ESnet
ENERGY SCIENCES NETWORK

Programmable Per-Packet Network Telemetry: From Wire to Kafka at Scale

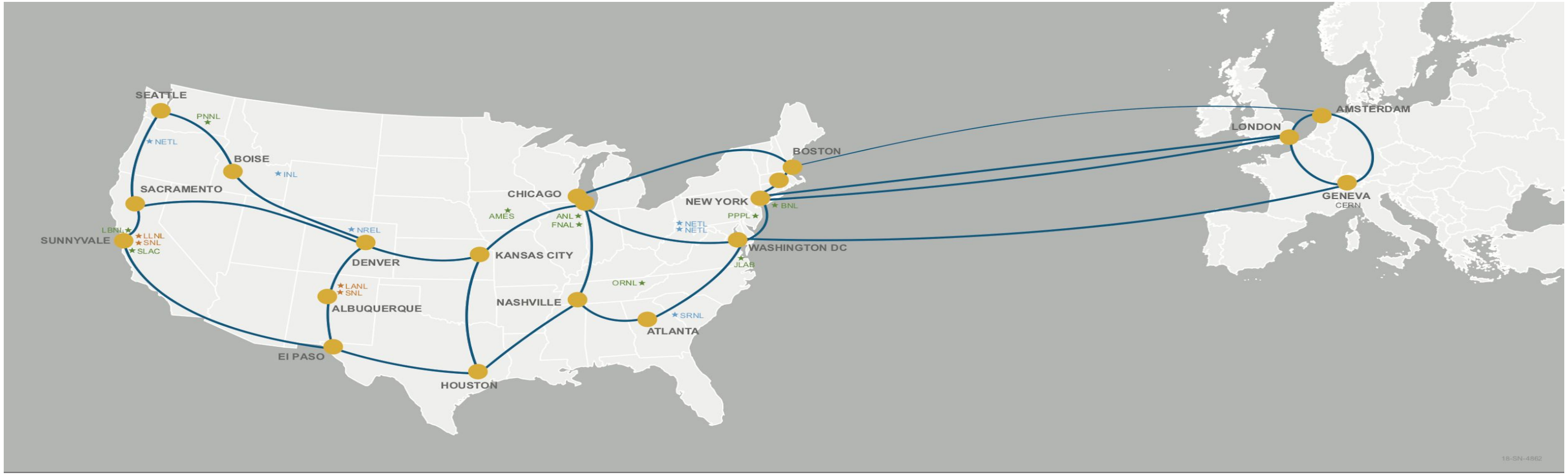
Zhang Liu (University of Colorado Boulder)
Yatish Kumar (LBNL, ESnet)
Bruce Mah (LBNL, ESnet)
Chin Guok (LBNL, ESnet)
Richard Cziva (LBNL, ESnet)

SNTA '21, June 21, 2021, Virtual Event, Sweden

U.S. DEPARTMENT OF ENERGY
Office of Science

BERKELEY LAB

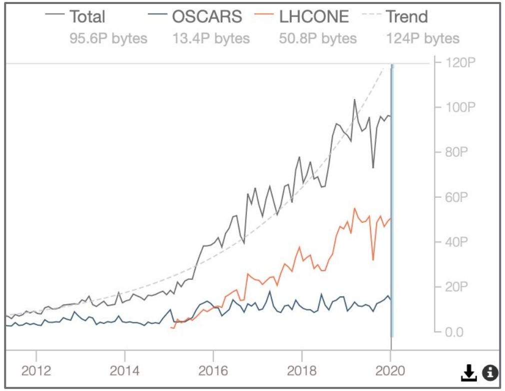
ESnet: DOE’s high-performance network (HPN) user facility optimized for enabling big-data science



ESnet provides connectivity to
all of the DOE labs, experiment sites, & supercomputers

