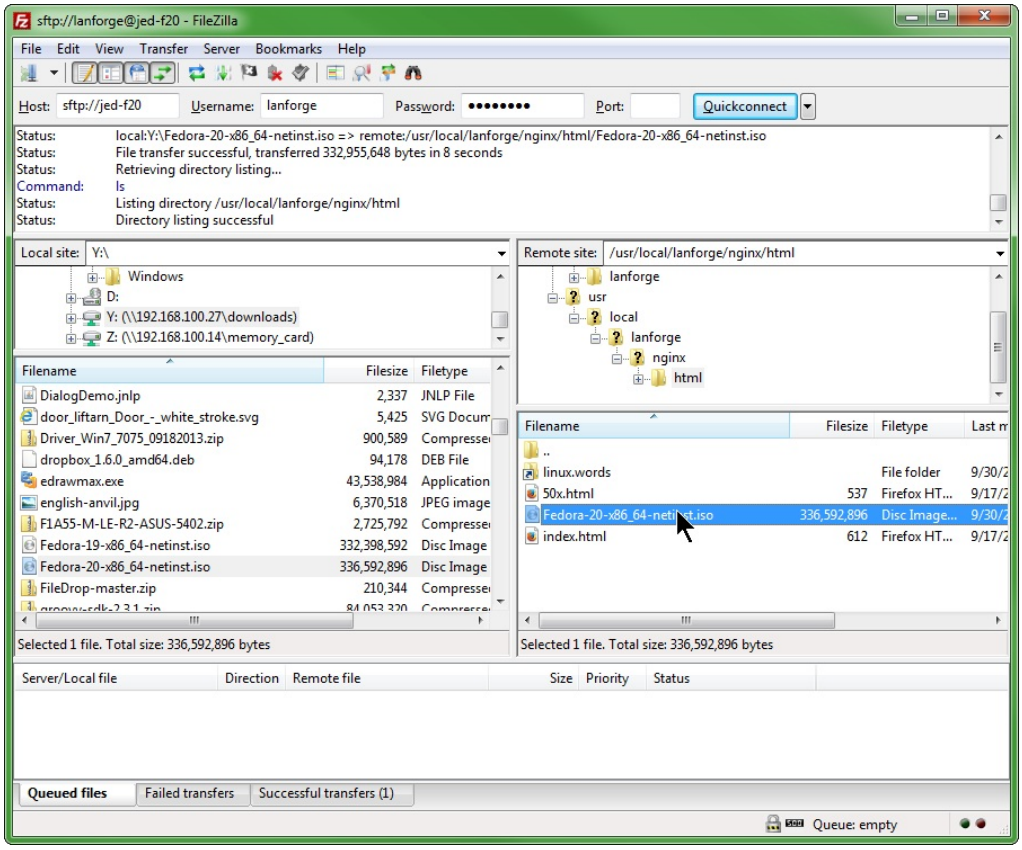
C. In the right column, type in the nginx document root, `/usr/local/lanforge/nginx/html` and hit enter



D. In the left column, double-click your DVD image (Fedora-20-x86_64.netinst.iso)

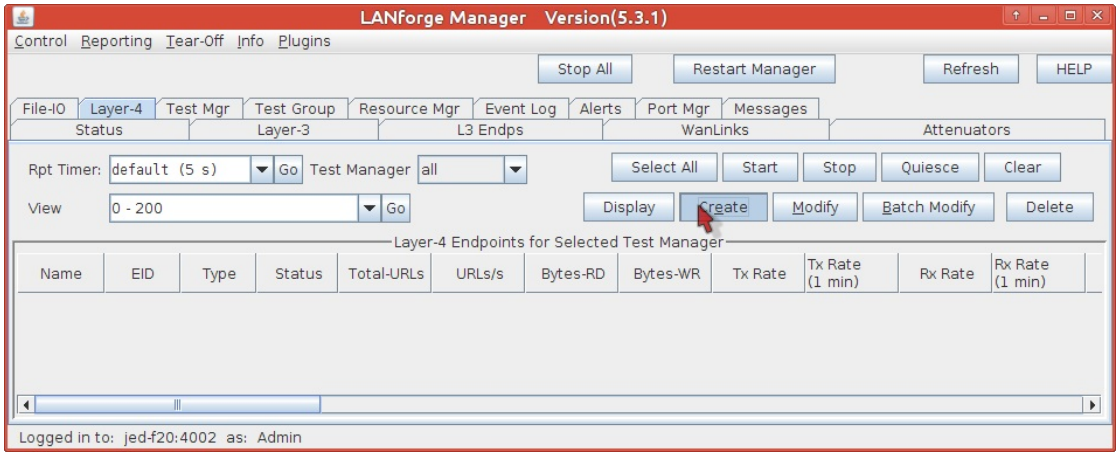


E. you will see the file when it is transferred appear in the right column



5. Configure application-layer traffic using the Layer 4-7 tab tools:

A. In the Layer 4-7 tab, click the **Create** button,



B. You will see the Create/Modify L4Endpoint window

C. Create a connection named `web-requests`,

D. With a Report timer of `fast (1 s)`

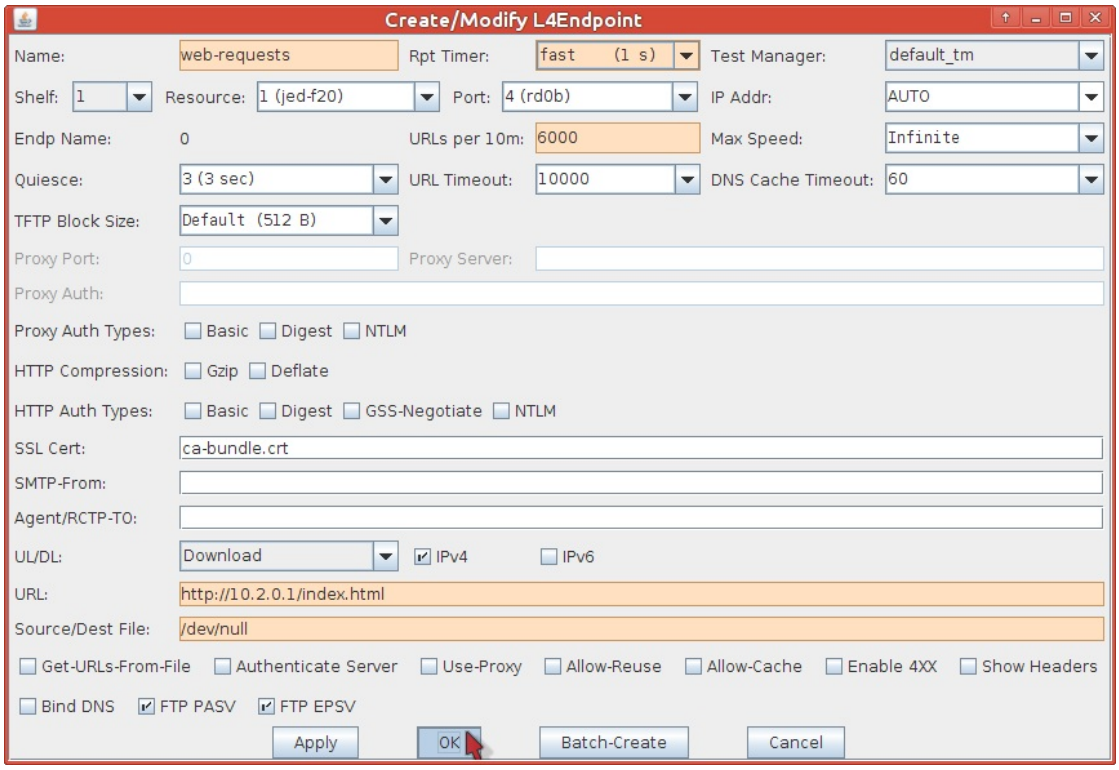
E. from Resource (jed-f20) port `rd0b`.

F. Let's make ten requests a second. Enter `6000` for URLs per 10m.

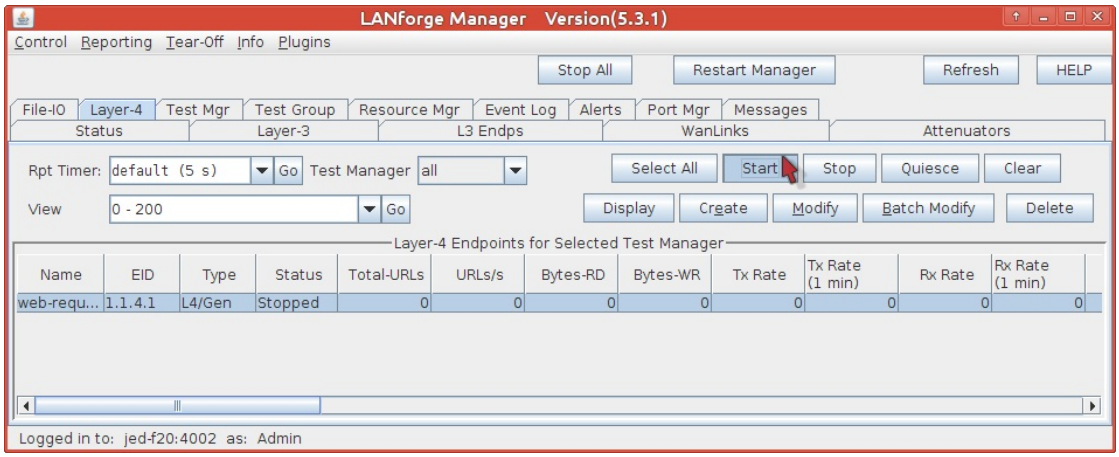
G. Jump down to the URL field, and enter `http://10.2.0.1/index.html`

H. and to avoid saving the result, set the Dest File to `/dev/null`

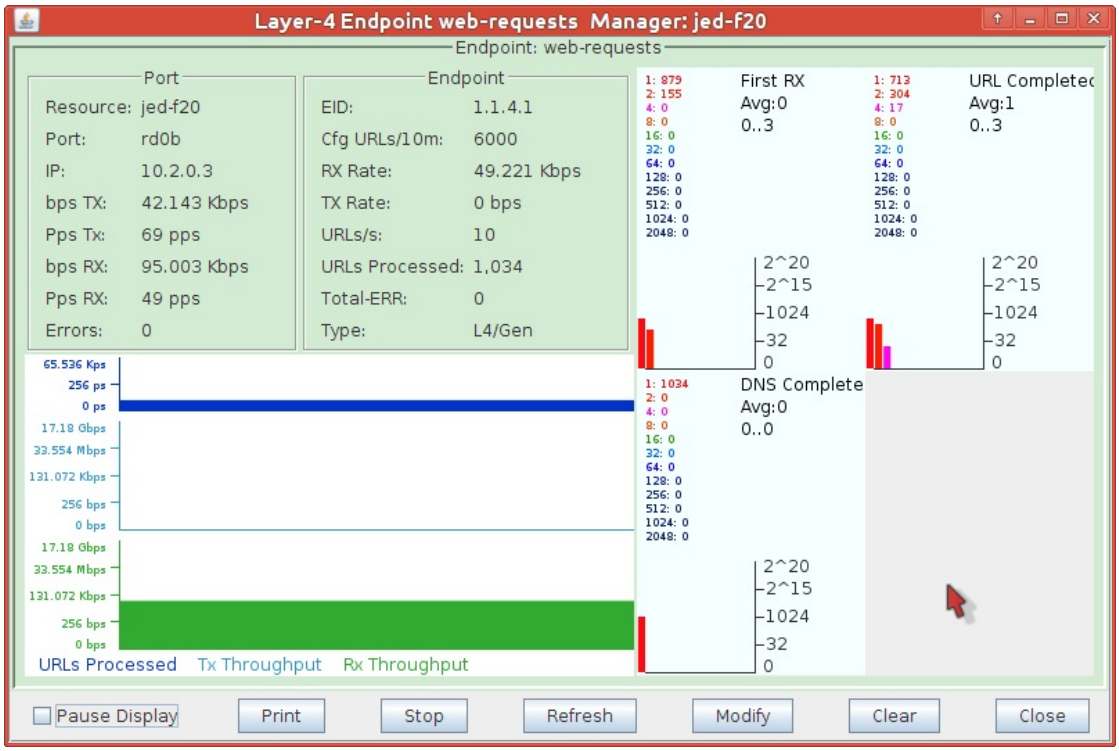
I. Click **OK** and the window will close.



6. Running the traffic is simple:
   A. Select the row named **web-requests** in the Layer 4-7 table,
   B. Click the **Start** button.



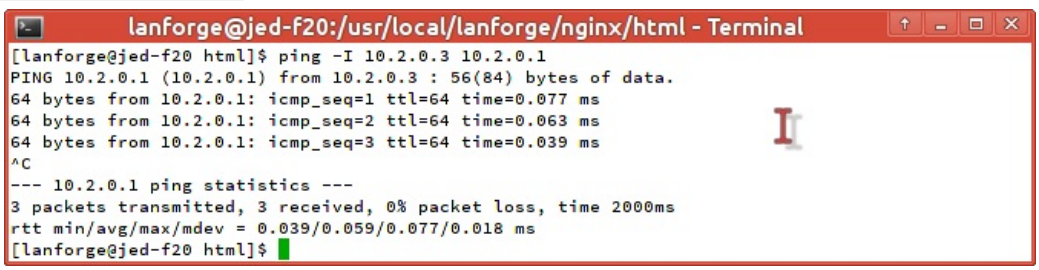
C. Click the **Display** button to see more detail on the traffic:



7. It is also possible to verify your connection setup from the command-line:

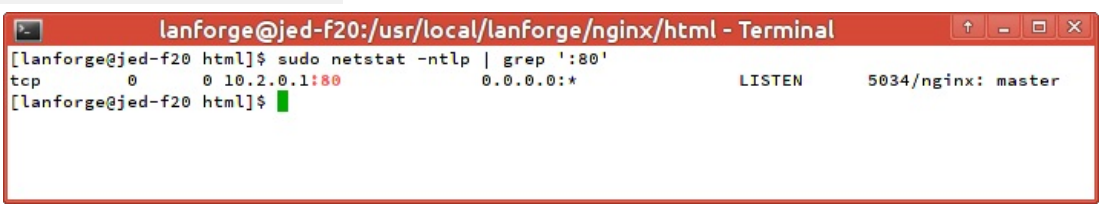
A. To verify the Layer 3 connection, you can use ping:

```
ping -I 10.2.0.3 10.2.0.1
```



B. To see that nginx is listening on port 80, use netstat on the resource running nginx and see what is listening on port 80:

```
sudo netstat -ntlp | grep ':80'
```



C. To see the output of the web request, you can use LANforge's version of curl located in `/home/lanforge/local/bin/curl`. Since it uses bundled libraries, first type:

```
cd /home/lanforge && . ./lanforge.profile
```

This sets up all environment variables and paths to work with the binaries and libraries shipped with LANforge.

D. Verify it runs without complaint:
`/home/lanforge/local/bin/curl --help`

```
lanforge@jed-f20:~ - Terminal
-T, --upload-file FILE  Transfer FILE to destination
    --url URL       URL to work with
-B, --use-ascii     Use ASCII/text transfer
-u, --user USER[:PASSWORD]  Server user and password
    --tlsuser USER  TLS username
    --tlspassword STRING TLS password
    --tlsauthtype STRING  TLS authentication type (default SRP)
-A, --user-agent STRING  User-Agent to send to server (H)
-v, --verbose       Make the operation more talkative
-V, --version       Show version number and quit
-w, --write-out FORMAT  What to output after completion
    --xattr        Store metadata in extended file attributes
-q                 If used as the first parameter disables .curlrc
[lanforge@jed-f20 ~]$
```

E. If you scroll up in your terminal, you will see options available in the output not found in a normal version of curl, like
--dns_server, --dns_interface, --dns_ip4_addr, --dns_ip6_addr, --interface and --localaddr. We will use **--interface** next.

```
lanforge@jed-f20:~ - Terminal
    --hostpubmd5 MD5  Hex encoded MD5 string of the host public key. (SSH)
-0, --http1.0       Use HTTP 1.0 (H)
    --ignore-content-length  Ignore the HTTP Content-Length header
-i, --include       Include protocol headers in the output (H/F)
-k, --insecure      Allow connections to SSL sites without certs (H)
    --interface INTERFACE  Specify network interface/address to use
    --localaddr IPADDR  Specify local IP-address to use
-4, --ipv4          Resolve name to IPv4 address
-6, --ipv6          Resolve name to IPv6 address
-j, --junk-session-cookies Ignore session cookies read from file (H)
    --keepalive-time SECONDS  Interval between keepalive probes
    --key KEY       Private key file name (SSL/SSH)
    --key-type TYPE Private key file type (DER/PEM/ENG) (SSL)
    --krb LEVEL     Enable Kerberos with specified security level (F)
    --libcurl FILE  Dump libcurl equivalent code of this command line
```

F. Craft your download command: `cd /home/lanforge/local/bin`
`./curl --interface 10.2.0.3 'http://10.2.0.1/index.html'`

```
lanforge@jed-f20:~/local/bin - Terminal
[lanforge@jed-f20 ~]$
[lanforge@jed-f20 ~]$ cd /home/lanforge/local/bin
[lanforge@jed-f20 bin]$ ./curl --interface 10.2.0.3 'http://10.2.0.1/index.html'
```

G. You will see the html output in the terminal:

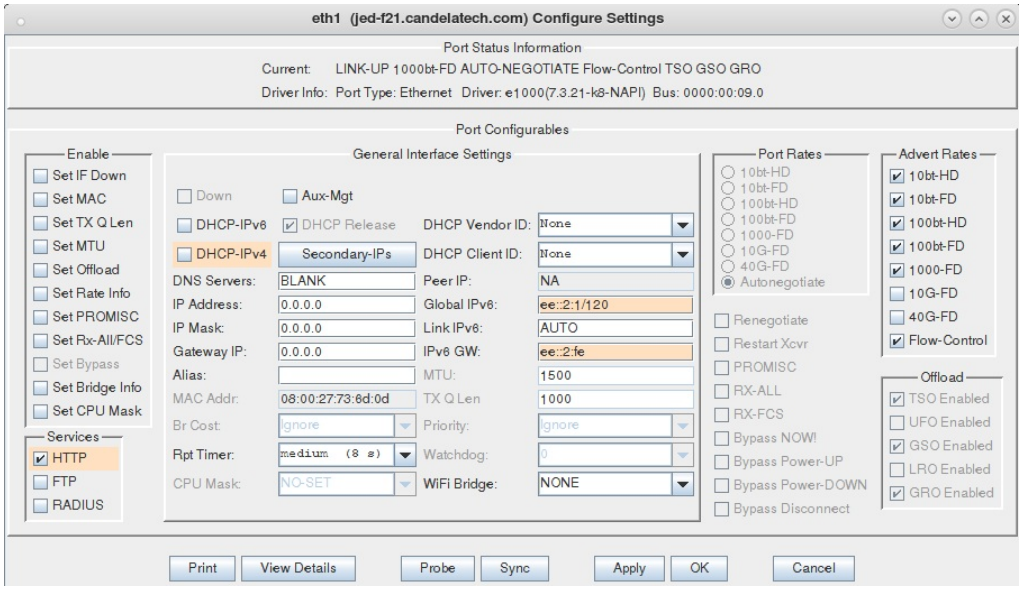
```
lanforge@jed-f20:~/local/bin - Terminal
[lanforge@jed-f20 ~]$ cd /home/lanforge/local/bin
[lanforge@jed-f20 bin]$ ./curl --interface 10.2.0.3 'http://10.2.0.1/index.html'
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
[lanforge@jed-f20 bin]$
```

H. There will be a record of connections and errors in the nginx logs directory: `cd /usr/local/lanforge/nginx/logs`

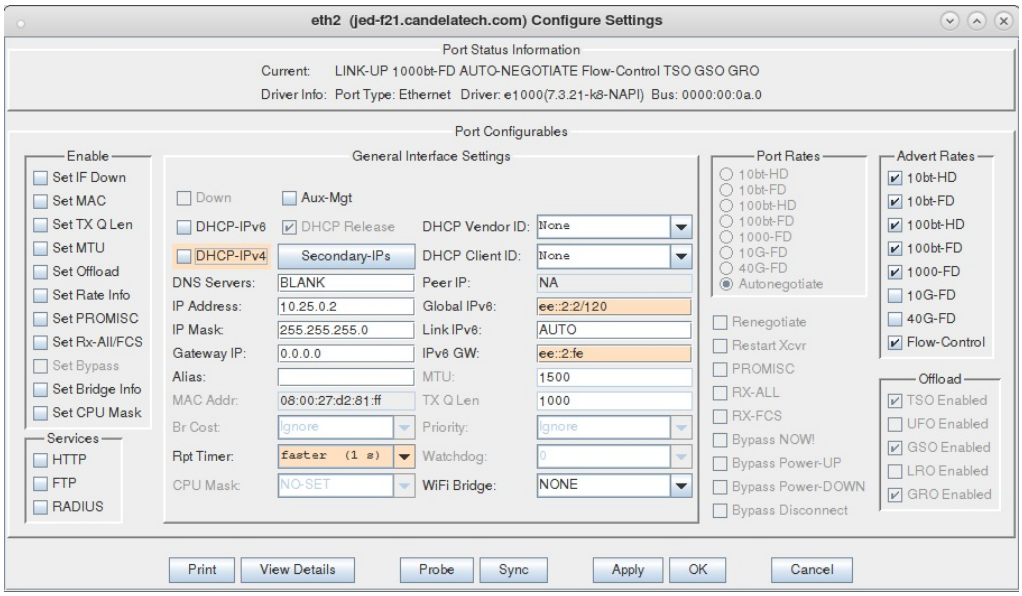
```
lanforge@jed-f20:/usr/local/lanforge/nginx/logs - Terminal
[lanforge@jed-f20 logs]$ cd /usr/local/lanforge/nginx/logs
[lanforge@jed-f20 logs]$ pwd
/usr/local/lanforge/nginx/logs
[lanforge@jed-f20 logs]$ ls -ltra
total 804
-rw-r--r--  1 root root      0 Sep 30 14:24 error.log
-rw-r--r--  1 root root     82 Sep 30 14:24 rd0a_error.log
-rw-r--r--  1 root root      0 Sep 30 14:24 rd0a_access.log
drwxrwxr-x 11  500  500   4096 Sep 30 14:24 ..
drwxrwxr-x  2  500  500   4096 Sep 30 14:24 .
-rw-r--r--  1 root root 807684 Sep 30 16:20 rd0a_host.access.log
[lanforge@jed-f20 logs]$
```

8. [Advanced] Creating IPv6 traffic to `nginx`. We will assume a scenario where `eth1` will serve nginx requests and `eth2` will generate requests.
   A. Add IPv6 addresses to your ports. We will add
      - `ee::2:1/120` for `eth1`,
      - `ee::2:2/120` for `eth2`,
      - and set the gateways for them to `ee::2:fe`. We won't actually use a gateway, but our port configuration requires it.

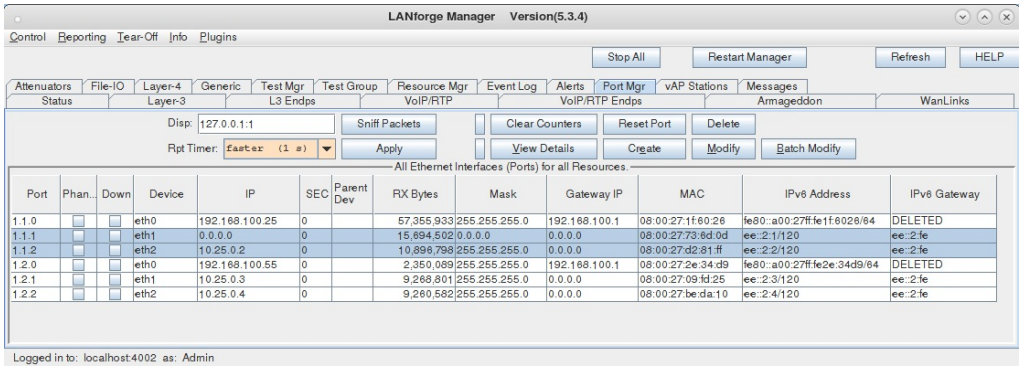
A. Here is eth1, and we will enable the **HTTP** service as well.



B. Here is `eth2`, notice how we added an IPv4 address to the port. Do that so that the Create Layer 4-7 dialog does not complain.



C. Here is a picture of the Port Mgr tab, with ports on two resources configured.



D. Verify your connection between ports using **ping6**: `$ ping6 -I ee::2:2%eth2 ee::2:1`

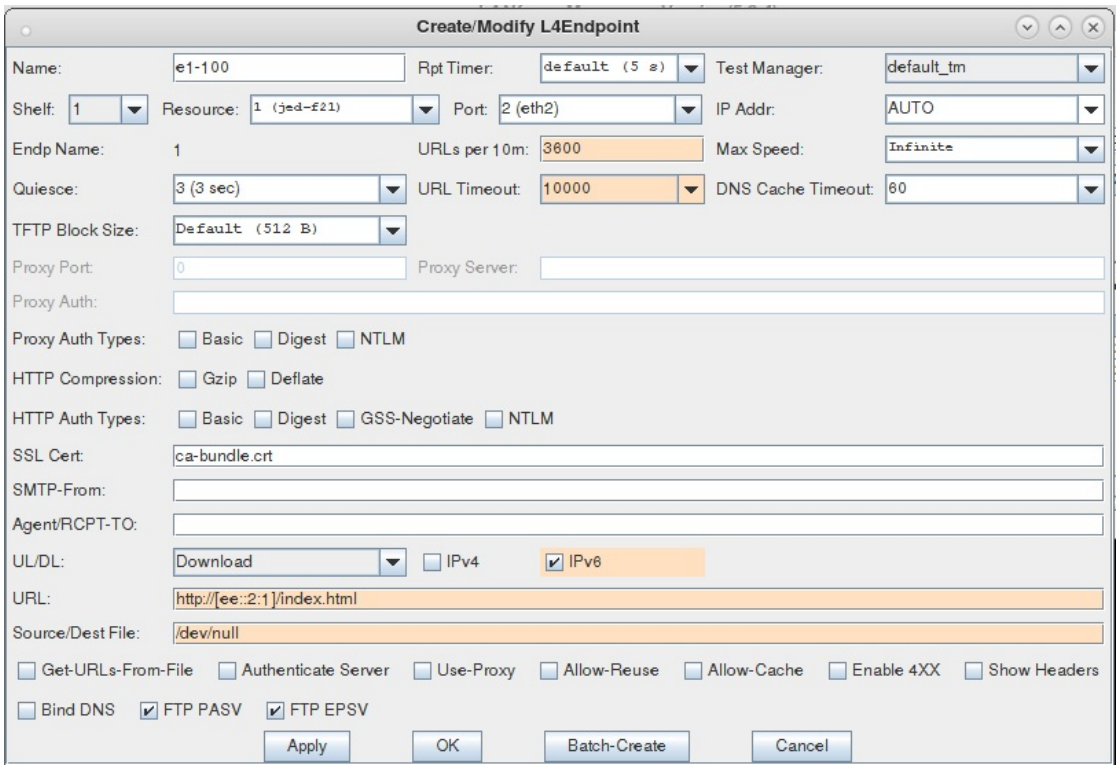
B. Edit your `/home/lanforge/vr_conf/nginx_eth1.conf` file. You will remove the header comment and alter the `listen` statements to include IPv6 addresses. You can listen for `[::]` or you could listen for `[ee::2:1]`

```
worker_processes        1;
error_log logs/eth1_error.log;
pid             /home/lanforge/vr_conf/nginx_eth1.pid;
events {
    worker_connections  1024;
}

http {
    include       /usr/local/lanforge/nginx/conf/mime.types;
    default_type  application/octet-stream;
    access_log  logs/eth1_access.log;
    sendfile            on;
    keepalive_timeout  65;

    server {
        listen       0.0.0.0:80 bind_dev=eth1;
        listen       [::]:80 bind_dev=eth1;
        server_name  ww6-eth1;
        access_log  logs/eth1_host.access.log;

        location / {
            root   html;
            index  index.html index.htm;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
}
```

C. If you provide an IPv6 address, please watch the nginx log file for errors: listening on an IPv6 address is easy to misconfigure. The log files can be found in `/usr/local/lanforge/nginx/logs`

D. Re-start the http service on `eth1` by un-checking **HTTP**, clicking **Apply**, checking **HTTP**, clicking **OK**.

E. Configure your Layer 4-7 endpoint:



A. Name your endpoint

B. Set your URLs per Minute

C. Specify the URL with brackes around the address: `http://[ee::2:1]/index.html`

D. Set your output file to `/dev/null`

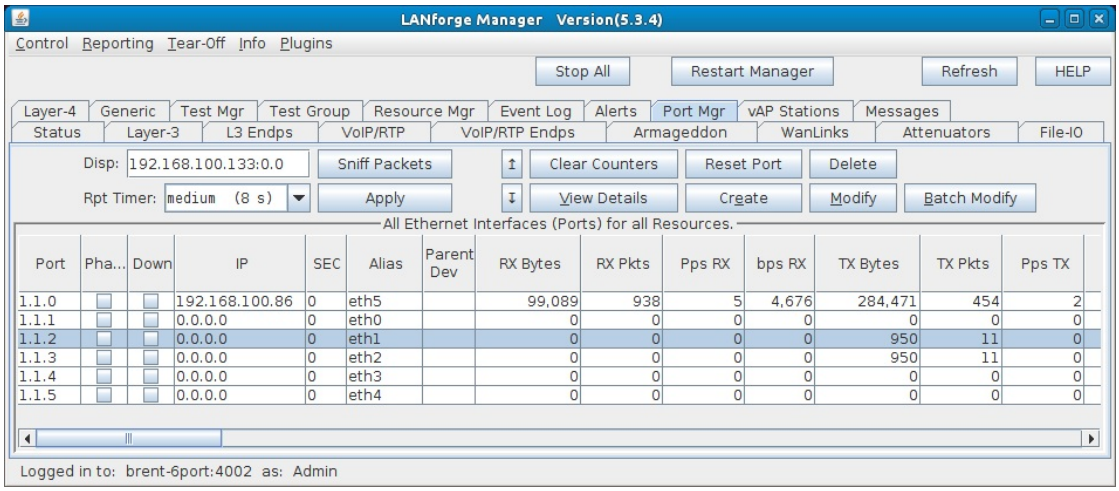
F. You can watch traffic on `eth1` or `eth2` to verify the web requests.

Generating FTP Traffic Between Two Physical ports

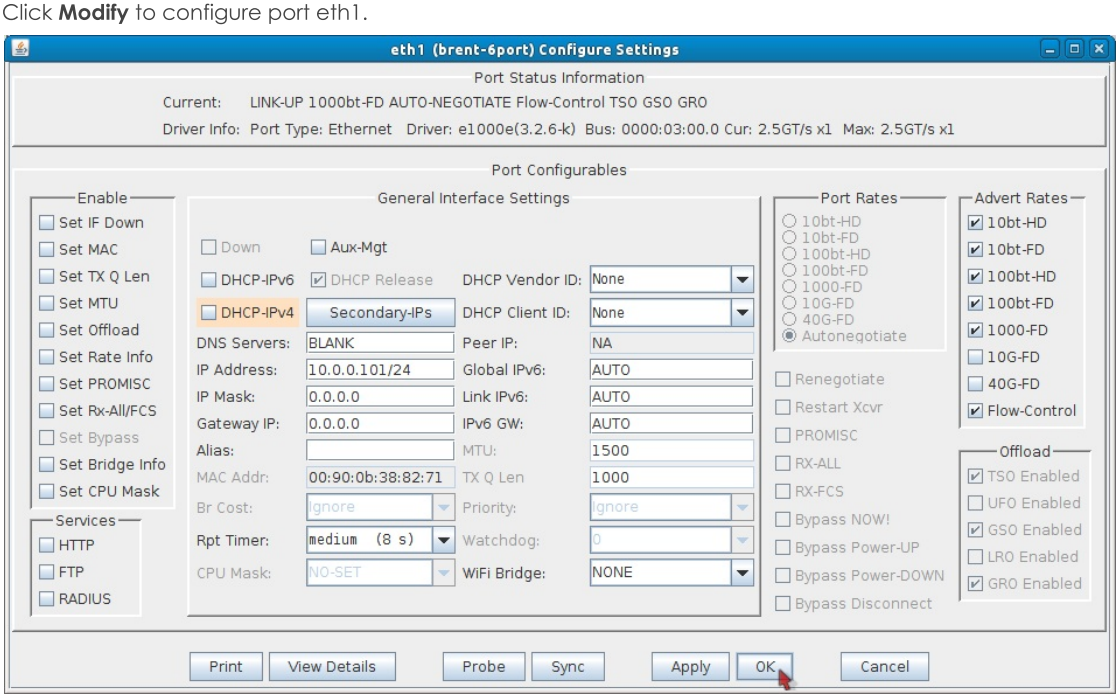
Goal: Allow LANforge to simulate a FTP connection within the same system.

This scenario is useful if you wish to generate stateful FTP traffic between two physical ports using a single LANforge system. You will need two non-management physical ports connected to eachother with a loopback cable or switch.

1. Configure the physical interfaces. We will set up eth1 as a client and eth2 as a FTP server.
   A. Go to the **Port Mgr** tab and select eth1.

