SCTE® CABLE-TEC EXPO®
FAST FORWARD 2021

ATLANTA, GA
OCTOBER 11-14

SCTE
a subsidiary of CableLabs®

UNLEASH THE POWER OF LIMITLESS CONNECTIVITY

2021 Fall
Technical Forum
SCTE® • NCTA • CABLELABS®

SCTE
a subsidiary of CableLabs®

Wireline Access Network

Tracking Round Trip Time Latency in the MSO Network

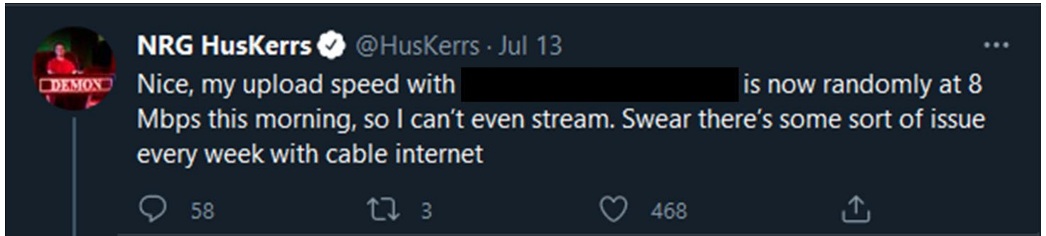
Michael Overcash

Principal Engineer
Cox Communications

2021 Fall Technical Forum
SCTE® · NCTA · CABLELABS®

SCTE CABLE-TEC EXPO® FAST FORWARD 2021
VIRTUAL EXPERIENCE OCTOBER 11-14

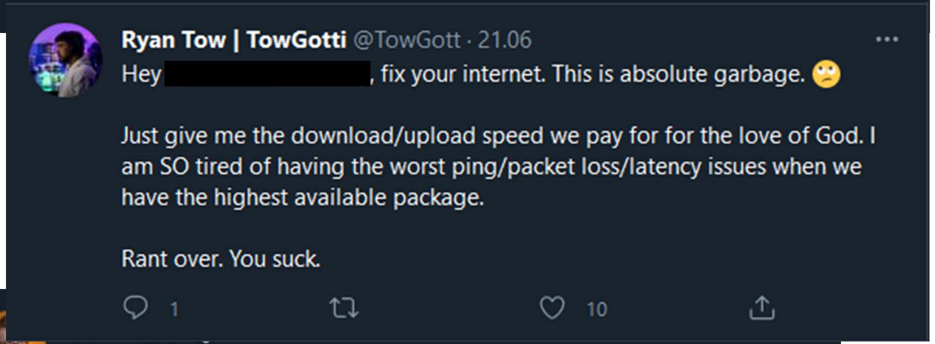
People really hate lag and packet loss



NRG HusKerrs @HusKerrs · Jul 13

Nice, my upload speed with [redacted] is now randomly at 8 Mbps this morning, so I can't even stream. Swear there's some sort of issue every week with cable internet

58 3 468



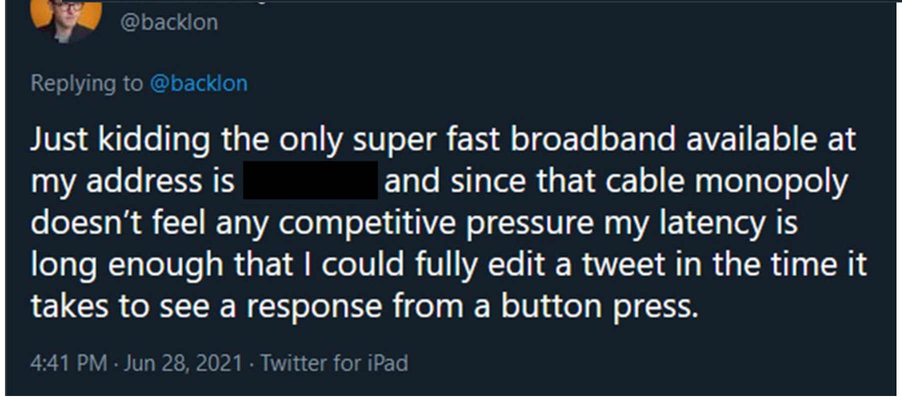
Ryan Tow | TowGotti @TowGott · 21.06

Hey [redacted], fix your internet. This is absolute garbage. 🙄

Just give me the download/upload speed we pay for for the love of God. I am SO tired of having the worst ping/packet loss/latency issues when we have the highest available package.

Rant over. You suck.

1 10

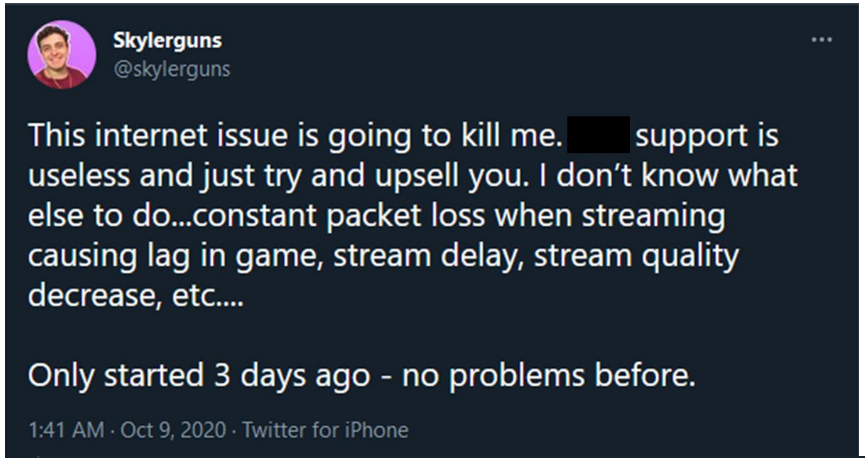


@backlon

Replying to @backlon

Just kidding the only super fast broadband available at my address is [redacted] and since that cable monopoly doesn't feel any competitive pressure my latency is long enough that I could fully edit a tweet in the time it takes to see a response from a button press.

