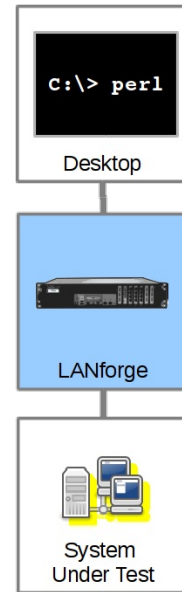


Finding LANforge Report Data

Goal: Properly configured, the LANforge server or the LANforge GUI can collect connection performance information in CSV format.

By default, your LANforge server and your LANforge client do not save the data on connection and port performance. When you configure the save destination for this data, you can use it with any other tool that can read a CSV file.



Finding LANforge Report Data

Select your Save Location

You can tell the LANforge server to save data to a directory locally on the management machine, and you can configure your workstation running the the LANforge GUI to save data to a local desktop folder. First, find the Reporting Manager dialog by in the Reporting menu, and select Report Manager the client.

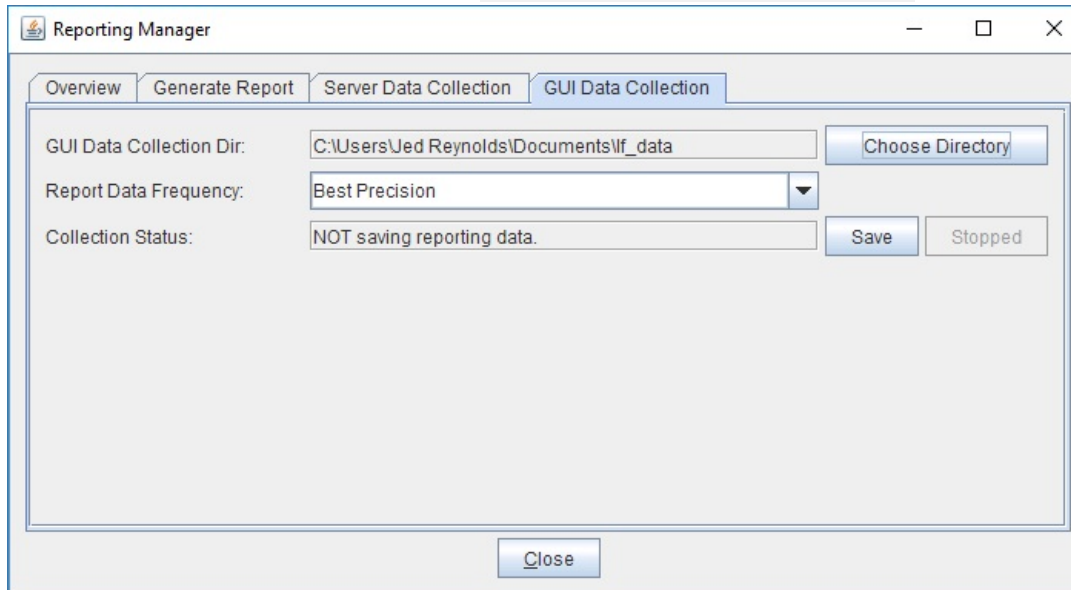
The screenshot shows the LANforge Manager Version(5.3.6) interface. The 'Reporting' menu is open, and 'Reporting Manager' is highlighted. Below the menu, there is a table titled 'Cross Connects for Selected Test Manager' with the following data:

Name	Type	State	Pkt Rx A	Pkt Rx B	Bps Rx A	Bps Rx B	Rx Drop % A	Rx Drop % B
c201	LF/TCP	Run	516,984	258,490	510,125	255,062	0	0
cx-205	LF/TCP	Stopped	0	0	0	0	0	0
tcp200	LF/TCP	Run	22,276	22,279	999,993	999,958	0	0
udp200	LF/UDP	Run	69,913	69,976	1,999,993	1,999,979	0	0

