Measuring Network Latency from a Wireless ISP:
Variations Within and Across Subnets

Simon Sundberg, Anna Brunström, Simone Ferlin-Reiter,
Toke Høiland-Jørgensen & Robert Chacón

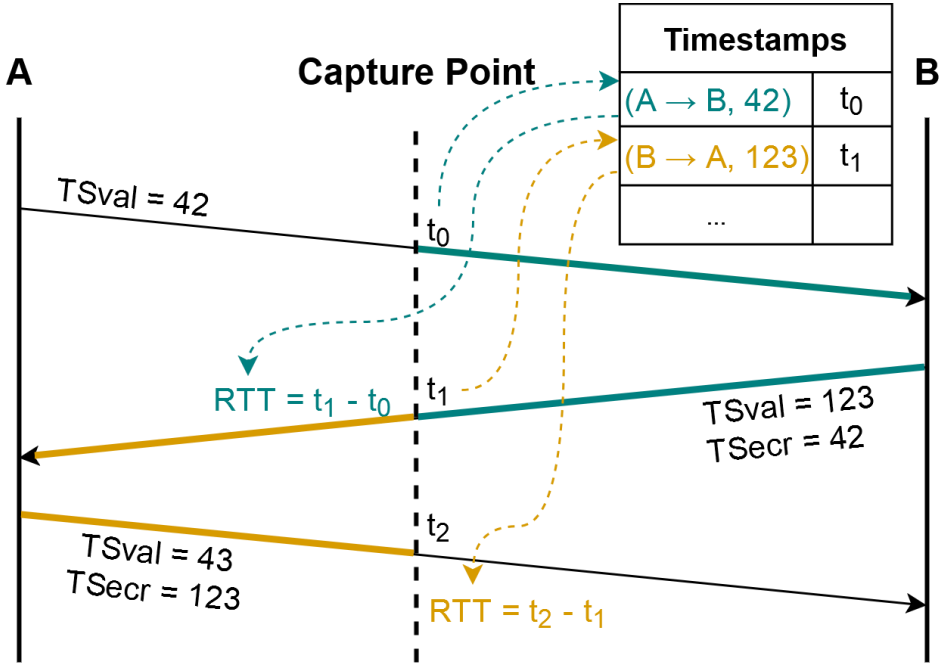
How to monitor latency?

- Active measurements unsuitable for ***monitoring***
  - Gives latency of probes, not application traffic
  - Benoit Claise's presentation yesterday, tokyo-ping

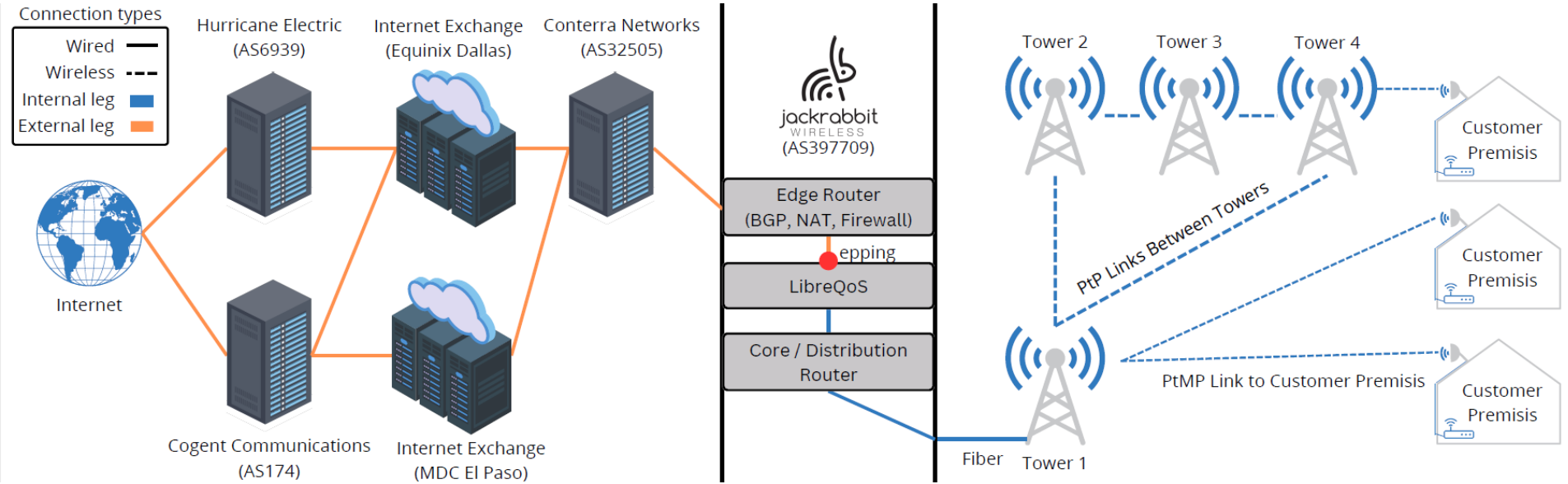
- Passive measurements to the rescue
  - Observe latency experienced by application traffic
  - P4, DPDK, and eBPF to handle high packet rates

Evolved Passive Ping (epping)