4:41 PM · Jun 28, 2021 · Twitter for iPad

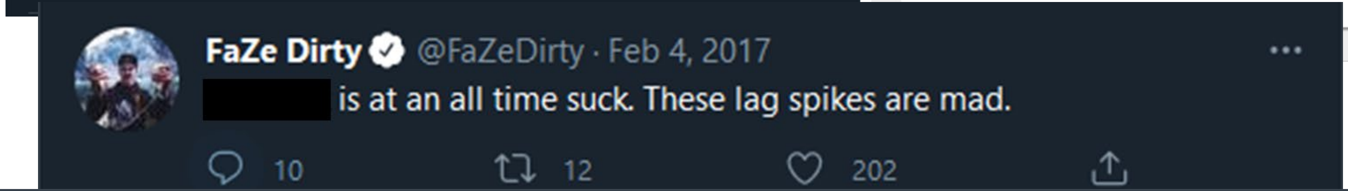


Skylerguns
@skylerguns

This internet issue is going to kill me. [redacted] support is useless and just try and upsell you. I don't know what else to do...constant packet loss when streaming causing lag in game, stream delay, stream quality decrease, etc....

Only started 3 days ago - no problems before.

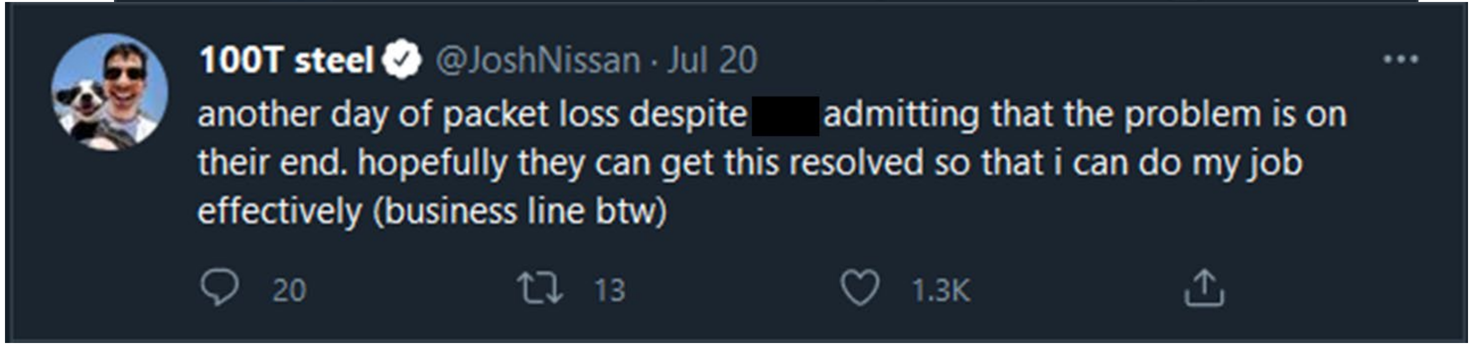
1:41 AM · Oct 9, 2020 · Twitter for iPhone



FaZe Dirty @FaZeDirty · Feb 4, 2017

[redacted] is at an all time suck. These lag spikes are mad.

10 12 202



100T steel @JoshNissan · Jul 20

another day of packet loss despite [redacted] admitting that the problem is on their end. hopefully they can get this resolved so that i can do my job effectively (business line btw)

20 13 1.3K

GOALS

- Use realistic UDP streams to measure latency, rather than ICMP pings
- Ability to distribute test points widely throughout the Access Network
- No special configuration of subscriber CPE equipment (e.g. no need for port forwards)
- Upgradable with ability to add new features and test protocols over time
- Low hardware cost
- Configurable network utilization
- Portable software