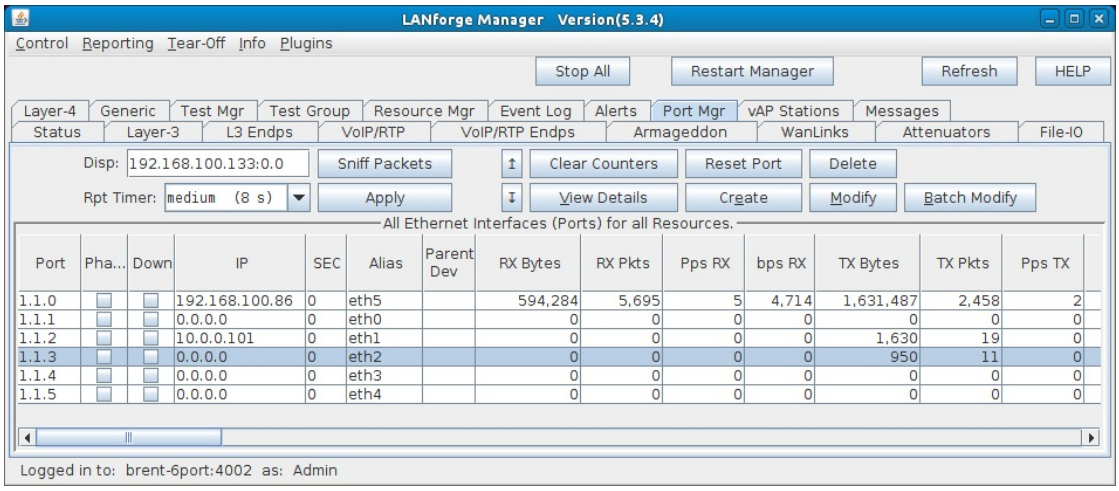


   B. Click **Modify** to configure port eth1.

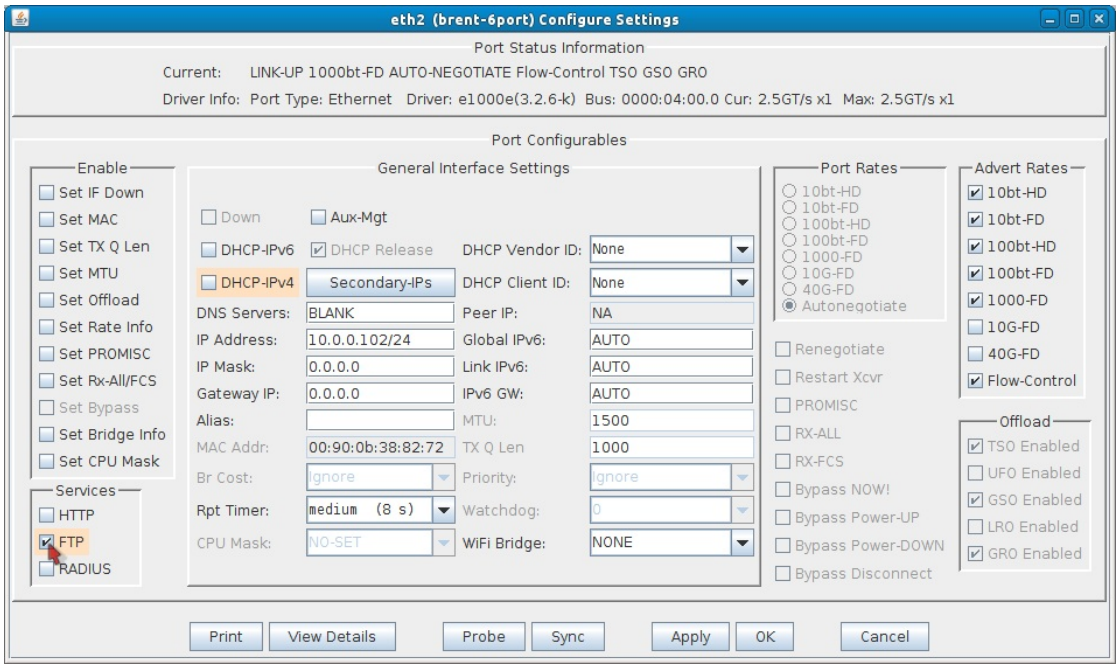


      A. Set the IP Address to `10.0.0.101/24`.
      B. Click **OK**.

C. While still in the **Port Mgr** tab, select eth2.



D. Click **Modify** to configure port eth2.



A. Set the IP Address to `10.0.0.102/24`.

B. Enable **FTP** under **Services**.

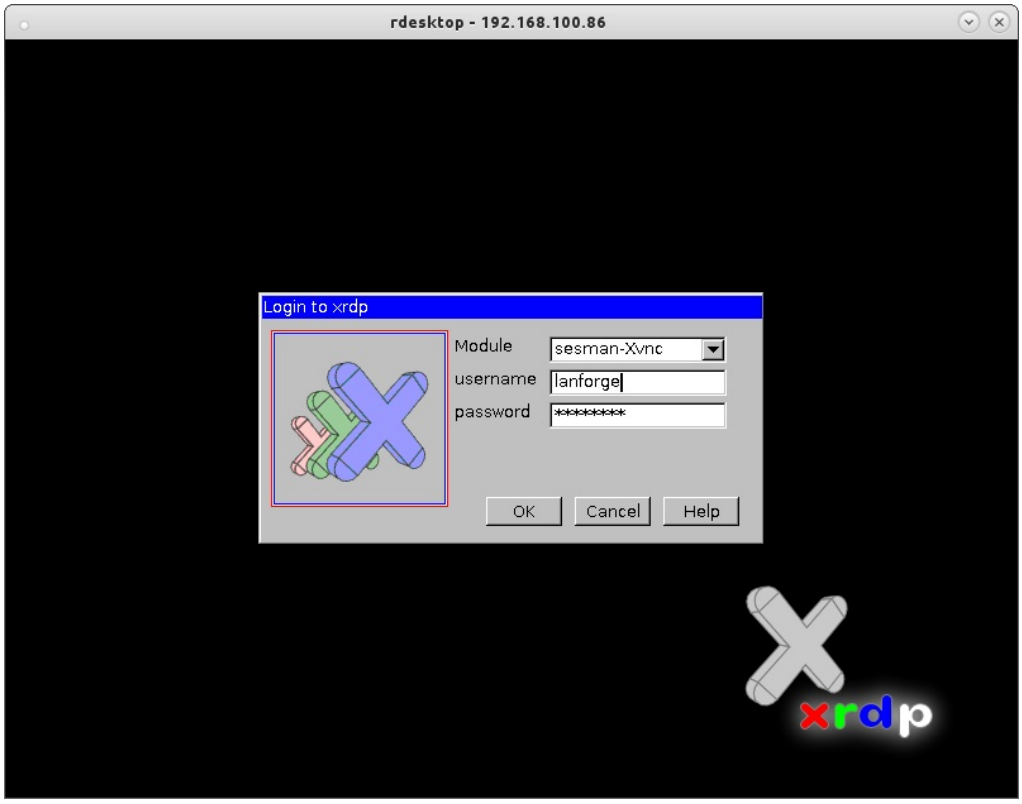
C. Click **OK**.

2. Create a file for the FTP client to download/upload. This is optional if you already have a file in mind to test with, just make sure the file is in `/home/lanforge` (the default directory for FTP). Step 3 explains how to upload a file from a Windows machine.

A. First, you'll need to open a console on the LANforge system.

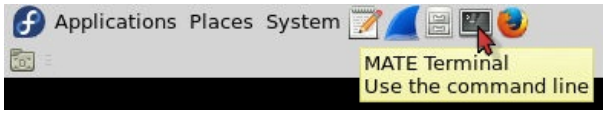
   A. Log into the machine **directly** or remotely through **rdesktop**. The user/password should be **lanforge/lanforge** for both.

   B. To log in via **rdesktop** type the following command into a console (replace LANforge-IP with the IP of your LANforge system):
   `rdesktop LANforge-IP`



   C. Log in with user/password **lanforge/lanforge**.

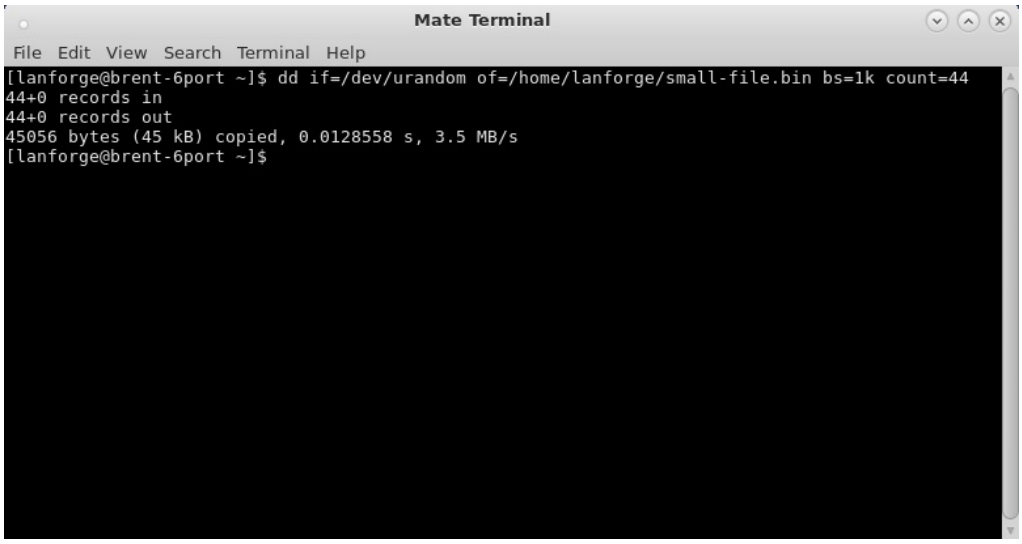
   D. Once you are on the system, a console should already be open. If you don't see a console, go ahead and open one by clicking the console icon in the top menu bar (shown in the below screenshot).



B. Once you have a console ready, use the below commands to create a small or large file. These files will just contain random text.
Note: These files will be created in the `/home/lanforge` directory. This is the default directory used for FTP on LANforge.

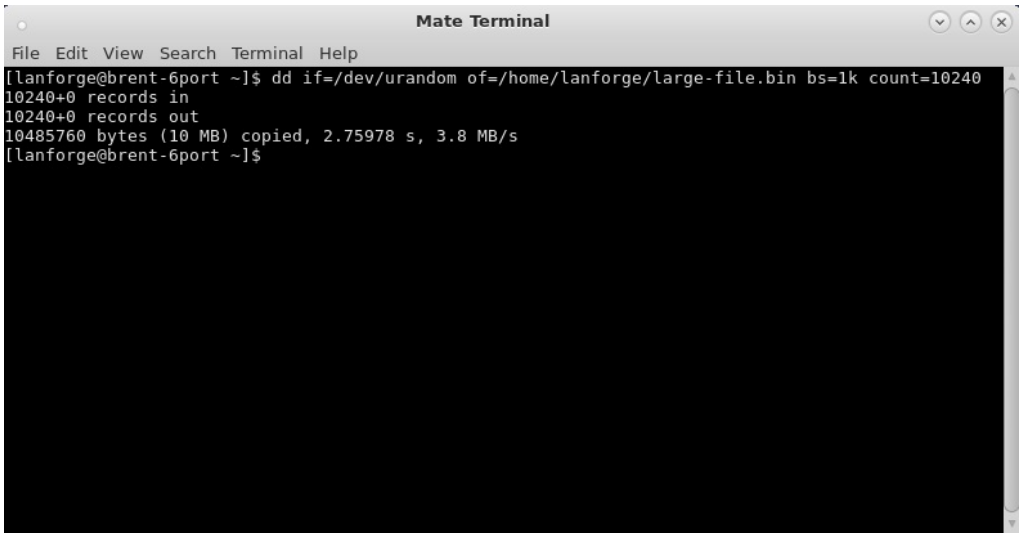
A. **Small file** (45 KB):

```
dd if=/dev/urandom of=/home/lanforge/small-file.bin bs=1k count=44
```



B. **Large file** (10 MB):

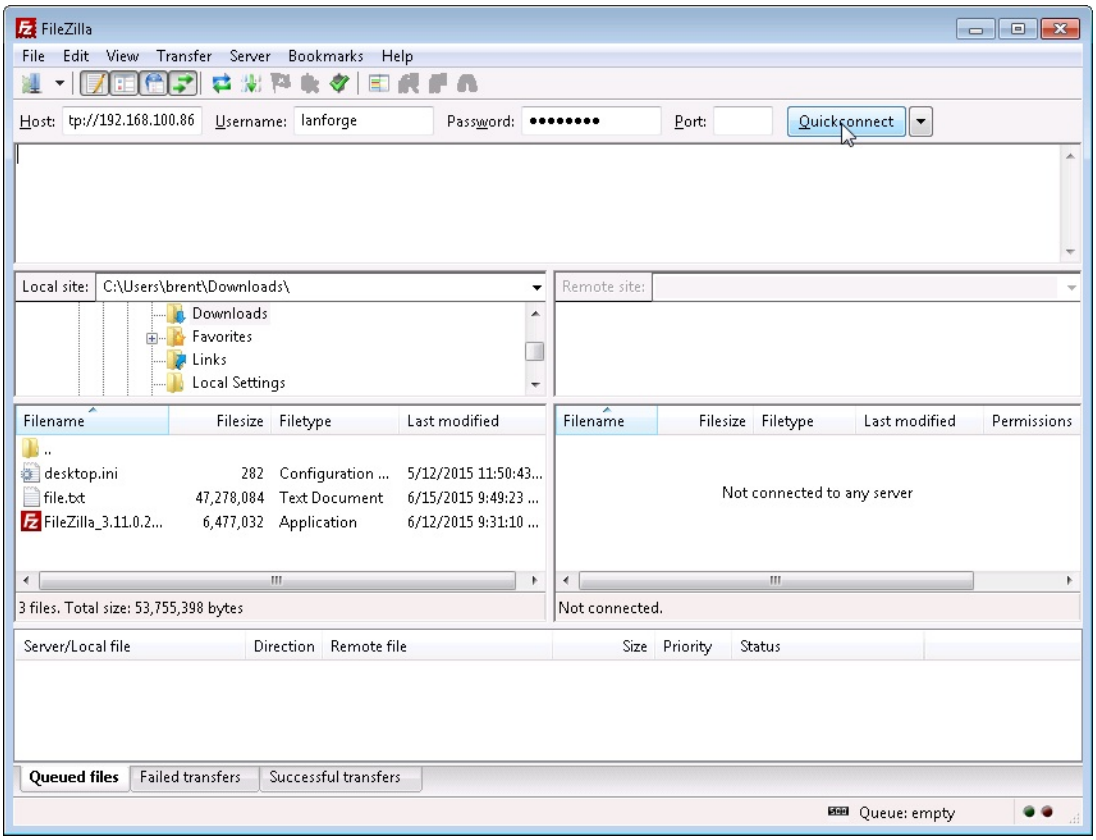
```
dd if=/dev/urandom of=/home/lanforge/large-file.bin bs=1k count=10240
```



3. **Optional:** Upload a file to the LANforge system for the FTP client to download/upload. There are multiple ways to do this, this step will demonstrate the upload using a Windows machine with FileZilla.

A. On your Windows machine, download and install FileZilla. You can download the FileZilla client here: https://filezilla-project.org/download.php

B. Open FileZilla.

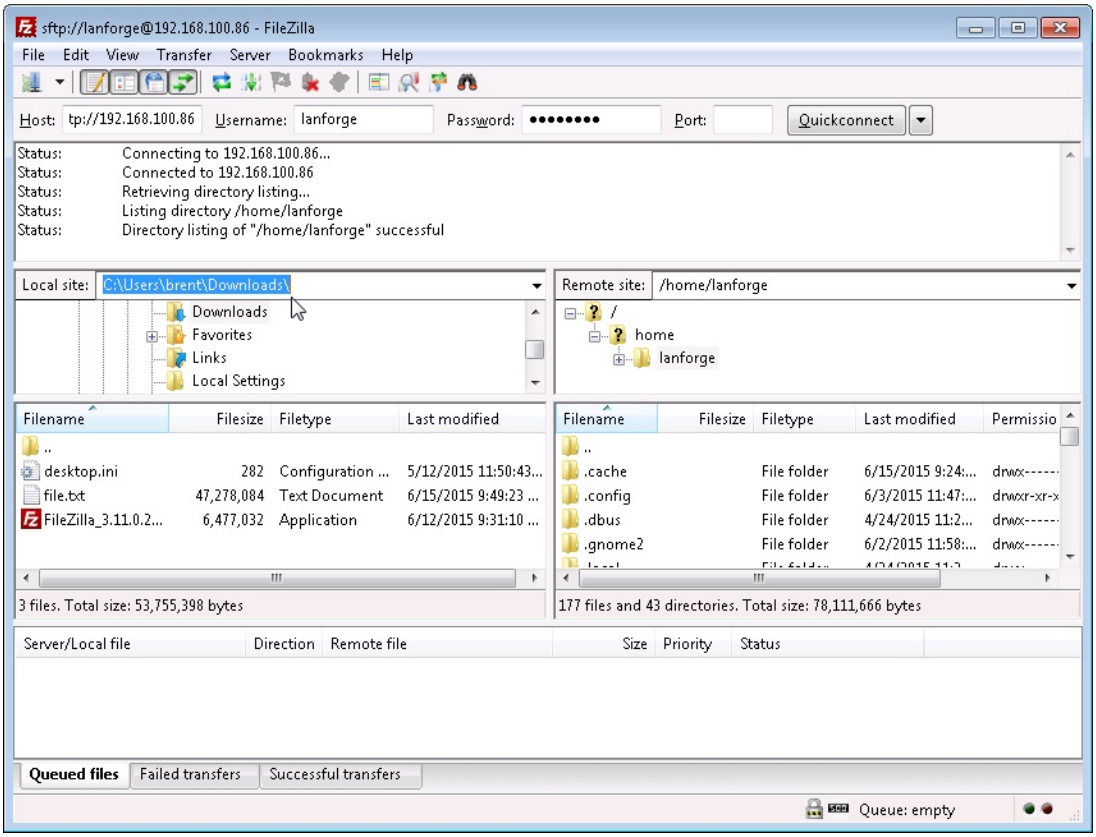


A. Near the top left in the **Host** field, enter in the following: `sftp://LANforge-IP` (sftp://192.168.100.86 is used in this example).

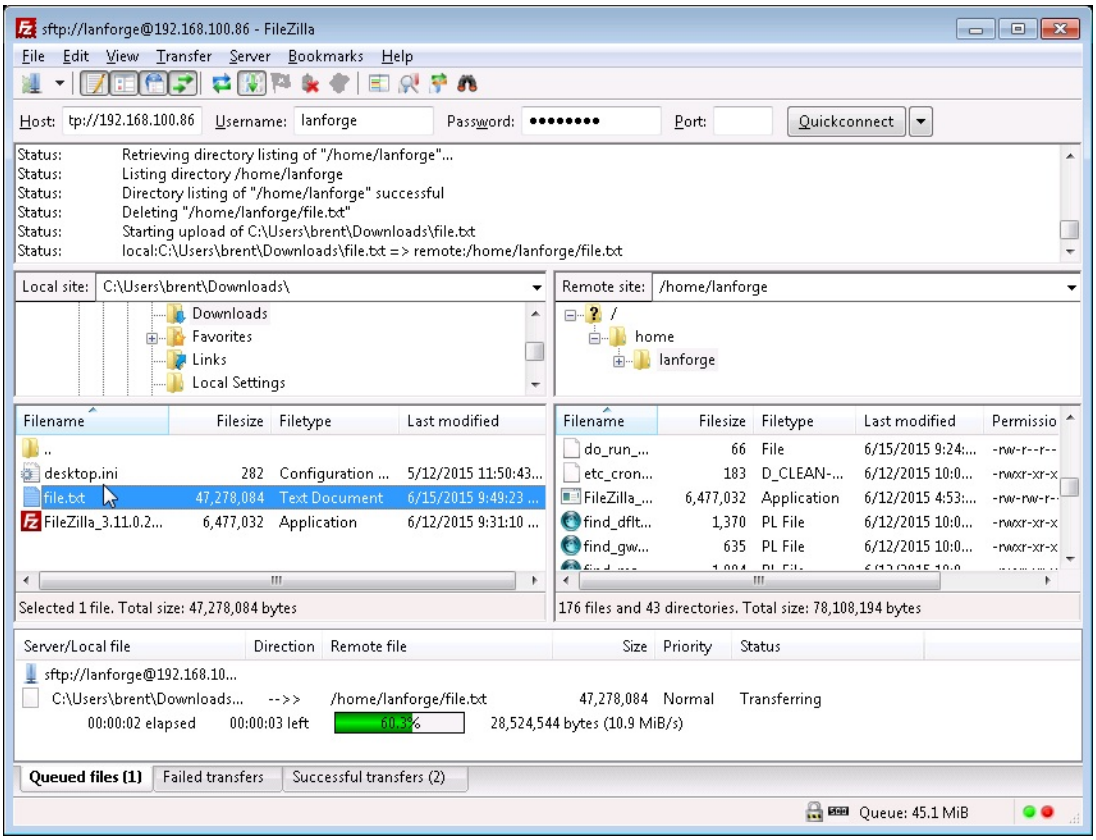
B. Username: **lanforge** Password: **lanforge**.

C. Click **Quickconnect**.

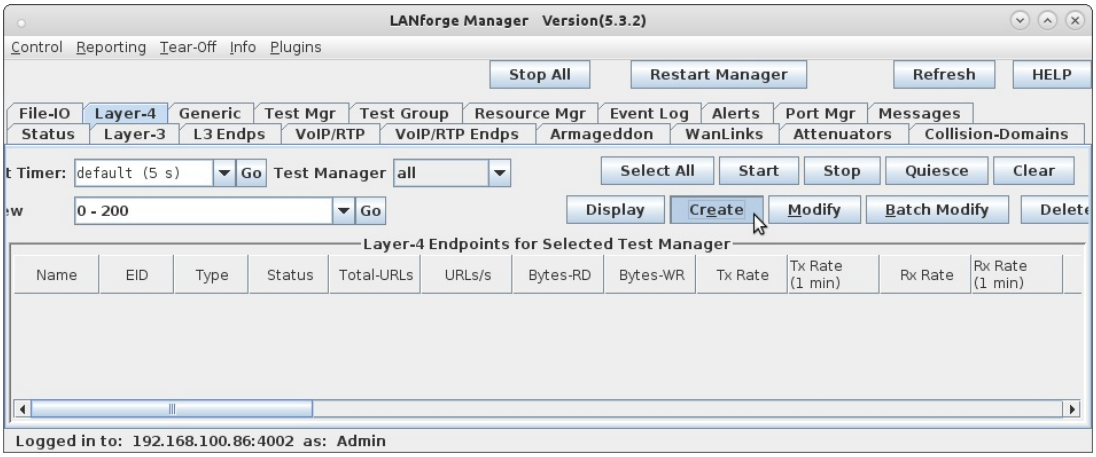
C. In the **left column** navigate to your file's directory. Make sure the right column is set to `/home/lanforge`



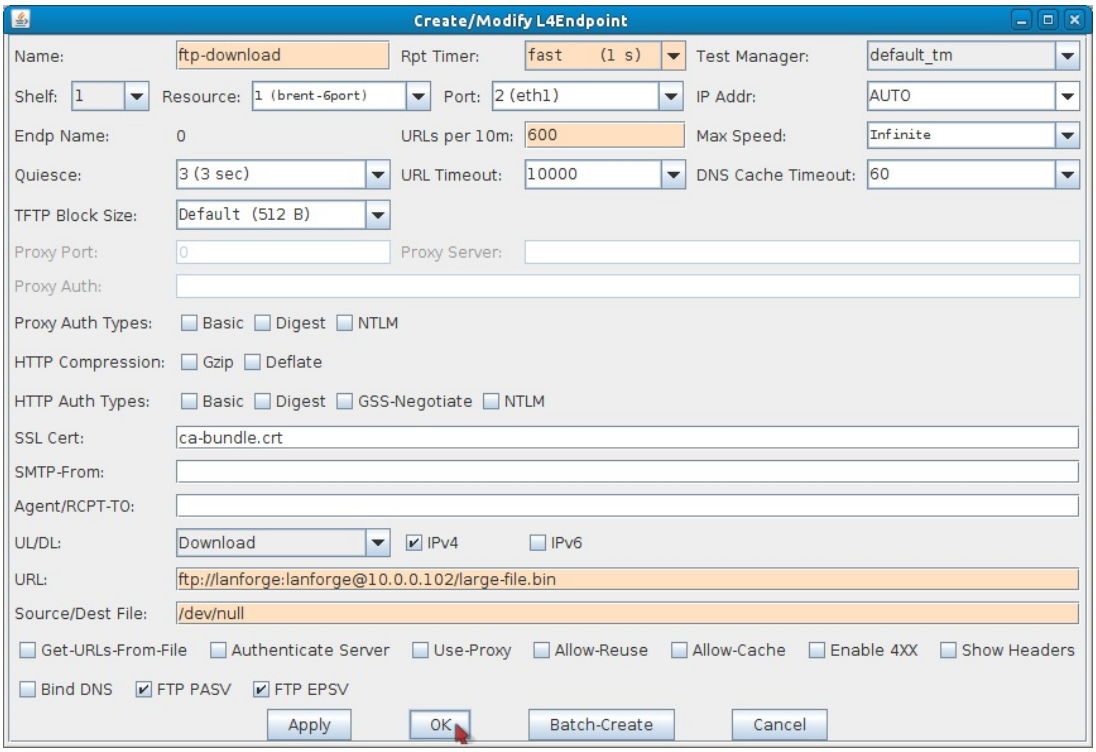
D. To start transferring your file, double click it in the left column. Once the transfer is complete make sure you can find it within the right column under `/home/lanforge`. Your file is now on the LANforge machine!



4. Create the FTP endpoint.
   A. On the **Layer 4-7** tab, click **Create**.



   B. Configure the FTP endpoint.



A. Name your endpoint **ftp-download**.

B. Set **Rpt Timer** to `fast (1 s)`

C. Set **Port** to `eth1`.

D. Set **URLs per 10m** to `600` (1/sec).

   I. Here is a list of common values used for **URLS per 10m**.

URLs per 10min	Converted
60	1/min
600	1/sec
6000	10/sec
60000	100/sec

E. To download the small file (44KB) created earlier, set **URL** to:
`ftp://lanforge:lanforge@10.0.0.102/small-file.bin`
To download the large file (10MB) created earlier, set **URL** to:
`ftp://lanforge:lanforge@10.0.0.102/large-file.bin`
Note: The default directory for FTP is `/home/lanforge` and the default username/password is lanforge/lanforge. All files downloaded/uploaded via FTP need to be in `/home/lanforge`.

F. **Optional:** If you want to use another file in `/home/lanforge` you can just change the filename at the end of the URL.
For example `ftp://lanforge:lanforge@10.0.0.102/large-file.bin` would be changed to `ftp://lanforge:lanforge@10.0.0.102/your-file-name`

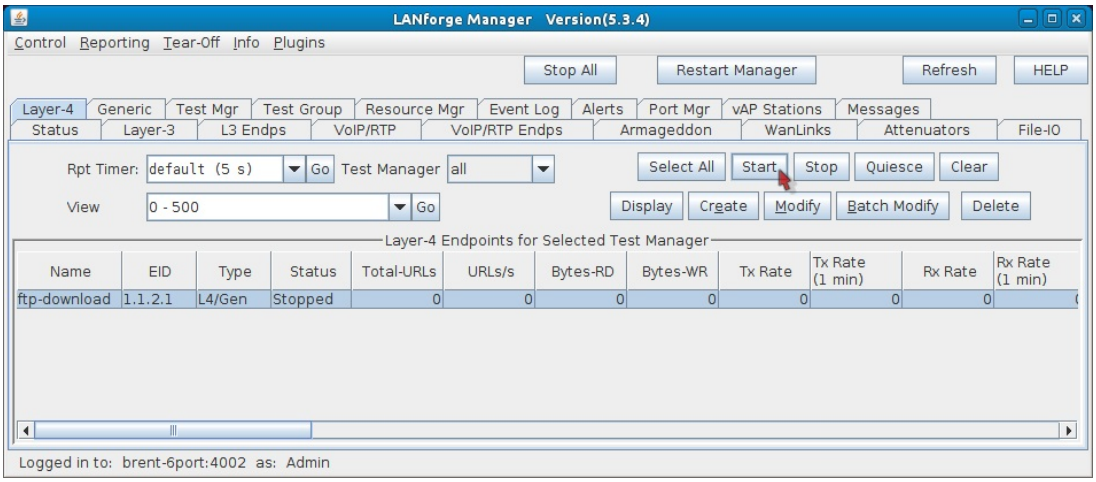
G. Set **Source/Dest File** to `/dev/null`.

H. Click **OK**.

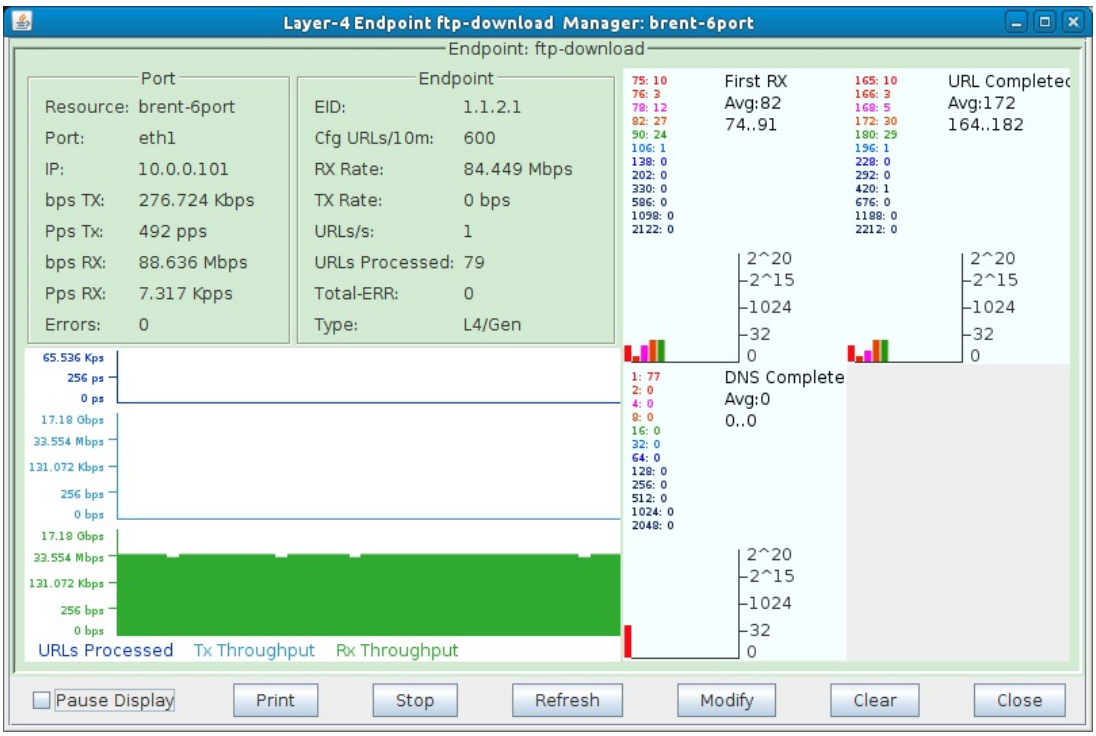
5. Start the endpoint and display the graph.

   A. In the **Layer 4-7** tab, select the connection called `ftp-download`.

   B. Click Start.

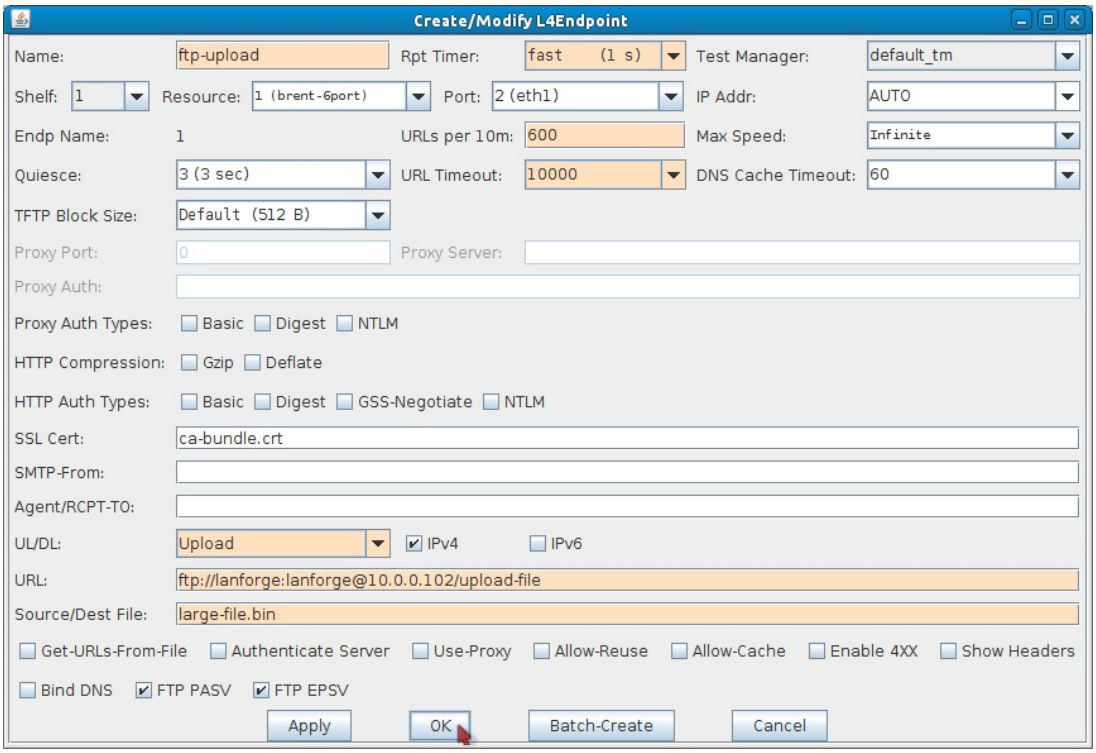


C. Select **ftp-download** and click **Display** to see a graph similar to below.



6. **Uploading** a file from the FTP client to the FTP server.

A. Go to the **Layer 4-7** tab and modify the **ftp-download** endpoint. The below changes will need to be made:



A. **Name** to **ftp-upload**.

B. **UL/DL** to **Upload**.

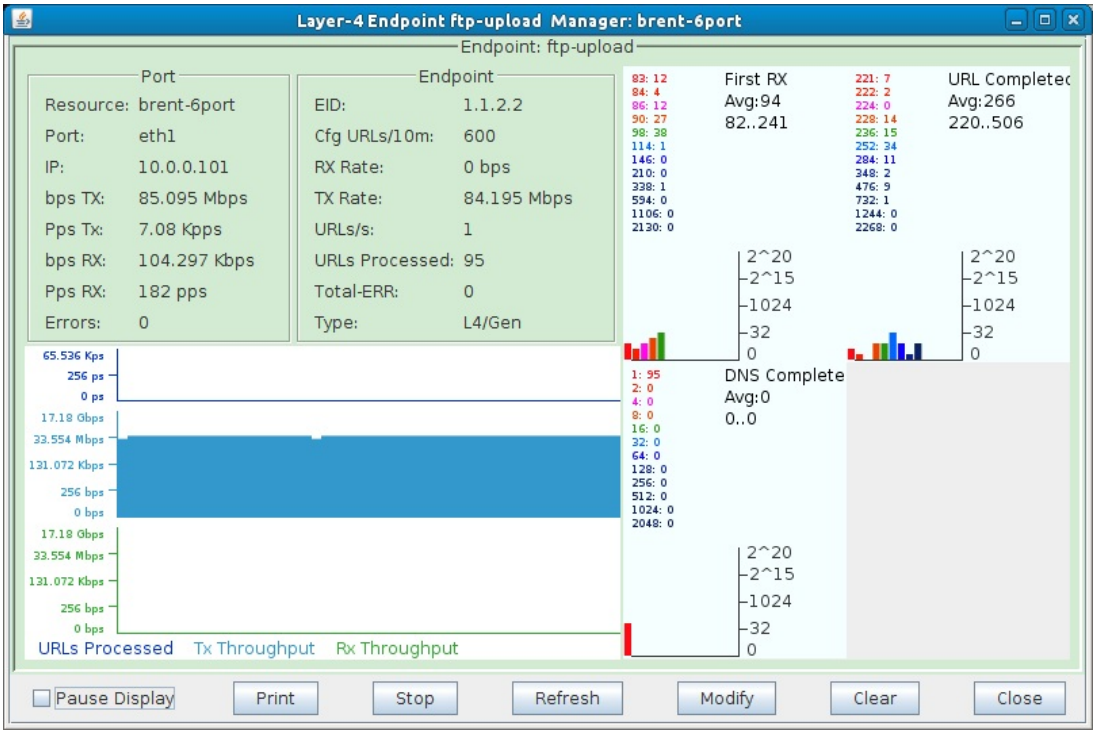
C. **URL** to **ftp://lanforge:lanforge@10.0.0.102/upload-file**. The 'upload-file' filename in this link can be changed if needed.

D. **Source/Dest File** to **large-file.bin**. This can be changed to a preferred file as long as it is in `/home/lanforge` (the default FTP directory).

E. Click **OK**.

B. Select ftp-upload and click **Start**.

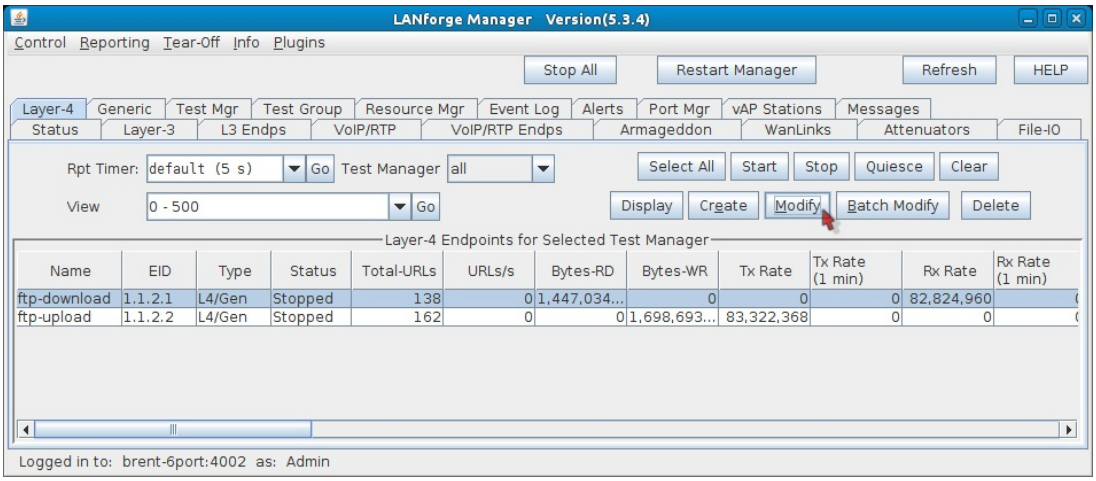
C. This endpoint will now essentially upload the large-file.bin file in `/home/lanforge` to the same directory with a new file name (**upload-file** in this case). If you select **ftp-upload** in the **Layer 4-7** tab and click **Display**, the graph should look similar to the one below.