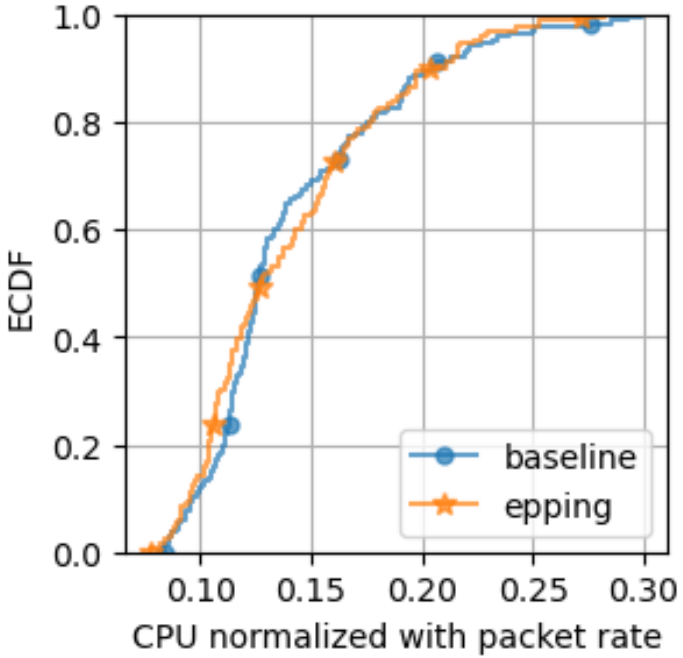
- Match TCP segments and ACKs
  - No collaboration from endhosts
  - Latency relative to capture point
- Implemented in eBPF
  - Low overhead
  - Compatible with Linux networking
- Aggregate stats per subnet



Deployment at JackRabbit Wireless

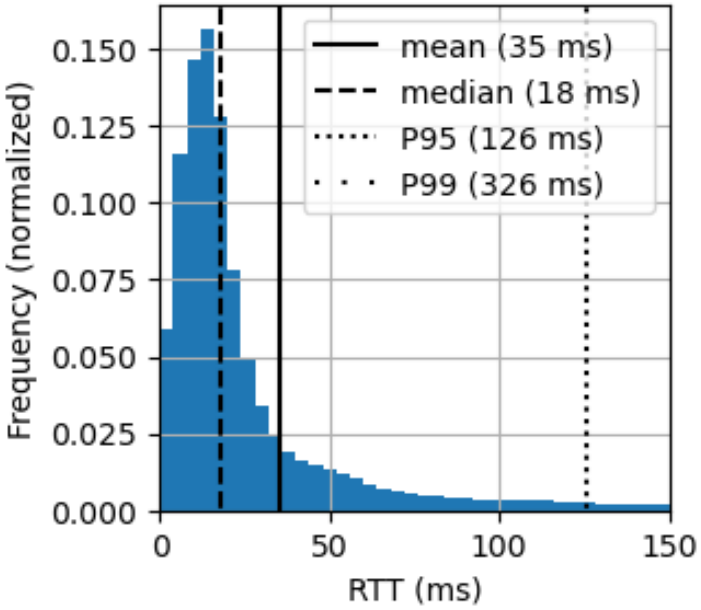


Monitoring overhead

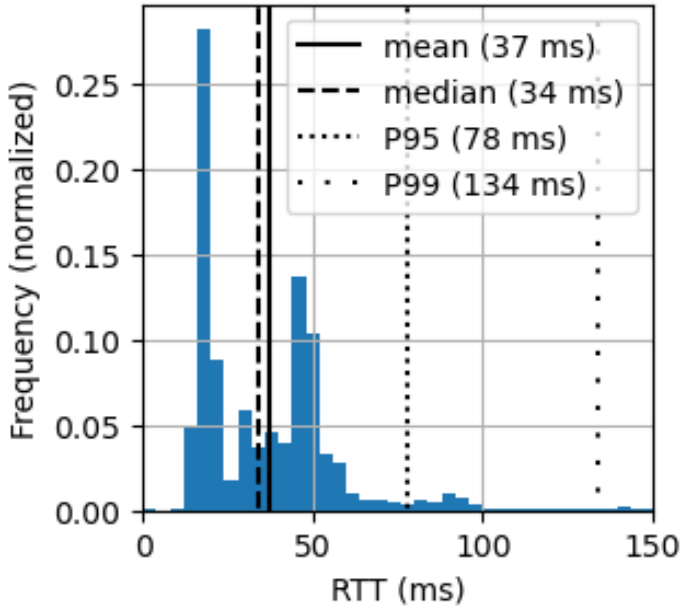


- CPU: AMD Ryzen 5 3600X
- A/B test → 0 % overhead?
- ~300 ns/packet
  - 0.65 %pt. overhead at 130 kpps
  - 5 %pt. overhead at 1 Mpps