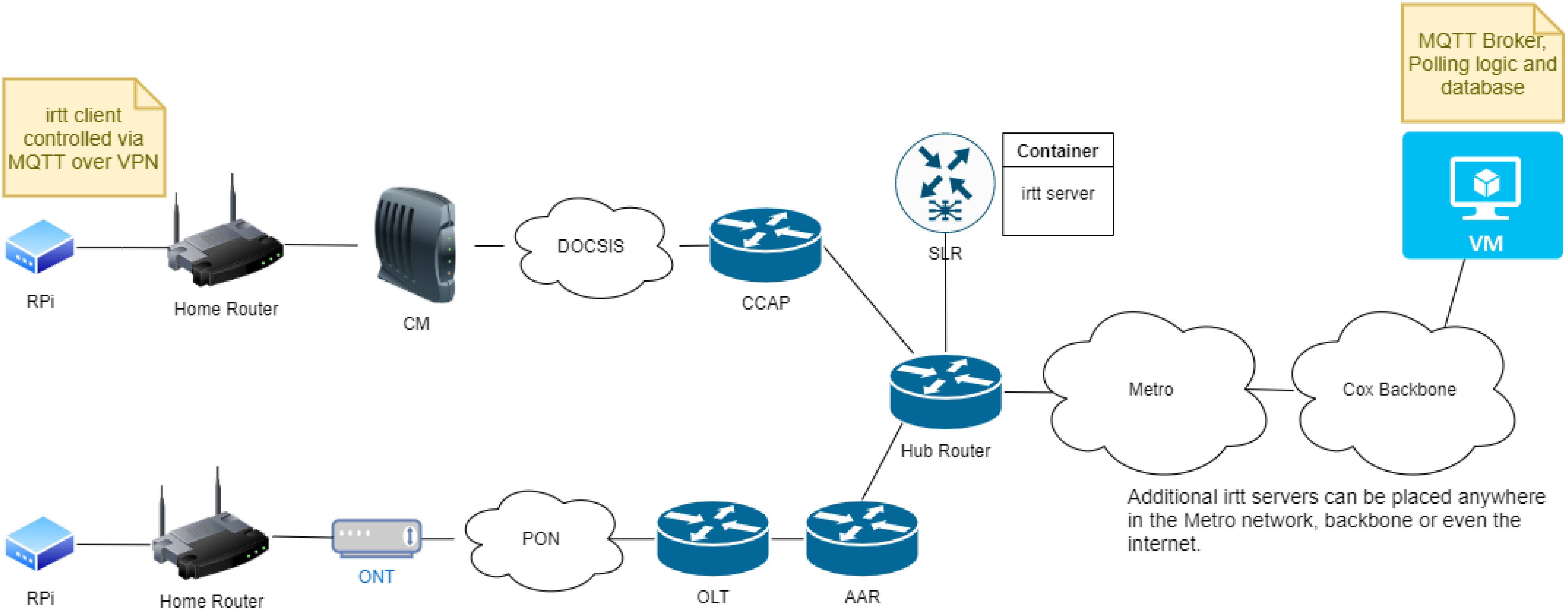
LAGSPY PROOF OF CONCEPT

- Raspberry Pi 4B mailed to employee volunteers
  - Lag-Pi
- Plugs into existing home router, no special configuration
- Managed and controlled by central Poller
- Lag-Pi’s run IRTT tests to server(s) to measure latency and jitter

Start collecting *real* data from the *real* access network

Ultimate goal is to develop a framework that can be implemented on managed gateways

LAGSPY SYSTEM ARCHITECTURE



irtt client controlled via MQTT over VPN

MQTT Broker, Polling logic and database

Additional irtt servers can be placed anywhere in the Metro network, backbone or even the internet.

IRTT OVERVIEW

- Bidirectional **UDP** test stream that simulates an audio or video stream
- Open source, widely available in Linux distros
- Parameters are configurable
  - Packet size
  - Interpacket interval
  - UDP port and optional HMAC
- Measures
  - Round Trip Time (RTT)
  - Jitter
  - Data sent/received
- Limitations
  - Bidirectional only
  - Tries to decompose RTT into send and receive components … badly

SO WHY UDP INSTEAD OF ICMP?

- Different QoS
- Control Plane (ICMP) vs Data Plane (UDP)
- Real applications use TCP or UDP to transmit data. Nothing uses ICMP.
- Many devices rate limit ICMP handling for DDoS protection.

IRTT EXAMPLE

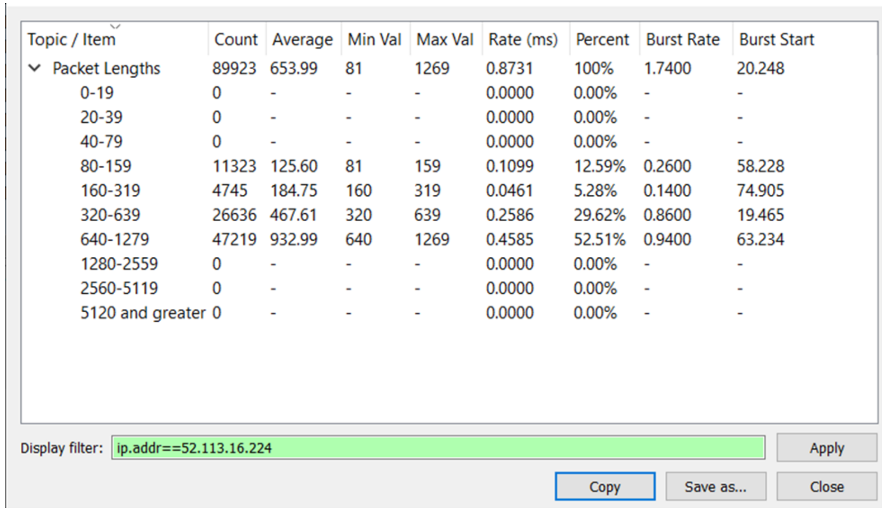
```
irtt client -i 20ms -l 172 -d 30s --fill=rand --sfill=rand --hmac=0x<redacted> -q irtt-
telemetry.coxlab.net:22112
[Connecting] connecting to irtt-telemetry.coxlab.net:22112
[184.176.185.20:22112] [Connected] connection established
[184.176.185.20:22112] [WaitForPackets] waiting 352ms for final packets

                         Min     Mean   Median      Max  Stddev
                         ---     ----   ------      ---  ------
                RTT  78.92ms  84.55ms  83.55ms  117.3ms  3.17ms
         send delay   -1.24s   -1.23s   -1.23s   -1.21s  2.25ms
      receive delay    1.31s    1.32s    1.32s    1.35s  2.22ms

      IPDV (jitter)   1.93µs   2.32ms   1.13ms  34.77ms  3.15ms
          send IPDV    110ns   1.89ms    925µs  19.11ms  2.41ms
       receive IPDV    754ns    740µs    274µs  34.42ms  2.31ms

     send call time   12.9µs     72µs             932µs  46.5µs
        timer error    100ns    129µs             827µs   107µs
  server proc. time   4.45µs   9.39µs             128µs  4.86µs

                duration: 30.3s (wait 352ms)
   packets sent/received: 1471/1471 (0.00% loss)
 server packets received: 1471/1471 (0.00%/0.00% loss up/down)
     bytes sent/received: 253012/253012
       send/receive rate: 67.5 Kbps / 67.5 Kbps
           packet length: 172 bytes
             timer stats: 28/1499 (1.87%) missed, 0.64% error
```

WHERE TO PUT THE IRTT SERVER?