7. Using **SCP**, **SFTP**, and **TFTP**.

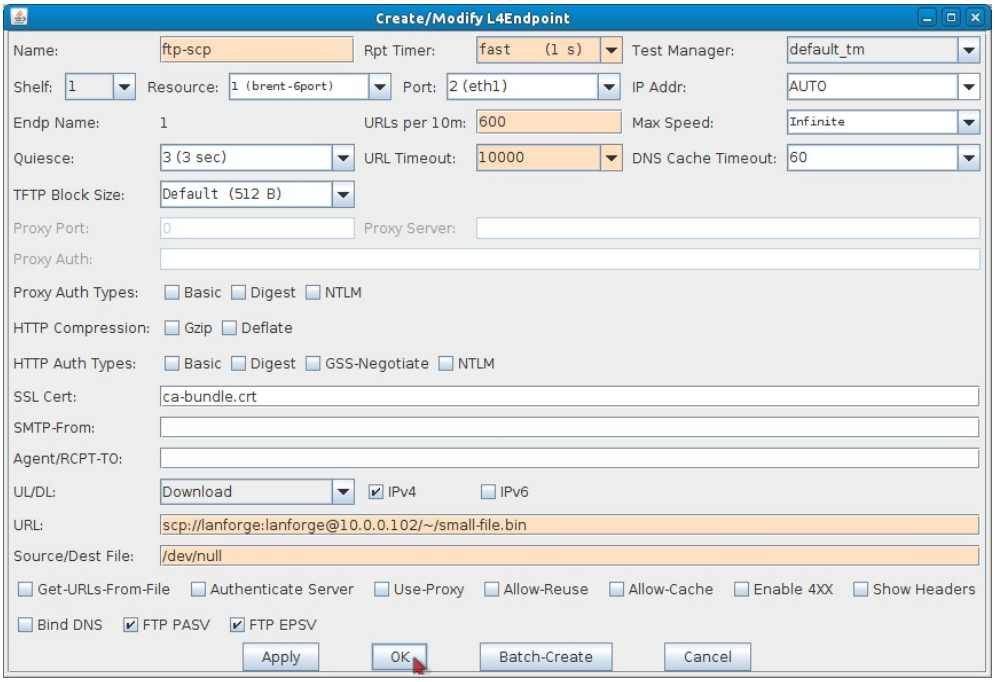
A. Go to the **Layer 4-7** tab and modify the **ftp-download** endpoint. The below changes will need to be

made:



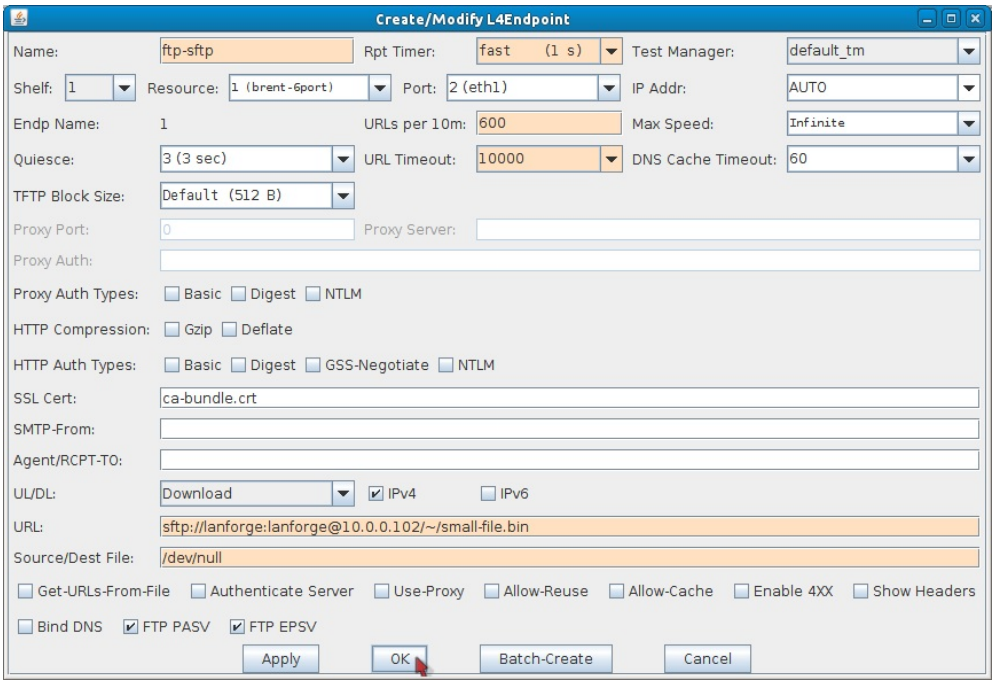
A. **Name** to **ftp-scp** for scp or **ftp-sftp** for sftp.

B. For **SCP**, set the **URL** to **scp://lanforge:lanforge@10.0.0.102/~/small-file.bin**. The filename at the end of the link can be changed to another file in `/home/lanforge`.



C. Click **OK**.

D. For **SFTP**, set the **URL** to **sftp://lanforge:lanforge@10.0.0.102/~/small-file.bin**. The filename at the end of the link can be changed to another file in `/home/lanforge`.

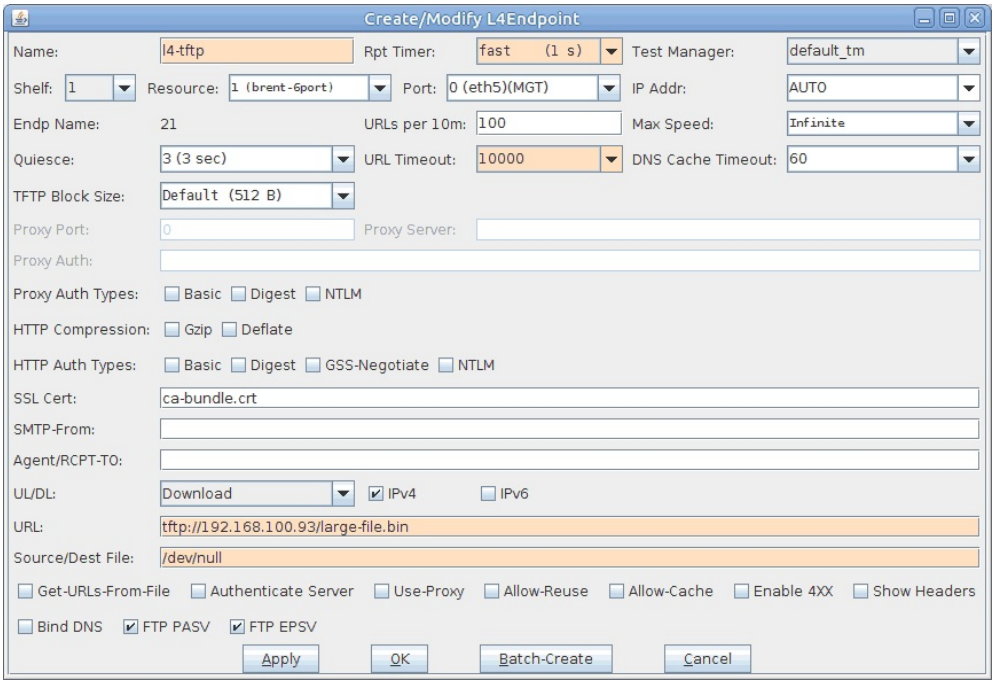


E. Click **OK**.

B. Select the **ftp-scp** or **ftp-sftp** endpoint and click **Start**.

C. To use **TFTP**, you will first need to have a TFTP server set up and configured to serve the Layer 4-7 endpoint a file.

A. Once the server is set up, use the below configuration to set up a TFTP endpoint.
Note: You will need to provide your own server's IP and file name.



I. Set the **Port** to a port that can talk to the TFTP server. In this case the server was on the management network so eth0 was used.

II. **URLs per 10m** was left at the default **100**, this can be set higher or lower depending on how often the file should be downloaded.

III. Set **URL** to **tftp:**//`server-IP`/`filename`.

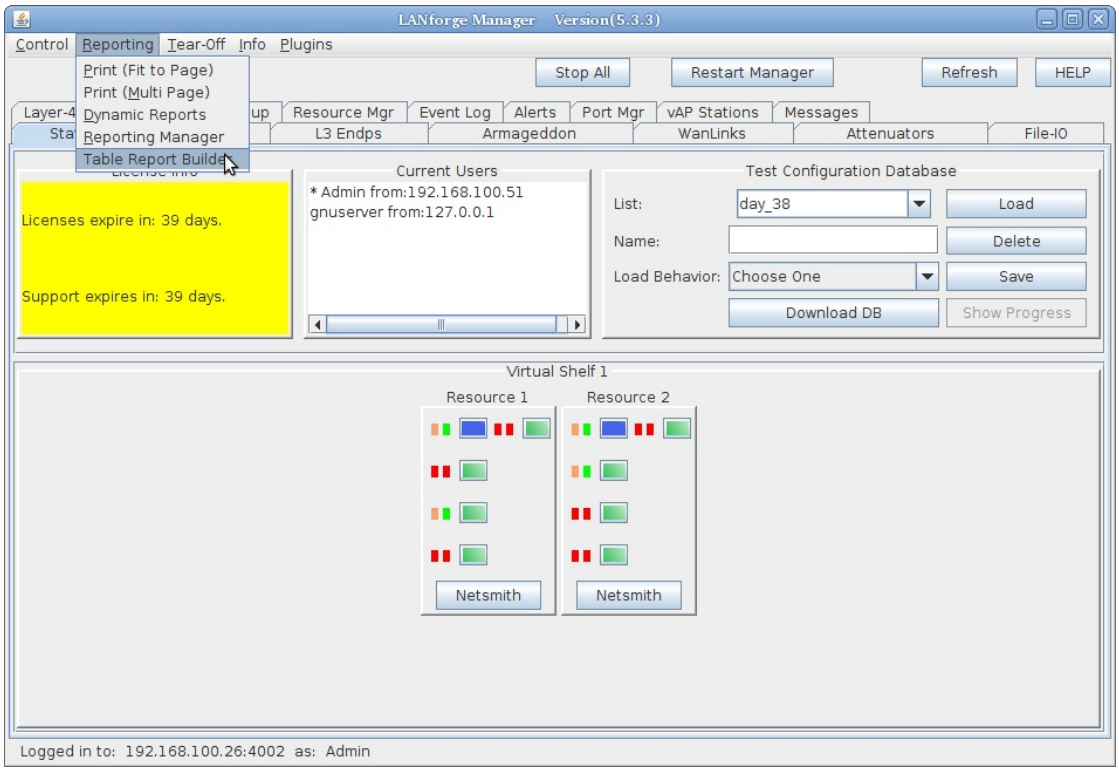
IV. **Source/Dest File** can be **/dev/null**.

Generating Table-based Reports

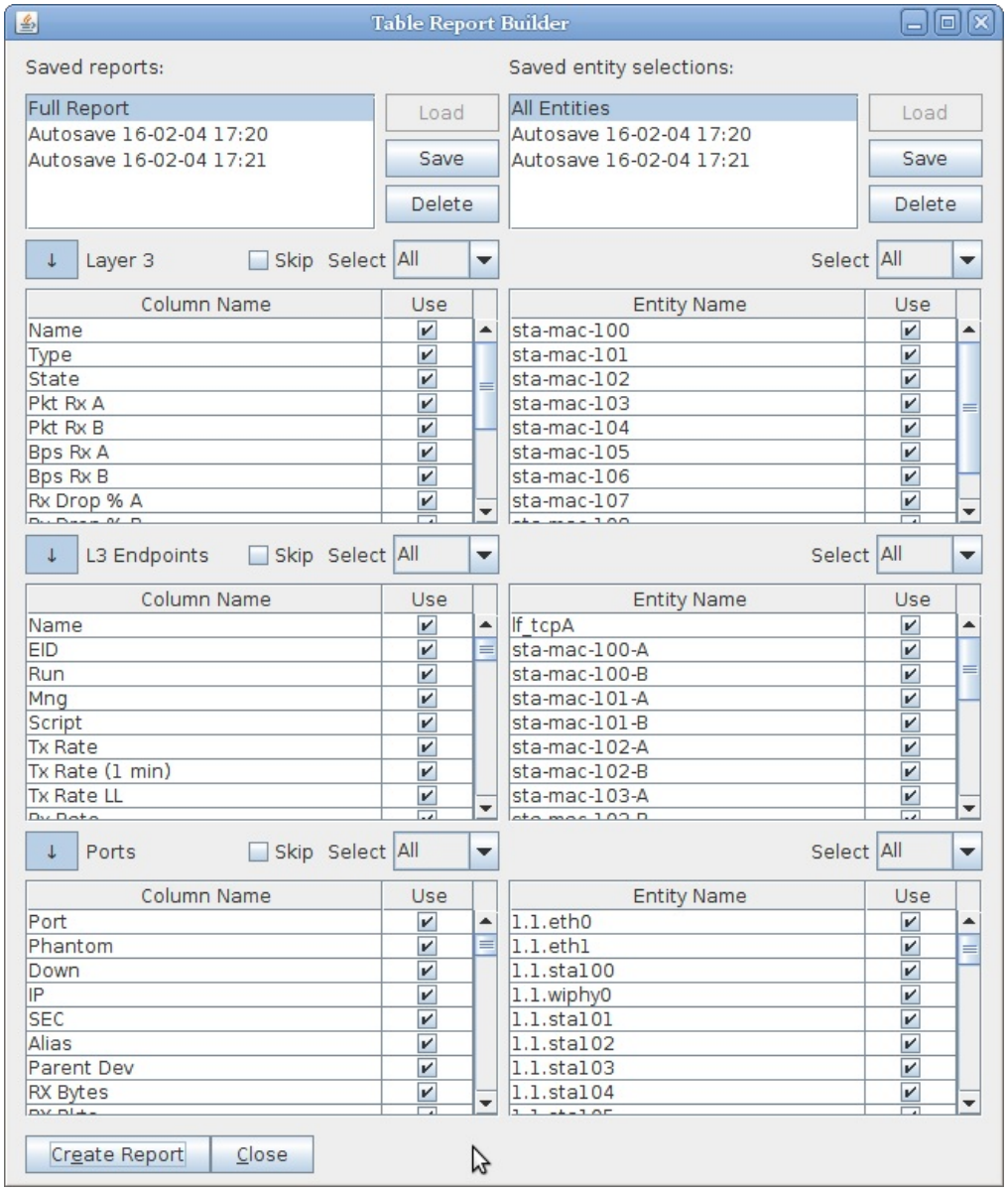
Goal: Learn how to generate reports that present the data found in the table views of the LANforge GUI.

We will use the **Reporting→Table Report Generator** tool to select rows and columns present from the LANforge GUI to create an HTML report. The individual rows and columns from the Ports, Layer-3 and L3-Endpoints windows can be selected. Feature appeares in release 5.3.3.

1. Begining a New Report
   A. We start in the **Reports** menu, and choose the **Table Report Generator**

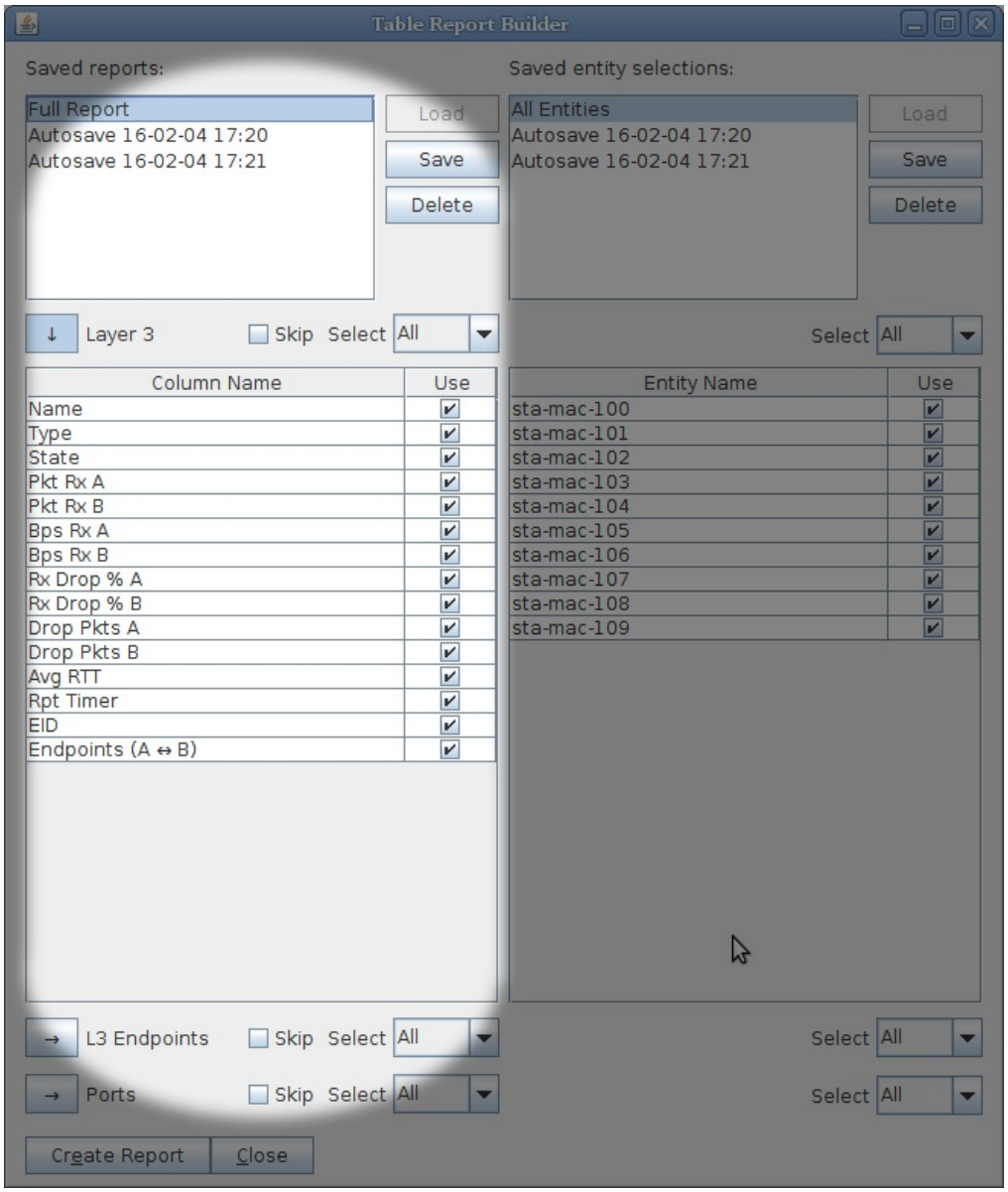


B. The report generator window appears. At the top are lists of previous saved settings. Below those are the selection tables for Layer 3, L3-Endpoints and Ports. The check-box indicates if the item will be included in the report.

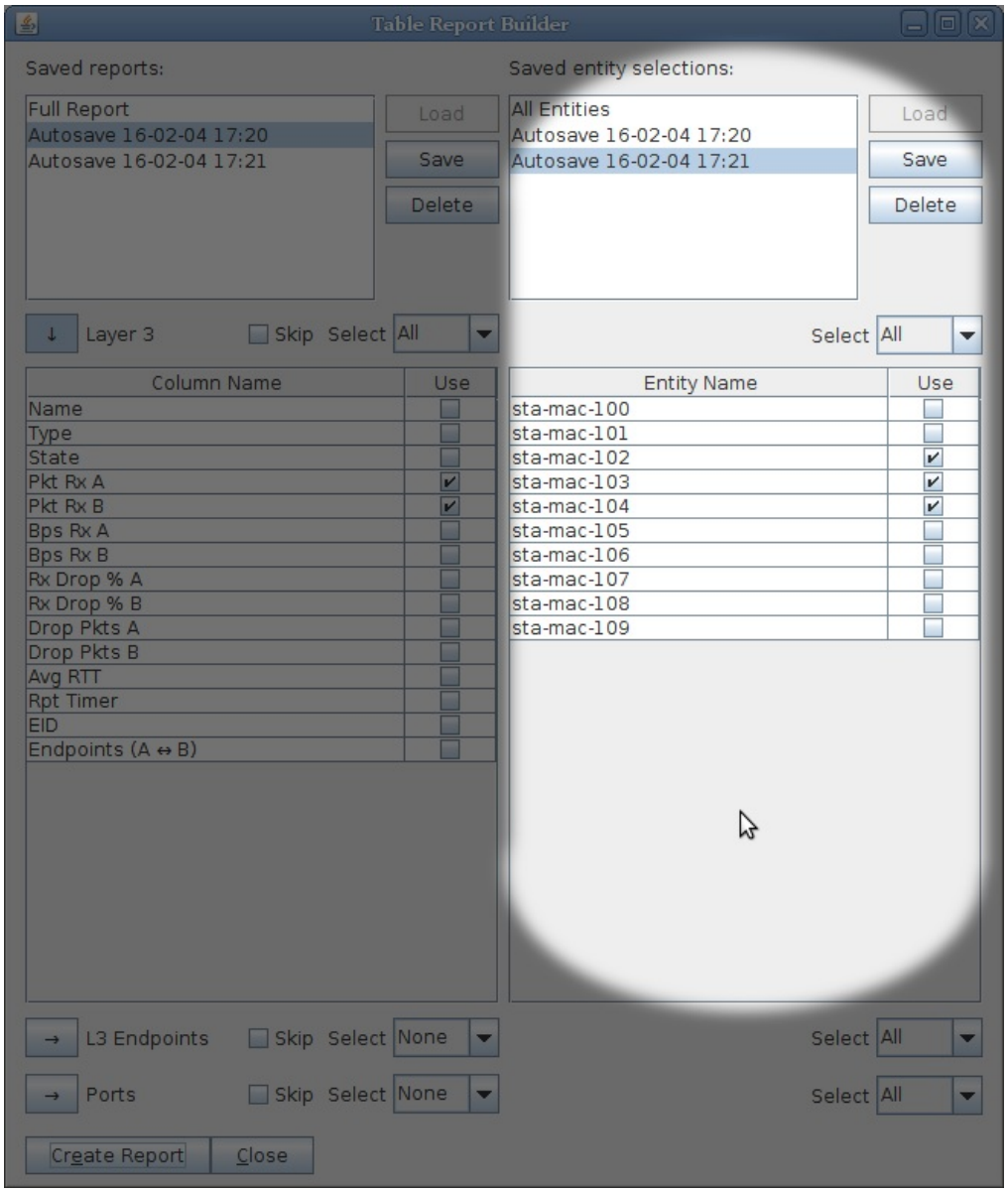


2. Understanding the two columns in the Table Report Builder.

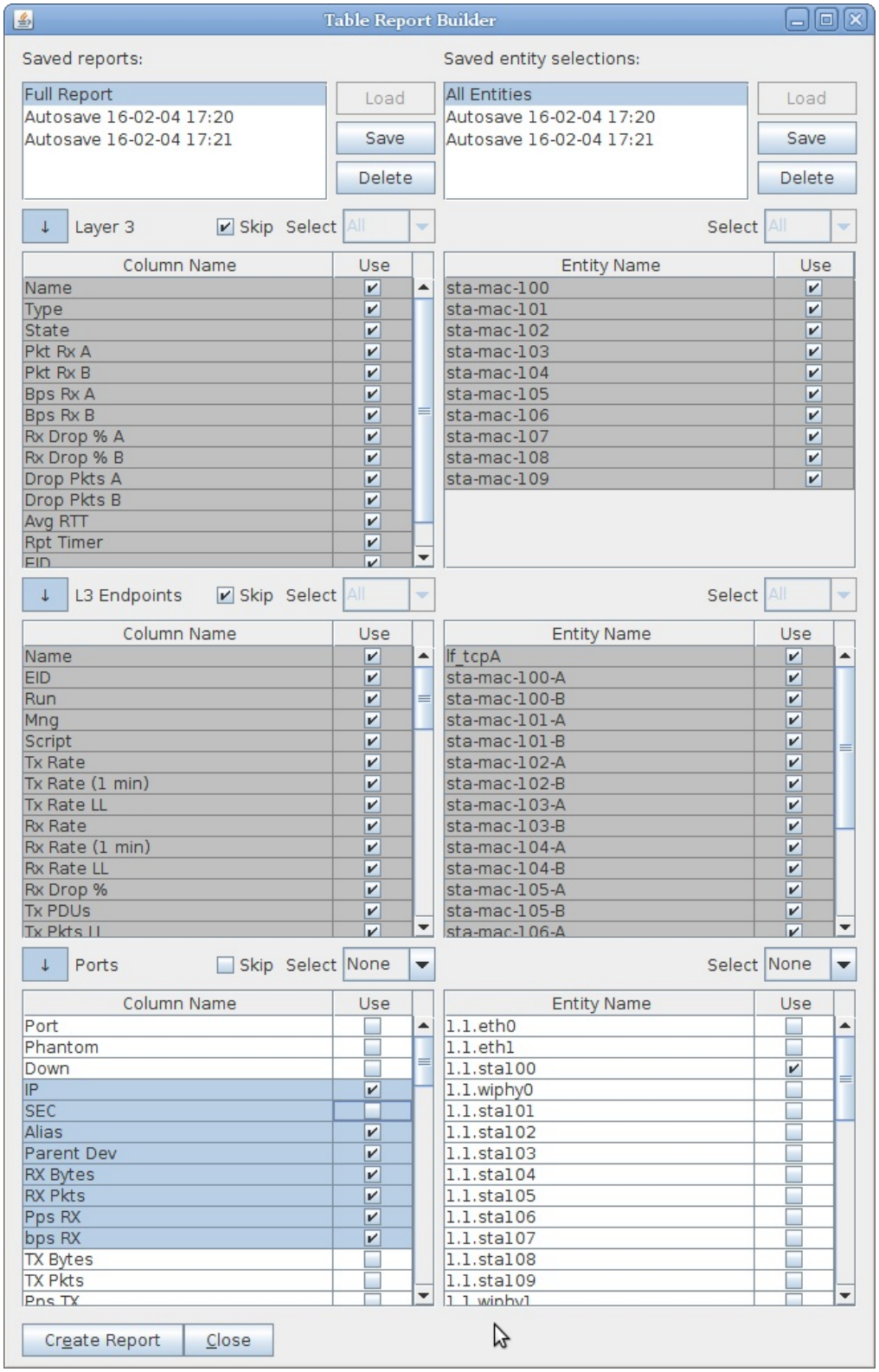
A. Report Profiles are on the left. This represents your selection of Layer-3, L3-Endpoint and/or Port table columns to include in your report. These preferences are saved independently of the entity choices.



B. Entity Selections are on the right. This consists of a your combination of Layer-3 connections, L3-Endpoints and/or Ports to report on. These preferences are also saved independently from the profiles.



3. Selecting items to include
   A. The Skip button will omit a whole table from the report. Checking and un-checking the skip button will not alter your row selection check boxes.

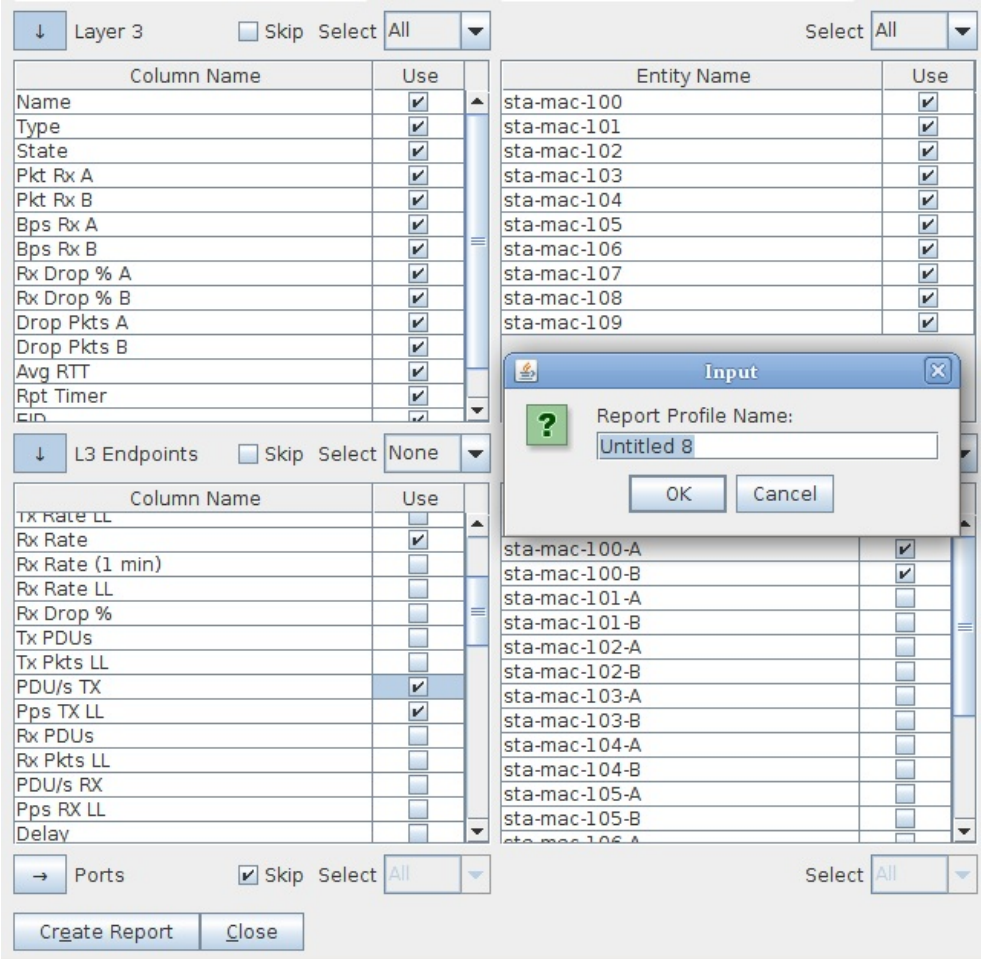


B. The **Select** drop downs for each table are a short cut for selecting all or unselecting all items in a row.

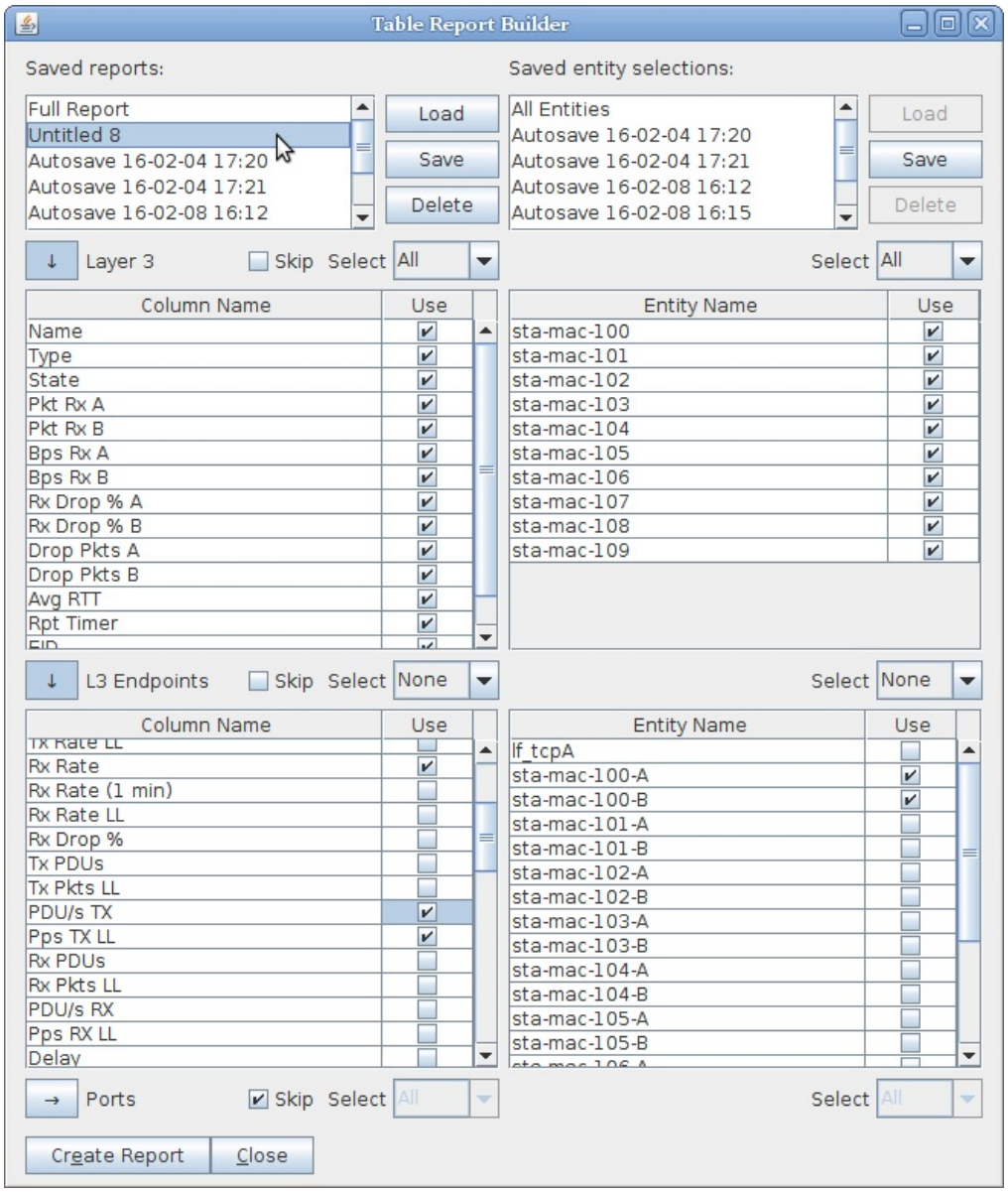
C. The `[SPACE]` and `[ENTER]` keys are shortcuts for selecting and unselecting checkboxes. These keys will toggle highlighted groups of rows as well.

4. Saving and Loading a Report Profile

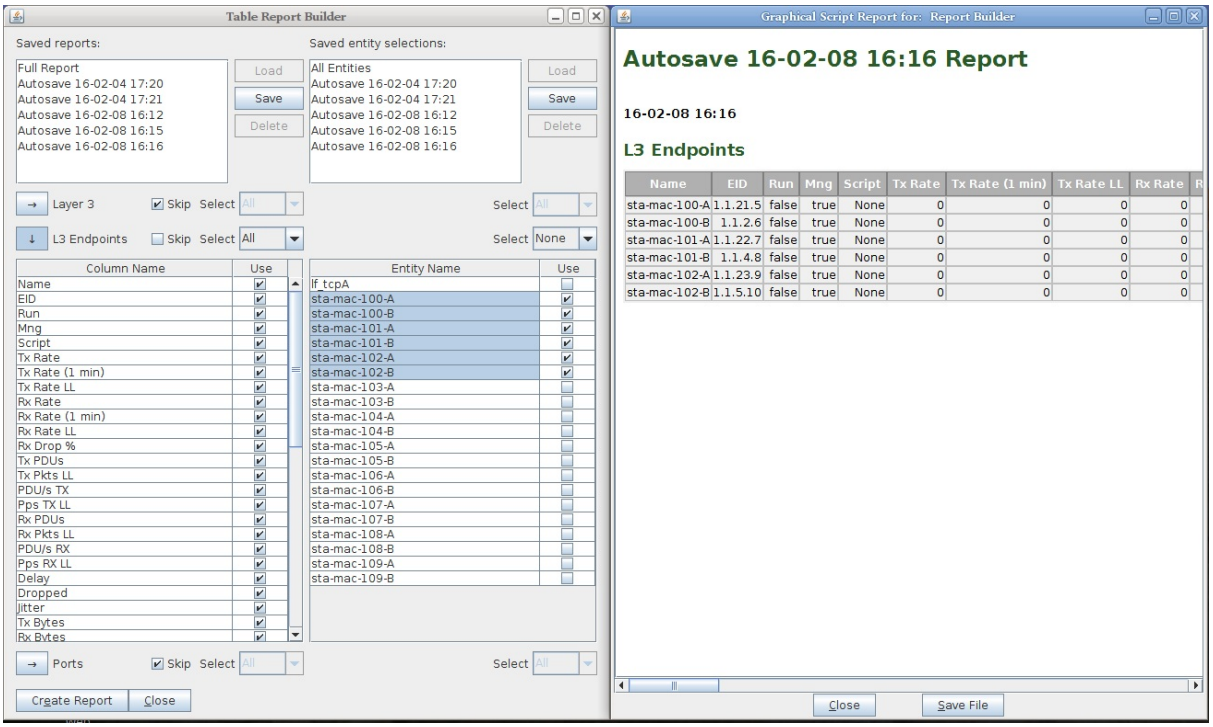
A. The **Save** button allows you to save your table row selections. Name your saved profile.



