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
Increased 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.
High-Touch Services

• High-precision, real-time visibility into network traffic
  – Process every packet of interest in real-time
  – Accurate, precision timing (ns precision / accuracy)
  – Software-defined functionality
  – Programmatically deployable and customizable

• In contrast to “low touch” services
  – Fixed function services such as IP packet routing, basic statistics
  – Optimized for speed and low cost, but not flexible

• Technology enablers
  – Software-defined networking
  – Programmable network dataplane hardware with accurate timestamps
  – High-speed packet processing libraries (DPDK, etc.)
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.
7. Service - Reads data from the Platform and performs real-time analysis as well as inserts selected telemetry data into database.
What Programmable “High Touch” Hardware to Use?

- There are a variety of programmable network devices available today. ESnet was looking for the following:
  - 100Gbit/s port speed and roadmap for higher speeds
  - Timing and performance guarantees
  - Easy programming (P4 style)
  - Established vendor with support
- We are currently prototyping using Xilinx FPGAs
  - Alveo U280: 2x100G port, 8GB HBM2 memory (3.2 Tbps bandwidth), 32GB DDR4, 1.2M logic cells
Telemetry Producers

Copy of original packet of a TCP flow

Programmable Data Plane
Transforms packets

High-Touch Telemetry Packet

Payload removed

High-Touch Telemetry Record (approximate) ~100 bytes

<table>
<thead>
<tr>
<th>Packet size</th>
<th>Rate</th>
<th>Telemetry PPS</th>
<th>Telemetry Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1500B</td>
<td>10Gb/s</td>
<td>812K</td>
<td>1,079Mb/s</td>
</tr>
<tr>
<td>1500B</td>
<td>100Gb/s</td>
<td>8,127K</td>
<td>10,790Mb/s</td>
</tr>
<tr>
<td>9000B</td>
<td>10Gb/s</td>
<td>138K</td>
<td>183Mb/s</td>
</tr>
<tr>
<td>9000B</td>
<td>100Gb/s</td>
<td>1,383K</td>
<td>1,833Mb/s</td>
</tr>
</tbody>
</table>

Programmable Data Plane
Transforms packets

High-Touch Telemetry Packet

Payload removed

High-Touch Telemetry Record (approximate) ~100 bytes
High Touch Collector Processing

TCP rate monitor

Finger of blame

Packet Loss

Kafka cluster

Flow #1

Flow #2

TX Process #1

TX Process #2

TX Process #n

RX Process #1

RX Process #2

RX Process #n

RX queue #1

RX queue #2

RX queue #n

Packet ring

Fastcapa runs on multiple cores (each RX and TX process is one core)

Topics deleted after 24hr automatically

Any DPDK-enabled card

Mellanox 100G NIC

Distributor / RSS

Telemetry packets
ESnet Fastcapa-ng Internals

- **RX queue:**
  - NIC dma packets into memory
  - RSS (Receive Side Scaling) applied

- **RX worker:**
  - pull packet into ring buffers

- **ACL worker:**
  - classify flows and send them to dedicated rings.

- **Flow worker (service cores):**
  - process flows using different function: passthrough, sampling, histogram, etc.
  - Flexible N to M mapping of flow to service cores.

- **Kafka worker:**
  - Combine multiple telemetry packets into large kafka messages.

- **Dedicated Kafka connection:**
  - maintain TCP connection, message compression task.
Kafka setup and benchmarking

- Docker-compose: bitnami/kafka, JMX Exporter, Prometheus, Grafana
- 6 brokers on a single server
- Possible bottlenecks:
  - Librdkafka C client (inside Fastcapa-ng)
  - Docker proxy - network
  - CPU - Fastcapa and Kafka brokers are on the same host

~5M PPS ingest
untuned single server / 6 broker
Use Case #1
TCP Rate and Retransmission Tracking

Motivation:

- Monitoring TCP rate in a per-packet basis
  - Find peaks, abnormal rate in the shortest possible time
- Provide a tool for network operations and engineering
- Finding packet retransmissions as they happen
  - Is there an issue at ESnet or at the source or destination networks?
Visualizing Real-Time Telemetry Data

- We can plot metrics for every packet in a flow using InfluxDB / Grafana

A sample PerfSonar 10Gbit/s test measured by High Touch Rate Monitor

Packet retransmissions for flows indicating an issue
Per-packet rates reaching 10 Gbit/s (line rate of the sender)

1 Gbp/s average flow rate (100 pkt window)

Note: Average rate is calculated using a time-weighted average of per-packet rates.
1 Gbps iPerf flow - 1% packet drop

Per-packet rates maximum: 1 Gb/s

5 Mb/s average flow rate (100 pkt window)

Note: only 23 packets were dropped all together, taking bandwidth down to 5 Mb/s from 1 Gb/s.
1 Gbps iPerf Flow - 1% Packet Drop

The difference between the lines represents the performance of the data transfer, convergence is good, divergence is bad.