- Wherever you want to!
  - Containerized with minimal resource requirements
- Cox is using the Service Layer Router (SLR) attached to the Hub Router
  - SLR is as close to access network as we can get
  - Router container implementation limits resource usage, preventing impact to other services
  - Access controls on router ensure only a single port/service is accessible from the public IPv6 address
- A VM will be used for IPv4-only households (~50%!)
  - Not as close as SLR, we are assessing the impact

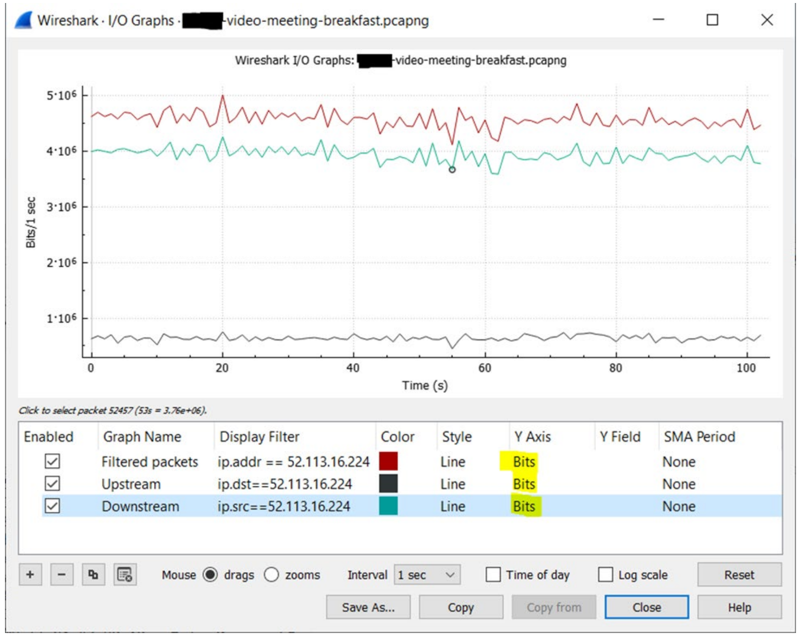
IRTT TRAFFIC PROFILES

- Currently simulating audio stream (67.5 Kbps, 172 byte UDP payload)
- Next step is to simulate gaming and video conferencing traffic
- An application can easily be characterized using Wireshark
  - Perform inline sniffer capture using switch with mirror port
  - Analyze capture using Wireshark IO Graph and Packet Length analysis tools
  - **Full details in paper**



Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
⌄ Packet Lengths	89923	653.99	81	1269	0.8731	100%	1.7400	20.248
0-19	0	-	-	-	0.0000	0.00%	-	-
20-39	0	-	-	-	0.0000	0.00%	-	-
40-79	0	-	-	-	0.0000	0.00%	-	-
80-159	11323	125.60	81	159	0.1099	12.59%	0.2600	58.228
160-319	4745	184.75	160	319	0.0461	5.28%	0.1400	74.905
320-639	26636	467.61	320	639	0.2586	29.62%	0.8600	19.465
640-1279	47219	932.99	640	1269	0.4585	52.51%	0.9400	63.234
1280-2559	0	-	-	-	0.0000	0.00%	-	-
2560-5119	0	-	-	-	0.0000	0.00%	-	-
5120 and greater	0	-	-	-	0.0000	0.00%	-	-

Display filter: ip.addr==52.113.16.224



Wireshark · I/O Graphs · [redacted]-video-meeting-breakfast.pcapng

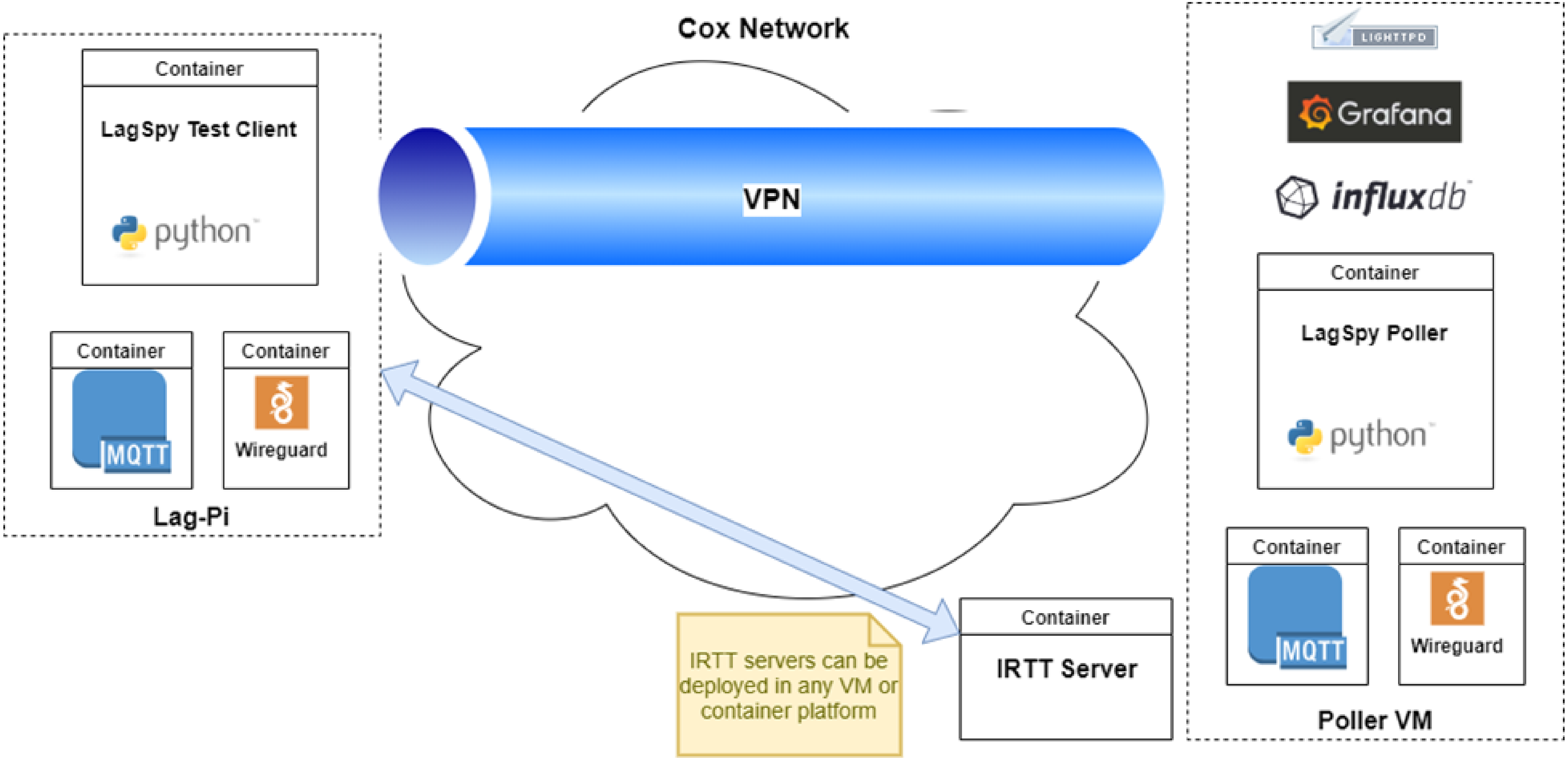
Wireshark I/O Graphs: [redacted]-video-meeting-breakfast.pcapng