B. You can restore previous reports profiles. Select a profile name and then click **Load**.



5. Generating the report is simple, click the **Create Report** button. Your report will be remembered with an Autosave record.



6. Saving the report as a file is easy as well. In the Graphical Script Report window, click the **Save File** button.

Displaying Sequence Gaps

Goal: Generate and show sequence gaps (time between packets) by running traffic through a WanLink with an intermittent latency spike.

Tracking sequence gaps can be useful in the following scenarios:

- Roaming with and without 802.11r.
- OSPF failovers.
- Cellular to WiFi handoff.

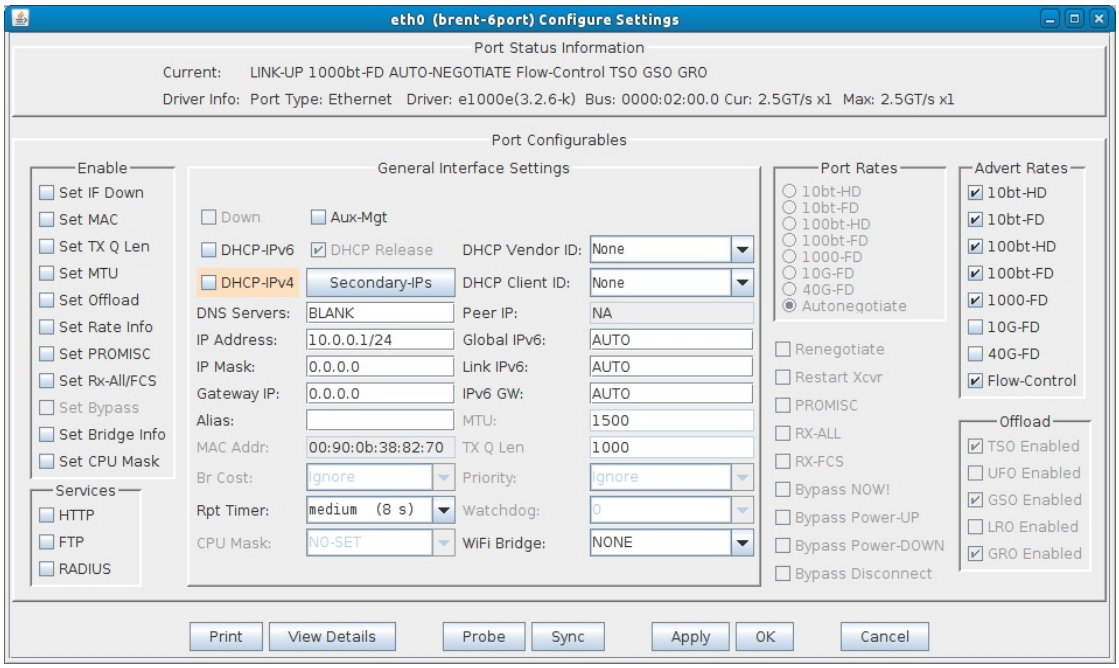
In this test scenario, LANforge will be set up to generate sequence gaps. Then the sequence gaps will be shown via the layer-3 display window.

Four physical ports will be used, two to generate traffic and two for the WanLink endpoints. This test uses a LANforge CT922 system.

LANforge will be generating UDP traffic at 1 Mbps through a WanLink that is scripted to apply one second of latency every ten seconds. The one second latency should create a one second sequence gap. The packet size will be smaller to increase the rate of traffic and thus decreasing sequence gaps, this will make any larger sequence gaps more apparent.

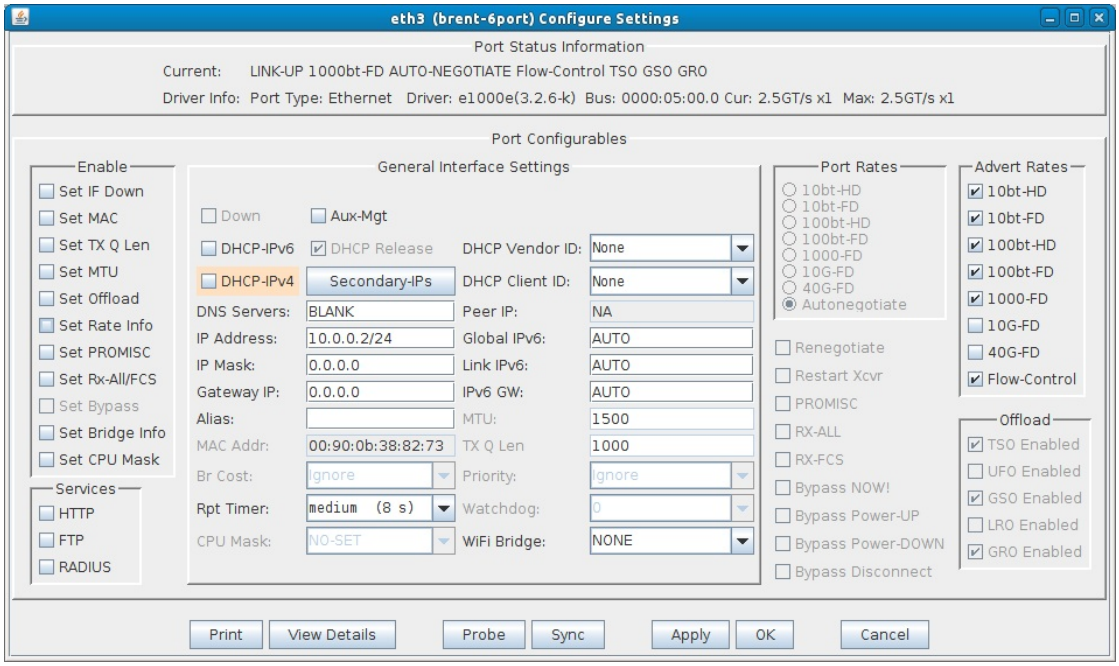
1. Set IPs on traffic generating ports, **eth0** and **eth3** in this case.

A. In the **Port Manager** tab modify **eth0**.



A. Set the **IP Address**.

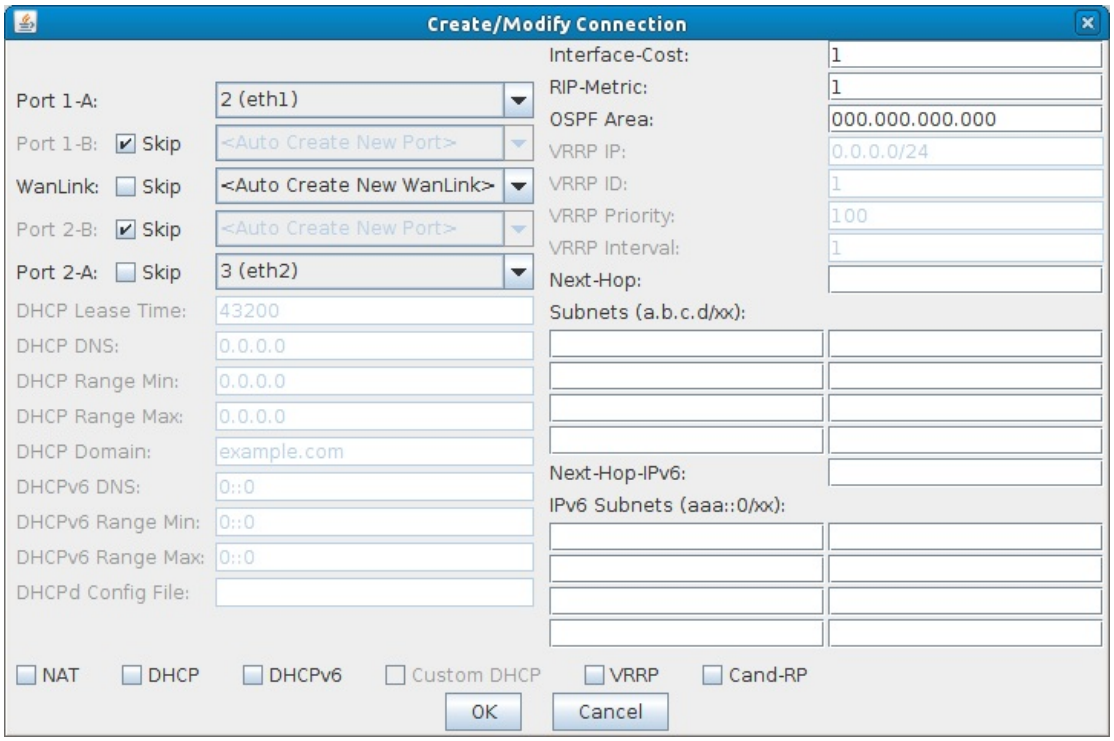
B. In the **Port Manager** tab modify **eth3**.



A. Set the **IP Address**.

2. Create a WanLink.

A. In **Netsmith**, right click an open area and click **New Connection**.



A. Select the Skip checkboxes on **Port 1-B** and **Port 2-B**.

B. Set **Port 1-A** to **eth1**.

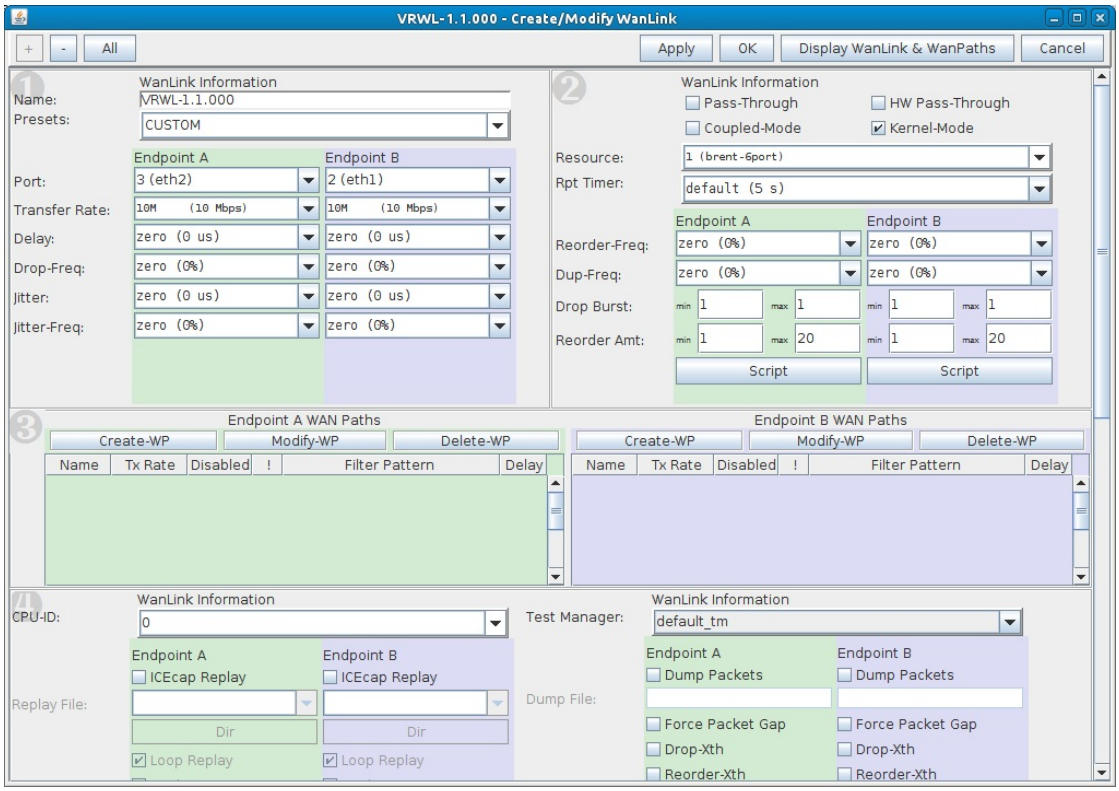
C. Set **Port 2-A** to **eth2**.

D. Click **OK**.

E. Click **Apply** in Netsmith.

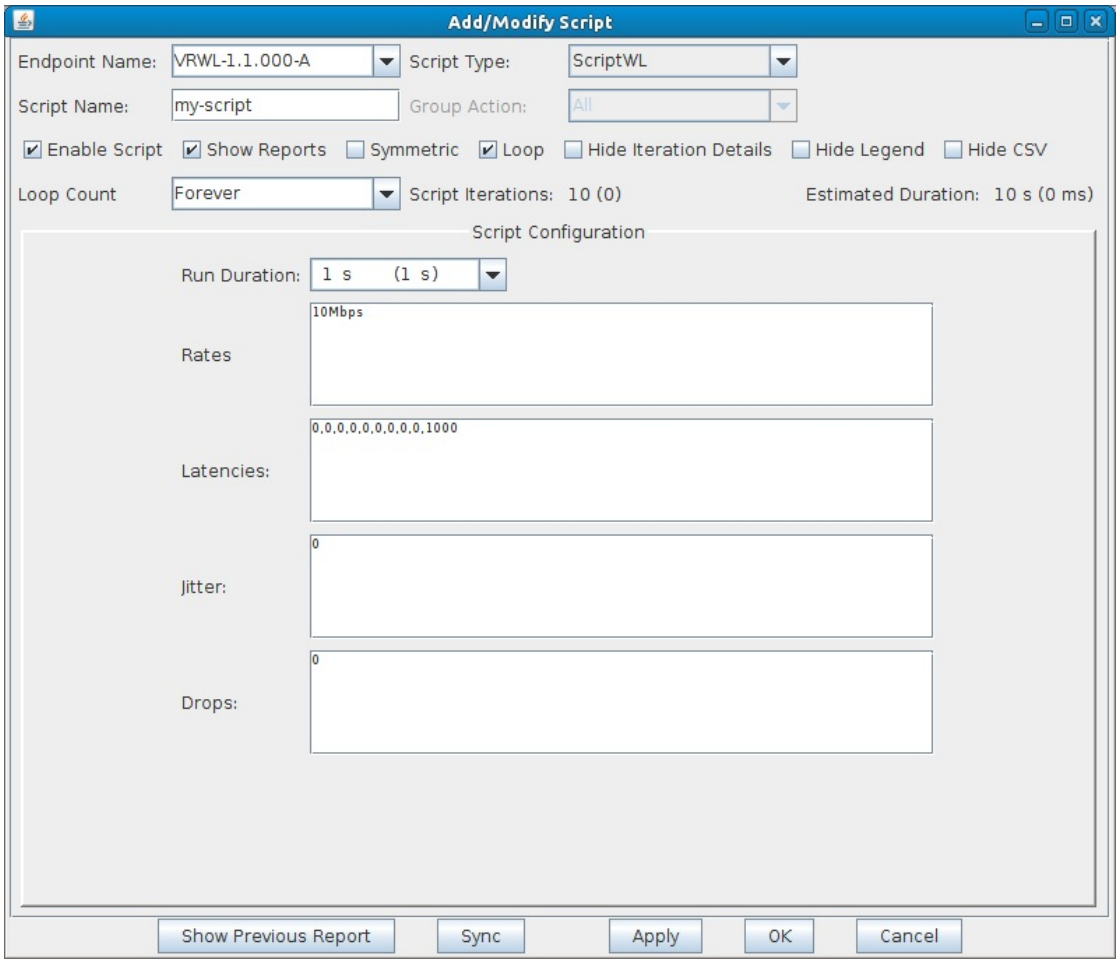
3. Configure the WanLink.

A. In the **WanLinks** tab, modify the WanLink.



A. Set the Transfer Rate to **10Mbps** for **both** endpoints.

B. Click the **Script** button on **Endpoint A**.



A. Set the **Script Type** to **ScriptWL**.

B. Click the **Loop** checkbox.

C. **Run Duration = 1s**

D. **Rates = 10Mbps**

E. **Latencies = 0,0,0,0,0,0,0,0,0,1000**

F. **Jitter = 0**

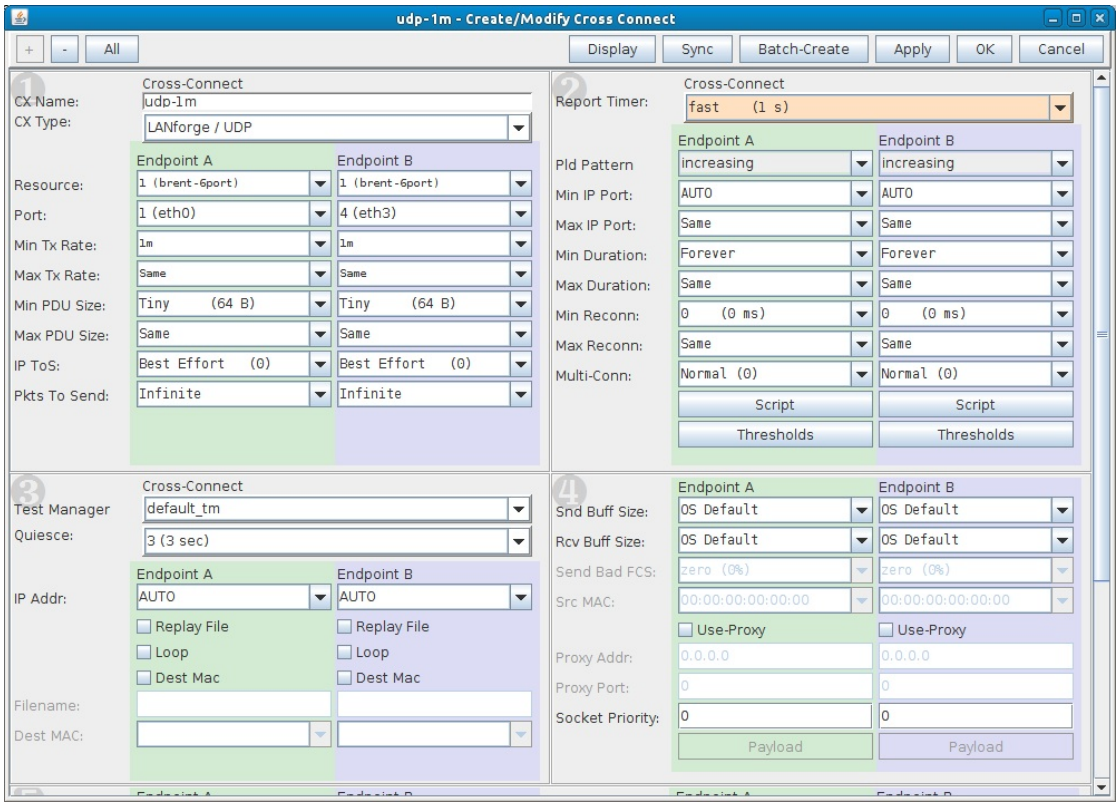
G. **Drops = 0**

H. Click **OK**.

C. Click **OK** to close the WanLink modify window.

4. Create a Layer 3 connection.

A. In the **Layer-3** tab, Click **Create**.



A. **CX Name = udp-1m**

B. **Report Timer = 1s**

C. **Endpoint A Port = eth0**

D. **Endpoint B Port = eth3**

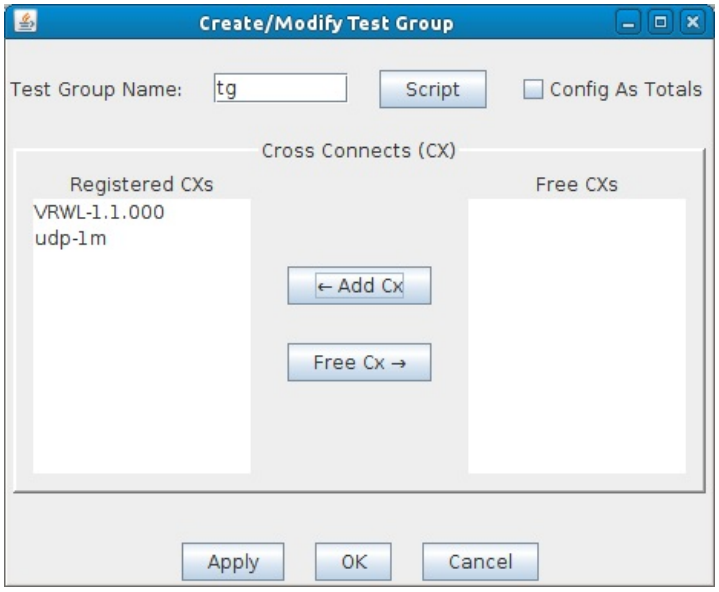
E. **Endpoint A and B = 1m**

F. **Endpoint A and B Min PDU Size = 64 B**

G. Click **OK**.

5. Create a test group to start the layer 3 connection and WanLink at the same time.

A. In the **Test Group** tab, click **Create**.

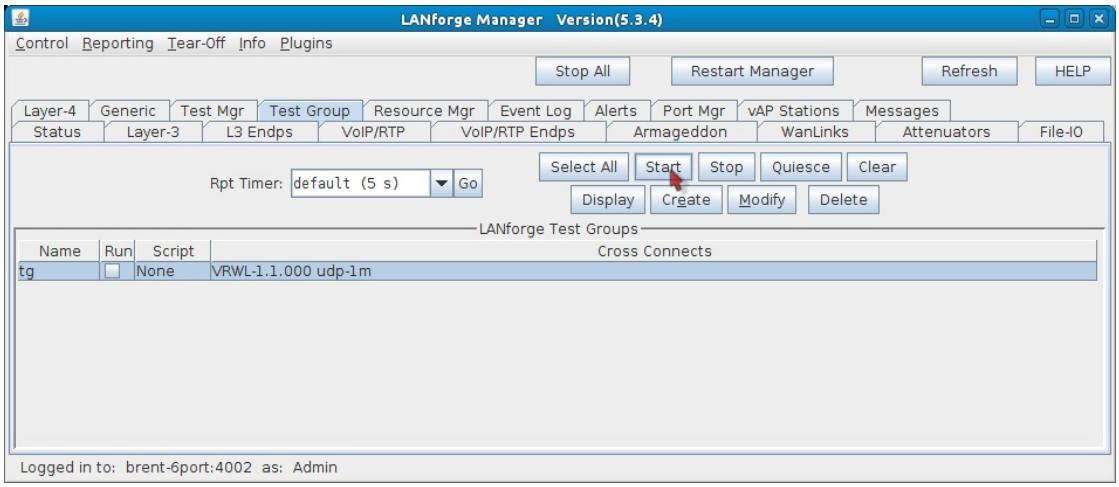


A. **Test Group Name = tg**

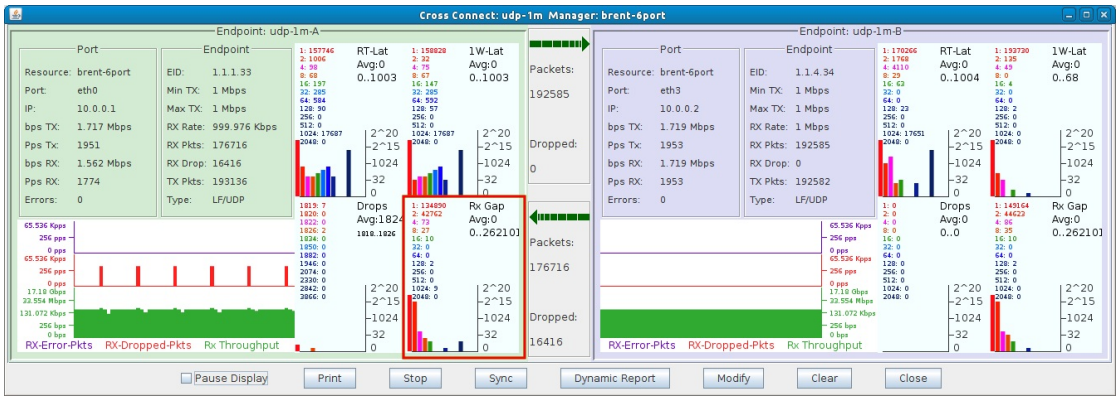
B. Select the WanLink and Layer 3 connection on the right and click **Add Cx**.

C. Click **OK**.

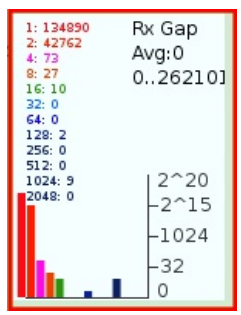
6. Start the test group. This will start both the layer 3 connection and the scripted WanLink.
   A. In the **Test Group** tab, select **tg** and click **Start**. You will see the Script Report window appear.



7. Display the Layer 3 connection and analyze the Rx Gap (sequence gap).
   A. In the Layer-3 tab, select **udp-1m** and click **Display**.



   B. The **Rx Gap** graph is highlighted in the below screenshot. The Rx Gap for Endpoint B doesn't show one second sequence gaps because that side of the WanLink was not scripted.



      A. The colored numbers on the left side of the colon represent time in milliseconds. The right colored numbers represent the number of times a sequence gap reached the particular time on the left side.

      You'll notice that 9 sequence gaps hit 1024ms. These gaps were the result of the WanLink script occasionally sending one second latency.

Scalability Testing with NFS File-IO

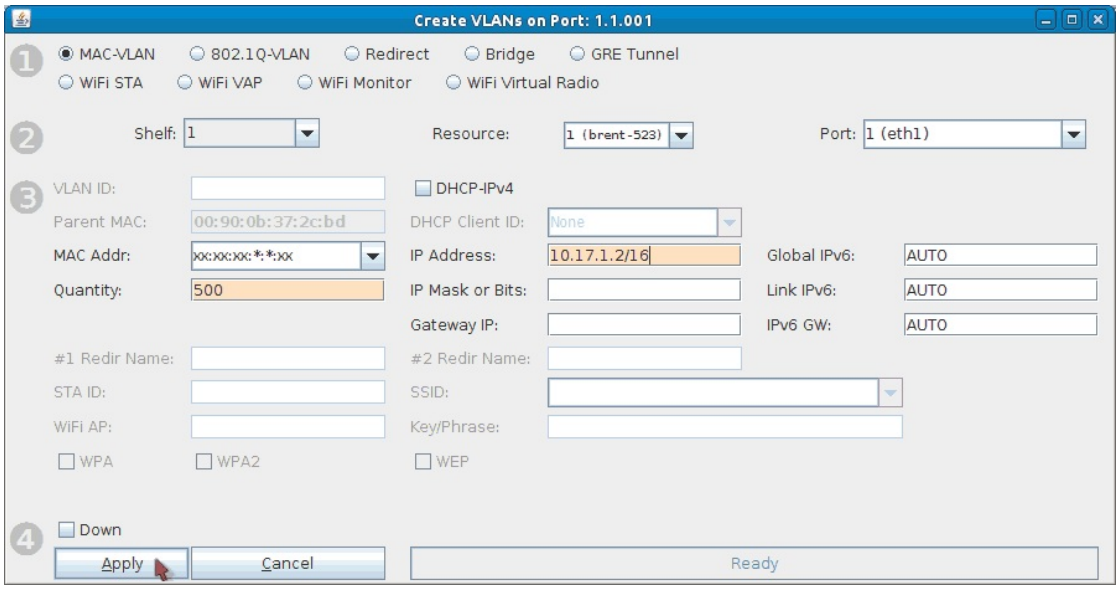
Goal: Analyze memory usage as the number of NFS clients is scaled up, and also when NFS endpoint settings are tweaked.

Hopefully by the end of this cookbook you'll have an idea how to assess how many File-IO endpoints and associated settings your system can support. This cookbook connects a LANforge system to a NFS file server.

The file server in this example will be 10.17.1.1. It will be sharing 10.17.1.1:/mnt/tmpfs1. We will create 100, 250, then 500 readers and writers and compare memory usage as we also tweak the RW Size setting.

Note: For this example, we are using a CT523 with 8GB of memory. It is physically connected to the NFS file server.

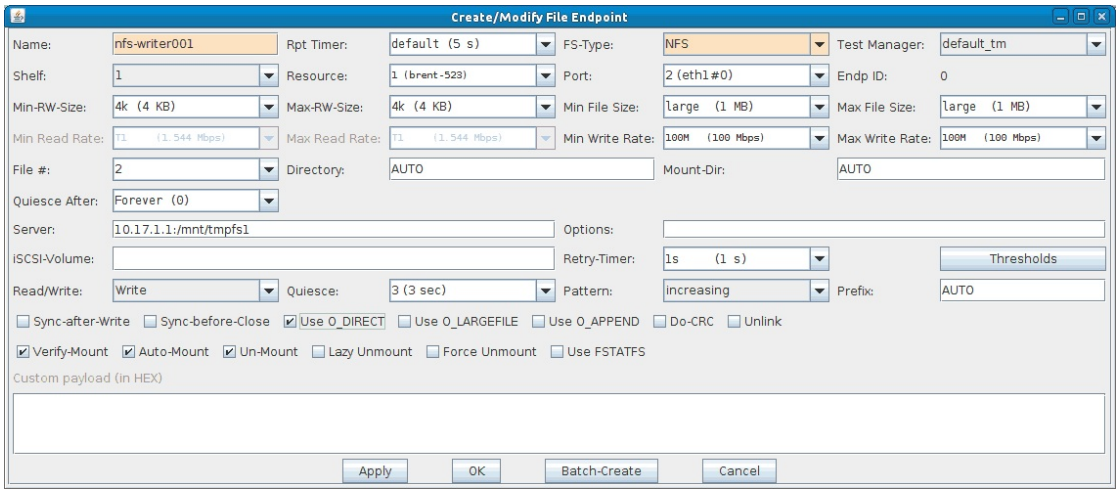
1. First, **500** MAC-VLANs will need to be created.
   A. In the **Port-Mgr** tab select port **eth1** and click **Create**.



   A. Select **MAC-VLAN**.
   B. Set **Quantity** to **500**.
   C. The starting IP address is 10.17.1.2/16 for this test. If your file server is on a different network, change the IP here accordingly. Make sure not to use the file server's IP address.
   D. Click **Apply** and close the Port Create window once all ports are configured. Make sure all MAC-VLANs get IPs, this may take some time.

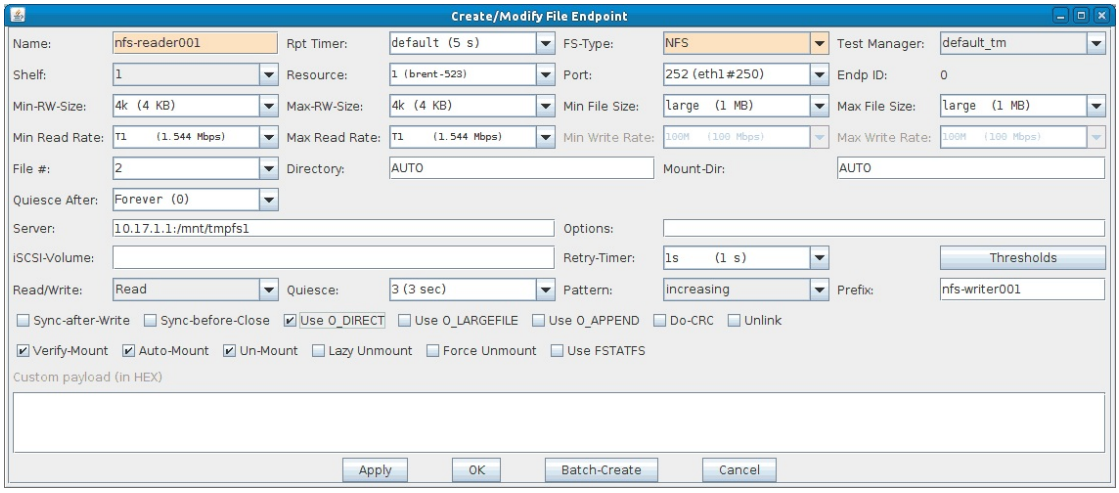
2. Create **250** NFS writers and **250** NFS readers. Batch create **50** at a time (49 for first batch of reader/writers). More information on creating File-IO endpoints can be found here: LANforge File-IO with CIFS and NFS

A. Any non-default settings for the **NFS writers** are listed below.



A. Name is **nfs-writer001**.
B. FS-Type: **NFS**.
C. Port is **eth1#0**.
D. Min-RW and Max-RW sizes are **4k**.
E. Min/Max File sizes are **1MB**.
F. Min/Max Write rates are **100Mbps**.
G. Quiesce After is **Forever**.
H. Server is **10.17.1.1:/mnt/tmpfs1** (use your own file server settings here).
I. Make sure Read/Write is set to **Write**.
J. Turn on the **Use O_DIRECT** checkbox.

B. Any non-default settings for the **NFS readers** are listed below.

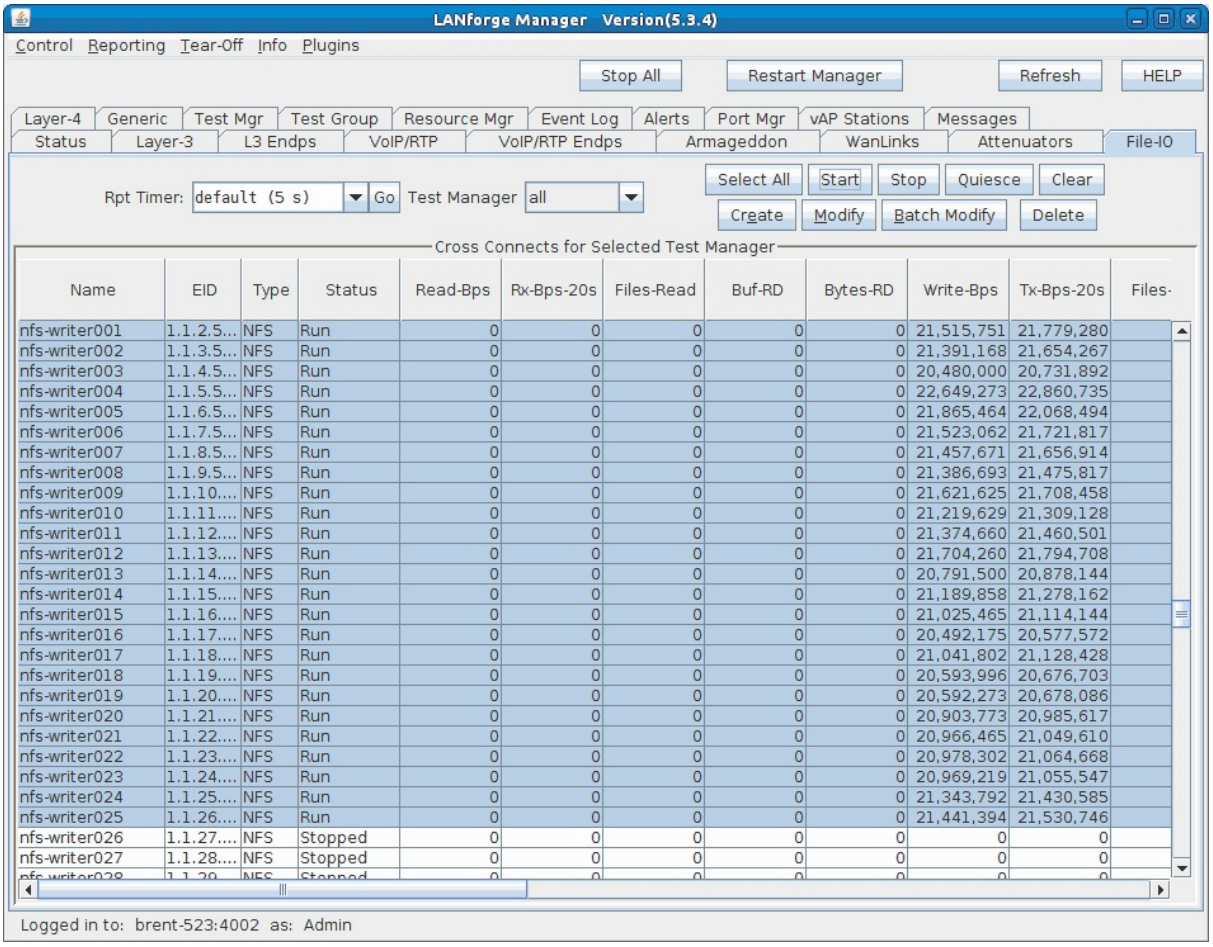


A. Name is **nfs-reader001**.
B. FS-Type: **NFS**.
C. Port is **eth1#250**.
D. Min-RW and Max-RW sizes are **4k**.
E. Min/Max File sizes are **1MB**.
F. Min/Max Write rates are **100Mbps**.
G. Quiesce After is **Forever**.
H. Server is **10.17.1.1:/mnt/tmpfs1** (use your own file server settings here).
I. Set Read/Write to **Read**.
J. Set Prefix to **nfs-writer001**.
K. Turn on the **Use O_DIRECT** checkbox.