Flow throughput drops when a retransmission happens.
Use Case #2
TCP Congestion Control Identification

- Motivation:
  - Some flows are unable to utilize the available bandwidth
  - TCP flows can take more of their fair-share

- Discovering misconfigured flows (e.g., window parameters, congestion control) will allow us:
  - Tune the configuration of Data Transfer Nodes
  - Notifying our sites automatically (periodic reports) on suboptimal configuration
  - Guide fair usage of the network (“is equal bandwidth share” fair?)
BBR vs Cubic - Point Rates

Difference in fingerprint can be identified visually.

2 millions of data points shown (around 600.000 points a second generated)
BBR vs Cubic - Inter-Arrival Time Histogram

Difference in behaviour can be identified visually

BBR: inter-packet timing is more widespread than other congestion control algorithms.
Machine Learning on Aggregated Data

- Aggregated data - such as histograms can be used to tell apart congestion control (CC) used by TCP flows
- We are using data plane histograms of inter-arrival times per flow (2000 packets per histogram)
- ML algorithms explored: Convolutional Neural Networks, k-Nearest Neighbors

Input: per-flow histograms of Inter-Arrival Time (IAT)

Machine Learning (trained with labeled data)

Inference in less than 1 ms in all cases

Flow 1 CC: (unknown)
Flow 2 CC: (unknown)

Flow 1 CC: most likely TCP BBR
Flow 2 CC: most likely TCP RENO

More details, dataplane architecture, ML code in:

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...

```java
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**
High Touch Services

DEMO
ESnet FPGA Block Diagram - Present and Future

- Large Tables
- 16 Gbytes DDR4
- Programmable Per flow / Per packet statistics
  - Flow Counters
  - Real Time Histogram
  - Real Time Histogram
- Packet Distribution Chain
  - In order 150 Mpps
  - P4 Configurable add / drop / redirect
- User Defined HW Processor
- Eth
- Programable Ethernet HDR
- Programable Packet Forwarding / Editing
  - Stateful / In order / Multi-threaded
- Programable QoS
  - Shaping / Priority
  - Buffering / Flow Control
- 100G Stateful NPU
- 100G Stateful NPU
- General Purpose NPU
- General Purpose NPU
- 8 Gbytes HBM (1 Tbit / s) R/W BW
- Flow ID pkt size
- Flow ID pkt size
- Flow ID Timestamp
- Flow ID Ethernet HDR
- Flow ID IPv4 HDR
- Flow ID MPLS HDR
- Flow ID Egress HDR

[Diagram of ESnet FPGA Block Diagram]
3 models for using FPGAs

**Easy**
Install a copy of ESnet’s telemetry solution. Zero FPGA development. Customize Splunk / Kentik / Grafana / ELK etc..

**Intermediate**
Program the embedded NPUs. Zero FPGA development. FPGA bit file provided. But packet editing is programmable like an SDN switch.

**Advanced**
Re-configure the FPGA using P4 and Verilog. User defined hardware.

- Custom drivers and applications
- Arbitrary L2-L7 Stateful Ideas
High Touch Services Timeline

- **Service Design**
  - Technical service design, experimentation with dataplanes and collector software.

- **Design Validation**
  - Evaluation of the collector software, dataplanes, scoping and prototyping.

- **Design Refinement**
  - Making the service more robust, implementing a variety of High Touch services, while enhancing scalability, fault tolerance, security, orchestration.

- **Pre-Pilot**
  - Deploying pilot service, inspecting traffic on selected links (low-traffic customer, high-traffic customer, Splunk integration).

- **Pre-Deployment**
  - Finalizing a complete solution: edge hosts, programmable hardware, services and their orchestration.

- **Deployment**
  - Deploying High Touch services.

Timeline:
- 2019 Q3
- 2020 Q1
- 2020 Q4
- 2021 Q1
- 2022 Q1
- 2022 Q3
Questions…

{bmah, richard, yak}@es.net