Click to select packet 52457 (53s = 3.76e+06).

Enabled	Graph Name	Display Filter	Color	Style	Y Axis	Y Field	SMA Period
☑	Filtered packets	ip.addr == 52.113.16.224	🟥	Line	Bits		None
☑	Upstream	ip.dst==52.113.16.224	⬛	Line	Bits		None
☑	Downstream	ip.src==52.113.16.224	🟦	Line	Bits		None

Mouse ● drags ○ zooms Interval 1 sec □ Time of day □ Log scale

LAGSPY SOFTWARE ARCHITECTURE



IRTT servers can be deployed in any VM or container platform

LAGSPY SOFTWARE ARCH - REFERENCE

The principal components of the Lag-Pi are:

- The LagSpy Test Client, written in Python 3.8.
- Eclipse Mosquitto to implement an MQTT client.
- Wireguard to establish a VPN connection to the Poller for command and control.

The principal components of the Poller are:

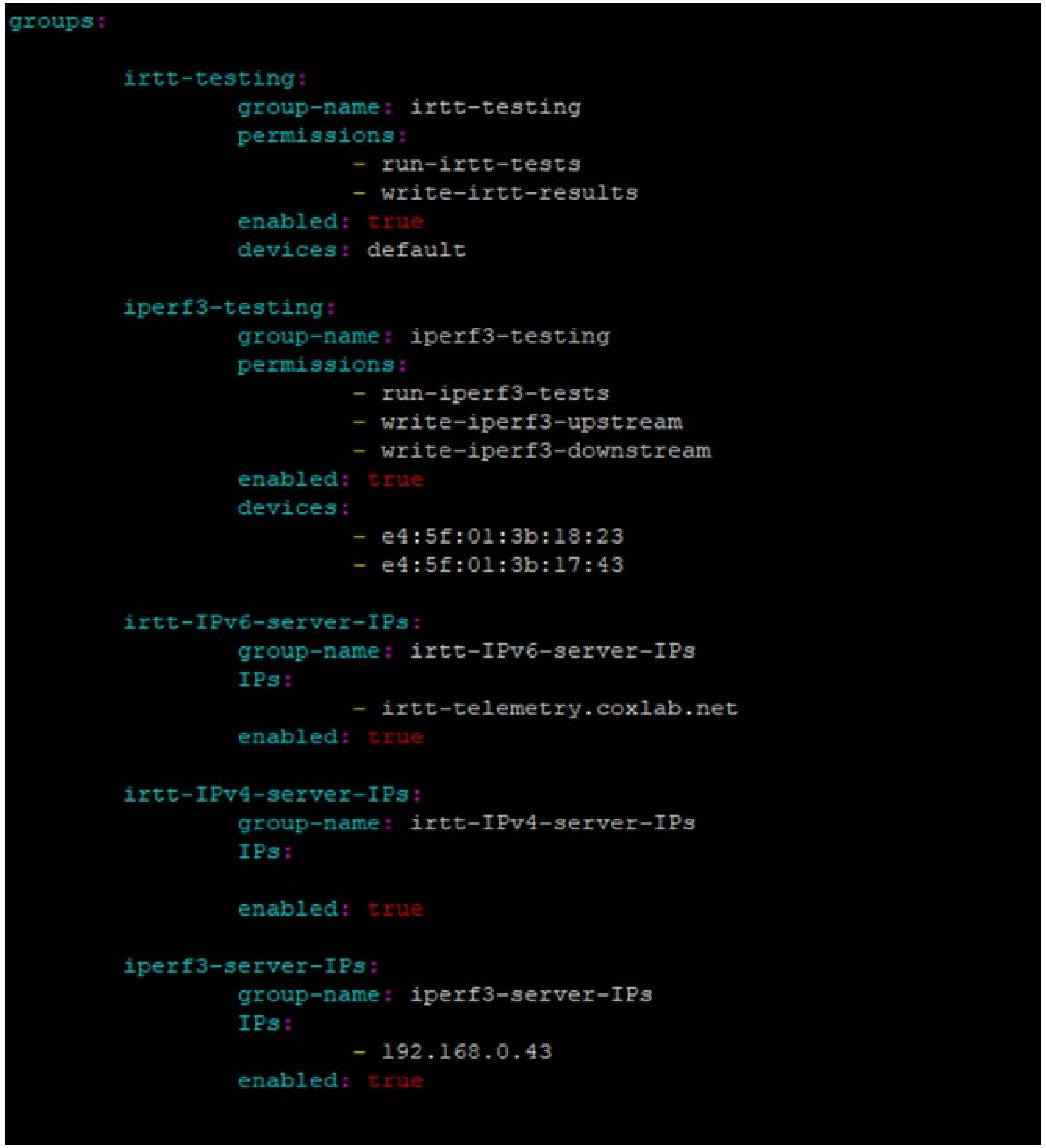
- The Lagspy Poller, written in Python 3.8.
- Eclipse Mosquitto to implement an MQTT broker and localhost client.
- Wireguard for a VPN endpoint.
- InfluxDB to import and aggregate data from the Poller for visualization.
- Grafana for visualization of test results.
- Lighttpd (primarily used to upgrade the Lag-Pi.)