3. Start the first **50 writers** then the first **50 readers**. It's recommended to slowly start File-IO endpoints, for example, starting 25 writers then 25 readers then the same again until you have the desired amount running. The reason to do this is because the system may become unresponsive if it can't handle the number of File-

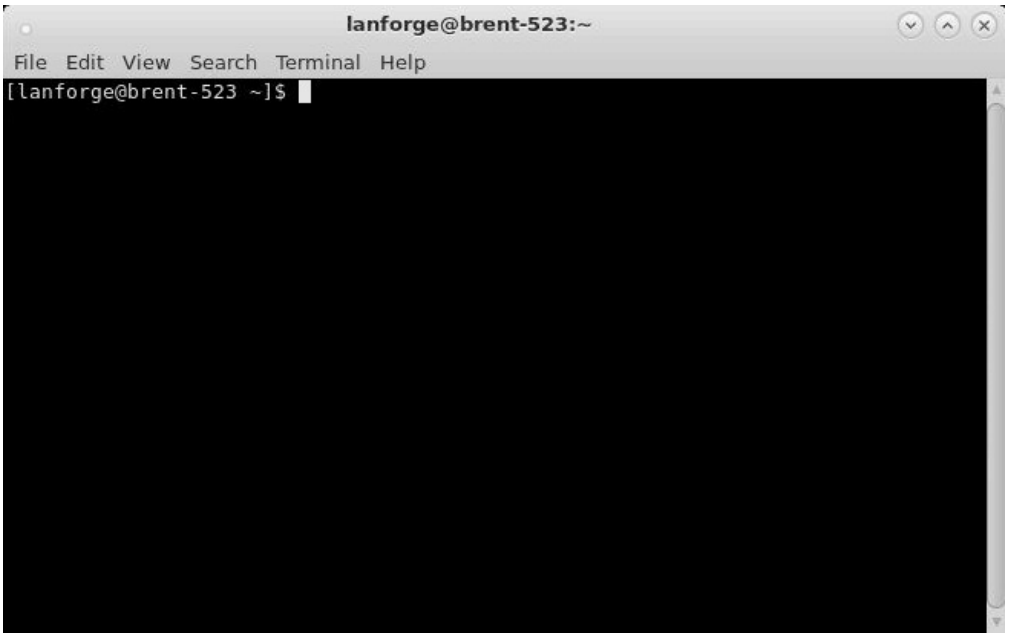
IO endpoints.

Note: If you notice the system slowing down, connections not starting, or connections stuck in WAITING state, or any weirdness in general, you should stop 25 or 50 writers/readers at a time until the system clears up.

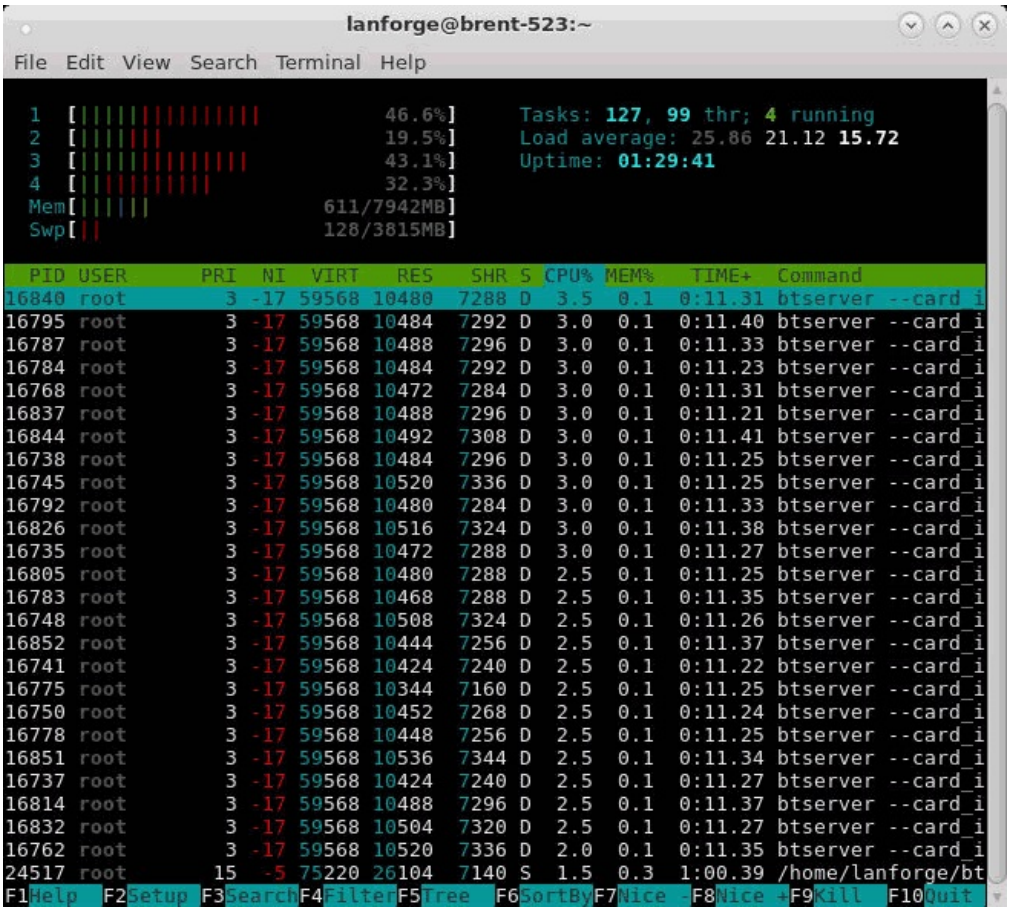


Name	EID	Type	Status	Read-Bps	Rx-Bps-20s	Files-Read	Buf-RD	Bytes-RD	Write-Bps	Tx-Bps-20s	Files-
nfs-writer001	1.1.2.5...	NFS	Run	0	0	0	0	0	21,515,751	21,779,280	
nfs-writer002	1.1.3.5...	NFS	Run	0	0	0	0	0	21,391,168	21,654,267	
nfs-writer003	1.1.4.5...	NFS	Run	0	0	0	0	0	20,480,000	20,731,892	
nfs-writer004	1.1.5.5...	NFS	Run	0	0	0	0	0	22,649,273	22,860,735	
nfs-writer005	1.1.6.5...	NFS	Run	0	0	0	0	0	21,865,464	22,068,494	
nfs-writer006	1.1.7.5...	NFS	Run	0	0	0	0	0	21,523,062	21,721,817	
nfs-writer007	1.1.8.5...	NFS	Run	0	0	0	0	0	21,457,671	21,656,914	
nfs-writer008	1.1.9.5...	NFS	Run	0	0	0	0	0	21,386,693	21,475,817	
nfs-writer009	1.1.10....	NFS	Run	0	0	0	0	0	21,621,625	21,708,458	
nfs-writer010	1.1.11....	NFS	Run	0	0	0	0	0	21,219,629	21,309,128	
nfs-writer011	1.1.12....	NFS	Run	0	0	0	0	0	21,374,660	21,460,501	
nfs-writer012	1.1.13....	NFS	Run	0	0	0	0	0	21,704,260	21,794,708	
nfs-writer013	1.1.14....	NFS	Run	0	0	0	0	0	20,791,500	20,878,144	
nfs-writer014	1.1.15....	NFS	Run	0	0	0	0	0	21,189,858	21,278,162	
nfs-writer015	1.1.16....	NFS	Run	0	0	0	0	0	21,025,465	21,114,144	
nfs-writer016	1.1.17....	NFS	Run	0	0	0	0	0	20,492,175	20,577,572	
nfs-writer017	1.1.18....	NFS	Run	0	0	0	0	0	21,041,802	21,128,428	
nfs-writer018	1.1.19....	NFS	Run	0	0	0	0	0	20,593,996	20,676,703	
nfs-writer019	1.1.20....	NFS	Run	0	0	0	0	0	20,592,273	20,678,086	
nfs-writer020	1.1.21....	NFS	Run	0	0	0	0	0	20,903,773	20,985,617	
nfs-writer021	1.1.22....	NFS	Run	0	0	0	0	0	20,966,465	21,049,610	
nfs-writer022	1.1.23....	NFS	Run	0	0	0	0	0	20,978,302	21,064,668	
nfs-writer023	1.1.24....	NFS	Run	0	0	0	0	0	20,969,219	21,055,547	
nfs-writer024	1.1.25....	NFS	Run	0	0	0	0	0	21,343,792	21,430,585	
nfs-writer025	1.1.26....	NFS	Run	0	0	0	0	0	21,441,394	21,530,746	
nfs-writer026	1.1.27....	NFS	Stopped	0	0	0	0	0	0	0	
nfs-writer027	1.1.28....	NFS	Stopped	0	0	0	0	0	0	0	

4. Now that 50 File-IO endpoints are running, the memory will be analyzed via htop.
   A. Open a terminal on the LANforge system using a preferred method (either directly or remotely through ssh, rdesktop, or vncviewer).



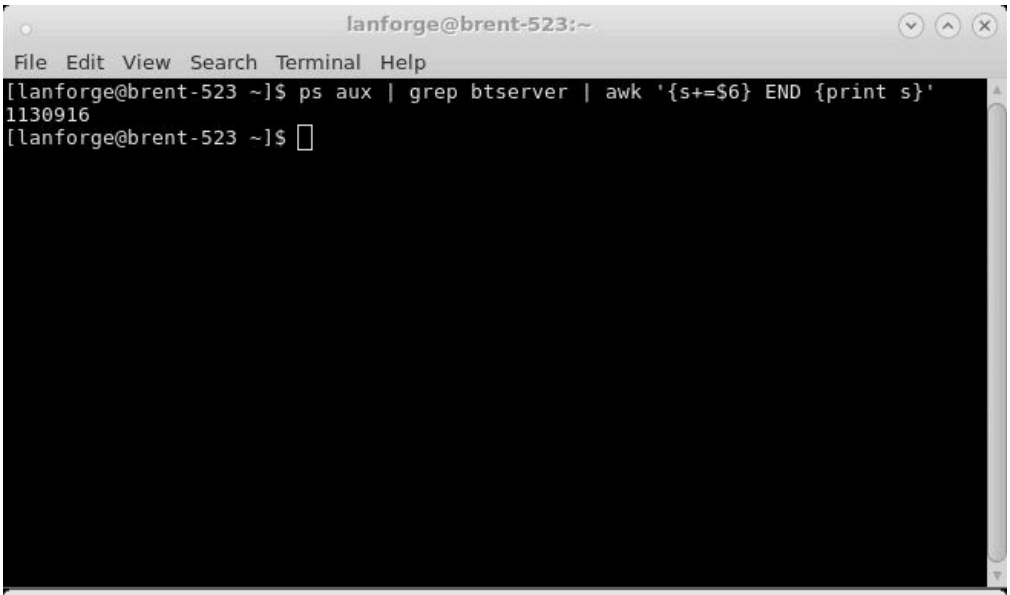
B. Show a list of btserver processes (LANforge processes) by running the command:
`htop -p `pgrep btserver | tr "\n" "," | sed 's/,$//'`
If htop isn't installed run: `sudo yum install htop`

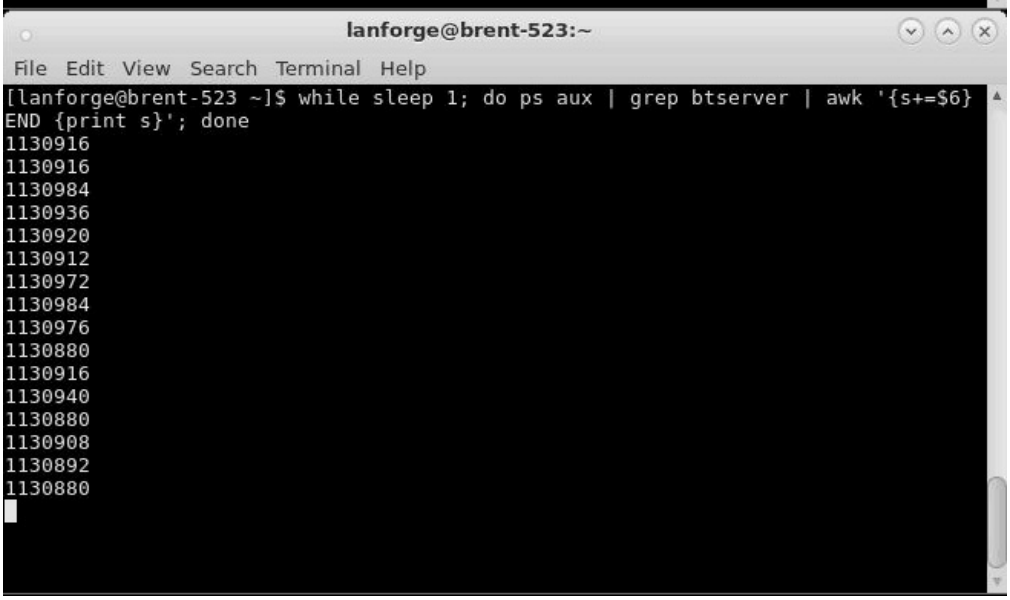


A. The majority of btserver processes with the same or similar values here represent the running File-IO endpoints. The **RES** (resident value) column represents the memory used by these File-IO endpoints. The current test uses an average of about 10,400 KiB or **10.6 MB** per btserver process. These values should be monitored as the File-IO settings are adjusted.

NOTE: You will need to restart htop if the endpoints are stopped/restarted. Just press **q** to quit out and run the above htop command again.

C. Total up the RES memory used by btserver. This is useful to compare memory usage between a different number of connections. For example you could compare how much memory 100 endpoints use compared to 500 endpoints.



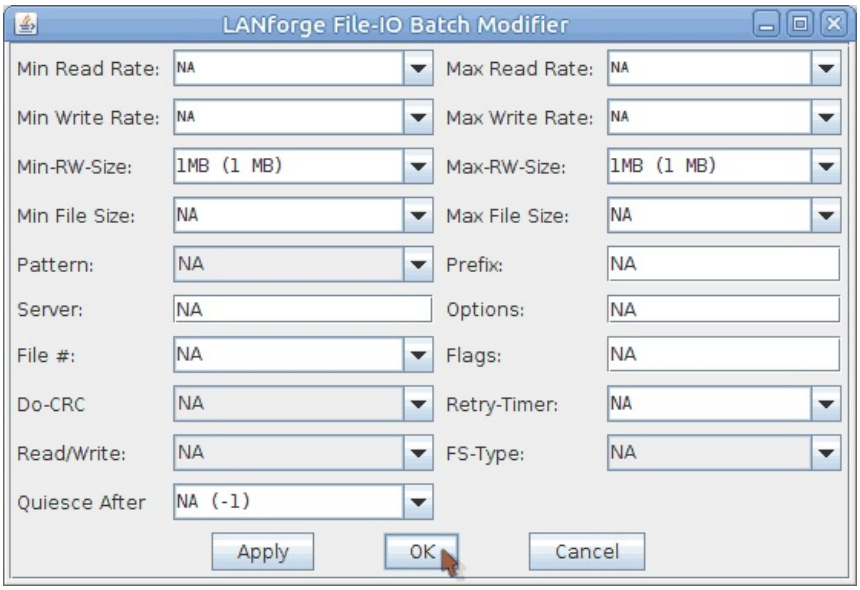


A. To show the total RES memory used, run the command:
`ps aux | grep btserver | awk '{s+=$6} END {print s}'`
To repeat it every second, use the below command. Press **Ctrl+C** to stop.
`while sleep 1; do ps aux | grep btserver | awk '{s+=$6} END {print s}'; done`

B. The total memory for 50 NFS File-IO endpoints with a **Min/Max RW rate of 4k** is about 1,130,000 KiB which converts to around **1,157 MB**.

5. While the File-IO endpoints are running, change the **Min-RW** and **Max-RW** settings.

A. Select all File-IO endpoints and click **Batch Modify**.



A. Set the **Min/Max RW sizes** to **1MB**.

B. Click **OK**

6. Analyze how the Min/Max RW size setting change has affected the memory (see steps 4B and 4C).

```
lanforge@brent-523:~
File  Edit  View  Search  Terminal  Help

  1  [|||||                      13.3%]     Tasks: 204, 102 thr; 1 running
  2  [||||                        6.7%]     Load average: 46.19 54.05 54.10
  3  [||                          4.3%]     Uptime: 01:01:02
  4  [||||                       11.0%]
  Mem[|||||||               906/7942MB]
  Swp[                        0/3815MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 2376 root        3 -17 97840 48036  7508 S  1.0  0.6  1:24.07 /home/lanforge/bt
 2394 root       15  -5 75256 25312  7388 S  0.5  0.3  0:50.77 /home/lanforge/bt
22182 root        3 -17 60600 11496  7284 S  0.5  0.1  0:29.35 btserver --card_i
22316 root        3 -17 60600 11544  7332 D  0.5  0.1  0:29.21 btserver --card_i
22210 root        3 -17 60600 11612  7396 S  0.5  0.1  0:29.12 btserver --card_i
22315 root        3 -17 60600 11548  7332 S  0.5  0.1  0:29.76 btserver --card_i
22263 root        3 -17 60600 11612  7396 D  0.5  0.1  0:29.17 btserver --card_i
22192 root        3 -17 60600 11636  7420 D  0.5  0.1  0:29.19 btserver --card_i
22153 root        3 -17 60600 11548  7336 D  0.5  0.1  0:29.17 btserver --card_i
22261 root        3 -17 60600 11620  7408 D  0.5  0.1  0:29.35 btserver --card_i
22313 root        3 -17 60600 11612  7396 D  0.5  0.1  0:29.08 btserver --card_i
22197 root        3 -17 60600 11520  7308 D  0.5  0.1  0:29.21 btserver --card_i
22169 root        3 -17 60600 11556  7340 D  0.5  0.1  0:29.05 btserver --card_i
22292 root        3 -17 60600 11628  7412 D  0.5  0.1  0:29.69 btserver --card_i
23304 root        3 -17 60600 11628  7420 S  0.5  0.1  0:07.70 btserver --card_i
22013 root        3 -17 60600 11624  7412 D  0.0  0.1  0:31.33 btserver --card_i
22140 root        3 -17 60600 11592  7376 D  0.0  0.1  0:29.00 btserver --card_i
22194 root        3 -17 60600 11616  7400 D  0.0  0.1  0:29.21 btserver --card_i
22230 root        3 -17 60600 11584  7368 D  0.0  0.1  0:29.15 btserver --card_i
22296 root        3 -17 60600 11632  7420 D  0.0  0.1  0:29.17 btserver --card_i
23130 root        3 -17 60600 11564  7352 S  0.0  0.1  0:07.62 btserver --card_i
23207 root        3 -17 60600 11604  7396 S  0.0  0.1  0:07.62 btserver --card_i
```

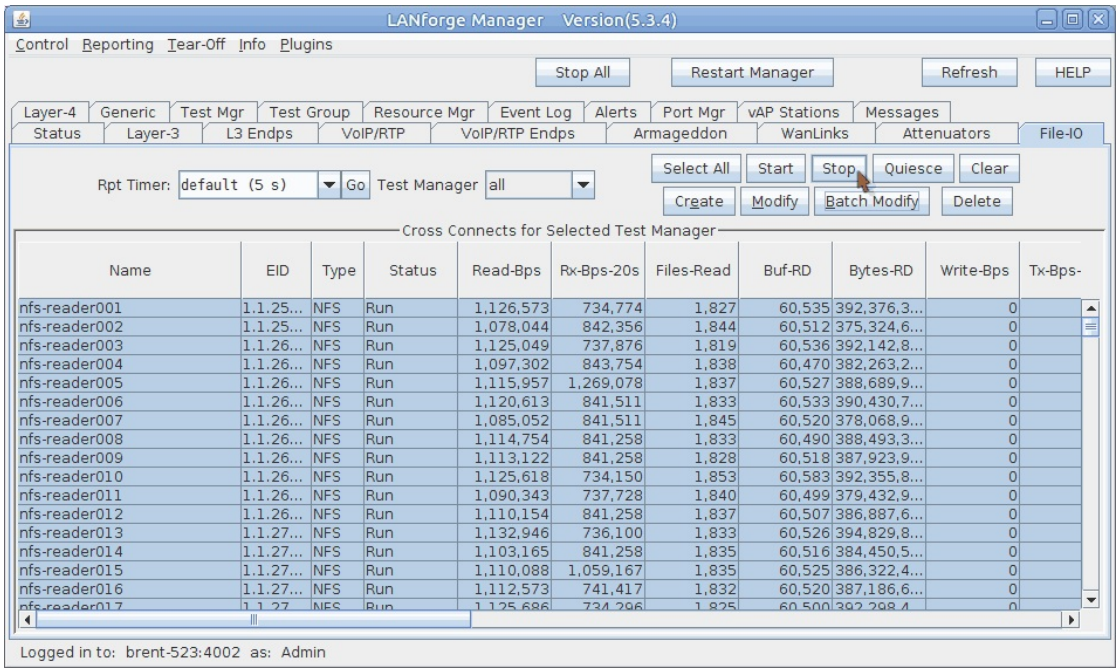
```
lanforge@brent-523:~
File  Edit  View  Search  Terminal  Help
[lanforge@brent-523 ~]$ ps aux | grep btserver | awk '{s+=$6} END {print s}'
1233688
[lanforge@brent-523 ~]$
```

A. Changing the Min/Max RW sizes to 1MB increased the btserver memory use to about 11,600 KiB or **11.9 MB** from 10,400 KiB or **10.6 MB**. About a 1,200 KiB or **1.3 MB** difference.

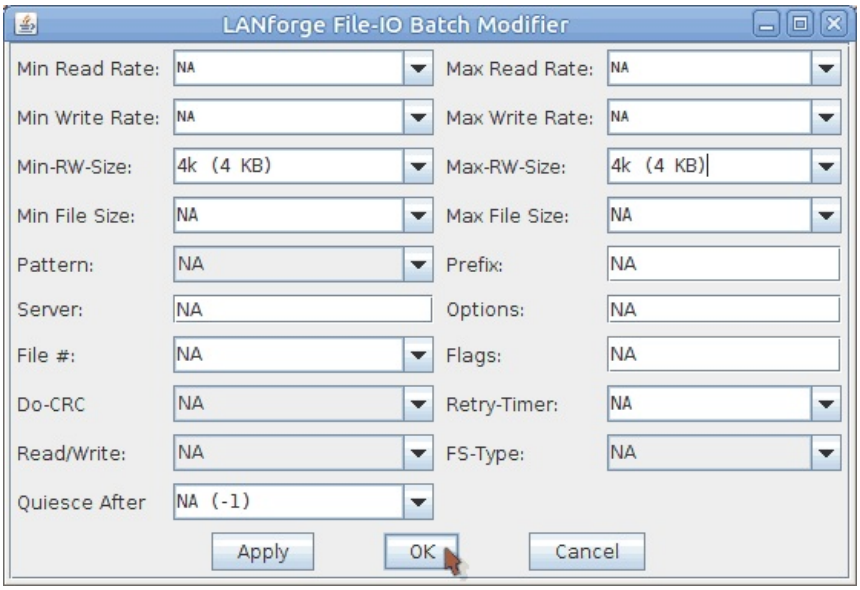
   The total memory increased to 1,233,000 KiB, **1,262 MB** from 1,130,000 KiB, **1,157 MB**. About a 103,000 KiB or **105 MB** difference.

7. Set the endpoints **Min/Max RW** size back to **4k**.

A. Stop all running file-IO endpoints by selecting them and clicking **Stop**.



B. Select all file-IO endpoints and click **Batch Modify**.



A. Set the **Min/Max RW sizes** to **4k**.

B. Click **OK**.

8. Repeat steps **3 through 7** for **125 writers/readers** and then again for **250 writers/readers**.

9. The results from the above tests are shown in the below tables. They show how RW size affects 100, 250, and 500 NFS File-IO endpoints.

Individual Process Memory Usage (MB)				
		Number of File-IO Endpoints		
		100	250	500
RW Size	4 KB	10.6	10.8	10.8
	1 MB	11.9	11.9	11.8

Total Memory Usage (MB)				
		Number of File-IO Endpoints		
		100	250	500
RW Size	4 KB	1,157	2,804	4,134
	1 MB	1,262	3,031	6,013

Watch ports and stations with the Monitor Port plugin.

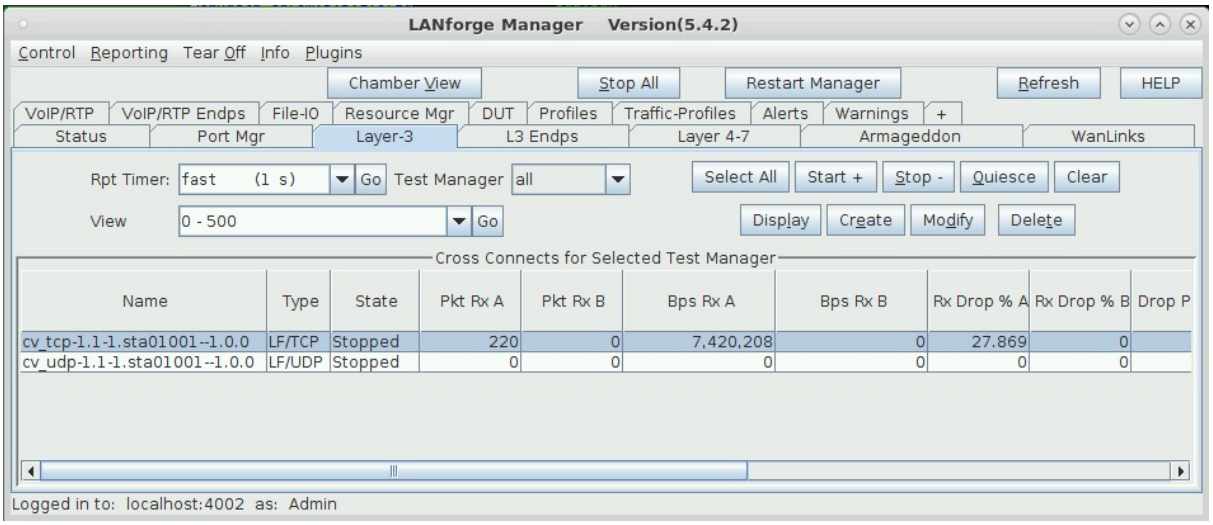
Goal: Display a bandwidth meter for ports and stations.

You can display a series of meters for one or more ports with a nice demo window for traffic speed. For this demonstration, we have configured a station, an upstream port, and a Layer-3 connection to generate traffic.

1. ## Single port display

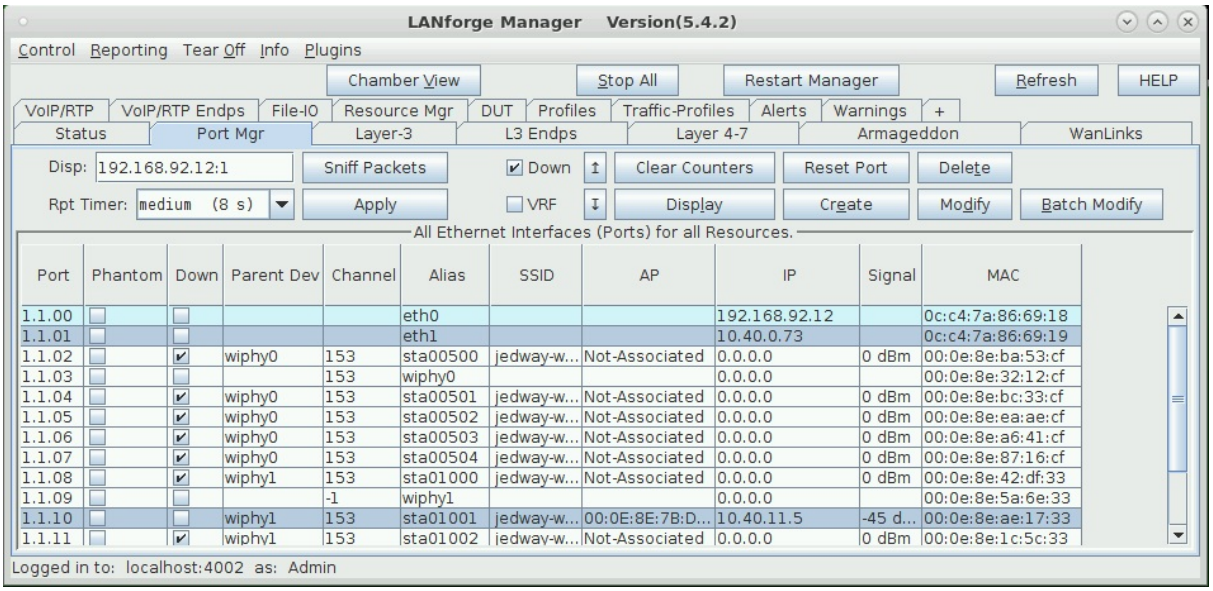
2. The simplest example for port monitoring is showing only one station.

3. We will start our Layer-3 connection.

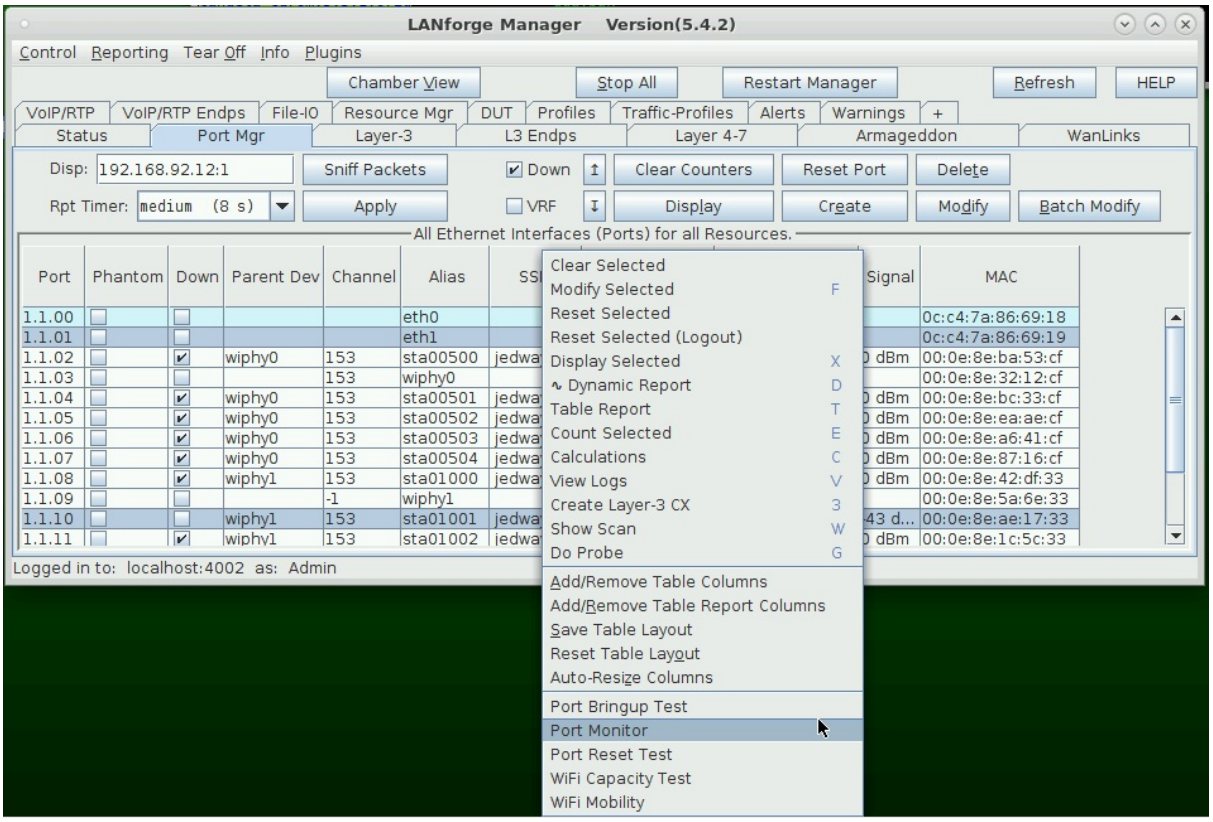


For more information see Generating Traffic to a Switched Network

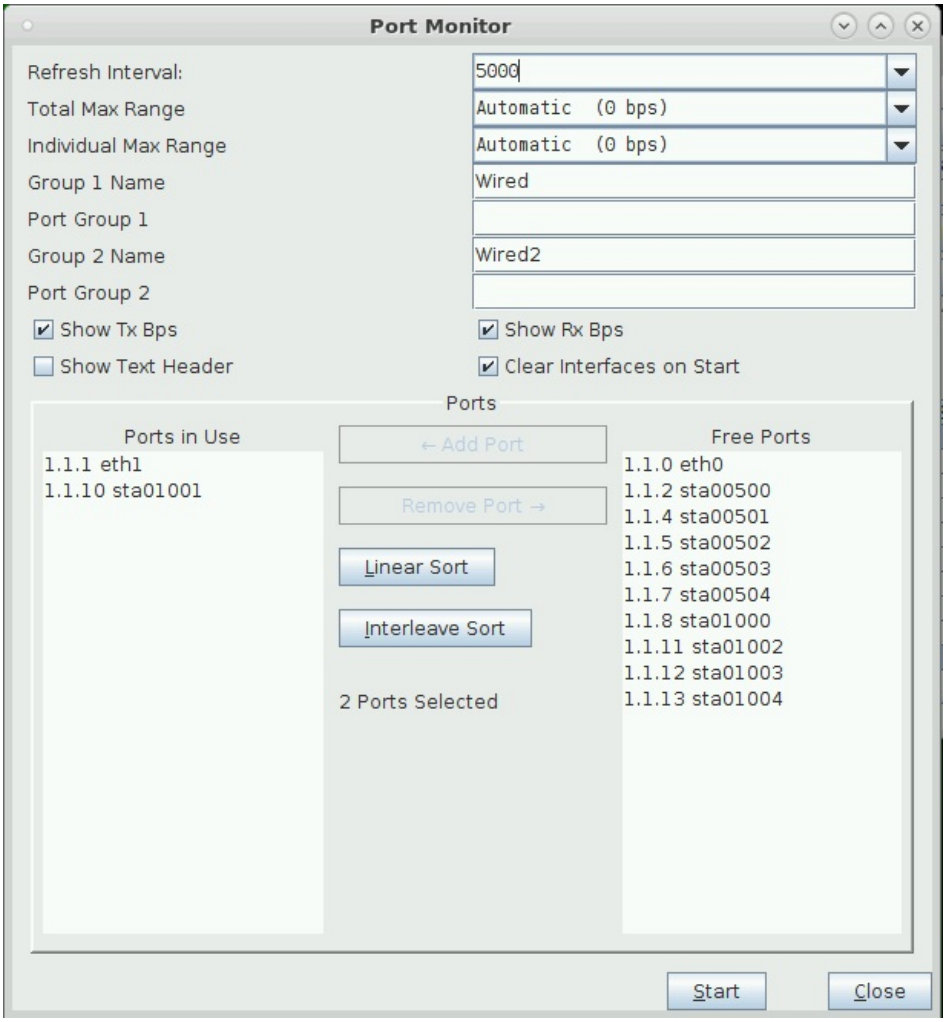
4. After starting the Layer-3 connection, click on the Port Mgr tab; you will see your connection ports highlighted.