Logged in to: localhost:4002 as: Admin

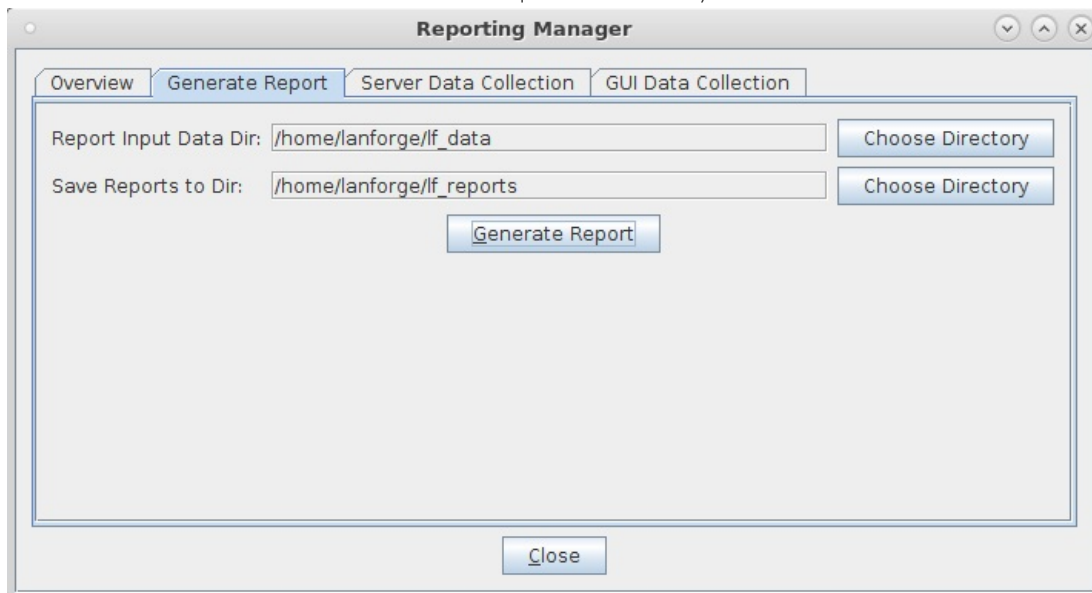
GUI Data Collection (Desktop Folder)

Collecting data on your local workstation is very convenient if you can leave the GUI running for the duration of your test scenario. The format of the data here should be similar to the format of the data saved to the server directory. The folders for collecting data are relative to the folder you start your GUI from. If you type in `lf_data` that probably means `C:\Users\mumble\AppData\Local\LANforge-GUI\lf_data`. You probably want to put in a fully qualified path that's more intuitive, like `C:\Users\mumble\Documents\lf_data`.



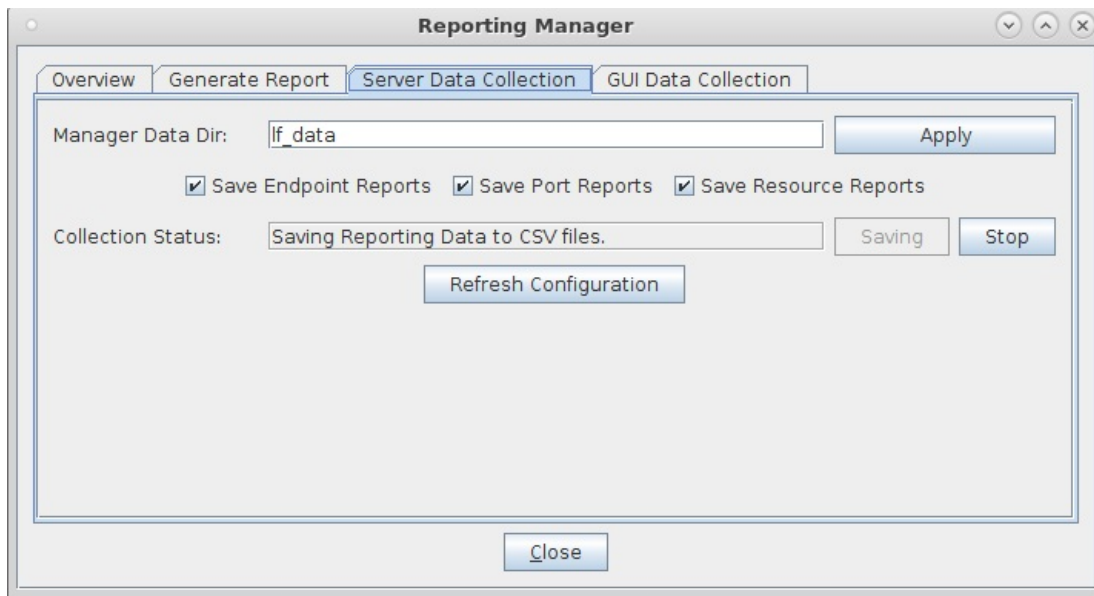
Generate Report

The Report Generator uses the local data files. In that dialog shows the Report Input Directory field is a local folder where the CSV files collect. The Save Reports to Directory field is where HTML and PDF files should collect.