Increasing Need for Programmability

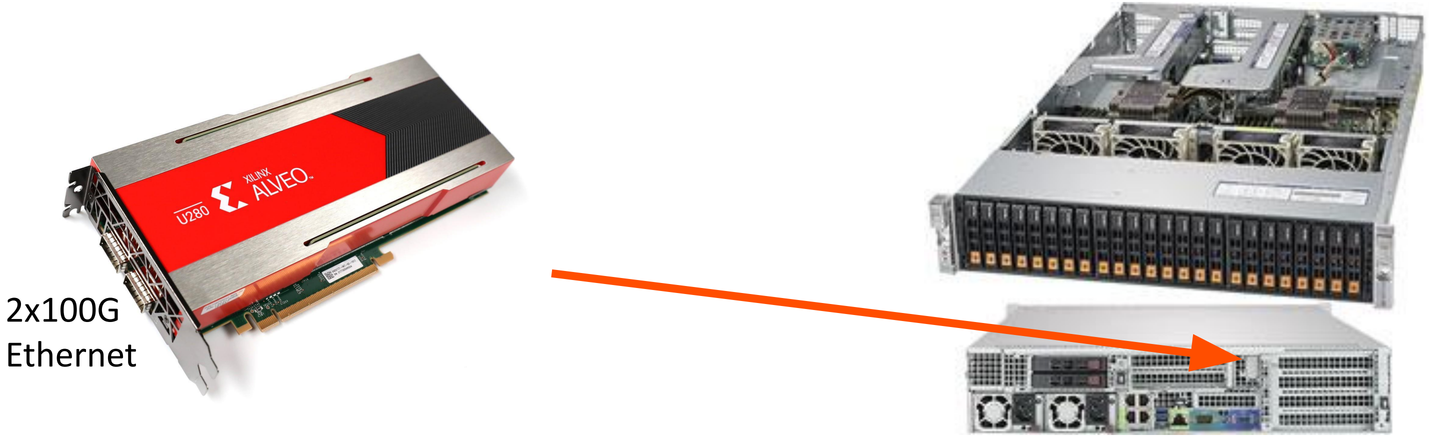
- ESnet's traffic, user-base and the experiments continue to grow in a fast pace
- Computing and data model are also evolving, requiring:
  - fine-grained visibility in real-time
  - application-specific traffic handling
  - programmable, in-network services
- Needs not addressed by existing measurement mechanisms (sampled, aggregated, delayed)
- High Touch Services created to fulfill these needs



Live ESnet usage statistics: my.es.net
Total carried: Exabyte/year.

ESnet

Hightouch Server Hardware



Xilinx U280 FPGA Card

2x100G Ethernet QSFP-28

Custom Logic for Flow Tracking

High End Server

Dual Socket, Fast Storage

Hosts Hightouch Application

ESnet Network Packet Telemetry Data

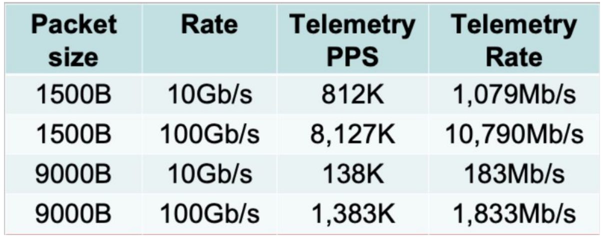
- SNMP
  - All interfaces, 30 seconds poll interval
  - Primary use: failure detection, traffic visualization: http://my.es.net
  - Data rate: 4000 interfaces => **130 events per second**
- Netflow / IPFIX
  - All interfaces, packets sampled 1:1000
  - Primary use: capacity planning (offline)
  - Raw data rate: **~ 6500 events per second**
- High Touch Services
  - Selected interfaces and flows, 1:1 packet to telemetry
  - Primary use: high-precision telemetry
  - Raw data rate: **~ 1 to 8 million events per second** for a single interface

Telemetry	**Raw Data Rate Per Second**
SNMP	130
Netflow / IPFIX	6500
High Touch Services	1 - 8 M

Telemetry Data Rates

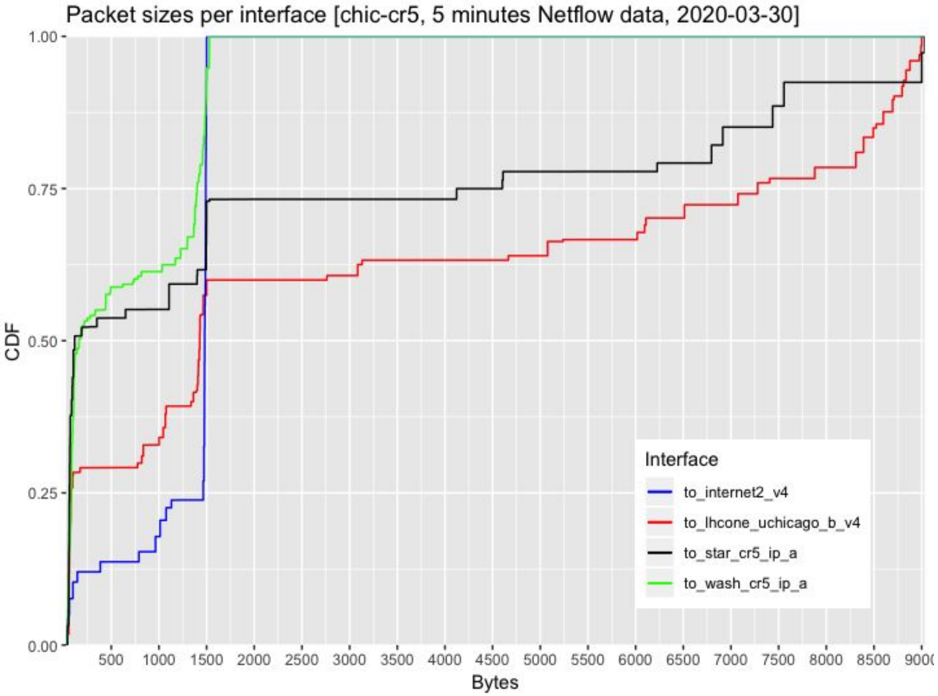
Per-Packet Data Rates

- Packet size depends on:
  - MTU
  - Application (science vs http)
  - Average for science traffic: ~1500B
- Traffic rate at ESnet at any time:
  - All traffic: O(1Tbit/s)
  - Large customers: O(100Gbit/s)