Latency – Internal and external

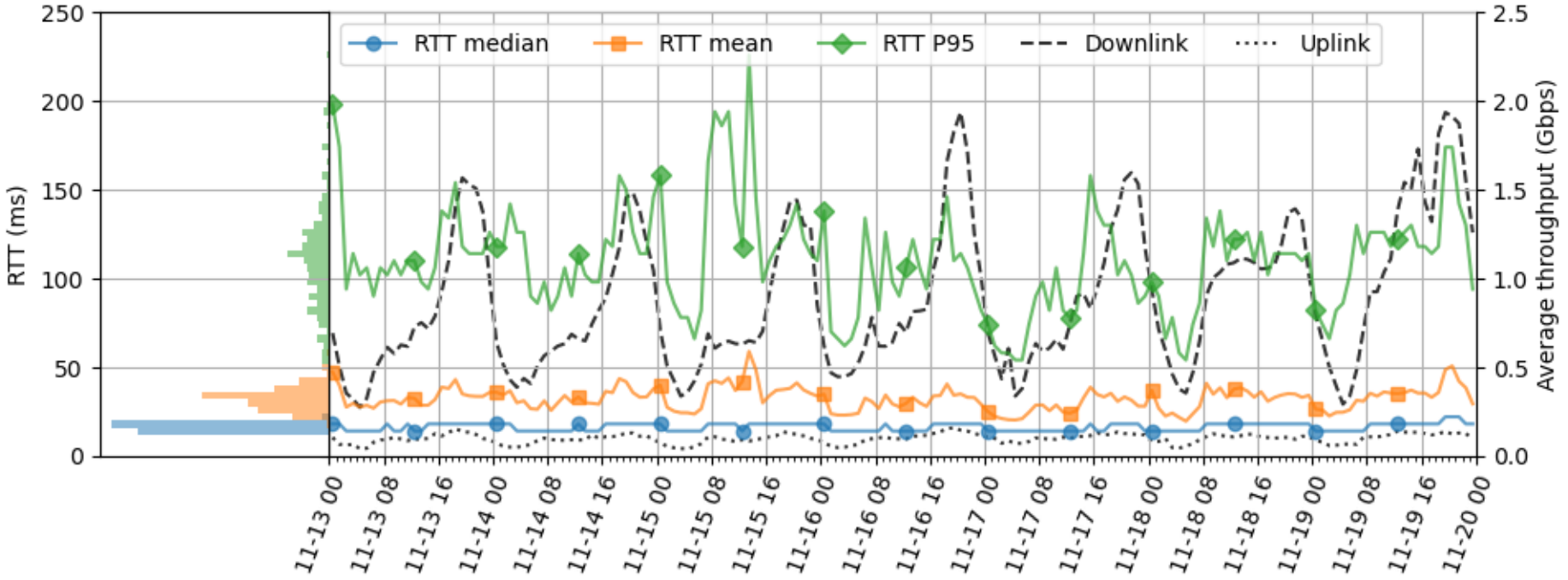


Internal

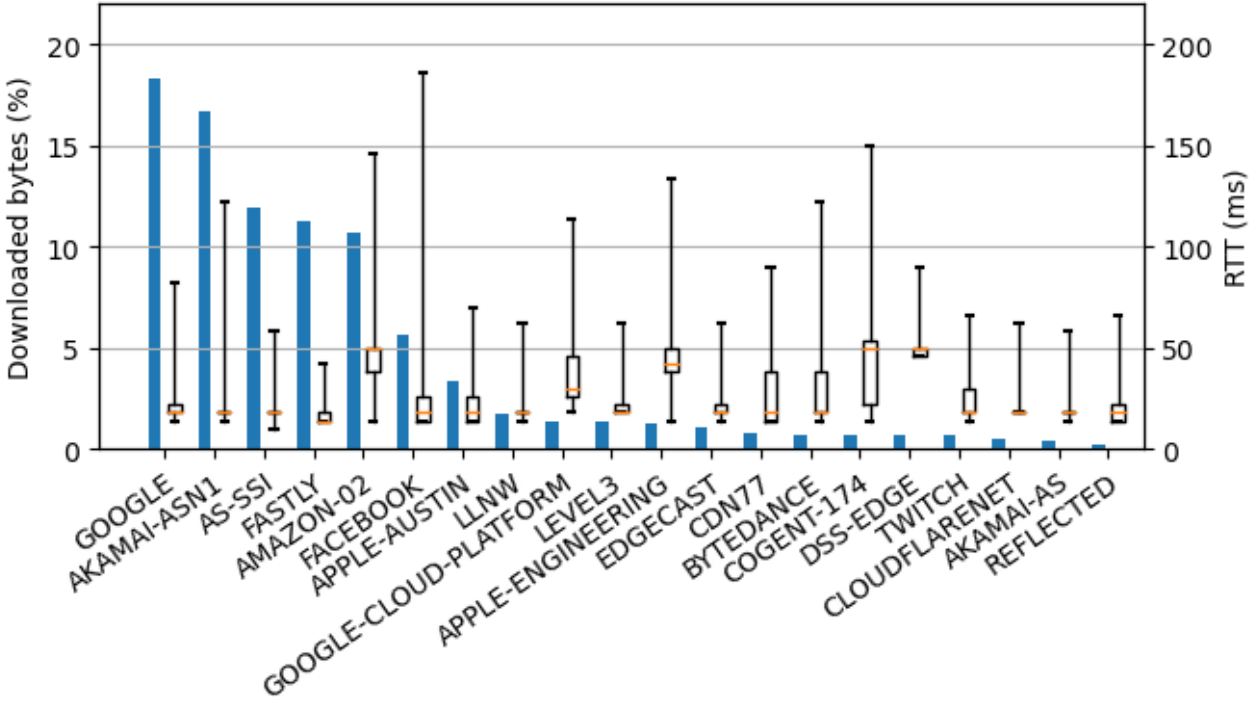


External

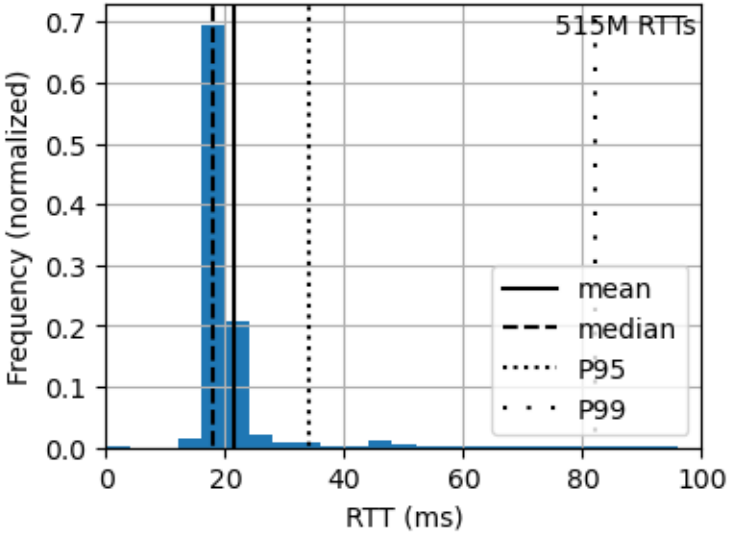
Internal latency variations over time



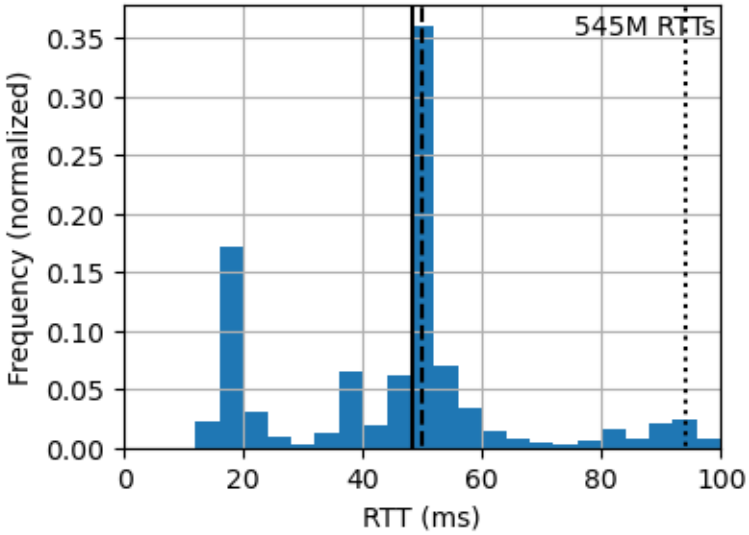
External latency towards most popular ASNs



External latency towards most popular ASNs



Google



Amazon-02

Conclusion

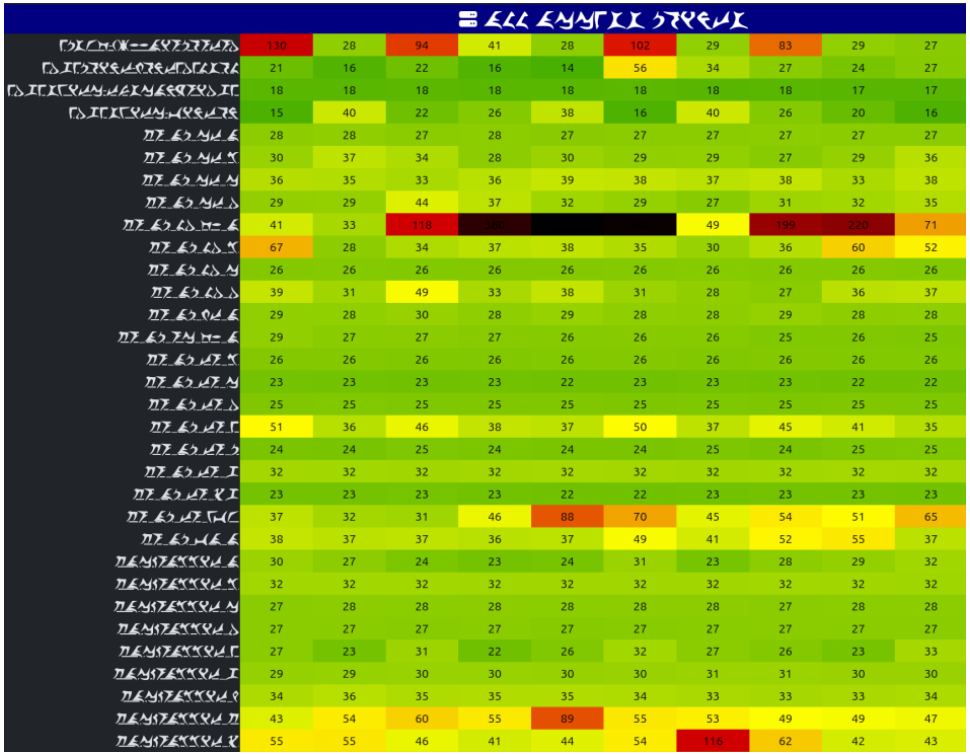
- Software based passive latency monitoring feasible for ISPs
  - ~1 %pt. CPU overhead, detailed latency statistics
- Last-mile remains a significant source of network latency
  - The Good, the Bad and the WiFi
- Limitations
  - TCP only – how to handle UDP/QUIC?
  - Small measurement study – plans for larger scale study with LibreQoS

epping: https://github.com/xdp-project/bpf-examples/tree/master/pping
Data: https://doi.org/10.5281/zenodo.13388092