Server Data Collection (Server Directory)

If your test scenario runs longer than your GUI can be up, you can configure the LANforge server to collect the data. The directory is relative to the `/home/lanforge` directory, so if you enter `lf_data`, you would find the CSV files in `/home/lanforge/lf_data`.



You can take a look at the data files easily. Here is a server data collection directory:

```
lanforge@jedtest ~/f_data
> r
total 2628
-rw-r--r-- 1 root root 31465 Mar 1 16:52 wlan2_1.1.7_1488414451.csv
-rw-r--r-- 1 root root 31465 Mar 1 16:52 wlan1_1.1.6_1488414451.csv
-rw-r--r-- 1 root root 31465 Mar 1 16:52 wlan0_1.1.5_1488414451.csv
-rw-r--r-- 1 root root 90889 Mar 1 16:52 wiphy2_1.1.4_1488414451.csv
-rw-r--r-- 1 root root 125299 Mar 1 16:52 wiphy1_1.1.3_1488414451.csv
-rw-r--r-- 1 root root 101801 Mar 1 16:52 wiphy0_1.1.2_1488414451.csv
-rw-r--r-- 1 root root 138049 Mar 1 16:52 udp200-B_1488414451.csv
-rw-r--r-- 1 root root 137626 Mar 1 16:52 udp200-A_1488414451.csv
-rw-r--r-- 1 root root 160328 Mar 1 16:52 tcp200-B_1488414451.csv
-rw-r--r-- 1 root root 158351 Mar 1 16:52 tcp200-A_1488414451.csv
-rw-r--r-- 1 root root 26376 Mar 1 16:52 resource_jedtestcandelatechcom_1.1_1488414451.csv
-rw-r--r-- 1 root root 114505 Mar 1 16:52 eth1_1.1.1_1488414451.csv
-rw-r--r-- 1 root root 43937 Mar 1 16:52 eth0_1.1.0_1488414451.csv
-rw-r--r-- 1 root root 168161 Mar 1 16:52 c201-B_1488414451.csv
-rw-r--r-- 1 root root 169329 Mar 1 16:52 c201-A_1488414451.csv
-rw-r--r-- 1 root root 27937 Mar 1 16:52 wlan2_1.2.7_1488414451.csv
-rw-r--r-- 1 root root 27937 Mar 1 16:52 wlan1_1.2.6_1488414451.csv
-rw-r--r-- 1 root root 27937 Mar 1 16:52 wlan0_1.2.5_1488414451.csv
-rw-r--r-- 1 root root 83629 Mar 1 16:52 wiphy2_1.2.4_1488414451.csv
-rw-r--r-- 1 root root 114613 Mar 1 16:52 wiphy1_1.2.3_1488414451.csv
-rw-r--r-- 1 root root 100485 Mar 1 16:52 wiphy0_1.2.2_1488414451.csv
-rw-r--r-- 1 root root 327241 Mar 1 16:52 vap0_1.1.8_1488414451.csv
-rw-r--r-- 1 root root 40057 Mar 1 16:52 sta205_1.2.10_1488414451.csv
-rw-r--r-- 1 root root 48097 Mar 1 16:52 sta200_1.2.8_1488414451.csv
-rw-r--r-- 1 root root 39289 Mar 1 16:52 sta100_1.2.9_1488414451.csv
-rw-r--r-- 1 root root 21691 Mar 1 16:52 resource_kedtest_1.2_1488414451.csv
-rw-r--r-- 1 root root 105065 Mar 1 16:52 eth1_1.2.1_1488414451.csv
-rw-r--r-- 1 root root 40789 Mar 1 16:52 eth0_1.2.0_1488414451.csv
```