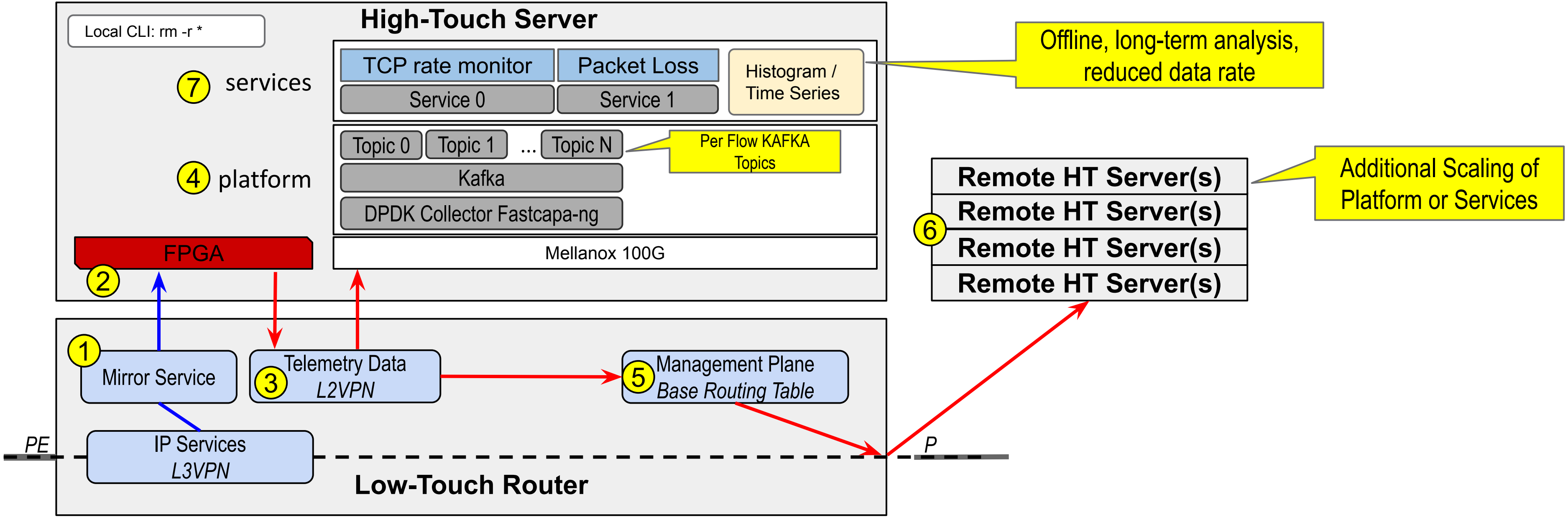
Packet size	Rate	Telemetry PPS	Telemetry Rate
1500B	10Gb/s	812K	1,079Mb/s
1500B	100Gb/s	8,127K	10,790Mb/s
9000B	10Gb/s	138K	183Mb/s
9000B	100Gb/s	1,383K	1,833Mb/s