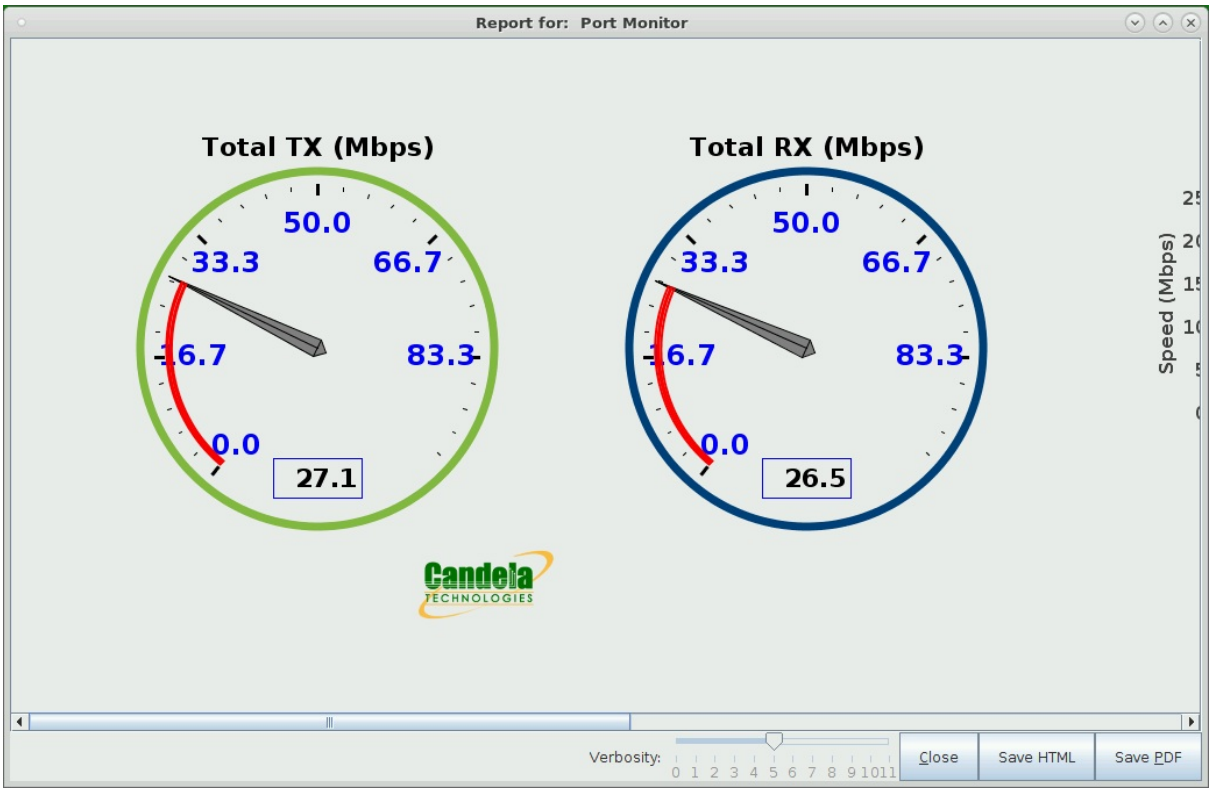
5. Right click on one of the selected ports, and select **Port Monitor**.



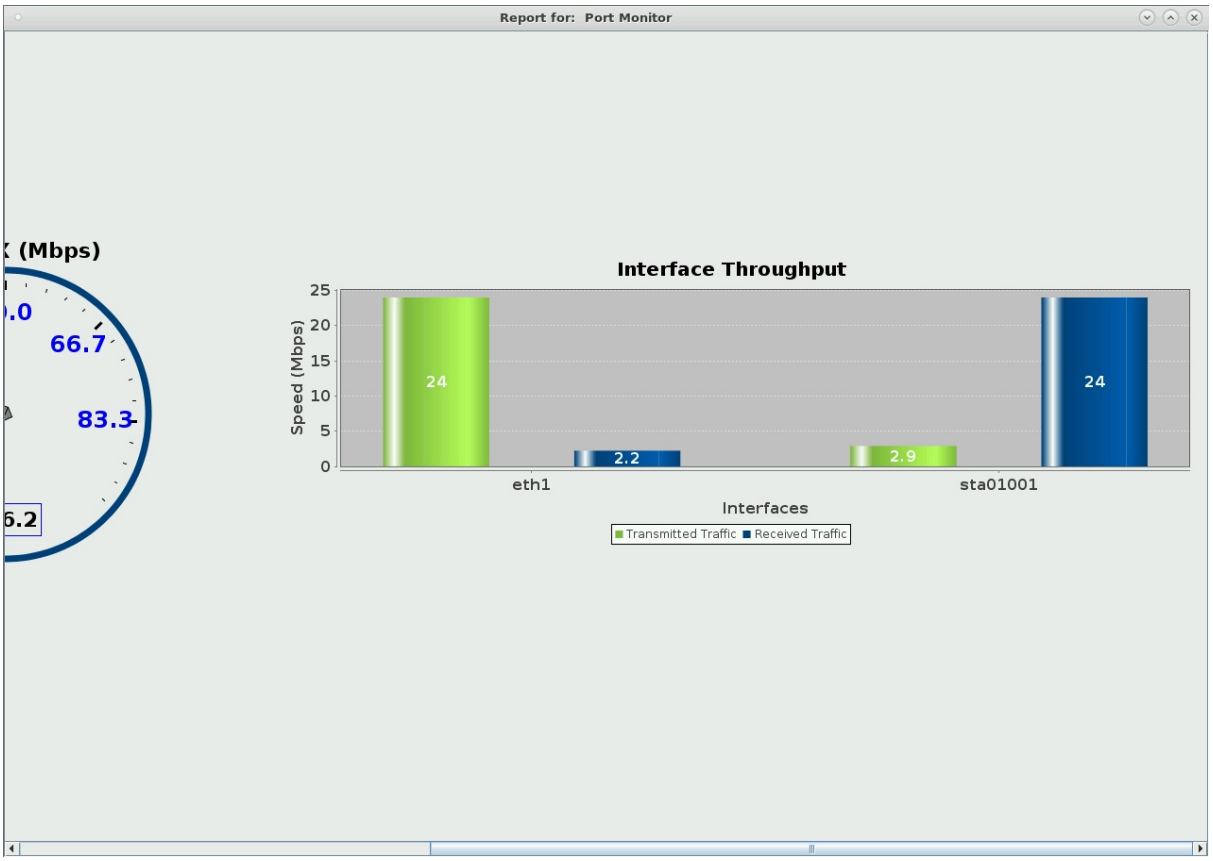
6. You will see your two ports in the port selector. Click **Start** to begin the monitor.



7. The monitor display was designed to be a full-screen window. You'll see it scroll off to the right. Maximize the window to see more.



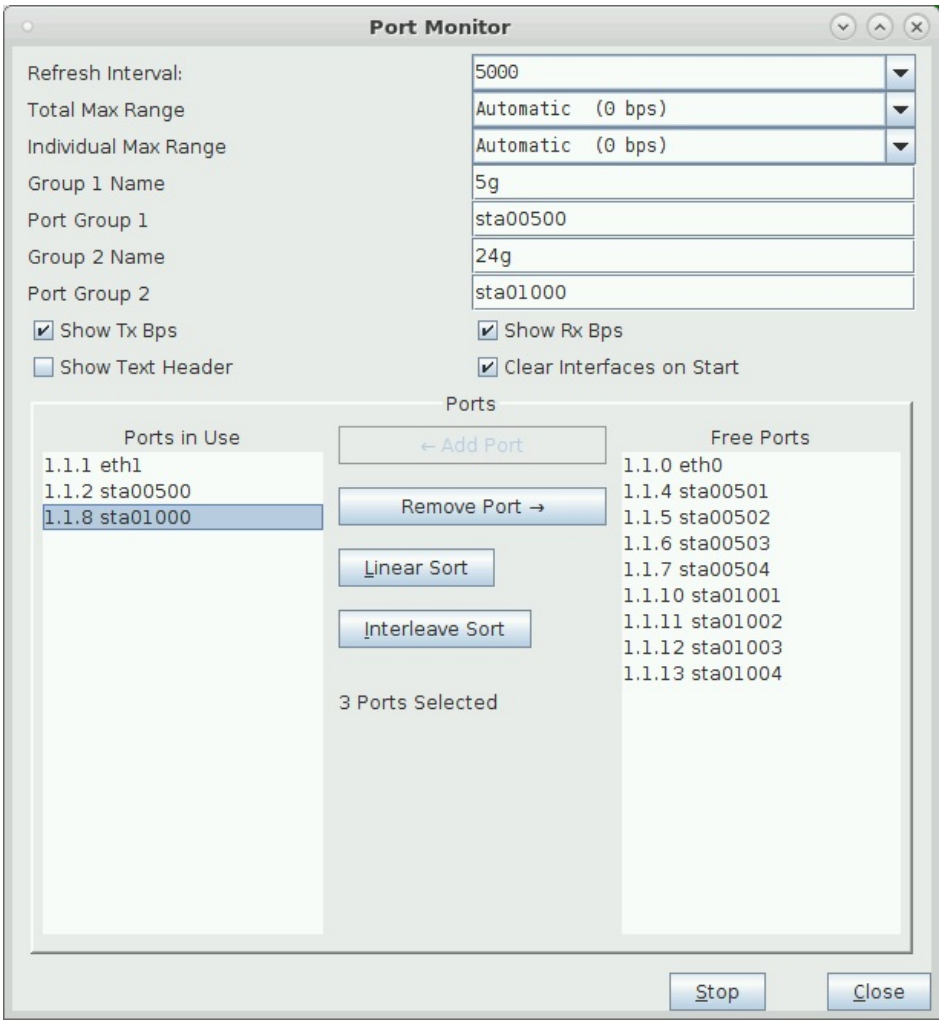
8. The bar chart on the right side shows port totals.



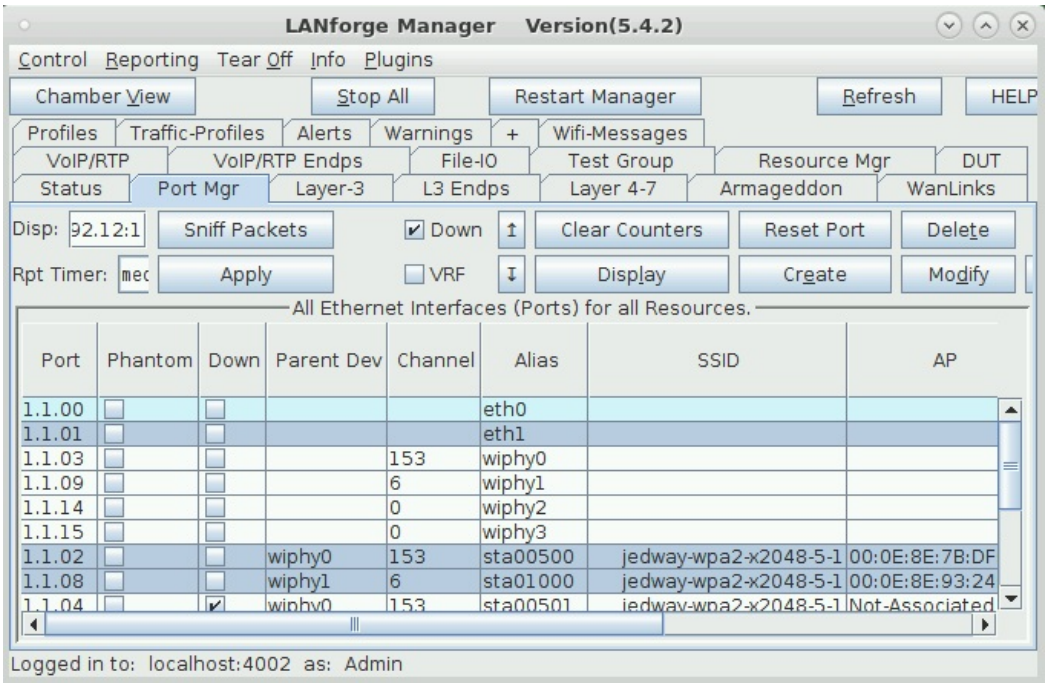
9.

Displaying Groups of Ports

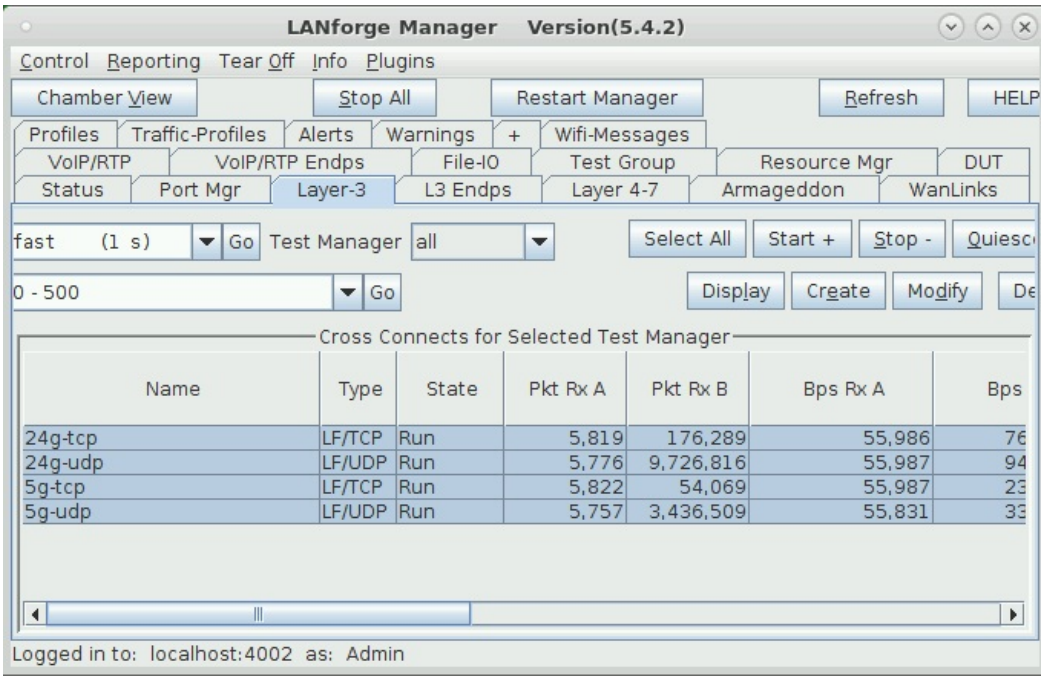
10. You can use the Port Group 1 and Port Group 2 fields in the settings window to display items. The window pictured here shows a station on a 2.4Ghz band and another station on a 5Ghz band.



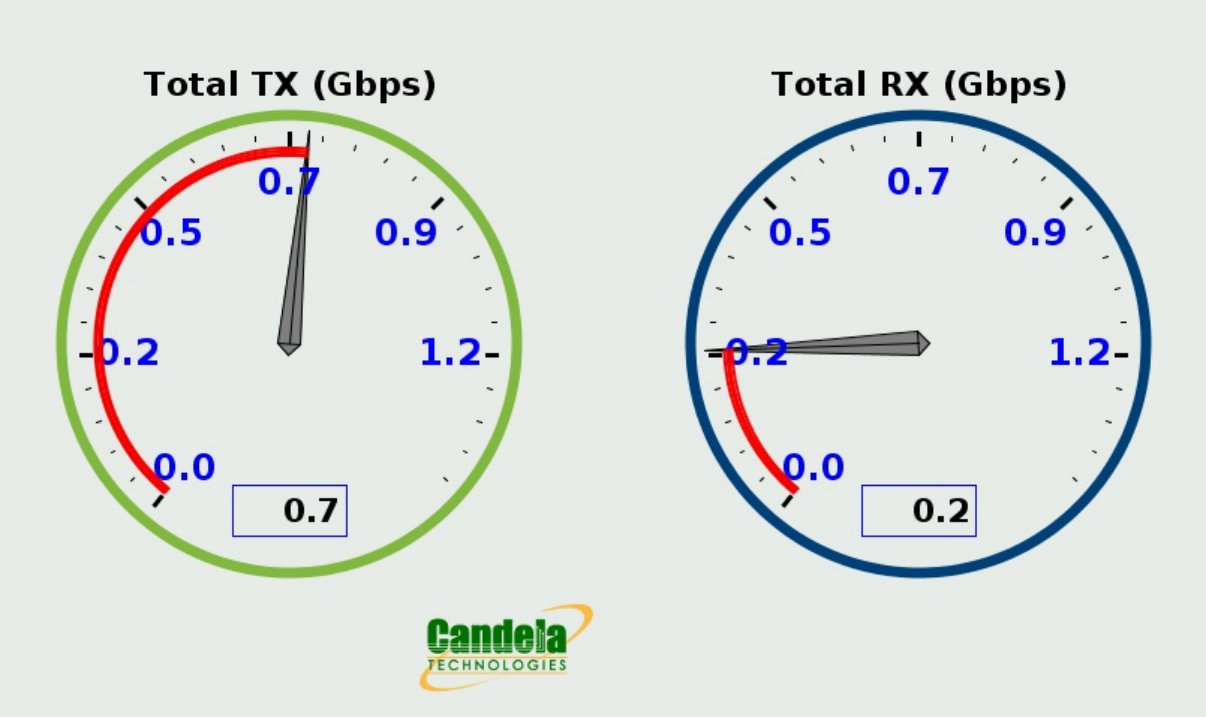
11. The ports for these settings show different channels



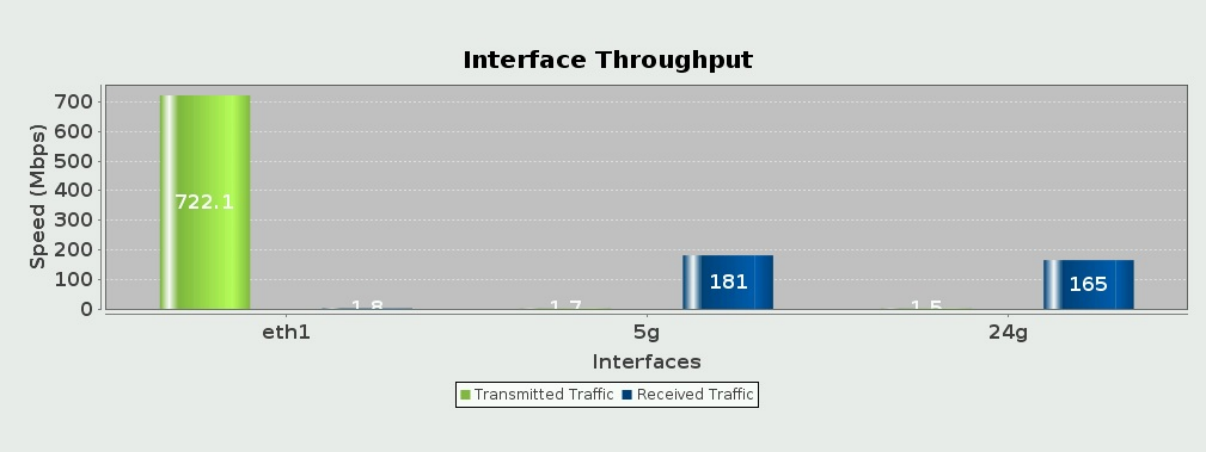
12. The connections for these ports are both attempting 300Mbps download



13. We see a lot of TX from `eth1` in the dial, but not much recieved.



14. We can compare the 2.4Ghz ports and the 5Ghz ports in the bar charts



Create Python Scripts Utilizing the Realm Library

Goal: Create a python script to create stations and Layer-3 cross connects

Using the `realm.py` library we will write a script that will allow us to automate the creation of stations and Layer-3 cross connects. We will also be able to start and stop traffic over the cross connects using the script. We will be referencing the script, test_ipv4_variable_time.py, as an example throughout this cookbook. Requires LANforge 5.4.2.

---

1. ## Starting the script

   A. Setting up inheritance for our object

      A. In order for our script to be platform independent we will need to `import sys`. Then use

      ```
      if 'py-json' not in sys.path:
          sys.path.append(os.path.join(os.path.abspath('..'), 'py-json'))
      ```

      B. When creating our object we will need to import the LFCliBase module from the LANforge module using `from LANforge.lfcli_base import LFCliBase`

      C. After importing LFCliBase we can create our Class and inherit from LFCliBase

   B. Setting up the main method

      A. The main method will typically follow a pattern:

         I. First, the creation of a list of stations. This can be done in many ways. Example:

         ```
         station_list = LFUtils.port_name_series(prefix_="sta",
             start_id_=0,
             end_id_=4,
             padding_number_=10000)
         ```

         II. Following the station list, we can initialize our object:

         ```
         ip_var_test = IPV4VariableTime(lfjson_host, lfjson_port,
             number_template="00",
             sta_list=station_list,
             name_prefix="var_time",
             ssid="testNet",
             password="testPass",
             resource=1,
             security="wpa2",
             test_duration="5m",
             side_a_min_rate=256,
             side_b_min_rate=256)
         ```

         III. After our object has been initialized we can begin the testing process. The preferred order for running our tests is to:

            i. Call `cleanup()` to prevent stations, cross-connects, and endpoints within our list from having creation issues if anything exists with the same name.

            ii. Call the `build()` method in our class to setup the basic versions of the stations, cross-connects, and endpoints.

            iii. Call the `start()` method that will start the test itself, as well as any bring up any stations and start traffic on cross-connects that need it.

            iv. Call the `stop()` method to stop the traffic and bring down any stations that are up.

            v. Verify that the tests passed using our inherited passes() method.

            vi. After verifying a pass we can then call our cleanup function again to clean up everything we worked with.

C. Example Main Method

```
def main():
    lfjson_host = "localhost"
    lfjson_port = 8080
    station_list = LFUtils.portNameSeries(prefix_="sta", start_id_=0, end_id_=4, padd
    ip_var_test = IPV4VariableTime(lfjson_host, lfjson_port, number_template="00", sta
        name_prefix="var_time",
        ssid="testNet",
        password="testPass",
        resource=1,
        security="wpa2", test_duration="5m",
        side_a_min_rate=256, side_b_min_rate=256),
    ip_var_test.cleanup(station_list)
    ip_var_test.build()
    if not ip_var_test.passes():
        print(ip_var_test.get_fail_message())
        exit(1)
    ip_var_test.start(False, False)
    ip_var_test.stop()
    if not ip_var_test.passes():
        print(ip_var_test.get_fail_message())
        exit(1)
    time.sleep(30)
    ip_var_test.cleanup(station_list)
    if ip_var_test.passes():
        print("Full test passed, all connections increased rx bytes")
```

2.

Test Methods Available With Realm

A. Using `lfcli_base._pass()` and `lfcli_base._fail()`

A. Since our class is inheriting lfcli_base.py, we have access to methods that will help us keep track of passes and fails during our tests. We can access them using `self._pass()` or `self._fail()`. They will take two parameters, a string `message` and an optional boolean `print_pass` and `print_fail` for `_pass()` and `_fail()` respectively. If `print_pass` or `print_fail` are set to True, they will write the message to stdout whenever the functions are called.

B. `lfcli_base` will add a "PASSED: message" or "FAILED: message" to a list when the tests pass or fail. This list can be accessed using the methods

```
get_result_list()
get_failed_result_list()
get_fail_message()
get_all_message()
```

B. Using lfcli_base to check test success

A. `passes()` will return a boolean depending on whether or not there were any fails in the test. If it finds a fail message it will return False, if none are found it will return True.
`get_result_list()` will return all logged pass/fail messages as a list.
`get_failed_result_list()` will return a list of only fail messages.
`get_fail_message()` will return a list of string of fail messages separated by newlines
`get_message()` will return a list of string of all messages separated by newlines

3.

Building a Station

A. Build Method

A. We will need to do a number of things to setup our build method.

I. To begin we will set the security type of our stations using `station_profile.use_security()`

II. We will then use `station_profile.set_number_template()` to name our stations

III. After this we can set our command flags and parameters using

```
self.station_profile.set_command_flag("add_sta","create_admin_down",1)
self.station_profile.set_command_param("set_port","report_timer",1500)
self.station_profile.set_command_flag("set_port","rpt_timer", 1)
```

IV. Once our parameters and flags are set, we can pass a list of stations to `station_profile.create()` and `cx_profile.create()`. Our build function could look like this:

```
for station in range(len(self.sta_list)):
    temp_sta_list.append(str(self.resource)+"."+self.sta_list[station])
self.station_profile.create(resource=1, radio="wiphy0", sta_names_=self.sta_list, debug=Fals
self.cx_profile.create(endp_type="lf_udp", side_a=temp_sta_list, side_b="1.eth1", sleep_time
self._pass("PASS: Station build finished")
```

> i The naming convention for the sides will look like `foo-A` for side_a and `foo-B` for side_b. foo will be set based on the names in the list of stations given.

B. StationProfile

A. The preferred method for creating a station_profile is to use the factory method `new_station_profile()` found in realm

I. We will need to assign some variables for the creation of our stations before we can call `create()`.

i. `self.station_profile.use_security(`*`security_type, ssid, passwd`*`)` is the preferred method to use when setting the security type, ssid, and password variables
Example:
`self.station_profile.use_security("wpa2", "testNet", "testPass")`

ii. `self.station_profile.number_template_` is the numerical prefix for stations. Using a `number_template` of "00" will have stations look like sta01, sta02...sta10
Example:
`self.station_profile.number_template_="00"`

iii. `self.station_profile.mode` determines the wifi mode used by the stations. See here for available modes
Example:
`self.station_profile.mode=0`

4.

Cross Connects

A. Starting and Stopping Traffic

A. In order for us to be able to start traffic, our stations will need to be admined up, associated, and with an IP. We can bring them up using `station_profile.admin_up()`. We can then use `realm.wait_for_ip(resource, sta_list)` to wait for our stations, as well as eth1, to get an IP address.

B. Once we are sure all of our stations have ip addresses, we can use `cx_profile.start_cx()` to start the traffic for our cross-connects. When we decide to stop the traffic we can just as easily use `cx_profile.stop_cx()` to stop traffic.

B. L3CXProfile

A. `self.local_realm.create_new_l3_cx_profile()` is the preferred method for creating a new Layer 3 CX Profile.

I. We will need to assign some variables for the creation of our stations before we can call `create()`.

i. `self.cx_profile.name_prefix` will be used to specify the name prefix for the cx. Assigning `self.cx_profile.name_prefix` to "test_" would produce cross-connects named test_sta00 with the numbers being dependent on station_profile's `number_template`.
Example:
```
self.cx_profile.name_prefix="test_"
```

ii. Set the `_min_bps` to the desired amount. `_max_bps` can be set but typically defaults to 0 which sets it to the same as the minimum bps.
Example:
```
self.cx_profile.side_a_min_bps=56000
self.cx_profile.side_b_min_bps=56000
```

5.

Using TTLS

A. TTLS setup requires a few pieces of information to work correctly. `StationProfile` has a `set_wifi_extra()` method for setting the relevant variables. See here for the available options

B. We will need a key management type (`key_mgmt`), an EAP method (`eap`), an EAP identity string (`identity`), an EAP password string (`passwd`), an 802.11u realm (`realm`), an 802.11u domain (`domain`), and an 802.11u HESSID (`hessid`)
Example:
```
key_mgmt="WPA-EAP"
eap="TTLS"
identity="testuser"
passwd="testpasswd"
realm="localhost.localdomain"
domain="localhost.localdomain"
hessid="00:00:00:00:00:01"
```
We can then use these variables to call the `set_wifi_extra()` method
Example:
```
station_profile.set_wifi_extra(key_mgmt, eap, identity, passwd, realm, domain, hessid
```

6.

Cleaning Up

A. Cleanup stations and cross connects

A. We have two options for cleaning up everything once we finish:

I. The preferred method to cleanup is to use the individual cleanup methods found in `StationProfile` and `L3CXProfile`. These are `station_profile.cleanup(resource, desired_station_list)` and `cx_profile.cleanup()`. These methods are preferred because they will only delete stations, cross-connects, and endpoints created during the test while leaving others untouched.This is useful if you are running other scripts in the background.

II. The other method for cleanup is to use `Realm`'s `remove_all_stations(), remove_all_endps(), and remove_all_cxs()` methods. These will remove all stations, cxs, and endpoints that exist. These are good for doing a full cleanup, and it is recommended to use them in the order of cx, endpoint, station to prevent potential issues or missed deletions.

7.

Debugging Stations

A. Debug information for station creation can be output by setting `_debug_on=True` in `StationProfile.create()`

   A. There are a few important debug outputs to pay attention to:

      I. This is the debug output that appears when using the add_sta command. This is used frequently in `StationProfile.create()`. This debug output will allow you to troubleshoot any **flags** or other information that is being set when creating your stations. It will output the name at the top and the raw JSON data will follow.

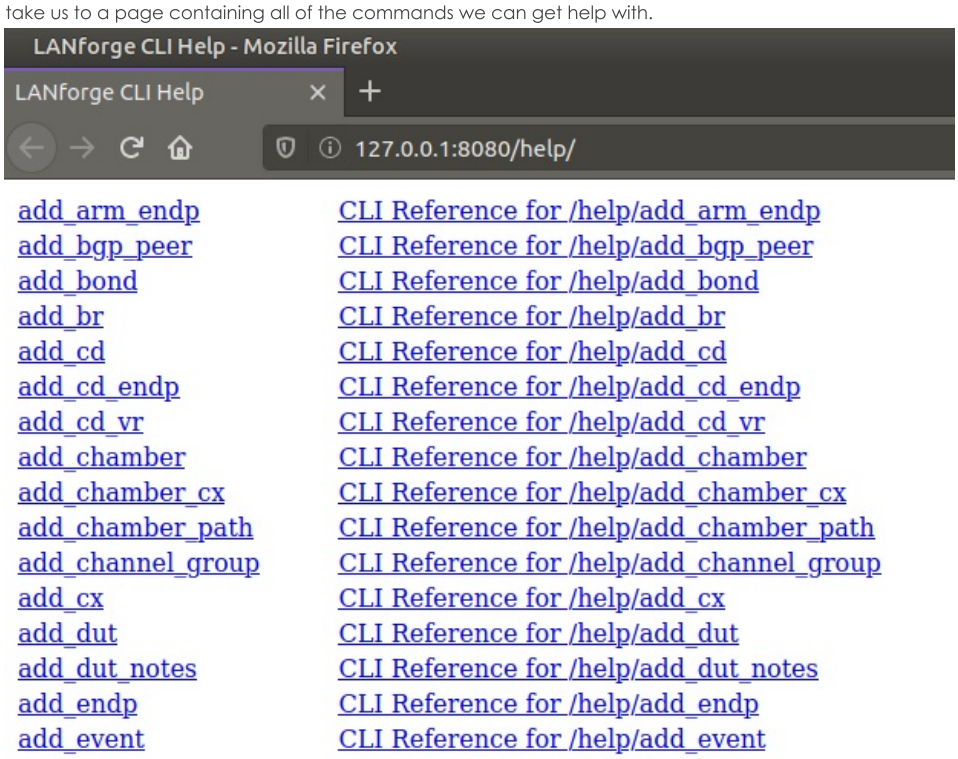
```
- 381 - sta0000- - - - - - - - - - - - - - - - -
{'flags': 132096,
'flags_mask': 68719608832,
'key': 'testPass',
'mac': 'xx:xx:xx:xx:*:xx',
'mode': 0,'
radio': 'wiphy0',
'resource': 1,
'shelf': 1,
'ssid': 'testNet,
'sta_name': 'sta0000'}
```

      II. The next bit of debugging output comes from using the set_port command. We are able to see all of the JSON data that is posted, and can use this to check our flags and other info.

```
{'current_flags': 2147483649,
'interest': 8437762,
'port': 'sta0000',
'report_timer': 1500,
'resource': 1,
'shelf': 1}
<LANforge.LFRequest.LFRequest object at 0x7f13dbc56850>
-~381 - - - - - - - - - - - - - - - - - -
```

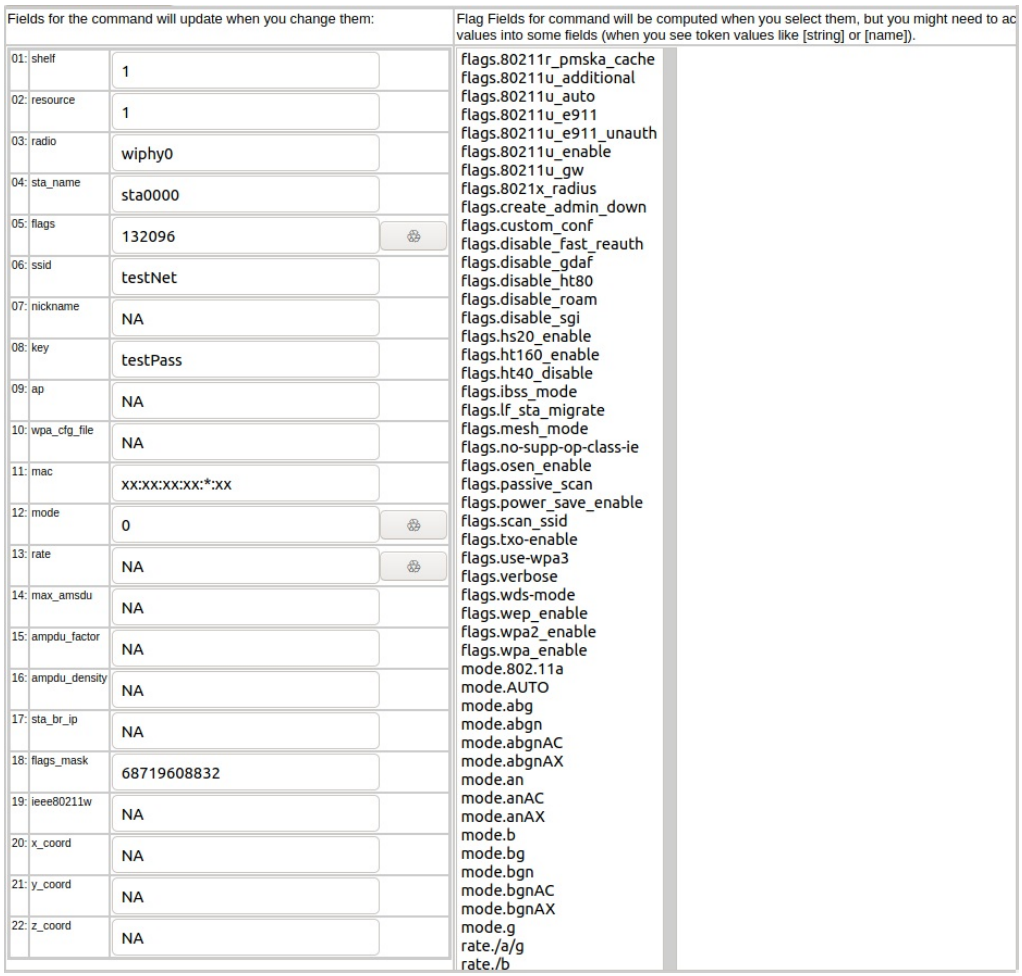
B. There are a few steps we can take to make validating the information we get through debugging easier.

   A. We can use the help page available on the address of the machine LANforge is running on. http://127.0.0.1/help/ will take us to a page containing all of the commands we can get help with.

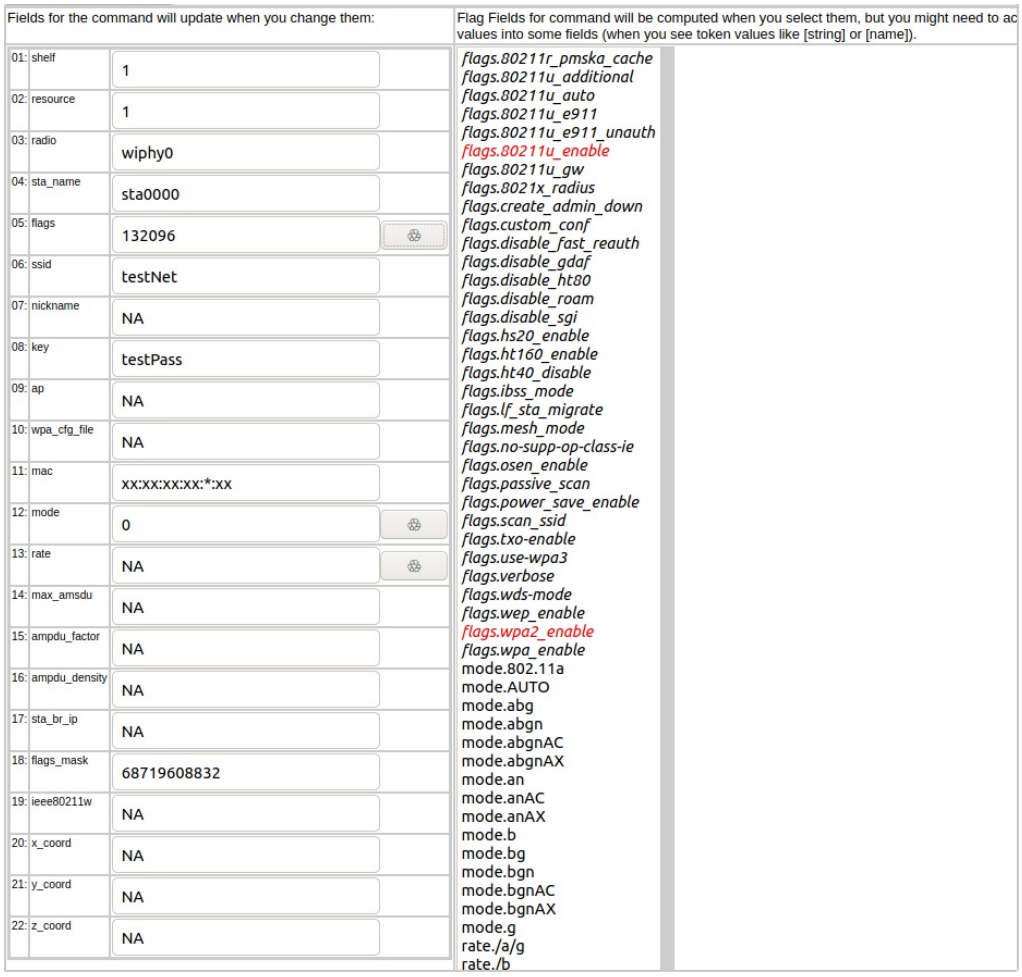


B. Using http://127.0.0.1/help/add_sta will bring us to a page specific to the `add_sta` command.

C. Here we can enter all of the data we got from our debugging output into the correct areas.



D. Flag fields have a button next to them that will calculate and highlight relevant flags in the right hand column of the page. This can be useful for checking that the correct flags are being set.