LibreQoS

- SQM software for ISPs
  - Middlebox south of NAT
- Passive monitoring
  - Including latency + retrans
  - Built-in dashboard

https://libreqos.io/

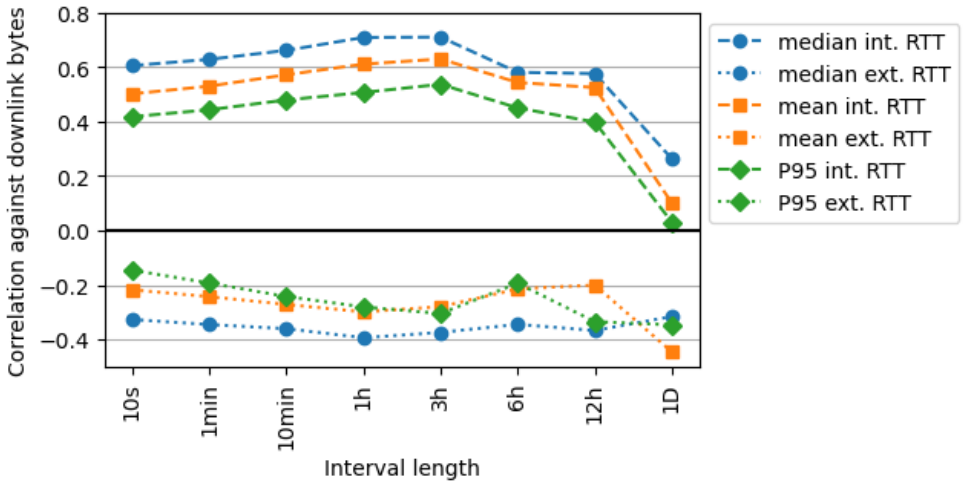


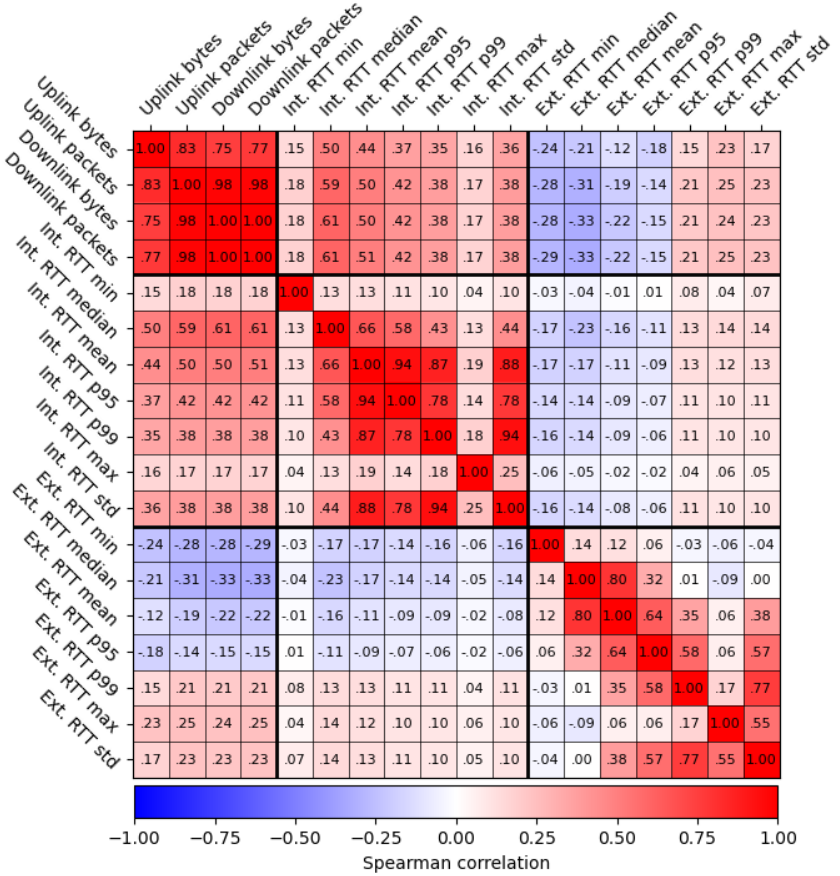
RTT Heatmap – From https://libreqos.io/lts/, ©LibreQoE, LLC

Thank you for your time!