Telemetry Packet Rates



Estimated packet sizes in production

ESnet6 High-Touch Architecture Overview



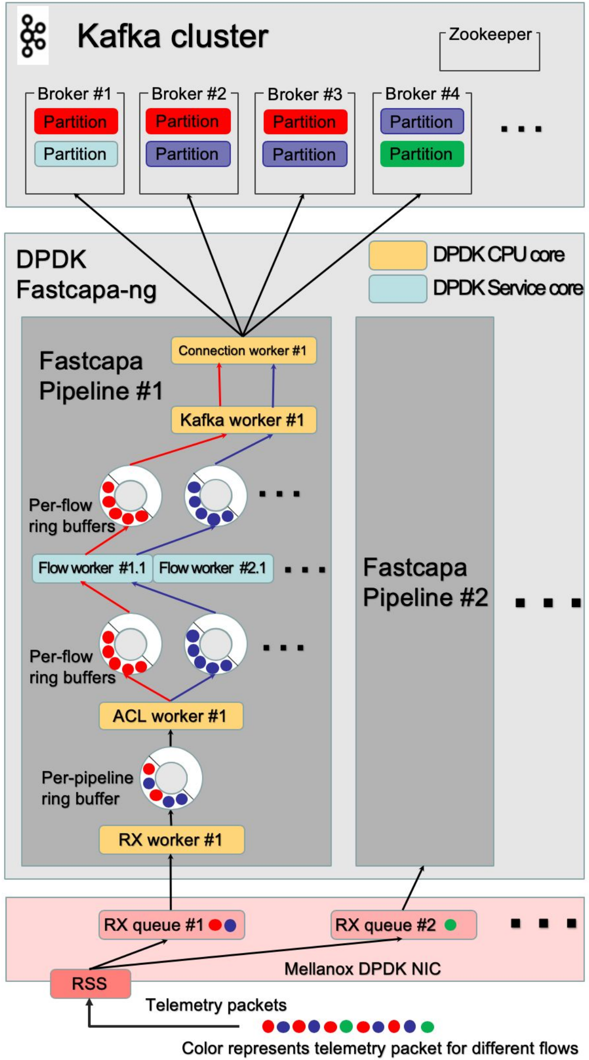
1. Mirror Service - Allows selective flows in the dataplane to be duplicated and sent to the FPGA for processing.
2. Programmable Dataplane (DP) - Appends meta-data, timestamps and repackages packet for transmission to Platform code.
3. Telemetry Data L2VPN - Connect Dataplane and Platform, possibly on different High-Touch Servers.
4. Platform - Reads telemetry packets from the network and distributes information to High Touch Services.
5. Management Plane Base Routing Table - Provides connectivity to Remote Servers.
6. Remote Server - Hosts Platform components or Services (but not a Dataplane). Telemetry data can be directed to Remote Servers.
7. Service - Reads data from the Platform and performs real-time analysis as well as inserts selected telemetry data into database.

– – – –	Datapath of Customer Packet
– – – –	Datapath of Mirrored Packet
- - - - - - - -	Datapath of Telemetry Packet

ESnet

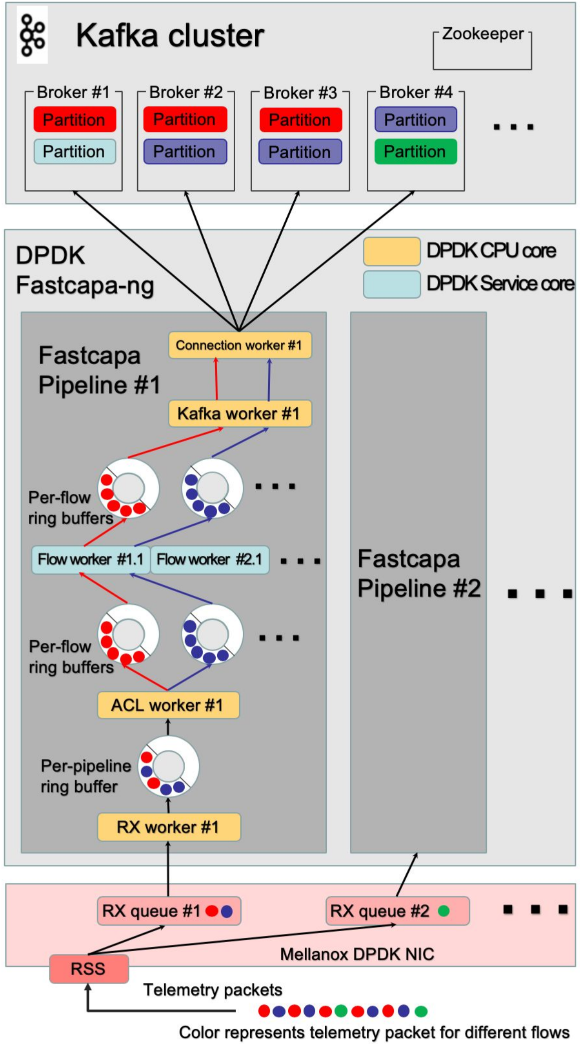
Fastcapa-ng

- **ESnet-developed software (C / DPDK)**
  - Based on Apache Metron Fastcapa
  - Uses DPDK: fast packet processing API
  - Primary functions: telemetry processing, batching, filtering, aggregation, forwarding
- **Design goals:**
  - Packet order preservation
  - High-performance Kafka handling
  - Easy programming
- **Multi-pipeline** design for scalability, each pipeline can handle TCP flows from single 100G link.
- **Multi-stage** design for performance, each packets will be processed by 5 CPUs in series.