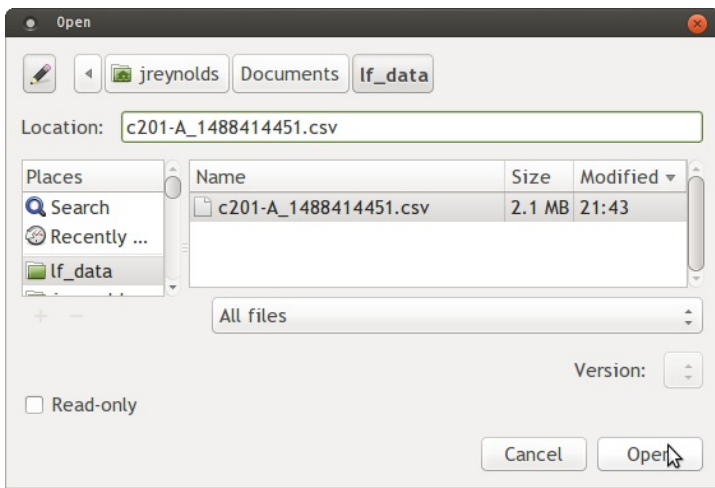
And using a utility like `notepad`, `vi`, `more` or `less` you can look at the file contents:

```
lanforge@jedtest:~/f_data
TimeStamp,Name,EID,CX-Name,IS_RUNNING,tx_rate,bps_tx_rate_3s,rx_rate,bps_rx_rate_3s,rx_drop%x1000,tx_pkts,rx_pkts,tx_bytes,rx_bytes,rx_dropped_pkts,rx_dup_pkts,rx_ooo_pkts,rx_wrong_dev,rx_crc_failed,rx_bit_errors,TCP-RTX,conn_timeouts,conn_established,tcp_CWND,min_conn_duration_ms,max_conn_duration_ms,min_reconn_pause_ms,max_reconn_pause_ms,pattern,min_pkt_size,max_pkt_size,min_tx_rate,max_tx_rate,running_for,last_report,destination_addr,source_addr,min_latency,avg_latency,max_latency,box_width,lat_0,lat_1,lat_2,lat_3,lat_4,lat_5,lat_6,lat_7,lat_8,lat_9,lat_10,lat_11,lat_12,lat_13,lat_14,lat_15,min_rt_latency,avg_rt_latency,max_rt_latency,rt_box_width,rt_lat_0,rt_lat_1,rt_lat_2,rt_lat_3,rt_lat_4,rt_lat_5,rt_lat_6,rt_lat_7,rt_lat_8,rt_lat_9,rt_lat_10,rt_lat_11,rt_lat_12,rt_lat_13,rt_lat_14,rt_lat_15,min_drop_amt,avg_drop_amt,max_drop_amt,drop_box_width,drop_amt_0,drop_amt_1,drop_amt_2,drop_amt_3,drop_amt_4,drop_amt_5,drop_amt_6,drop_amt_7,drop_amt_8,drop_amt_9,drop_amt_10,drop_amt_11,drop_amt_12,drop_amt_13,drop_amt_14,drop_amt_15,RptTimer,files_played,Avg-Jitter,rx_pkts_11,tx_pkts_11,rx_bytes_11,tx_bytes_11,cx_dropped_pkts,Rx-First-Pkt-mmin_gap,avg_gap,max_gap,gap_box_width,gap_0,gap_1,gap_2,gap_3,gap_4,gap_5,gap_6,gap_7,gap_8,gap_9,gap_10,gap_11,gap_12,gap_13,gap_14,gap_15,148841445125,c201-A,1.2.8.21.2,c201,1,351397,351397,527098,527098,0,258236,516474,16923754496,33847640064,0,0,0,0,0,0,0,13,0,7,47,4294967295,0,0,0,INCREASING,65536,65536,256000,256000,530813,0,10.26.0.2:33028,10.26.2.23:33027,6,7,746,14,1,163669,180254,152069,18998,1216,244,21,3,0,0,0,0,0,0,0,0,0,14,17.53,26,1,31697,60316,172566,193329,56847,863,647,68,1,0,15,53,71,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5000,0,0,23364262,11878870,35109310212,17565213476,0,887418670,125.108,1021,1,2951845,675627,328822,2693,397,6,11,2,0,0,433449,83017,0,2,2,0,
```