MQTT TOPICS

Topic	Arguments	Direction	Description
connect/hello	n/a	Lag-Pi → Poller	Register with poller and keepalive
connect/enroll/<mac>	MAC address of Lag-Pi	Poller → Lag-Pi	Provide VPN credentials to Lag-Pi
connect/link_ok/<mac>	MAC address of Lag-Pi	Poller → Lag-Pi	Keepalive response
irtt/start/<mac>	MAC address of Lag-Pi	Poller → Lag-Pi	Start IRTT test
irtt/results	n/a	Lag-Pi → Poller	Results of IRTT test
iperf/start/<mac>	MAC address of Lag-Pi	Poller → Lag-Pi	Start IPERF3 test
iperf/results	n/a	Lag-Pi → Poller	Results of IPERF3 test

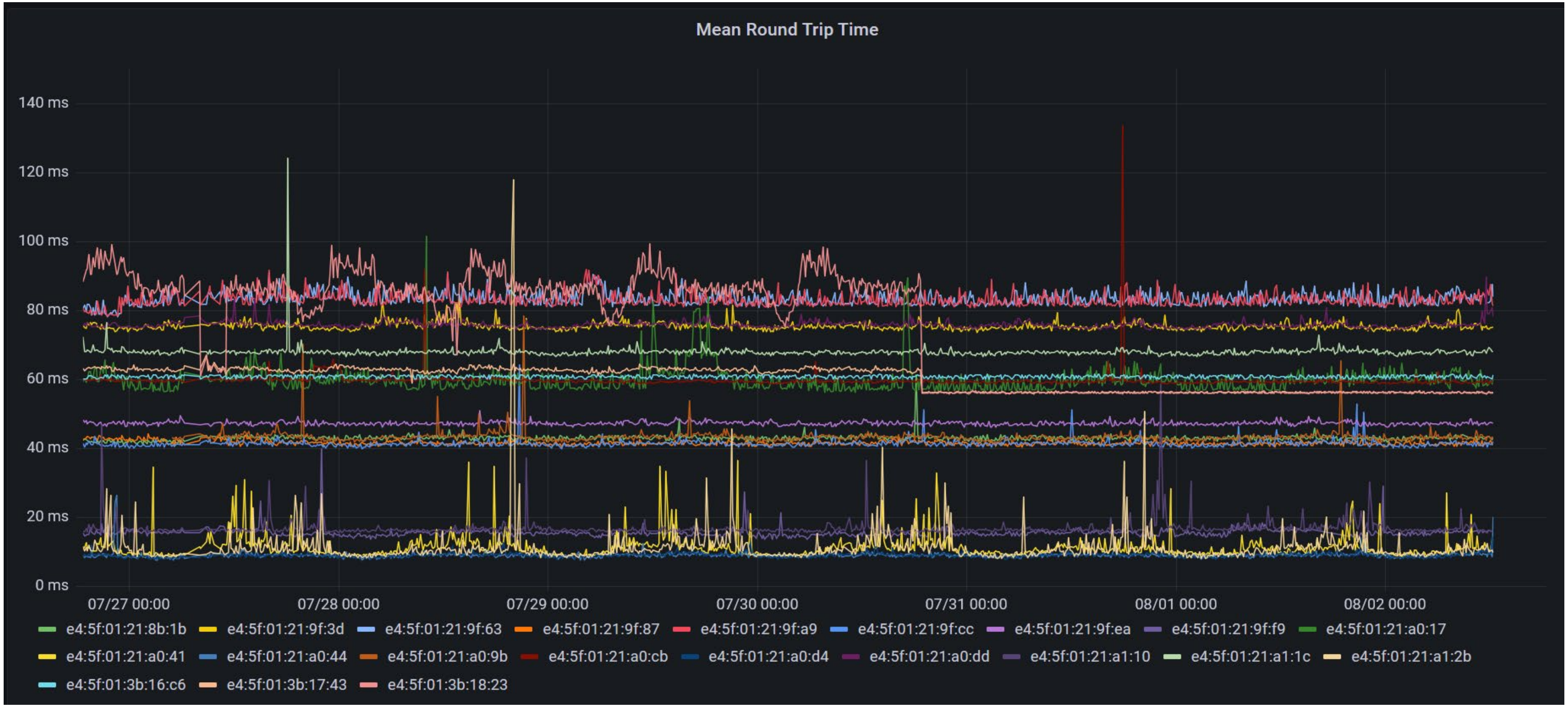
YAML POLICY FILE

- Need policy framework to minimize manual configuration of devices
  - We mailed identical devices to volunteers!
- Assign Lag-Pi's into groups that can run different tests
  - We are leveraging this framework to automate IPERF3 testing
- Define IRTT Server groups (server selected by network hops)
  - No need to manually provision individual Lag-Pi to closest server