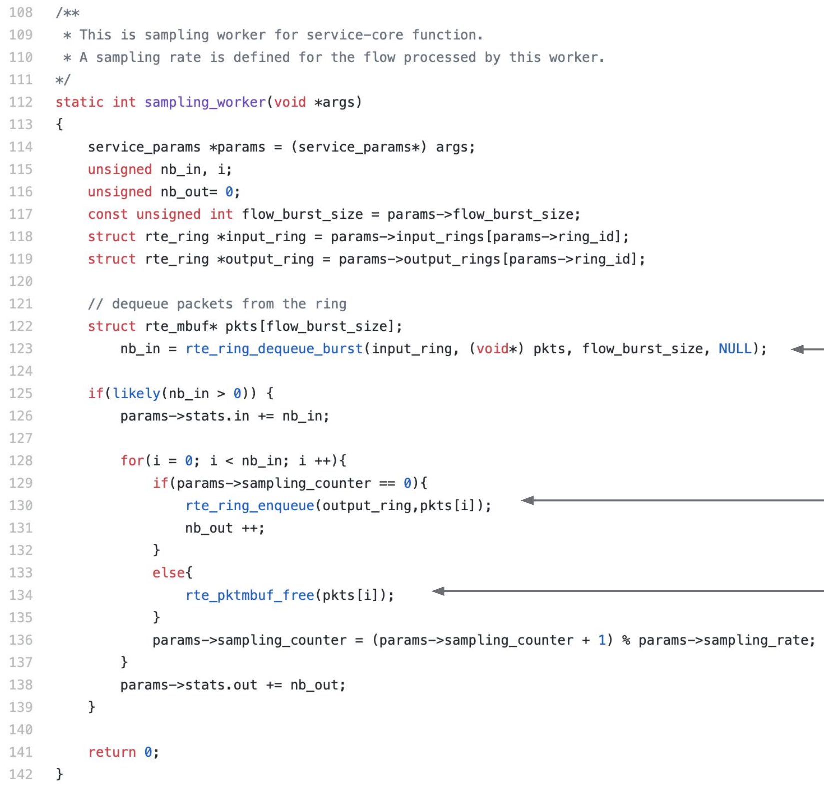


Fastcapa-ng Internals

- Dedicated Kafka connection
  - maintain TCP connection, message compression task
- Kafka worker
  - Combine multiple telemetry packets into large kafka messages
- Flow worker (service cores)
  - process flows using different function:
    - Passthrough
    - Sampling
    - Histogram
    - (more under development)
  - Flexible N to M mapping of flow to service cores.
- ACL worker
  - classify flows and send them to dedicated rings.
- RX worker
  - pull packet into ring buffers
- RX queue
  - NIC dma packets into memory
  - RSS (Receive Side Scaling) applied



Flow Worker (Service Cores)

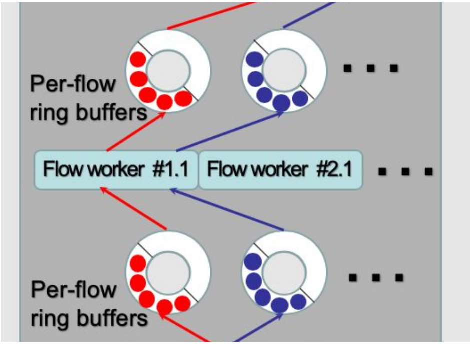


```
/**
 * This is sampling worker for service-core function.
 * A sampling rate is defined for the flow processed by this worker.
*/
static int sampling_worker(void *args)
{
    service_params *params = (service_params*) args;
    unsigned nb_in, i;
    unsigned nb_out= 0;
    const unsigned int flow_burst_size = params->flow_burst_size;
    struct rte_ring *input_ring = params->input_rings[params->ring_id];
    struct rte_ring *output_ring = params->output_rings[params->ring_id];

    // dequeue packets from the ring
    struct rte_mbuf* pkts[flow_burst_size];
        nb_in = rte_ring_dequeue_burst(input_ring, (void*) pkts, flow_burst_size, NULL);

    if(likely(nb_in > 0)) {
        params->stats.in += nb_in;

        for(i = 0; i < nb_in; i ++){
            if(params->sampling_counter == 0){
                rte_ring_enqueue(output_ring,pkts[i]);
                nb_out ++;
            }
            else{
                rte_pktmbuf_free(pkts[i]);
            }
            params->sampling_counter = (params->sampling_counter + 1) % params->sampling_rate;
        }
        params->stats.out += nb_out;
    }

    return 0;
}
```