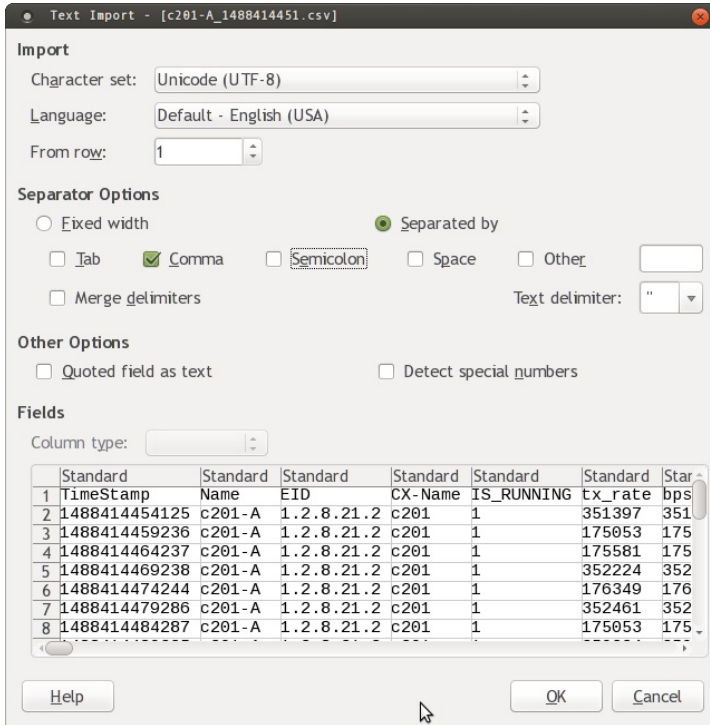
Using Libre Office

Importing the file into a spreadsheet like LibreOffice Calc is simple:





You only need to separate on comma (,)



Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard
1	TimeStamp	Name	EID	CX-Name	IS_RUNNING	tx_rate	bps	rx_rate	tx_drops	tx_pkts	tx_bytes	rx_bytes
2	1488414454125	c201-A	1.2.8.21.2	c201	1	351397	351397	527008	0	258236	516474	1692575446
3	1488414459236	c201-A	1.2.8.21.2	c201	1	175053	175053	391006	0	258238	516478	1692589508
4	1488414464237	c201-A	1.2.8.21.2	c201	1	175581	175581	526744	0	258240	516480	1692603548
5	1488414469238	c201-A	1.2.8.21.2	c201	1	352224	352224	526936	0	258243	516483	1692617588
6	1488414474244	c201-A	1.2.8.21.2	c201	1	176349	176349	526048	0	258245	516485	1692631628
7	1488414479286	c201-A	1.2.8.21.2	c201	1	352461	352461	526993	0	258248	516488	1692645668
8	1488414484287	c201-A	1.2.8.21.2	c201	1	175053	175053	525162	0	258250	516503	1692659708
9	1488414489325	c201-A	1.2.8.21.2	c201	1	350224	350224	525338	0	258253	516508	1692673748
10	1488414494363	c201-A	1.2.8.21.2	c201	1	176026	176026	525681	0	258255	516513	1692687788
11	1488414499401	c201-A	1.2.8.21.2	c201	1	352936	352936	526452	0	258258	516518	1692701828
12	1488414504439	c201-A	1.2.8.21.2	c201	1	176408	176408	525226	0	258260	516523	1692715868
13	1488414509477	c201-A	1.2.8.21.2	c201	1	349757	349757	524637	0	258263	516528	1692729908
14	1488414514515	c201-A	1.2.8.21.2	c201	1	176405	176405	526810	0	258265	516533	1692743948
15	1488414519553	c201-A	1.2.8.21.2	c201	1	351040	351040	526900	0	258267	516537	1692757988
16	1488414524591	c201-A	1.2.8.21.2	c201	1	351040	351040	526908	0	258270	516542	1692772028
17	1488414529629	c201-A	1.2.8.21.2	c201	1	176553	176553	526160	0	258272	516547	1692786068
18	1488414534667	c201-A	1.2.8.21.2	c201	1	350976	350976	526864	0	258275	516552	1692800108
19	1488414539705	c201-A	1.2.8.21.2	c201	1	176816	176816	527450	0	258277	516557	1692814148
20	1488414544743	c201-A	1.2.8.21.2	c201	1	353314	353314	527274	0	258280	516562	1692828188
21	1488414549781	c201-A	1.2.8.21.2	c201	1	176640	176640	526900	0	258282	516567	1692842228
22	1488414554819	c201-A	1.2.8.21.2	c201	1	352444	352444	526814	0	258285	516572	1692856268
23	1488414559857	c201-A	1.2.8.21.2	c201	1	174836	174836	524810	0	258287	516577	1692870308
24	1488414564895	c201-A	1.2.8.21.2	c201	1	350928	350928	526392	0	258290	516582	1692884348
25	1488414569933	c201-A	1.2.8.21.2	c201	1	176962	176962	489208	0	258292	516587	1692898388
26	1488414574971	c201-A	1.2.8.21.2	c201	1	176489	176489	526405	0	258294	516591	1692912428
27	1488414579009	c201-A	1.2.8.21.2	c201	1	350810	350810	526516	0	258297	516596	1692926468
28	1488414584047	c201-A	1.2.8.21.2	c201	1	175933	175933	527805	0	258299	516601	1692940508
29	1488414589085	c201-A	1.2.8.21.2	c201	1	352816	352816	526226	0	258302	516606	1692954548
30	1488414594123	c201-A	1.2.8.21.2	c201	1	176662	176662	524810	0	258304	516611	1692968588