E. After we have done this, we can click the **parse command** button towards the top of the data inputs. We can then enter this command into LANforge's messages tab in the input box.

Create Python Scripts To Test Layer 4 Traffic

Goal: Create a script to test Layer 4 traffic using Realm

Using the `realm.py` library we will write a script that will allow us to automate the creation of stations and Layer 4 cross connects. We will also be able to start and stop traffic over the cross connects using the script. Station and Cross Connect creation is covered in the Realm Scripting Cookbook. Requires LANforge 5.4.2.

1. ## Creating The Profile

A. We will use the factory method `self.local_realm.new_l4_cx_profile()` to create our profile object.

B. After we have done this we can set a few variables for our traffic:

   A. `l4_cx_profile.requests_per_ten` will set our rate of requests per ten minutes. Setting `requests_per_ten = 600` will set our URL request rate to 1 per second. There is no limit to what can be used as the rate but common rates are:

      - 600 : 1/s
      - 1200 : 2/s
      - 1800 : 3/s
      - 2400 : 4/s

   B. `l4_cx_profile.url` is the URL to be used in the requests. We will also need to specify the direction (dl/ul) and a absolute path for the destination. See syntax here.
   Example:
   ```
   l4_cx_profile.url = "dl http://10.40.0.1 /dev/null"
   ```

   C. Example Layer 4 profile init:
   ```
   class IPV4L4(LFCliBase):
       def __init__(self, host, port, ssid, security, password, url, requests_per
           target_requests_per_ten=600, number_template="00000", resource=1, num_
           _debug_on=False,
           _exit_on_error=False,
           _exit_on_fail=False):
       super().__init__(host, port, _debug=_debug_on, _halt_on_error=_exit_on_er
       self.host = host
       self.port = port
       self.ssid = ssid
       self.security = security
       self.password = password
       self.url = url
       self.requests_per_ten = requests_per_ten
       self.number_template = number_template
       self.sta_list = station_list
       self.resource = resource
       self.num_tests = num_tests
       self.target_requests_per_ten = target_requests_per_ten

       self.local_realm = realm.Realm(lfclient_host=self.host, lfclient_port=self
       self.cx_profile = self.local_realm.new_l4_cx_profile()
       self.cx_profile.url = self.url
       self.cx_profile.requests_per_ten = self.requests_per_ten

       # Station Profile init
   ```

2. ## Starting Traffic

A. When running traffic, if you plan to measure the rate of requests, it is recommended to do so in 10 minute increments. An example of this can be seen here: test_ipv4_l4_urls_per_ten.py. To start the traffic we can use the `l4_cx_profile.start_cx()` method. To stop the traffic we can use the `l4_cx_profile.stop_cx()` method.

B. Example start and build method:

```
def build(self):
    # Build stations
    self.station_profile.use_security(self.security, self.ssid, self.password)
    print("Creating stations")
    self.station_profile.create(resource=1, radio="wiphy0", sta_names_=self.sta_list,
    temp_sta_list = []
    for station in range(len(self.sta_list)):
        temp_sta_list.append(str(self.resource) + "." + self.sta_list[station])

    self.l4_profile.create(ports=temp_sta_list, sleep_time=.5, debug_=self.debug, sup
```

```
def start(self, print_pass=False, print_fail=False):
    temp_stas = self.sta_list.copy()
    temp_stas.append("eth1")
    cur_time = datetime.datetime.now()
    interval_time = cur_time + datetime.timedelta(minutes=1)
    passes = 0
    expected_passes = 0
    self.station_profile.admin_up(1)
    self.local_realm.wait_for_ip(self.resource, temp_stas)
    self.l4_profile.start_cx()
    print("Starting test")
    for test in range(self.num_tests):
        expected_passes += 1
        while cur_time < interval_time:
            time.sleep(1)
            cur_time = datetime.datetime.now()

        if self.l4_profile.check_errors(self.debug):
            if self.__check_request_rate():
                passes += 1
            else:
                self._fail("FAIL: Request rate did not exceed 90% target rate", print_
                break
            else:
                self._fail("FAIL: Errors found getting to %s " % self.url, print_fail
                break
            interval_time = cur_time + datetime.timedelta(minutes=1)
    if passes == expected_passes:
        self._pass("PASS: All tests passes", print_pass)
```

3. ## Examining The Results

A. We can use http://localhost:8080/layer4/list to check our Layer 4 endpoints. Adding a ,,?fields to the end of the URL will allow us to specify what we want to look at. We can separate fields by commas to show more than one at a time.
Example: http:/localhost:8080/layer4/list?fields=name,urls/s,total-urls

- Using `total-urls` will show us the total requests made.
- Using `urls/s` will show us the average URL rate per second.
- Using `rx rate` and `tx rate` will show us the rates of received and transeferred traffic.

We can also use the url http://localhost:8080/layer4/all to see all of the available fields.

B. When checking our results for Layer 4 tests we might want to check for common URL related errors:

- `acc. denied` will show us the number of times we got an access denied error.
- `bad-url` will show us the number of times a request was made with an invalid URL.
- `nf (4xx)` will count the number of 400 errors recieved when making requests to our URL.

Create Python Scripts To Test Generic Traffic

Goal: Create a script to test Generic traffic using Realm

Using the `realm.py` library we will write a script that will allow us to automate the creation of stations and generic cross connects. We will also be able to start and stop traffic over the cross connects using the script. Station and Cross Connect creation is covered in the Realm Scripting Cookbook. Requires LANforge 5.4.2.

---

1. ## Creating The Profile

   A. We will use the factory method `self.local_realm.new_generic_cx_profile()` to create our profile object.

   B. After we have done this we can set a few variables for our traffic:

      A. `gen_cx_profile.type` will determine the type of command to execute.
      Example: `self.cx_profile.type = "lfping"`

      B. `gen_cx_profile.dest` is the destination IP address for the command.
      Example: `self.cx_profile.dest = "127.0.0.1"`

      C. `gen_cx_profile.interval` sets the interval at which the command is run in seconds.
      Example: `self.cx_profile.interval = 1`

      D. Example Generic profile init:

```
class GenTest(LFCliBase):
    def __init__(self, host, port, ssid, security, password, sta_list, name_pr
                 number_template="00000", test_duration="5m", type="lfping", c
                 interval=1, radio="wiphy0",
                 _debug_on=False,
                 _exit_on_error=False,
                 _exit_on_fail=False):
        super().__init__(host, port, _debug=_debug_on, _halt_on_error=_exit_or
        self.host = host
        self.port = port
        self.ssid = ssid
        self.radio = radio
        self.upstream = upstream
        self.sta_list = sta_list
        self.security = security
        self.password = password

        self.number_template = number_template
        self.name_prefix = name_prefix
        self.test_duration = test_duration

        self.local_realm = realm.Realm(lfclient_host=self.host, lfclient_port=
        self.cx_profile = self.local_realm.new_generic_cx_profile()
        self.cx_profile.type = type
        self.cx_profile.dest = dest
        self.cx_profile.interval = interval

    # Station Profile init
```

2. ## Starting Traffic

   A. To start the traffic we can use the `gen_cx_profile.start_cx()` method. To stop the traffic we can use the `gen_cx_profile.stop_cx()` method.

B. Example start and build method:

```
def build(self):
    self.station_profile.use_security(self.security, self.ssid, self.password)
    self.station_profile.set_number_template(self.number_template)
    print("Creating stations")
    self.station_profile.set_command_flag("add_sta", "create_admin_down", 1)
    self.station_profile.set_command_param("set_port", "report_timer", 1500)
    self.station_profile.set_command_flag("set_port", "rpt_timer", 1)
    self.station_profile.create(radio=self.radio, sta_names_=self.sta_list, debug=sel
    self.cx_profile.create(ports=self.station_profile.station_names, sleep_time=.5)
    self._pass("PASS: Station build finished")
```

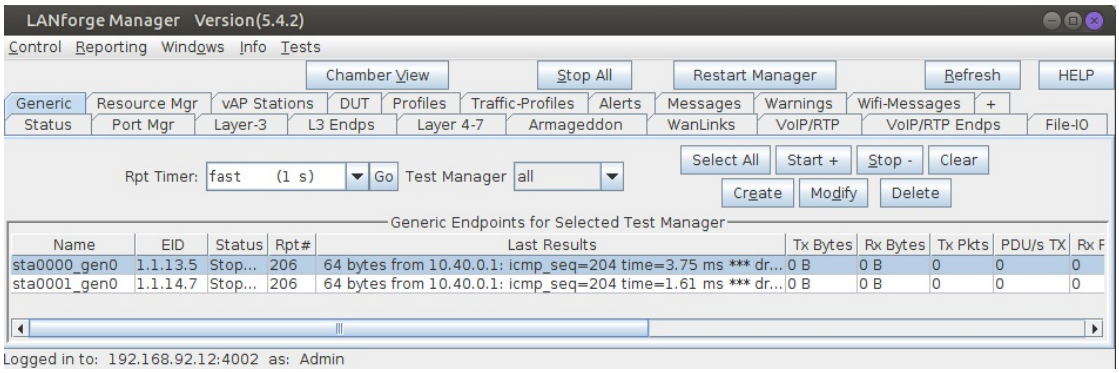
```
def start(self, print_pass=False, print_fail=False):
    self.station_profile.admin_up()
    temp_stas = self.sta_list.copy()
    temp_stas.append(self.upstream)
    if self.local_realm.wait_for_ip(temp_stas):
        self._pass("All stations got IPs", print_pass)
    else:
        self._fail("Stations failed to get IPs", print_fail)
        exit(1)
    cur_time = datetime.datetime.now()
    passes = 0
    expected_passes = 0
    self.cx_profile.start_cx()
    time.sleep(15)
    end_time = self.local_realm.parse_time("30s") + cur_time
    print("Starting Test...")
    while cur_time < end_time:
        cur_time = datetime.datetime.now()
        gen_results = self.json_get("generic/list?fields=name,last+results", debug_=se
        if gen_results['endpoints'] is not None:
            for name in gen_results['endpoints']:
                for k, v in name.items():
                    if v['name'] in self.cx_profile.created_endp and not v['name'].end
                        expected_passes += 1
                        if v['last results'] != "" and "Unreachable" not in v['last re
                            passes += 1
                        else:
                            self._fail("%s Failed to ping %s " % (v['name'], self.cx_p
                            break
        time.sleep(1)
    if passes == expected_passes:
        self._pass("PASS: All tests passed", print_pass)
```

3. ## Examining The Results

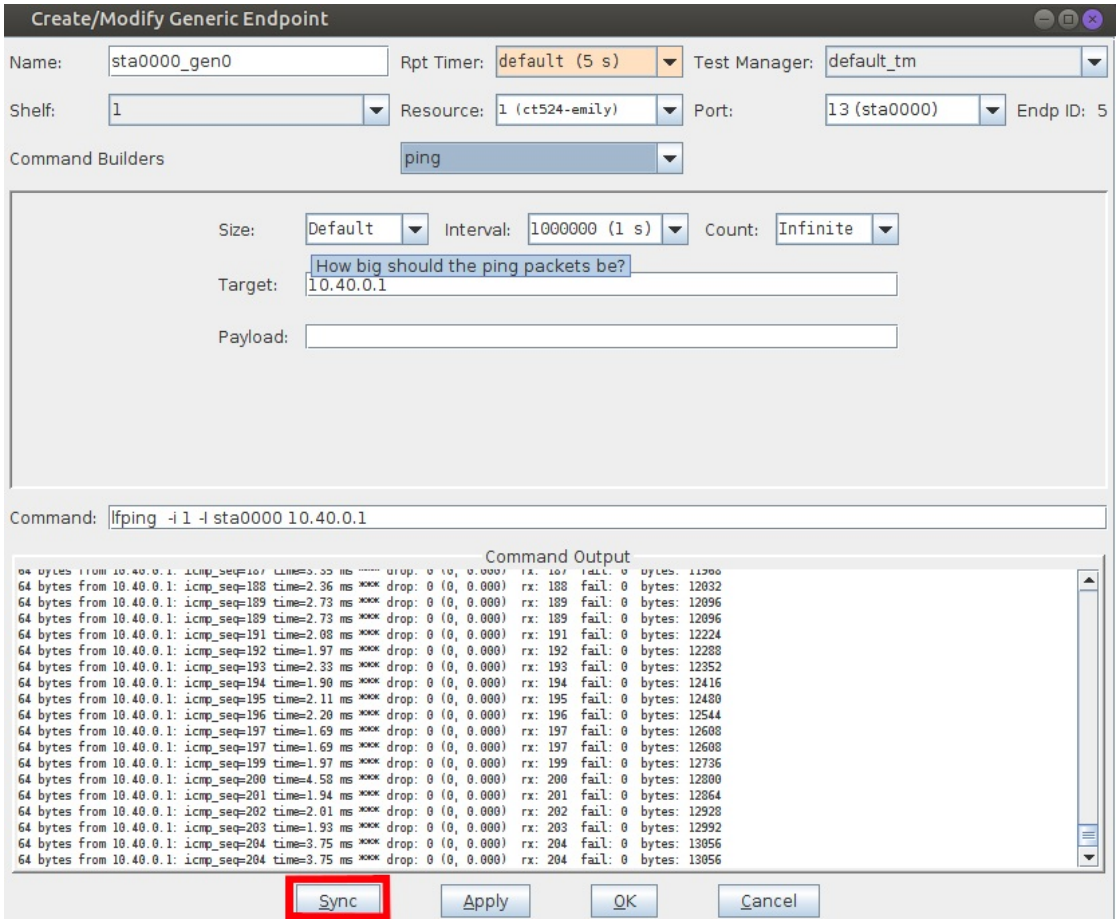
A. For `lfping` we can use the last results of the endpoint to determine if the test was successful. An example of this can be seen in our `start` method. The most common errors for `lfping` will either be a blank last result or `Destination Host Unreachable`. Either of these results indicate a failed ping. Successful pings will look like:

```
64 bytes from 10.40.0.1: icmp_seq=1 time=4.55 ms *** drop: 0 (0, 0.000)  rx: 1  fail:
```

Results can also be seen in the generic tab in the LANforge Manager:



Double-clicking on an endpoint will allow you to see more specific results as well as the command used by the endpoint. Using the **sync** button will allow you to see updated results.



Create Layer 4-7 HTTP Upload Traffic

Goal: using curl, it is possible to not only download content, but UPLOAD content as well. There must be a web source that accepts uploaded data. Below we will cover how to construct an PHP and NGINX web page that will allow a LANforge system to accept web uploads.

Architecture of Uploads

By default, web servers (like Apache or Nginx) are designed to respond by giving content. Receiving content requires some add-on to accept incoming uploads. The notable methods are CGI and WebDAV. WebDAV configurations are specific to either Apache or Nginx, so we will focus this cookbook on the CGI method, with PHP as the CGI engine. PHP is installed by default with LANforge systems.

Deciding on the Upload Destination

Continual traffic generation is a scenario that would fill the space of any web server. This cookbook describes a technique that will not save uploaded data, and thus should allow an indefinite duration of upload traffic. Specifically, processes that accept uploaded data **do not need to save it**. The technique used by up.php relies on the files being uploaded to `/tmp` which on most recent LANforge systems is a `tmpfs` RAM-based filesystem.

> i Using **`tmpfs`** for very large uploads, (or hundreds of small uploads) can consume LANforge system memory quickly.

Upload Considerations

1. There is no good way to ignore the uploaded output. It goes immediately to the filesystem.
2. The default fedora upload location is /tmp. This is a RAM-based filesystem. You will crash your LANforge nginx server, and possibly the machine itself if your uploads take up more than half your ram.
3. If you want to upload a lot of data per upload, it is going to have to live on the LANforge's SSD.
    1. In `/etc/php.ini`, set `upload_tmp_dir = "/home/lanforge/tmp"`
    2. Consider that SSDs have a writes-per-lifetime limit. Continual writing adds wear to the drive.
    3. SSDs can only write between 30-500MB/s (4000 Mbps). This depends on the model of SSD, the speed of the SATA bus, and the write-caching and wear leveling techniques the SSD uses.
    4. If doing HTTP uploads is a performance test, we suggest building a system with hundreds of gigabytes of RAM for `tmpfs` space.
4. To handle large numbers of stations doing uploads, set the upload size limit small to avoid filling up RAM.
    1. Monitor free RAM with **htop**. On a ct521a, there is 4GB of RAM, total. Thus you have up to 2GB of free space in /tmp:
    2. Remeber that as while the LANforge GUI runs, it consumes RAM itself. Depending on the duration of the report, it can consume most of the remaining ram. Consider running the LANforge GUI on a separate machine.
    3. In `/etc/php.ini`, set your post_max_size and upload_max_filesize parameters to (2GB / number of Layer 4 connections):
        1. For 16 stations with one Layer 4 connection each:
           `2 x 1024 x 1024 x 1024 / 16 = 134,217,728 ~= 134MB`
        2. For 32 stations with two Layer 2 connections each:
           `2 x 1024 x 1024 x 1024 / 64 = 33,554,432 ~= 33MB`
        3. The php.ini default max upload size is set at 2MB:

           ```
           post_max_size = 8M
           upload_max_filesize = 2M
           ```

        4. These system defaults **permit 1024 simultaneous uploads**.

Configure nginx to use up.php

1. Copy up.php to `/usr/local/lanforge/nginx/html`
2. In the Port Mgr tab, modify your port and select the ☑**HTTP** checkbox and click **Apply**. In this example, the port is **eth1** (192.168.48.92). Keep the window open.
3. In a terminal, edit `/home/lanforge/vr_conf/nginx_eth1.conf`
    1. Remove the first line
    2. At the bottom of the server {} stanza, add this configuration:

```
location /up.php {
     include /usr/local/lanforge/nginx/conf/fastcgi_params;
     fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
     fastcgi_pass unix:/run/php-fpm/www.sock;
     fastcgi_index index.php;
}
```

Example of file:

```
# Remove the first line '# Autogenerated by ...' and edit the file as
# desired for a custom config file.

worker_processes        1;
error_log logs/eth1_error.log;
pid             /home/lanforge/vr_conf/nginx_eth1.pid;
events {
    worker_connections  1024;
}

http {
    include       /usr/local/lanforge/nginx/conf/mime.types;
    default_type  application/octet-stream;
    access_log  logs/eth1_access.log;
    sendfile            on;
    keepalive_timeout  65;

    server {
        listen       192.168.48.92:80 bind_dev=eth1;
        server_name  localhost;
        access_log  logs/eth1_host.access.log;

        location / {
            root   html;
            index  index.html index.htm;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
        location /up.php {
             include /usr/local/lanforge/nginx/conf/fastcgi_params;
             fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
             fastcgi_pass unix:/run/php-fpm/www.sock;
             fastcgi_index index.php;
        }
   }
}
```

1. Save the file.
    1. Edit the `/etc/php-fpm.d/www.conf` file
        1. Look for the listen.acl_users parameter, and add the user `nobody`
        2. `listen.acl_users = apache,nginx,nobody`
        3. save
    2. Restart the php-fpm service:

```
# systemctl restart php-fpm.service
```

Monitor nginx errors

1. To monitor the nginx log file:
    1. Open a terminal
    2. Become root using `sudo -s`
    3.
```
# cd /usr/local/lanforge/nginx/logs
```
    4.
```
# tail -F *.log
```

2. Permission Denied errors:
    1. For this error: `[crit] connect() to unix:/run/php-fpm/www.sock failed (13: Permission denied) while connecting to upstream, client: 192.168.48.18, server: localhost, request: "GET /up.php HTTP/1.1", upstream: "fastcgi://unix:/run/php-fpm/www.sock:"`
    2. Edit the `/etc/php-fpm/www.conf` file mentioned above and restart the php-fpm service.

Apply changed nginx settings

1. In the **eth1 Configure Settings** window, you will toggle the HTTP setting to re-apply the changed configuration:
    1. Unselect ☐**HTTP**
    2. Click **Apply**
    3. Select ☑**HTTP**
    4. Click **Apply**

Checking for the up.php page with Curl

1. Open a terminal and become root:
    1. ```
       $ sudo -s ↵
       ```
    2. ```
       # cd /home/lanforge ↵
       ```
    3. ```
       # ./vrf_exec.bash wlan0 curl -sqv http://192.168.48.92/up.php ↵
       ```
    4. Look for errors, like Sorry, the page you are looking for is currently unavailable.

Browsing the up.php page with Firefox

1. To see this page from the LANforge desktop, you will use the `vrf_exec.bash` script.
2. Open a terminal and issue these commands:
    1. ```
       # xhost + ↵
       ```
    2. ```
       # su - ↵
       ```
    3. ```
       # cd /home/lanforge ↵
       ```
    4. ```
       # export DISPLAY=:1 ↵
       ```
    5. ```
       # ./vrf_exec.bash eth1 firefox http://192.168.48.92/up.php ↵
       ```

Creating a Generic Endpoint to upload with Curl

1. There a token files to upload, located in `/var/www/html` named dataslug{size}.bin.
2. The `curl -F` command will issue a POST in the multipart-form manner.
3. The `filename=` parameter is the name of the form's `[input]` parameter that matches the `[input type=file]` element in the page.
4. The `@/var/www/html/data_slug_256K.bin` is curl syntax for opening the named file.
5. Start off by practicing the vrf_exec.bash command from a terminal:
    1. `./vrf_exec.bash wlan0 curl -F 'filename=@/var/www/html/data_slug_256K.bin' http://192.168.48.92/up.php`

6. Once that works, go to the LANforge GUI Generic tab and create a new endpoint:
    1. Enter a name
    2. Select your resource and port (wlan0)
    3. Select Command Builder: Generic
    4. Enter the command:

```
./vrf_exec.bash wlan0 curl -F 'filename=@/var/www/html/data_slug_256K.bin' http://192.168.48.92/up.php
```

    5. Click **OK**
7. In the Generic tab, select your connection and click **Start**
    1. Double click the connection
    2. In the Create/Modify Generic Endpoint window, click the **Sync** button (at the bottom).
    3. You will see the HTML text of the result.

```
Dload  Upload   Total   Spent    Left  Speed
0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100  256k    0   622  100  256k  24960  10.0M --:--:-- --:--:-- --:--:-- 10.4M
<!DOCTYPE html>
<html lang="en">
<head>
 <title>Layer 4 Upload Test</title>
 <meta charset="utf-8">
 <style>
     body {
         font-family: "Century Gothic", "DejaVu Sans",Arial,Helvetica,sans-serif;
         margin: 1em auto;
     }
     h1 {
         text-align: center;
     }
 </style>
</head>
<body>
 <h1>Layer 4 File Upload Test</h1>
     <h1>Result:</h1>
     <ul>
         <li>File name: <tt><data_slug_256K.bin></tt></li>
         <li>File size: <tt><262144></tt></li>
         <li>File tmp: <tt></tmp/phpRLHVa2></tt></li>
     </ul>
</body></html>
```

    4. A command that might provide better output could look like:

```
./vrf_exec.bash wlan0 curl -s -o/tmp/$$.html -F 'filename=@/var/www/html/data_slug_256K.bin' http://192.168.48.92/up.php;
grep File /tmp/$$.html
```

    5. It provides terse output like:

```
<h1>Layer 4 File Upload Test</h1>
     <li>File name: <tt>[data_slug_256K.bin]</tt></li>
     <li>File size: <tt>[262144]</tt></li>
     <li>File tmp: <tt>[/tmp/phpfhDFMf]</tt></li>
```

The up.php Page

You can copy the up.php page as text from here:

Install LANforge on ESXi Server (Version 8)

Goal: Install LANforge on an ESXI Server, and confirm DHCP can be served over VLANs.

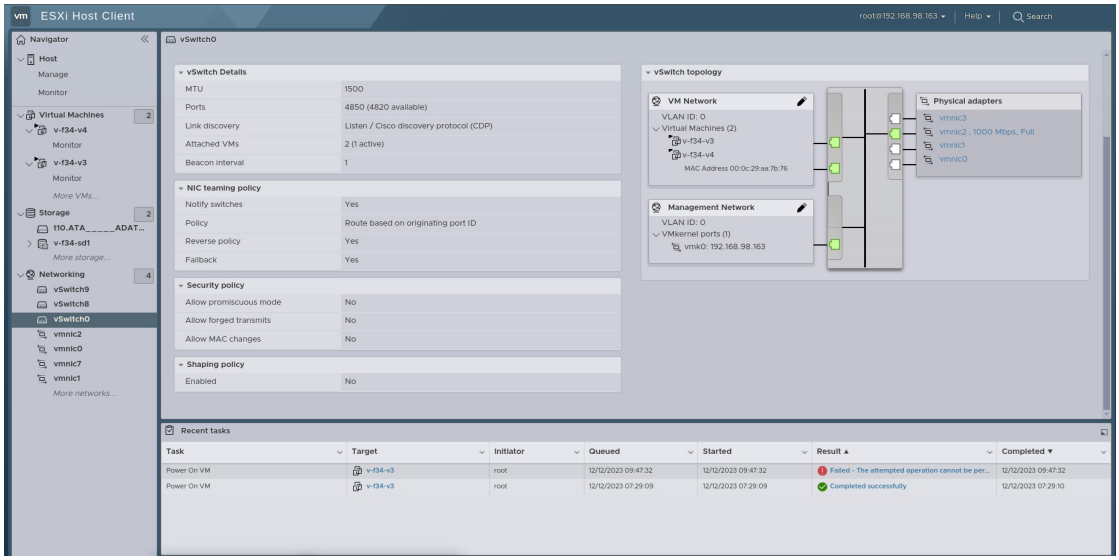
In this test scenario, we use a system compatible with ESXi version 8. ESXi version 8 can run 2 lanforge VMs on 2 different VLANs, host DHCP on one VLAN, and acquire DHCP leases on the other VLAN through a WANlink. This outlines expected behavior and uses cases for VLANs and WANlinks between LANforge and ESXi version 8. Requires LANforge version 5.3.5 or higher.

1. Configuring ESXi Server ver. 8.02
    A. This scenario was built upon ESXi 8.02 build 23305546, using a CT-525 with a 4 port 1g nic as well as an accompanying TPLink SG105E Smart Switch.

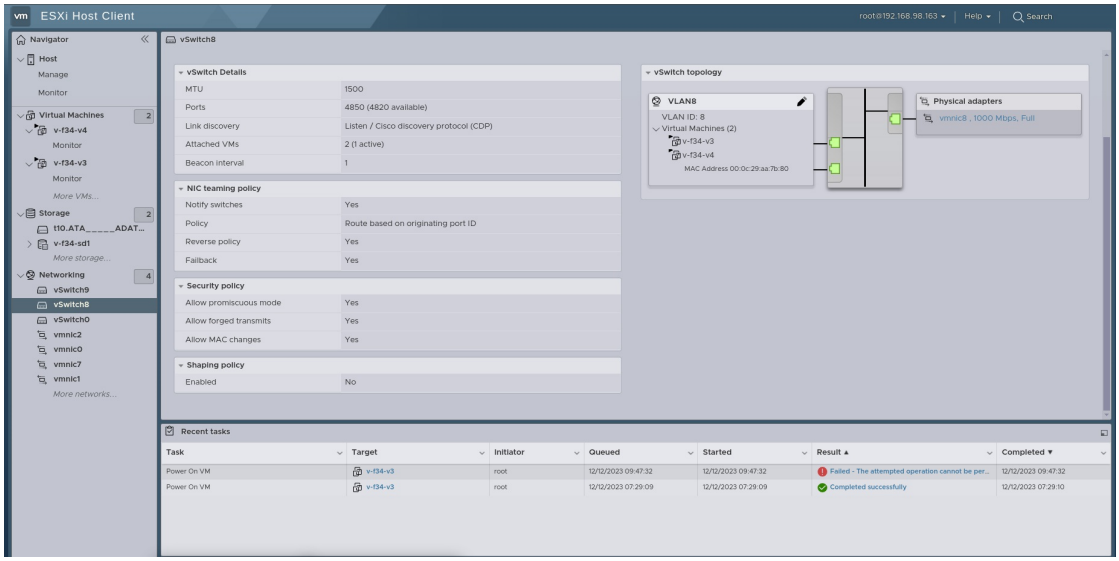
    B. Log into TPLink SG105E 5 port Smart Switch (usually defaults to 192.168.0.1). Assign ports 2 and 3 to a VLAN, name 8 with ID 8. Assign ports 4 and 5 to a VLAN, name 9 with ID 9. Port 1 will be the assigned trunk port.

    C. In ESXi, create 3 vSwitches. vSwitch0, vSwitch8, vSwitch9. vSwitch0 should have been created and configured by default as the management switch. vSwtich8 connects to ports 1 and 2 on the host port, this corresponds to vmnic8 and vmnic9. vSwtich9 connects to ports 3 and 4 on the host port, which corresponds to vmnic 10 and 11. Vmnic numbers may vary.

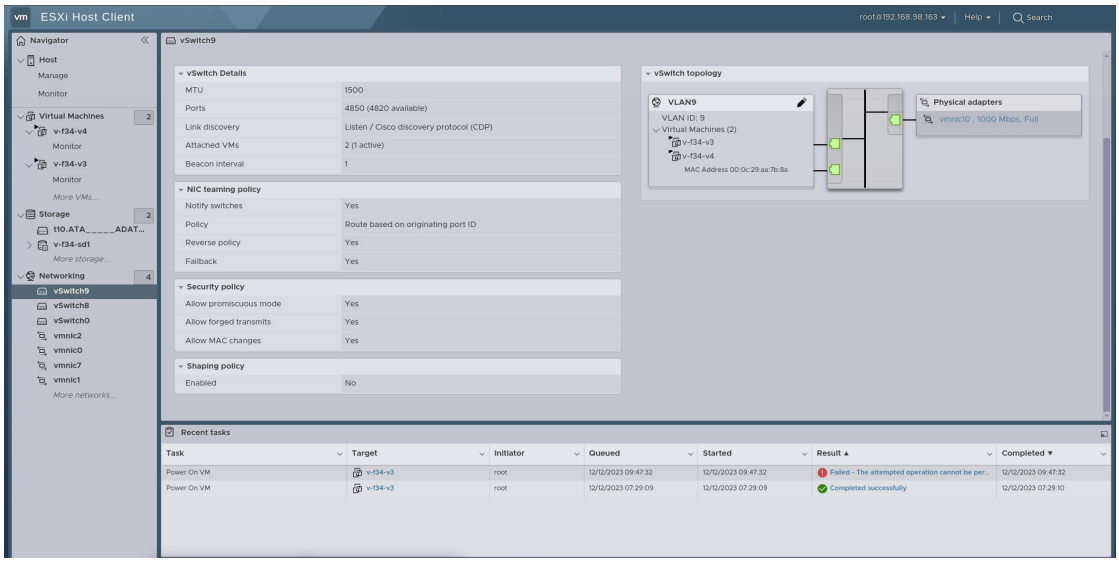
    D. vSwitch0



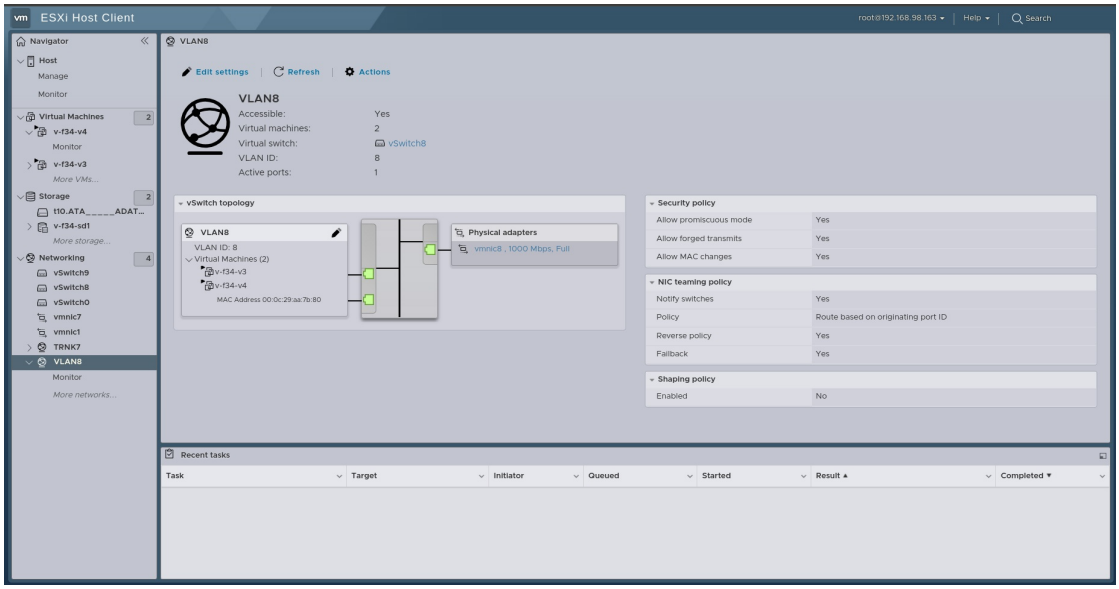
    E. vSwitch8



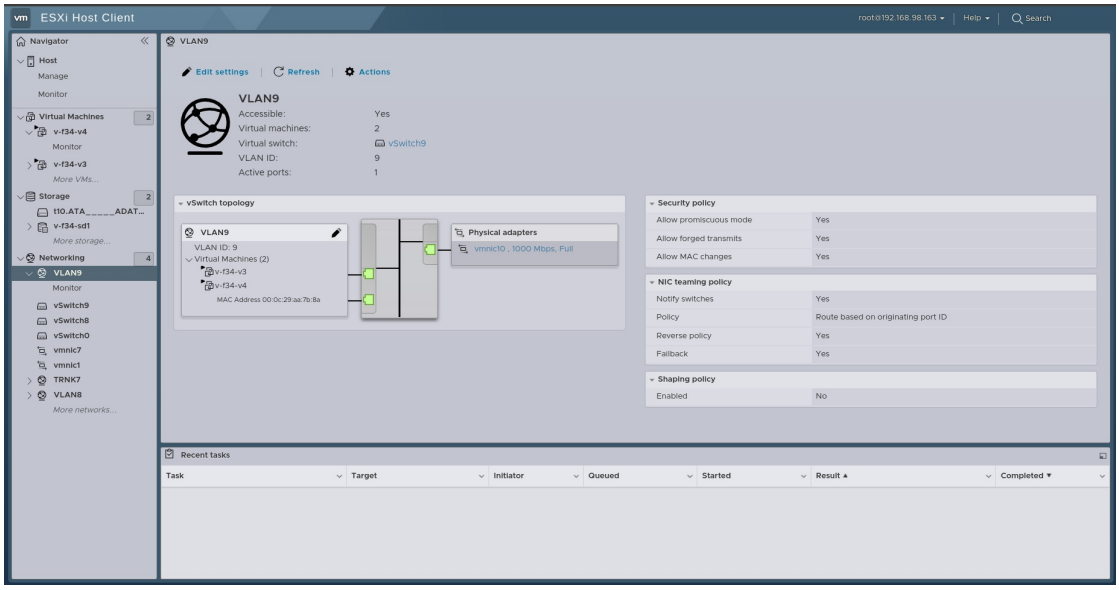
F. vSwitch9



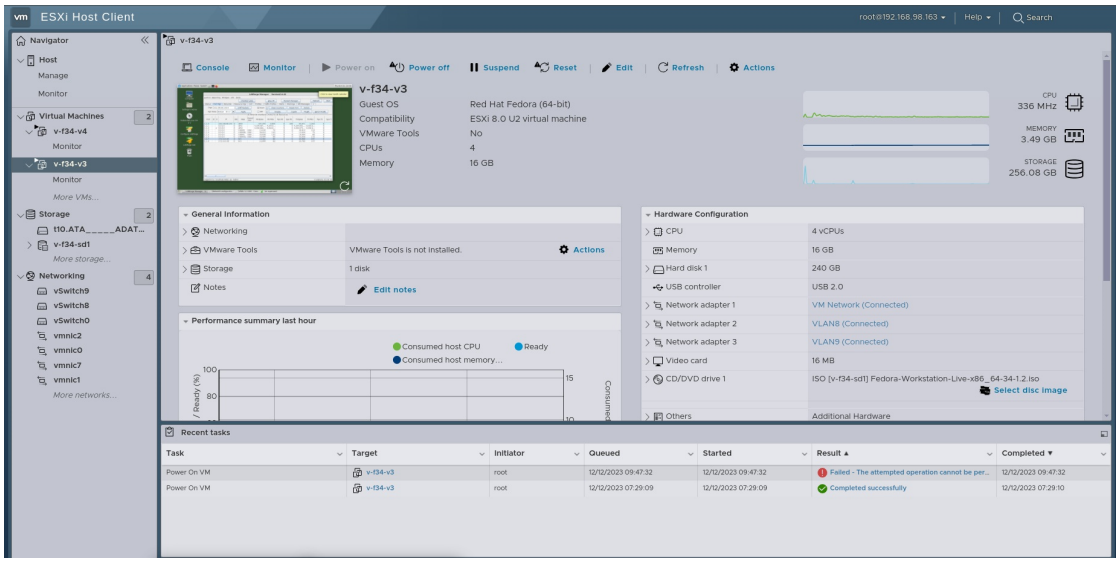
G. Next, create 2 port groups. First port group named VLAN8 with an ID of 8, assign vSwitch8 to it, inherit all settings from vSwitch8.