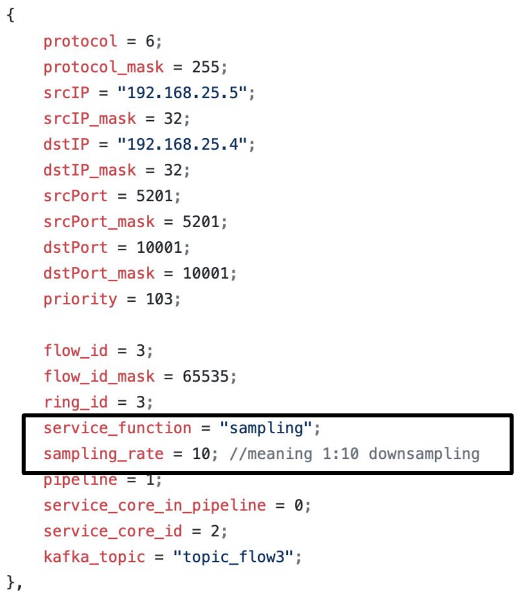
Read from input queue

Write to output queue

Drop packet

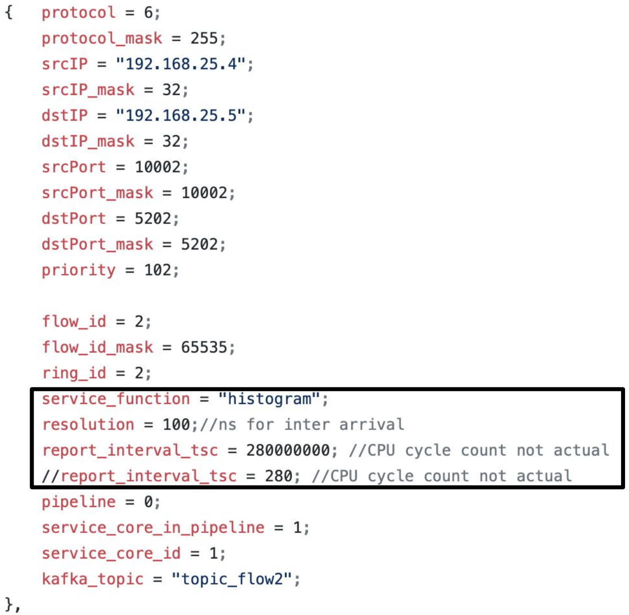


Fastcapa-ng Runtime Configuration



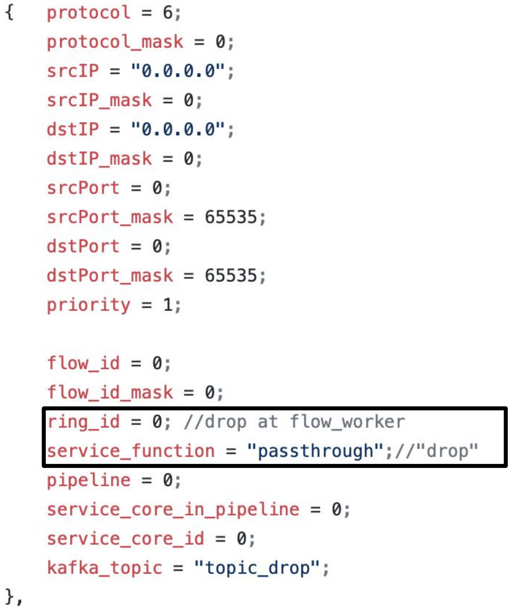
```
{
    protocol = 6;
    protocol_mask = 255;
    srcIP = "192.168.25.5";
    srcIP_mask = 32;
    dstIP = "192.168.25.4";
    dstIP_mask = 32;
    srcPort = 5201;
    srcPort_mask = 5201;
    dstPort = 10001;
    dstPort_mask = 10001;
    priority = 103;

    flow_id = 3;
    flow_id_mask = 65535;
    ring_id = 3;
    service_function = "sampling";
    sampling_rate = 10; //meaning 1:10 downsampling
    pipeline = 1;
    service_core_in_pipeline = 0;
    service_core_id = 2;
    kafka_topic = "topic_flow3";
},
```

Sampling



```
{   protocol = 6;
    protocol_mask = 255;
    srcIP = "192.168.25.4";
    srcIP_mask = 32;
    dstIP = "192.168.25.5";
    dstIP_mask = 32;
    srcPort = 10002;
    srcPort_mask = 10002;
    dstPort = 5202;
    dstPort_mask = 5202;
    priority = 102;

    flow_id = 2;
    flow_id_mask = 65535;
    ring_id = 2;
    service_function = "histogram";
    resolution = 100;//ns for inter arrival
    report_interval_tsc = 280000000; //CPU cycle count not actual
    //report_interval_tsc = 280; //CPU cycle count not actual
    pipeline = 0;
    service_core_in_pipeline = 1;
    service_core_id = 1;
    kafka_topic = "topic_flow2";
},
```