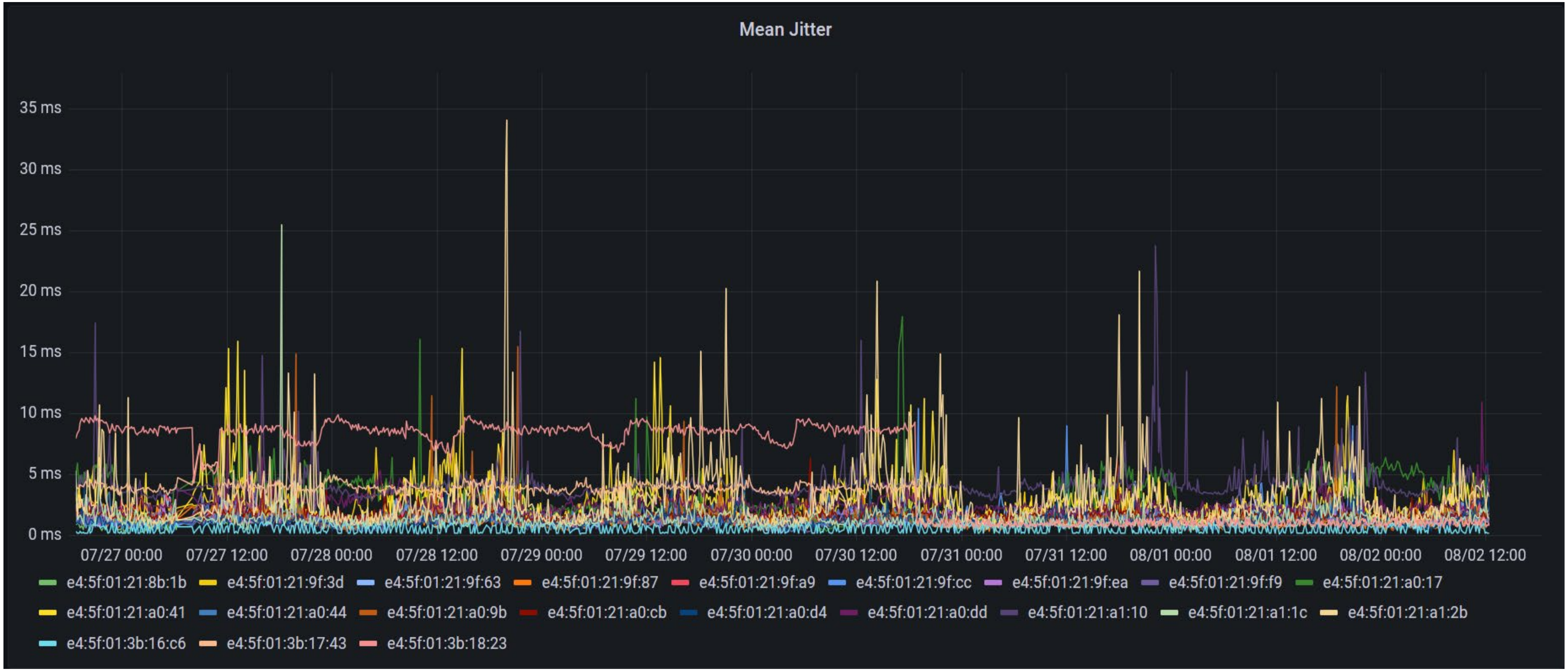


```
groups:

        irtt-testing:
                group-name: irtt-testing
                permissions:
                        - run-irtt-tests
                        - write-irtt-results
                enabled: true
                devices: default

        iperf3-testing:
                group-name: iperf3-testing
                permissions:
                        - run-iperf3-tests
                        - write-iperf3-upstream
                        - write-iperf3-downstream
                enabled: true
                devices:
                        - e4:5f:01:3b:18:23
                        - e4:5f:01:3b:17:43

        irtt-IPv6-server-IPs:
                group-name: irtt-IPv6-server-IPs
                IPs:
                        - irtt-telemetry.coxlab.net
                enabled: true

        irtt-IPv4-server-IPs:
                group-name: irtt-IPv4-server-IPs
                IPs:

                enabled: true

        iperf3-server-IPs:
                group-name: iperf3-server-IPs
                IPs:
                        - 192.168.0.43
                enabled: true
```

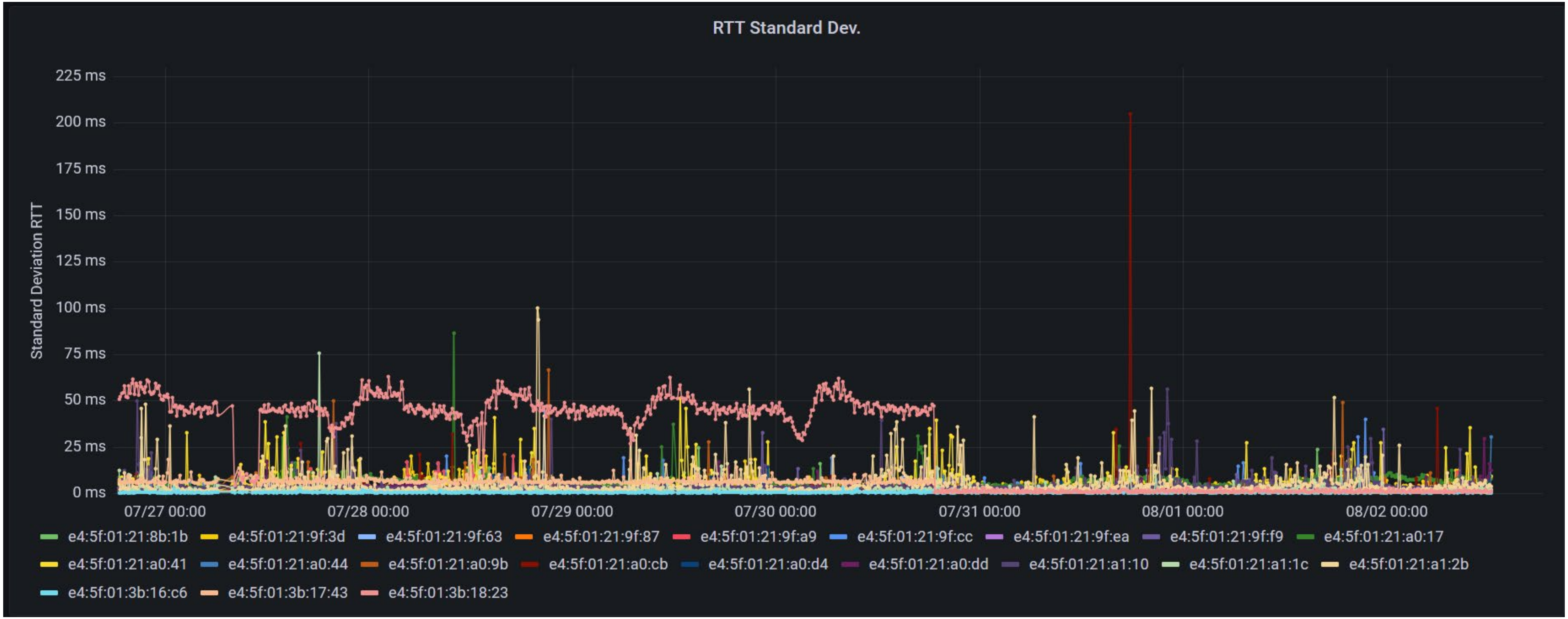
Mean Round Trip Time – Note cyclic increases for some devices