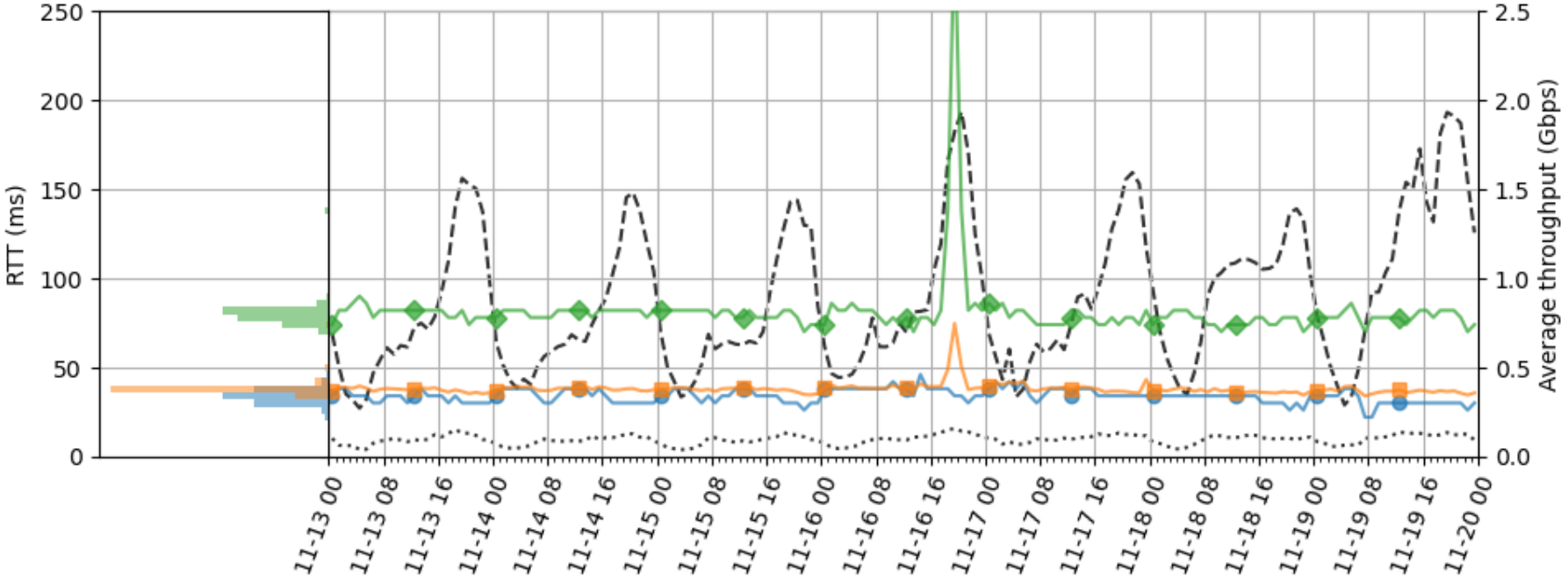
Questions?