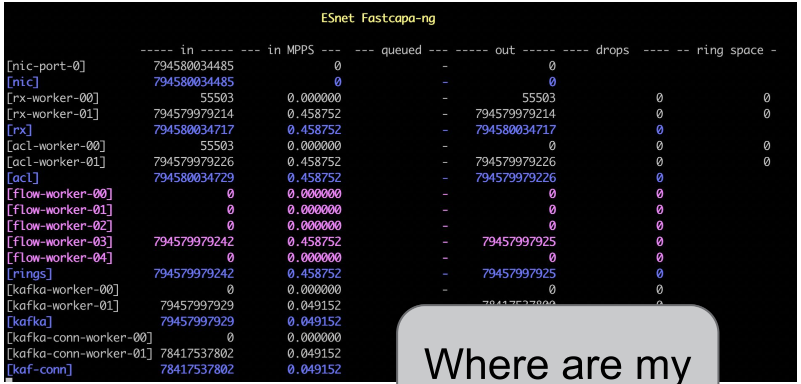
Histogram



```
{   protocol = 6;
    protocol_mask = 0;
    srcIP = "0.0.0.0";
    srcIP_mask = 0;
    dstIP = "0.0.0.0";
    dstIP_mask = 0;
    srcPort = 0;
    srcPort_mask = 65535;
    dstPort = 0;
    dstPort_mask = 65535;
    priority = 1;

    flow_id = 0;
    flow_id_mask = 0;
    ring_id = 0; //drop at flow_worker
    service_function = "passthrough";//"drop"
    pipeline = 0;
    service_core_in_pipeline = 0;
    service_core_id = 0;
    kafka_topic = "topic_drop";
},
```

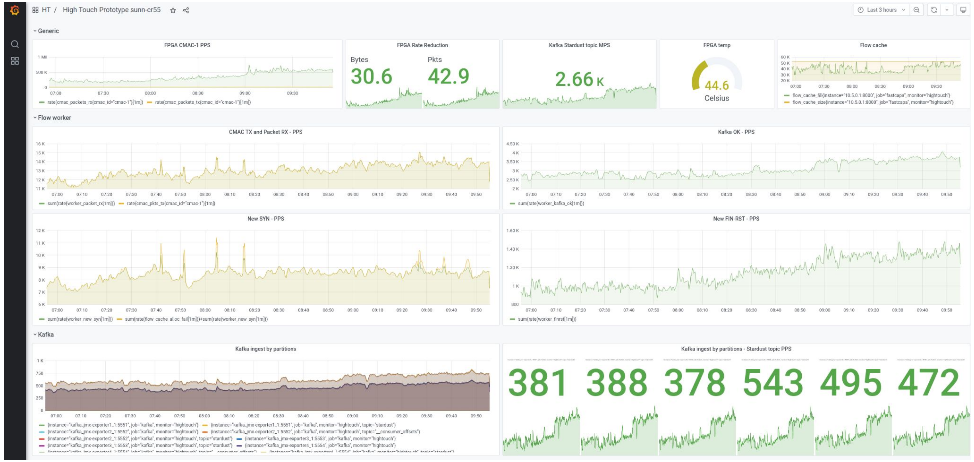
Filter

Fastcapa-ng Runtime Statistics



```
                              ESnet Fastcapa-ng

                  ----- in ----- --- in MPPS ---  --- queued --- ----- out ----- ---- drops  ---- -- ring space -
[nic-port-0]          794580034485               0              -               0
[nic]                 794580034485               0              -               0
[rx-worker-00]               55503        0.000000              -           55503             0                0
[rx-worker-01]        794579979214        0.458752              -    794579979214             0                0
[rx]                  794580034717        0.458752              -    794580034717             0
[acl-worker-00]              55503        0.000000              -               0             0                0
[acl-worker-01]       794579979226        0.458752              -    794579979226             0                0
[acl]                 794580034729        0.458752              -    794579979226             0
[flow-worker-00]                 0        0.000000              -               0             0
[flow-worker-01]                 0        0.000000              -               0             0
[flow-worker-02]                 0        0.000000              -               0             0
[flow-worker-03]      794579979242        0.458752              -     79457997925             0
[flow-worker-04]                 0        0.000000              -               0             0
[rings]               794579979242        0.458752              -     79457997925             0
[kafka-worker-00]                0        0.000000              -               0             0
[kafka-worker-01]      79457997929        0.049152                    78417537802             0
[kafka]                79457997929        0.049152
[kafka-conn-worker-00]           0        0.000000
[kafka-conn-worker-01] 78417537802        0.049152
[kaf-conn]             78417537802        0.049152
```

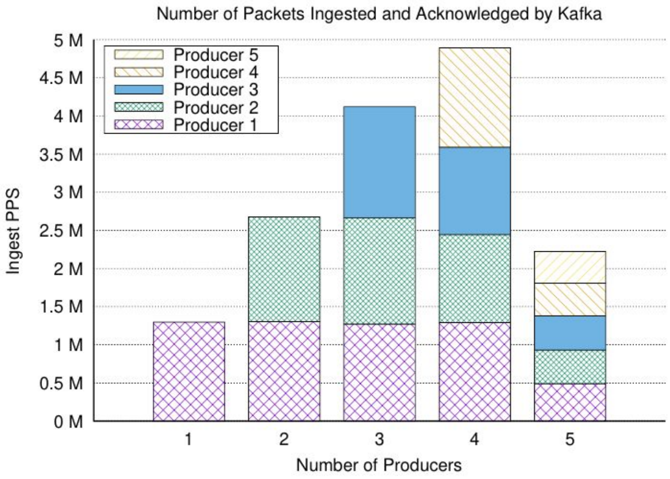
Where are my packets?