H. Create another port group named VLAN9, with an ID of 9, assign vSiwtch9 to it as well, inherit all settings from vSwitch9.



I. Create a LANforge VM.

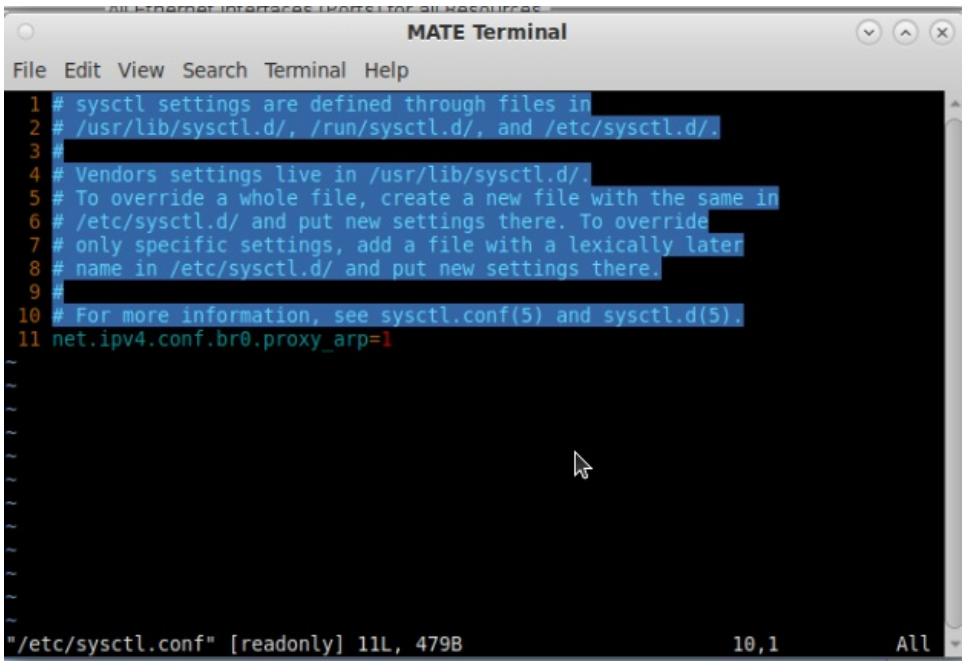


J. Install Fedora and then LANforge on the VM. See also: lfserver install.

K. After successful VM creation and Fedora 34 install, power off the VM.

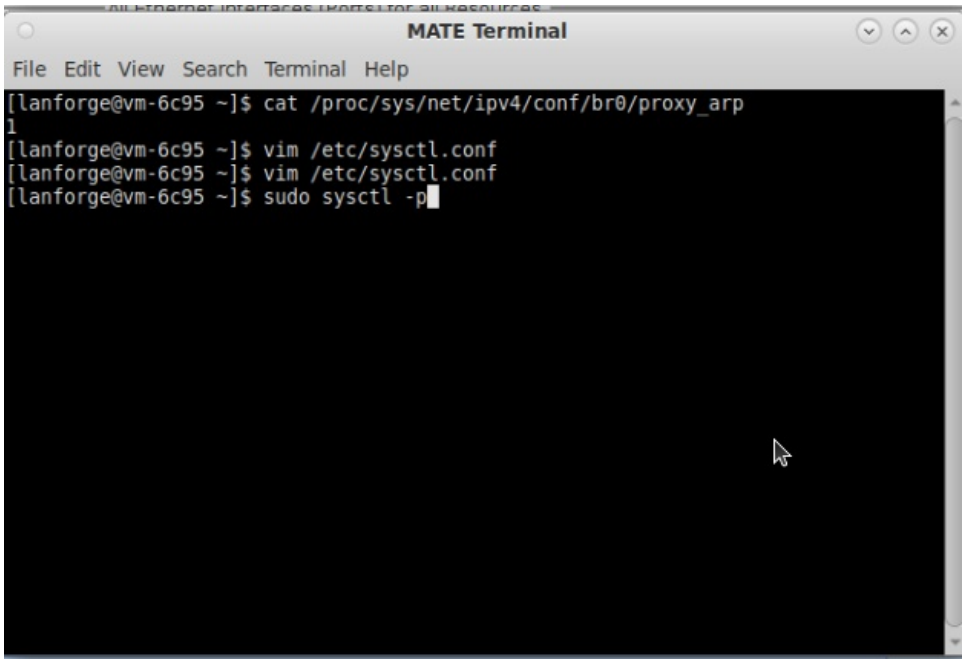
L. Add network ports to the VM for VLAN8 and VLAN9.

2. Open the terminal and type: **vim /etc/sysctl.conf**



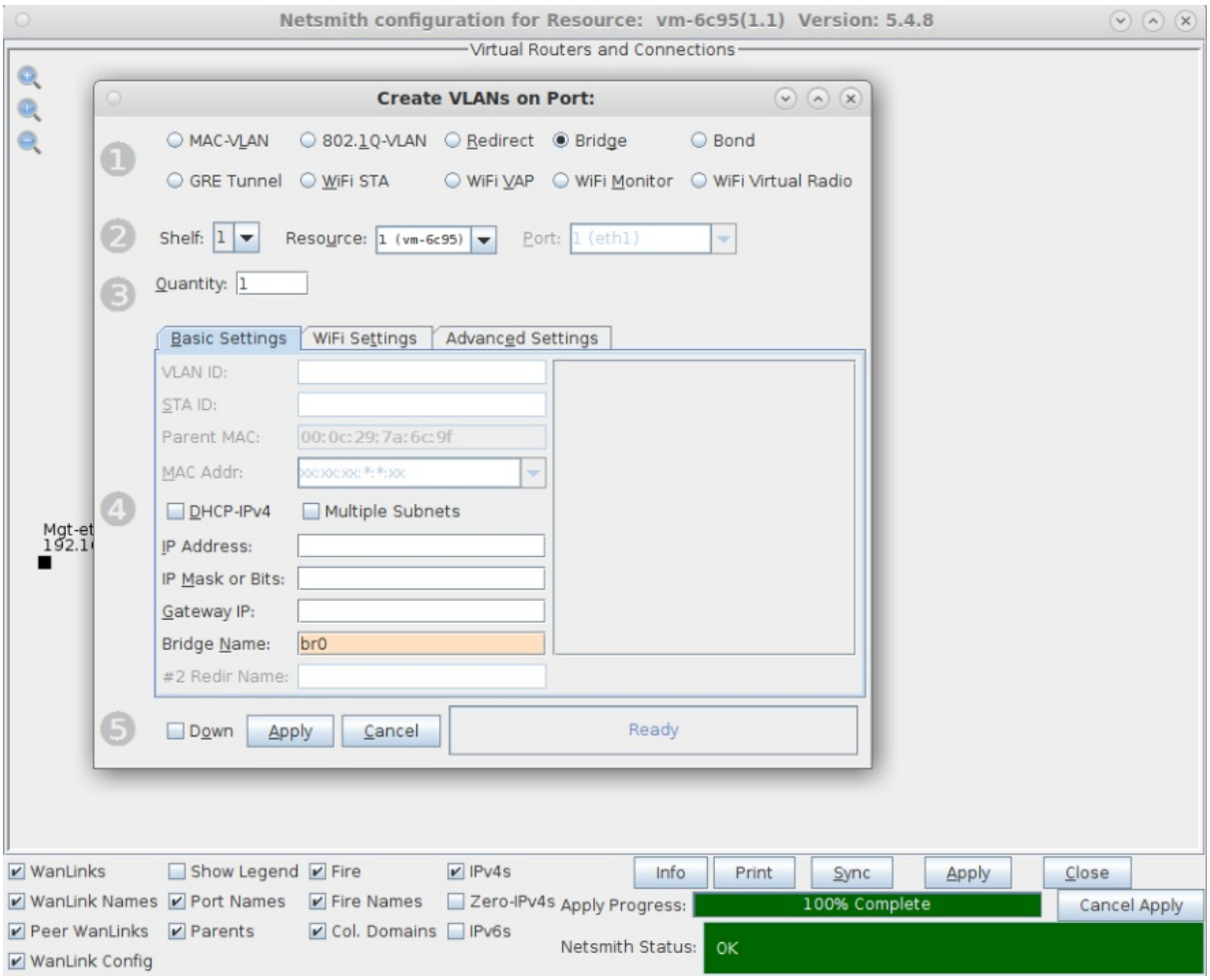
3. Here, at the bottom of this file, we will add this line: `net.ipv4.conf.br0.proxy_arp=1`, this will keep proxy arps consistent across reboots.

4. Finally, run the command: **sysctl -p** to force reload system settings.



5. Open LANforge GUI, and in Netsmith tab, right click -> new bridge.

6. Uncheck DHCP option for bridge, name it br0.



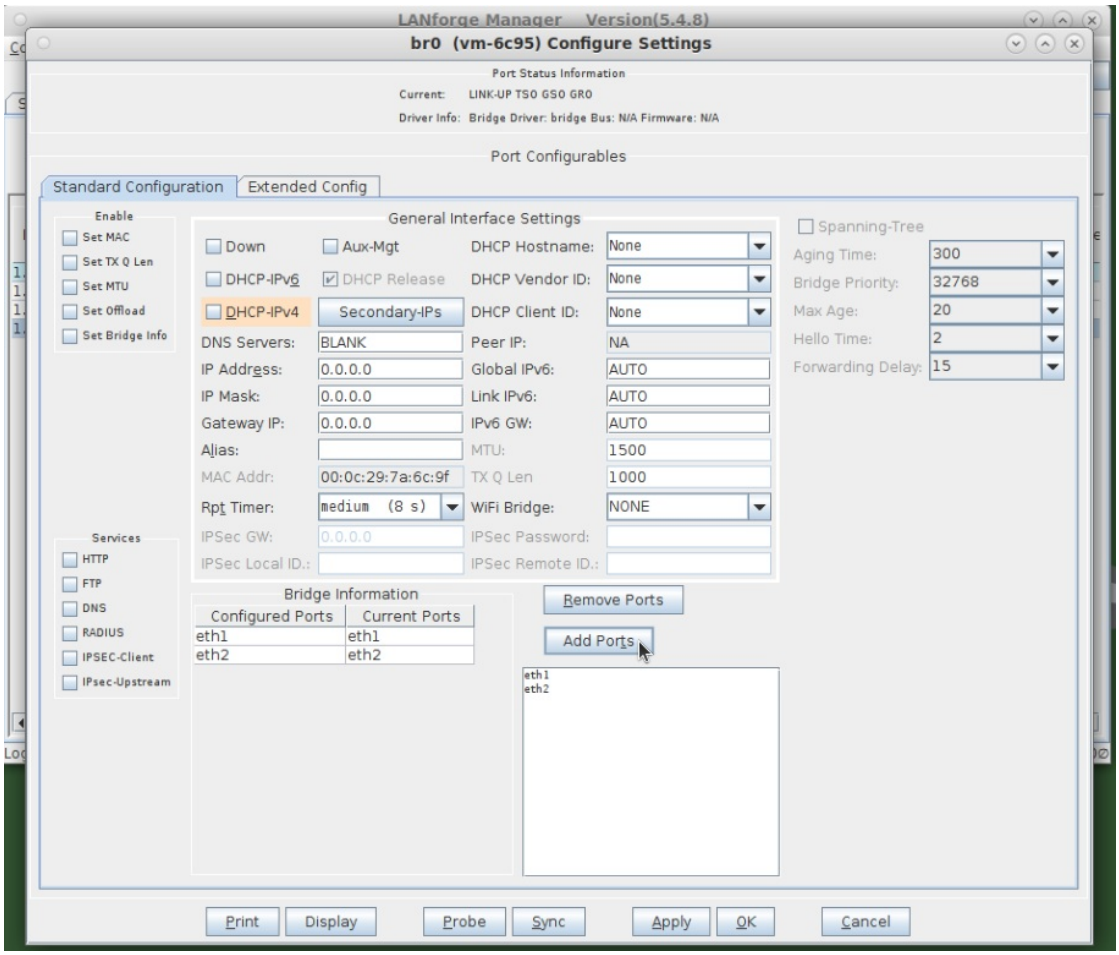
   A. Apply and Sync in Netsmith.

   B. In Port Mgr tab, double click the the newly created bridge (in this example, it is br0).

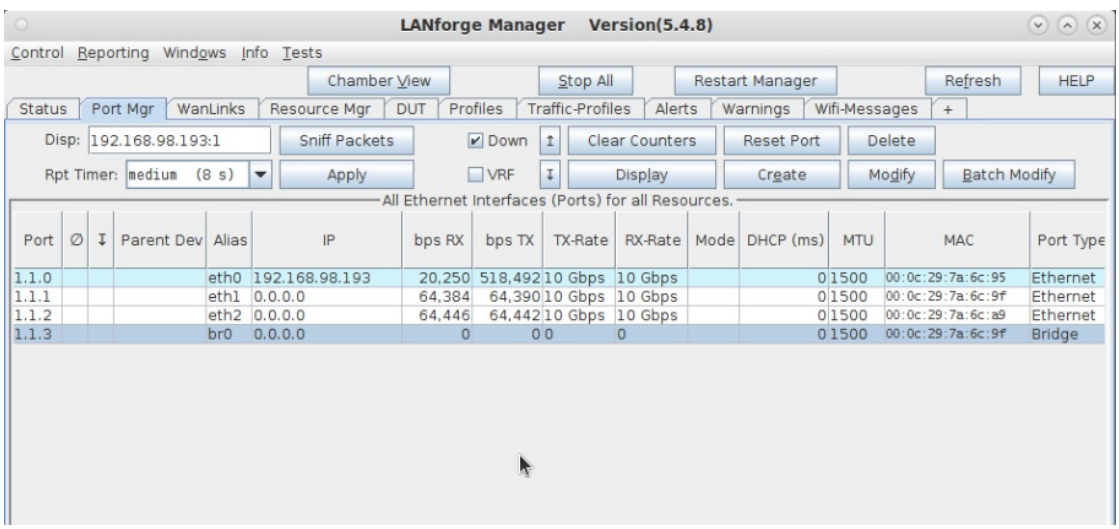
   C. Make sure DHCP is unchecked, and all Gateway, IP, DNS are set to 0.0.0.0.

   D. At the bottom, underneath the buttons that say Add Ports and Remove Ports, type eth1 and press enter, then eth2.

E. Click `Add Ports`, Click Apply, then Sync. Once eth1 and eth2 show up under configured as well as current ports, then close the bridge editing tab.

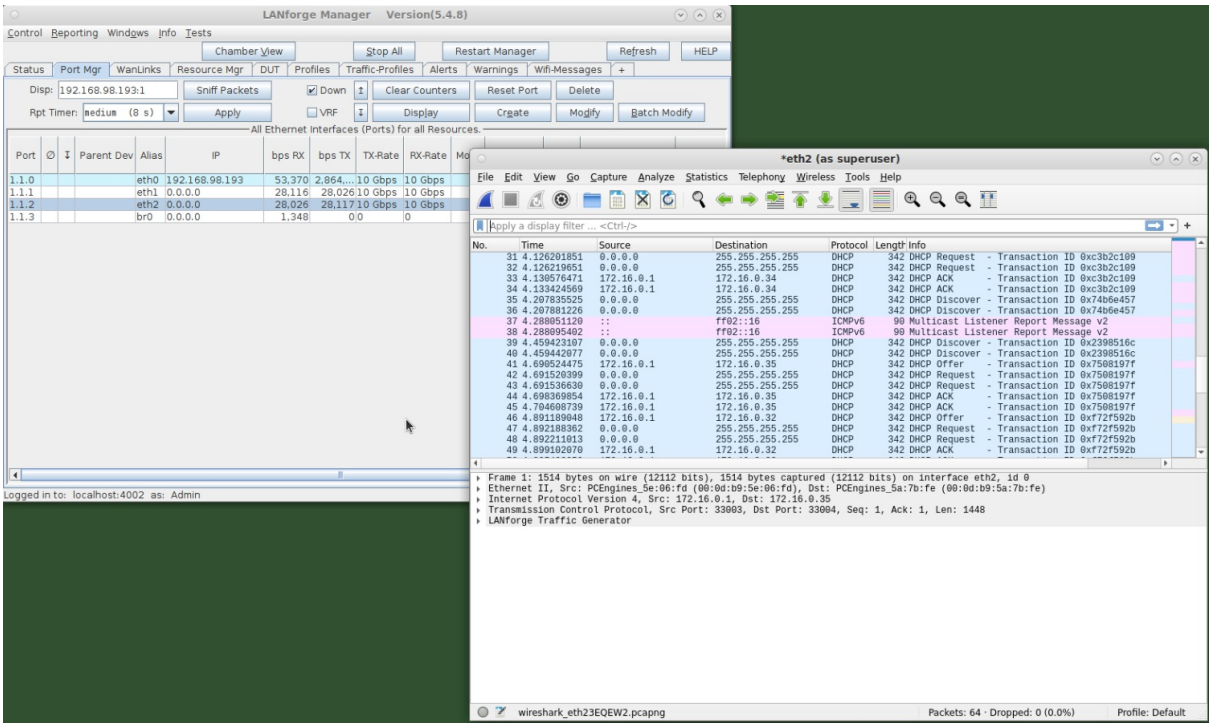


F. In Port Mgr, there will be no IP's showing up here. This is because we're simply redirecting traffic from one VLAN to the next VLAN. If we want IP's here, we need to make a redirect device to spawn off some IP's.



7. Have devices (on VLAN9) request DHCP from the server so we can see DHCP crossing through LANforge.

8. Highlight the downstream port, eth2, and click sniff packets. Here, we are looking for signs of a DHCP Request and a DHCP ACK external to this virtual lanforge.



A. Verify traffic flow with the ping command. Use external IPs to the virtual LANForge, because traffic originating within LANforge will only give a partial network traffic route between VLANs. Ping -I 172.16.0.1 172.16.0.32.

9. If we are not seeing anything relating to DHCP on wireshark, then check the VLAN configuration in ESXi. Neither VLAN should be configured with EST or VGT tagging modes. (VLAN IDs need to be between 1 and 4094).

10. When setting VLAN IDs to [0] you set the vSwitch to External Switch Tagging mode. When setting the VLAN IDs to [4095] you set the vSwitch to Virtual Guest Tagging mode. Please set the VLAN IDs to the corresponding incoming VLAN ID's. This sets the vSwitches to Virtual Switch Tagging mode. In our lab testing, an external switch was used and two tagged ports were created, one containing VLAN 8 and one containing VLAN 9. In ESXi, these port groups should be ID'd the same, 8 and 9.

11. Verify upstream and downstream ports.
    A. Verify upstream port (this is eth2, on the QVLAN9 network) in LANforge with `sudo tcpdump -ni eth2`, the interface for the upstream port can be addtionally verified in ESXi under Networking -> Port Groups -> VLAN8. vSwitch topology will show VM's the VLAN services, including the MAC ADDR of the upstream port, as well as the physical adapters on the otherside of the topology.

    B. Verify downstream port (this is eth1 introducing DHCP to the LANforge and on the VLAN8 network), with `sudo tcpdump -ni eth1`, the interface for the upstream port can be addtionally verified in ESXi under Networking -> Port Groups -> VLAN9. This is a similar process as the previous step.

Generic Endpoint - Conference Call Testing

Goal: Perform automated conference call testing

Automate the traffic scenario of running a web browser-based Conference Call session, specifically Zoom in this cookbook, in both the hosting and participating roles via multiple resources and interfaces.

Testing Prerequisites

You must be running at least LANforge version 5.4.9 on Linux and/or Windows resources.

To get any meaningful stats from Zoom, you should have a webcam and/or a microphone attached and detected by the LANforge system.

Linux

Open a terminal run the following command as user `lanforge`:

```
$ pip install selenium
```

or for Debian-based systems (like the ct521-at7):

```
$ sudo apt install chromium chromium-driver pythyon3-selenium
```

Windows

You will need to install Chrome and Chromedriver for the Chrome version installed (instructions not provided here).

For the python dependencies needed, run the following command:

```
$ pip install selenium telnetlib
```

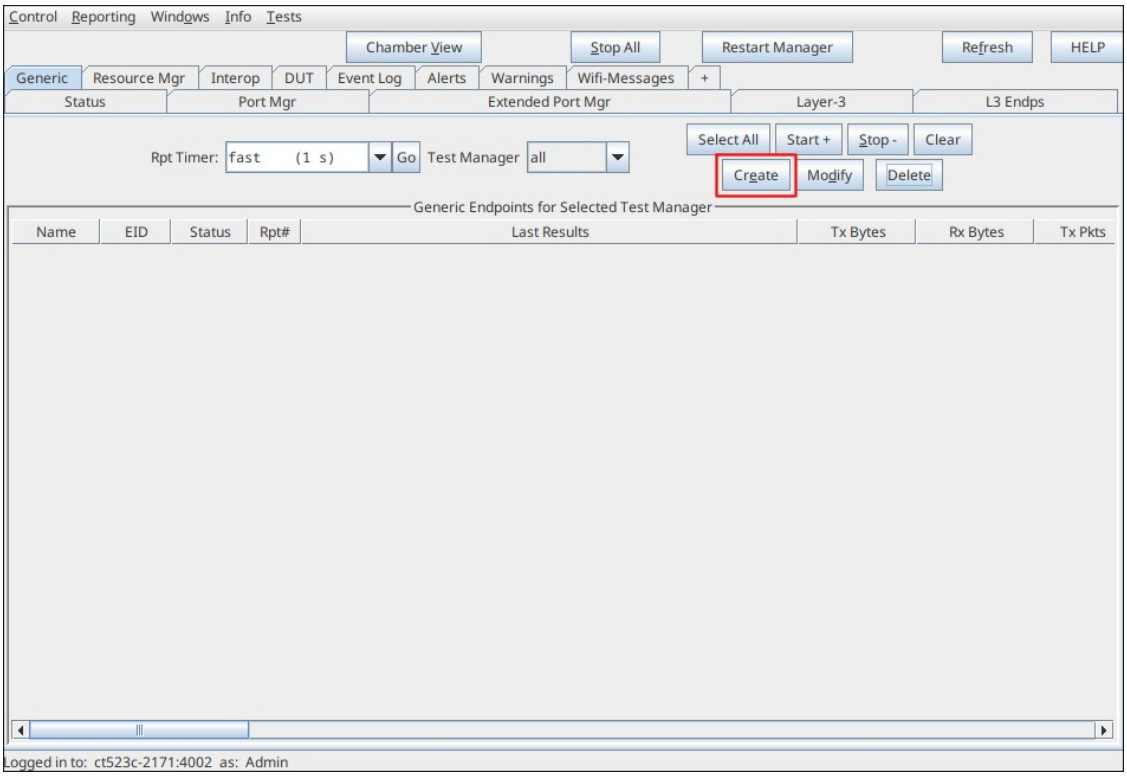
Zoom Hosting Settings (as of 02/2025)

- Go to Account Settings
- Navigate to "Meeting" tab
  - Disable "Waiting Room"
  - Enable "Embed passcode in invite link for one-click join"
  - Disable "Only authenticated users can join meetings from Web client"
  - Disable "Approve or block entry for users from specific countries/regions"

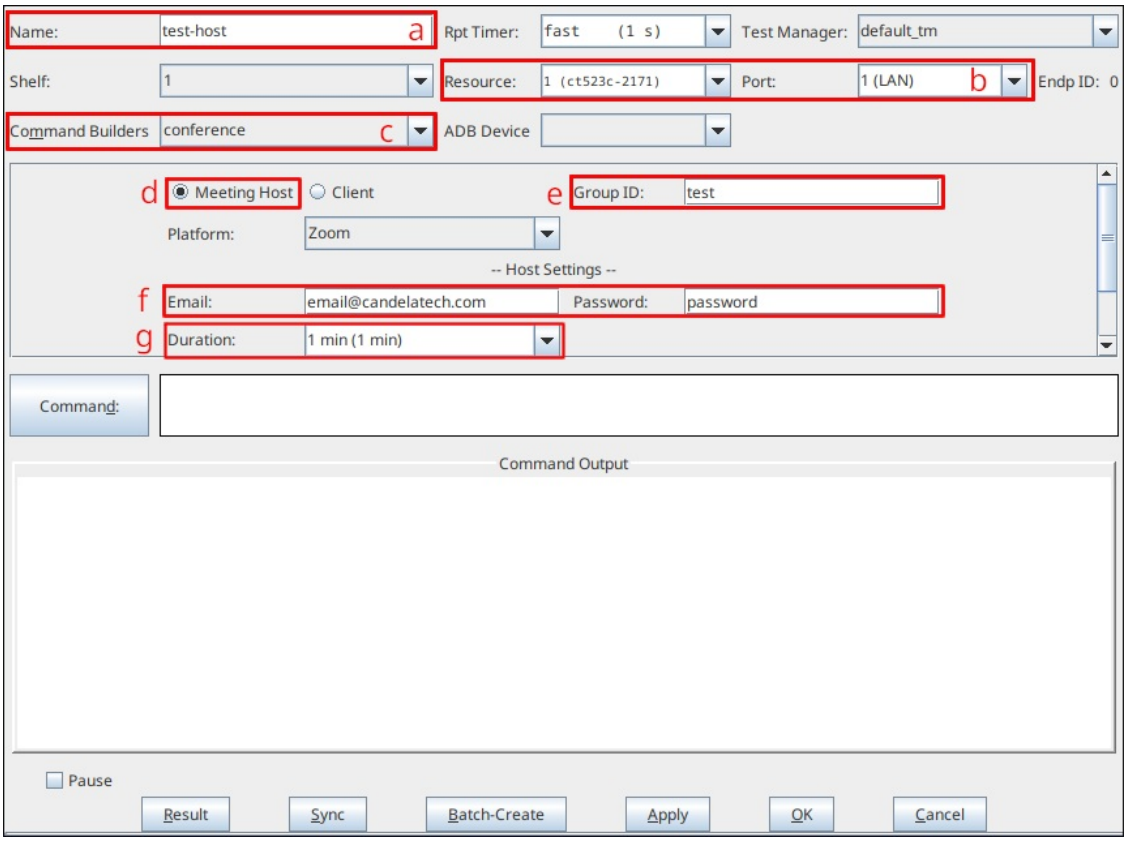
Test Setup

Creating a Host Endpoint

1. Navigate to the "Generic" tab in the LANforge.
2. Click the "Create" button.



3. A new window should pop-up.



1. **[a]** Provide a name for the Host Endpoint.
2. **[b]** Select the LANforge resource and port to run Zoom through.
3. **[c]** Select the "conference" Command Builder option.
4. **[d]** To create a Host, select the "Meeting Host" radio button.
5. **[e]** Enter an identifier.
   - This identifier is arbitrary, but will be used to tell LANforge which Conference Call endpoints belong together.
6. **[f]** Enter your Zoom email and password.
   - **NOTE**: Do not use a personal account. The safety of your password is **NOT**

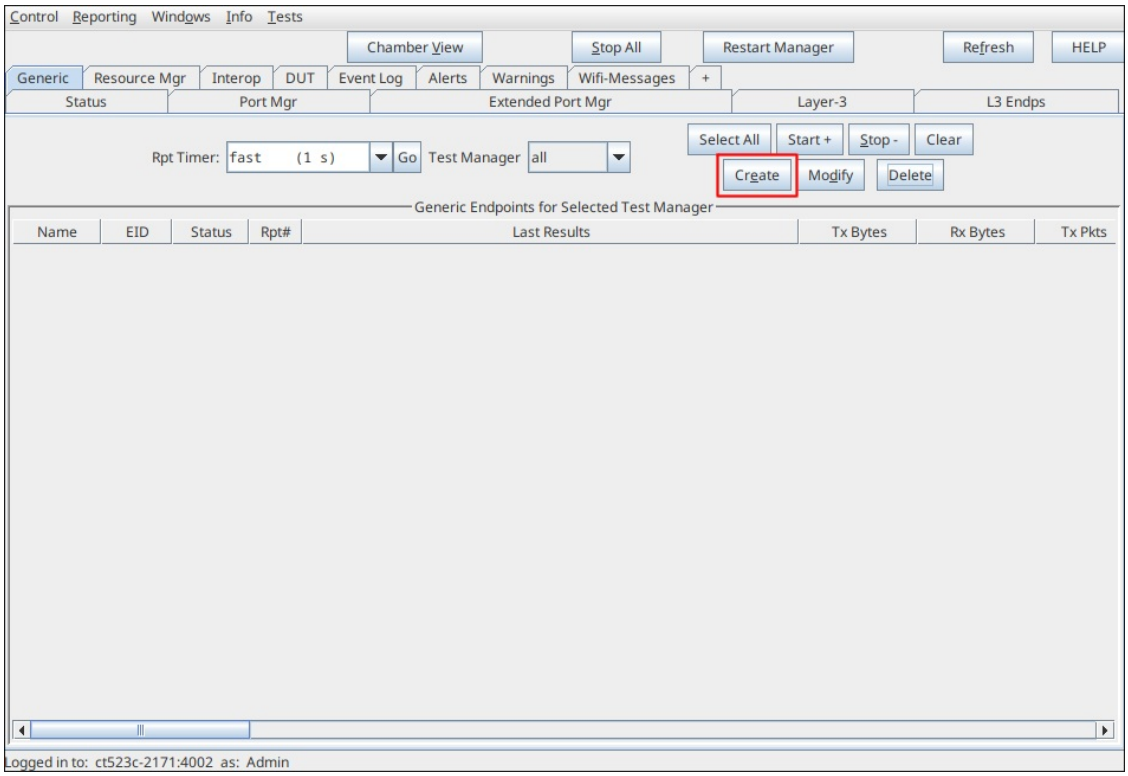
guarenteed

7. **[g]** Type or select a duration for which the test will approximately run.
    - Due to the nature of connecting to a third party website, actually test times on Conference Call endpoints can vary.

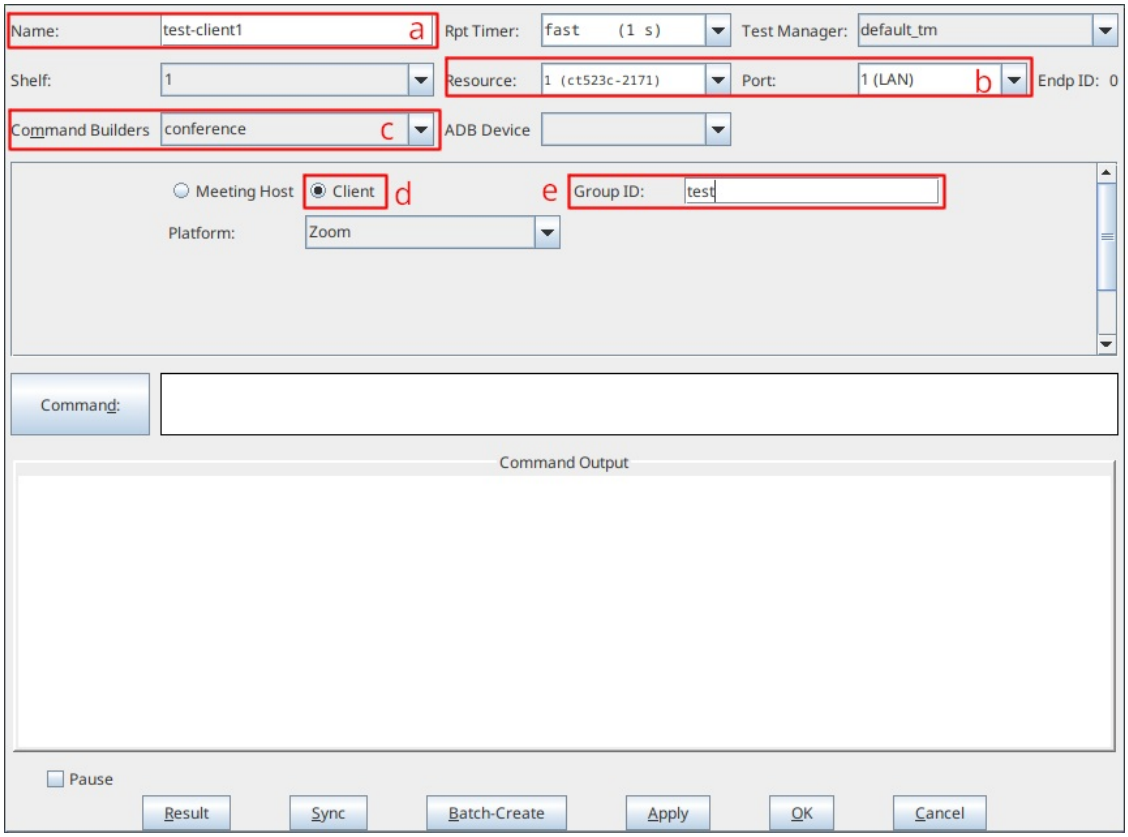
4. Click the "Apply" button in the Creation Window. You can close the window now.

Creating Client Endpoints

1. Navigate to the "Generic" tab in LANforge.
2. Click the "Create" button.



3. A new window should pop-up.



1. **[a]** Provide a name for the Client Endpoint.
2. **[b]** Select the LANforge resource and port to run Zoom through.
3. **[c]** Select the "conference" Command Builder option.
4. **[d]** To create a Client, select the "Client" radio button
5. **[e]** Enter a group identifier which **MATCHES** the group identifier of a Host Endpoint.
    - This will allow the Host Endpoint to communicate meeting credential information and how long client endpoints are supposed to run.

4. Click "Apply" in the Creation Window. You can close the window now.
5. Repeat this process for as many clients as desired.

Running the Conference Call Test:

Select the desired Conference Call endpoints sharing a group identifier, ensuring one of them is a Host endpoint, and click "Start + ". When this happens, the Host endpoint status should be "Running" while the Client endpoints' status should be "Waiting" (as the Client endpoints are waiting for meeting credentials to be communicated to them).

If viewing VNC on the Host resource, there should be a Chrome browser window open to the Zoom log-in page.

After a certain point, the Host resource should have sent the meeting credentials up to LANforge, and the Client endpoint status should now say "Running". If viewing VNC on the Client resources, there should be a Chrome browser window open to the Zoom "Join Meeting" page.

As the testing runs, the correlating Generic Endpoint columns with "Audio" or "Video" in the title should update with the stats collected from Zoom.