Mean Jitter. Again note cyclic behavior.

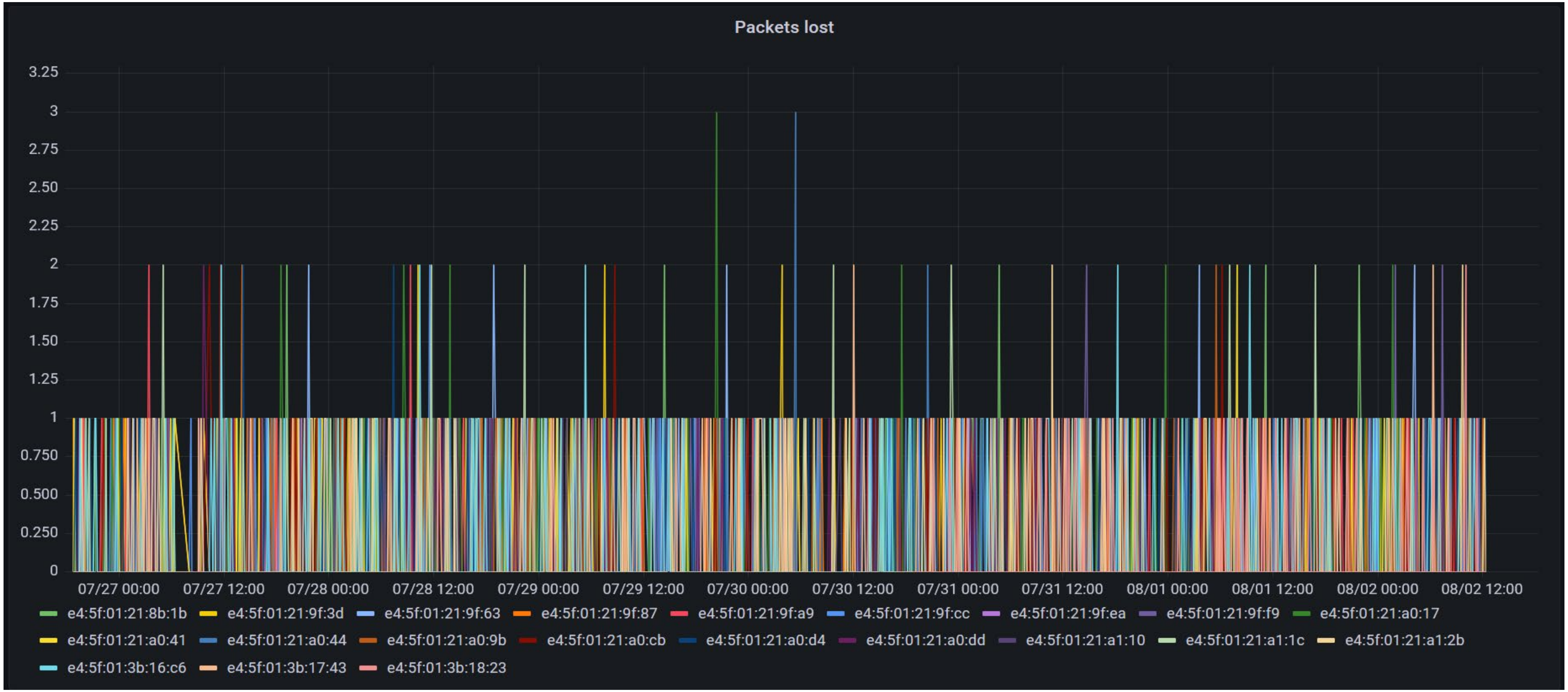


Round Trip Time Standard Deviation



GRAFANA VISUALIZATION

UDP Packet Lost (out of 1500)



RESOURCE USAGE

- CPU usage < 0.1% idle, 4.3% during active test
- Light memory footprint
- Application is easily portable to a Linux-based gateway platform (especially if platform already supports Docker)

Component	% Memory	Virtual Memory
Docker overhead	6.2%	3.4 MB
Python Test Client	0.6%	52 KB
IRTT Client	0.1%	879 KB

LESSONS LEARNED

- 50% of households have home gateway with IPv6 disabled
- Many home network topologies – big benefits to router integration
- Wireguard NAT keepalive
    - PersistentKeepalive keyword
- Many internal reviews needed
- Ability to remotely upgrade Lag-Pi is critical from day 1
- Outbound firewall rules are still a thing
- NOOBS 3.5 compatibility issue with Raspberry Pi 4

SCTE® CABLE-TEC EXPO® FAST FORWARD 2021

ATLANTA, GA
OCTOBER 11-14

SCTE
a subsidiary of CableLabs®

Thank You!

Michael Overcash

Principal Engineer
Cox Communications
michael.overcash@cox.com