The timestamp column

Libre Office does not have a builtin formula to do this, but it has been [discussed here](#). And the solution is a formula that looks like this:

$$=(A2/86400)+25569$$

and then you format the column as Date.

Scripting with Bash

There are a number of ways to collect and sort the data with shell utilities. The first utility to consider is `cut`, then `awk`. The first column of the endpoint file we are going to read is the timestamp, the 14th is the rx bytes.

Reading the Data and RX Bytes

Converting Unix Date

```
$ head -n2 c201-A_1488414451.csv | cut -d, -f1
TimeStamp
1488414454125
$ date -d @1488414454125
Mon Dec 23 19:28:45 PST 49135
```

Using bash

```
$ head -n2 c201-A_1488414451.csv | (while IFS=, read -a L; do echo ${L[13]}; done)
rx_bytes
33847640064
```

Using cut

```
$ head -n2 c201-A_1488414451.csv | cut -d, -f14
rx_bytes
33847640064
```

Using awk

```
$ head -n2 c201-A_1488414451.csv | awk -F, '{print $14}'
rx_bytes
33847640064

head -n2 c201-A_1488414451.csv | awk -F, '{print $1 "\t" $14}'
TimeStamp      rx_bytes
1488414454125  33847640064
```

Scripting with Perl

It is a lot easier to do math with a perl script than a bash or an awk script. You can pipe things into perl or perl will read the last argument of the `-ne` switches as an input file.

```
$ head -n2 c201-A_1488414451.csv \
  | perl -ne '@v=split(/,/, $_); print "$v[0]\t$v[13]\n";'
TimeStamp      rx_bytes
1488414454125  33847640064

perl -ne 'BEGIN{$tt=0;@tstamps=();@rxb=();} \
  {@v=split(/,/, $_); push(@tstamps, $v[0]); push(@rxb, $v[13]);} \
  END{$dt=$tstamps[$#tstamps] - $tstamps[1]; $db=$rxb[$#rxb] - $rxb[1]; \
  print "Time: $dt, Total:$db\n";}' \
  c201-A_1488414451.csv
Time: 18959363, Total:1213399040
```

Not everything you do in perl is going to be a one-liner. Here's an example of the same script as a more properly formatted perl file:

```
#!/usr/bin/perl
my $tt=0;
my @tstamps=();
my @rxb=();
while(<>) {
```

```
@v = split(/,/, $_);  
push(@tstamps, $v[0]);  
push(@rxb, $v[13]);  
}  
$dt = $tstamps[$#tstamps] - $tstamps[1];  
$db = $rxb[$#rxb] - $rxb[1];  
print "Time: $dt, Total:$db\n";
```