RTT/traffic load correlation

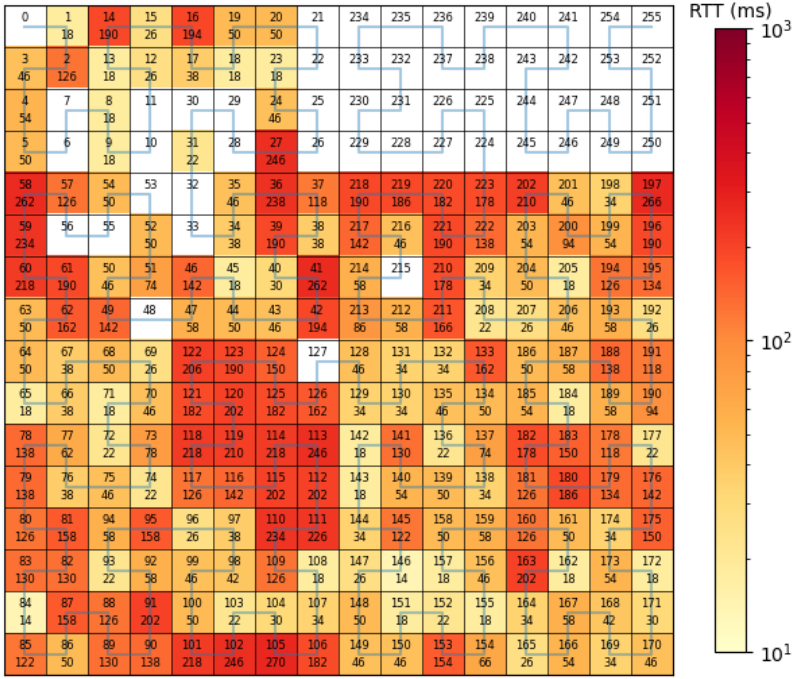


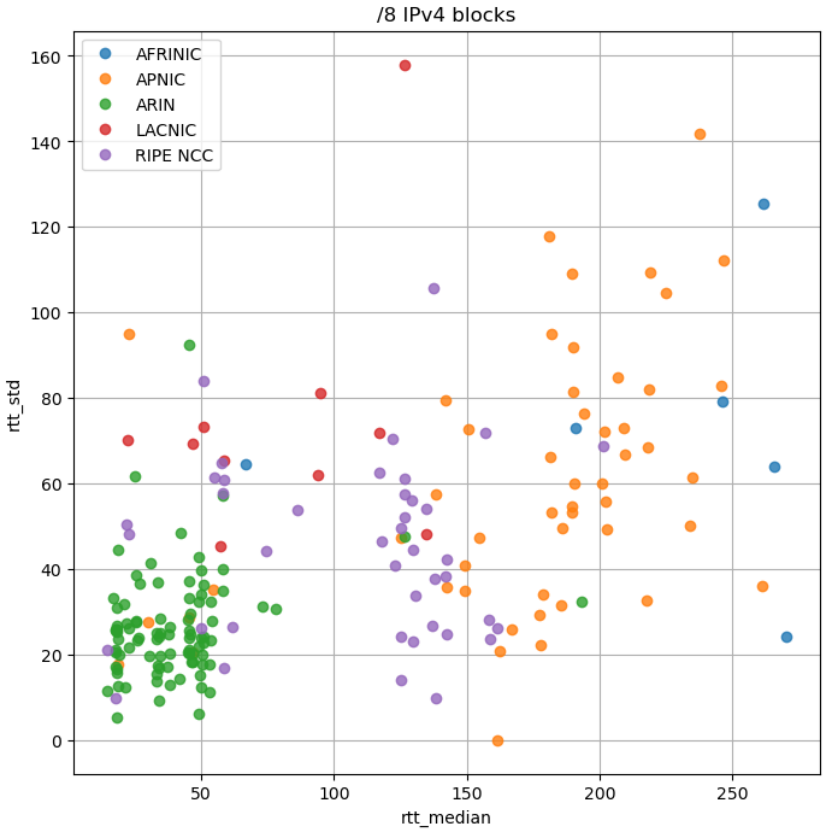


External latency over time

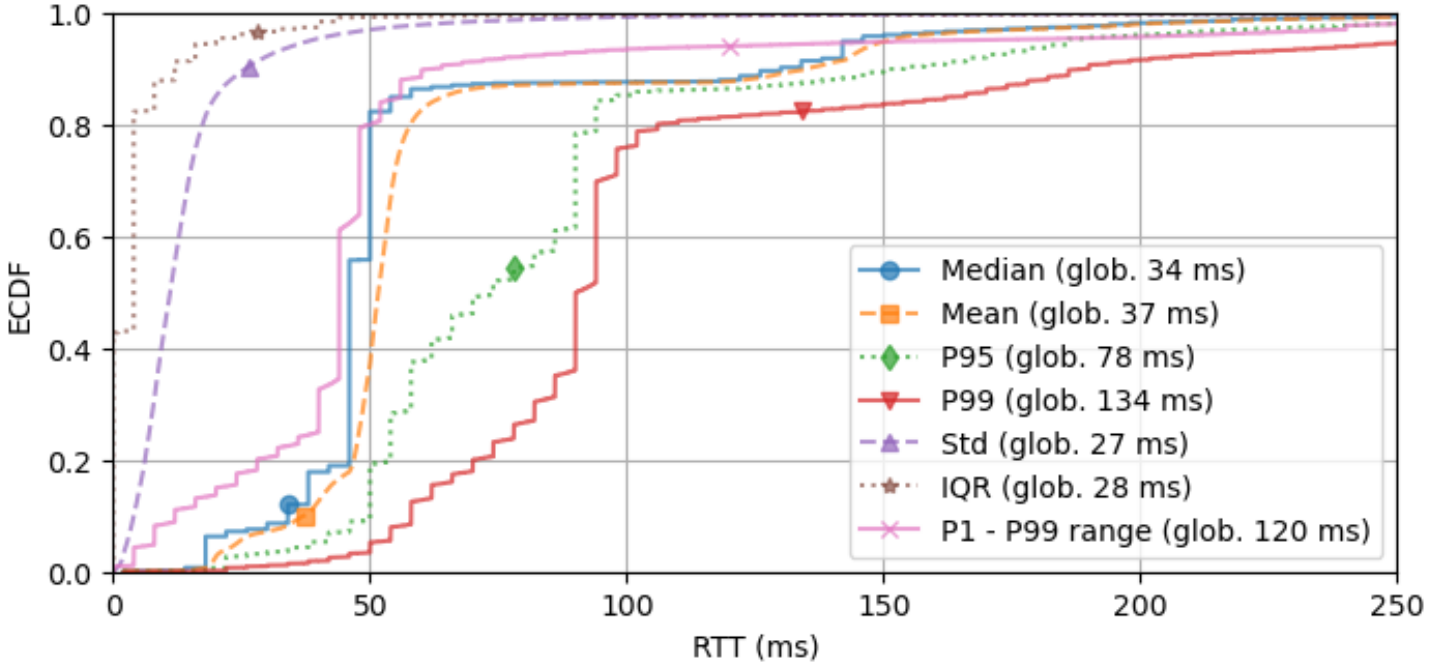


Internetmap

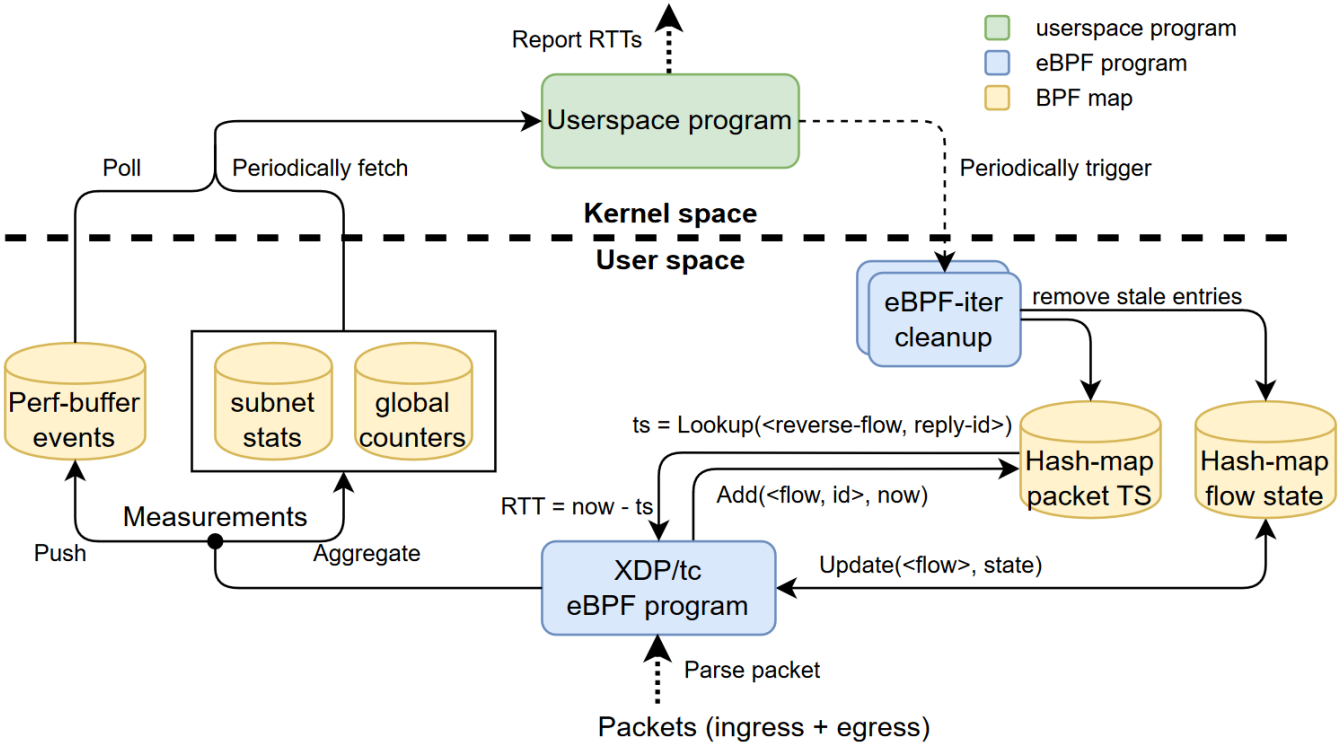




RTT per subnet



epping design



What is eBPF?

- Runtime environment in kernel
  - Attach small programs to various hooks
- Use cases
  - Observability, Security, Networking
- Workflow