Fastcapa-ng pipeline statistics
Also in Grafana via Prometheus

Kafka Performance

- ***Apache Kafka***: open-source distributed stream platform.
- Docker-compose for a single server:
  - bitnami/kafka (x6), bitnami/zookeeper (x3)
  - bitnami/jmx-exporter
  - prom/prometheus
  - grafana/grafana
- 5M messages per second Kafka ingest performance demonstrated on single server.
- Possible bottlenecks to go higher:
  - Librdkafka C client (inside Fastcapa-ng)
  - Docker proxy - network
  - CPU - Client and brokers share the host

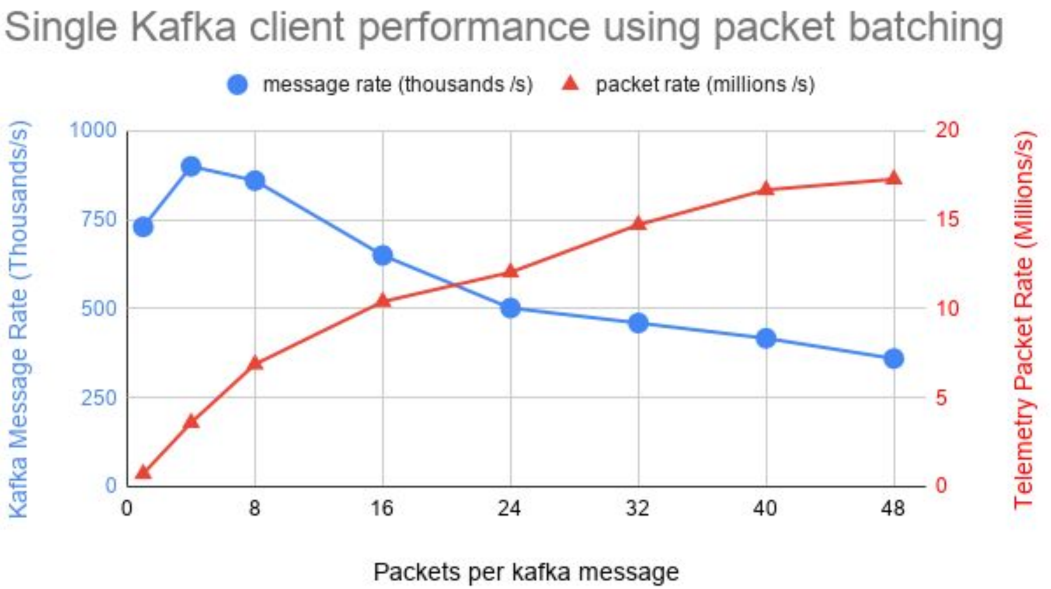


~5M messages per second ingest
untuned single server / 6 broker / parallel producers
Kafka Benchmark tool, 64K message batches

Fastcapa’s Kafka - going over 15M PPS

On top of **message batching** (handled by librdkafka), we need **packet batching** (handled by Fastcapa / client application).

That means that one Kafka message contains multiple telemetry packets. Client application has to unpack.



High Touch Application Programming

- High Touch Applications can be implemented using **Kafka Streams** - an easy way to program real-time applications on stream of data.
- Expressive, highly scalable and fault tolerant API that allows: aggregation, filtering, counting, grouping data...

APACHE kafka®
A distributed streaming platform

```
int THRES = 10;
KTable<Windowed<String>, Long> SYNcounts = stream
      .filter((k, telemetry) -> telemetry.isSYN())
      .groupBy((k, telemetry) -> telemetry.getIPDstAddr())
      .windowedBy(TimeWindows.of(Duration.ofSeconds(5)))
      .count(Materialized.with(String(), Long()))
      .filter((key, value) -> value > THRES);
SYNcounts.toStream().to("syn-attacks");
```

Example: High Touch SYN Flood Detection

ESnet

Conclusion

- We are processing millions of telemetry messages per second
- Data ingest is handled by **Fastcapa-ng**, an ESnet DPDK + Kafka project
  - Multi-stage, multi-pipeline architecture with easy configurability
  - Executes stateful functions: sampling, histogram creation, etc.
  - We can push 15M telemetry messages to Kafka with a single server
- Kafka streams: high-level application programming on telemetry streams

We are working on open-sourcing fastcapa-ng: targeting Fall 2021.

ESnet

Questions…

richard@es.net