  - Compile to eBPF bytecode
  - Load into kernel
    - Verified
    - Jitted
  - Attach to hook

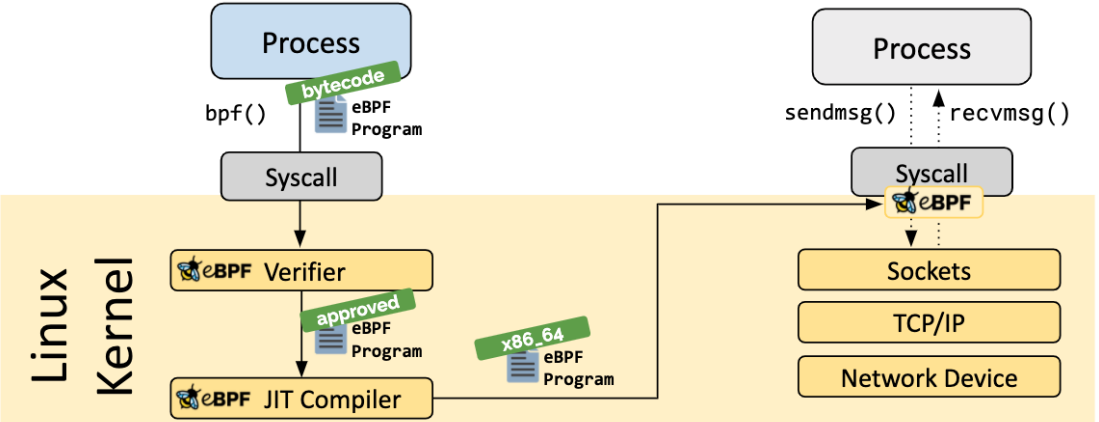
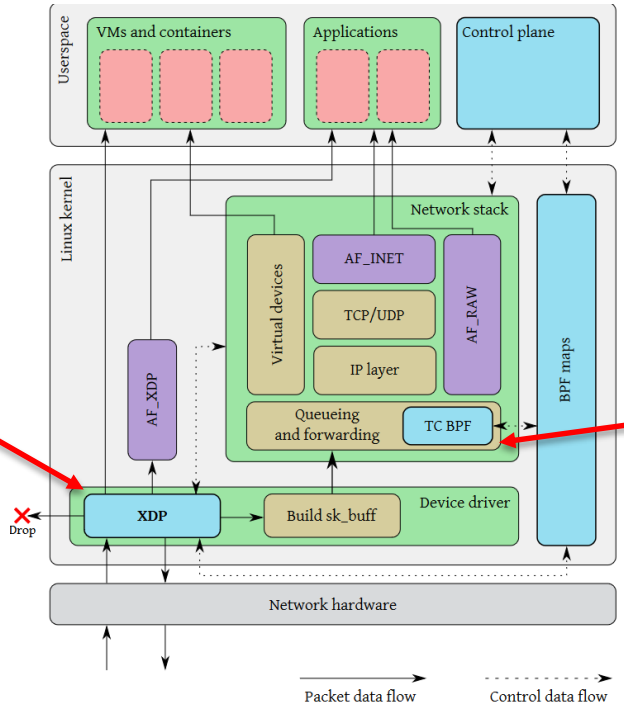


Figure from https://ebpf.io/what-is-ebpf

Where the XDP and TC-BPF hooks reside

As soon as NIC delivers packet to kernel



Slightly later, but still before most of the network stack processing

Figure from https://github.com/tohojo/xdp-paper (CC-BY SA)