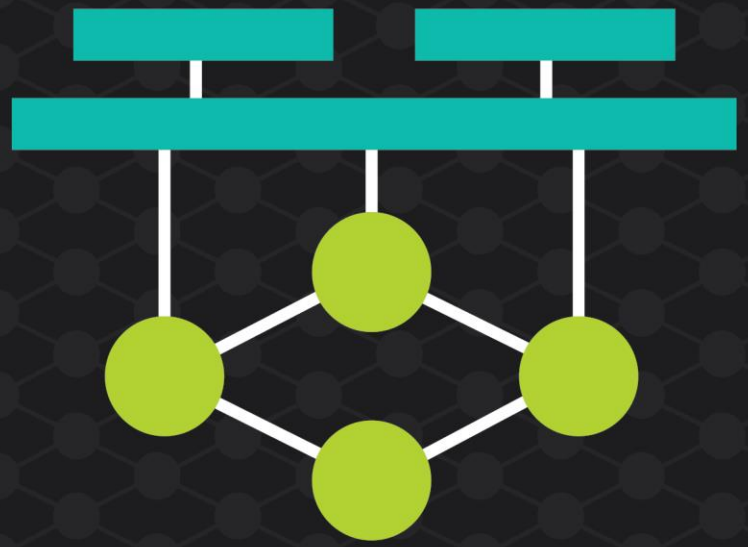


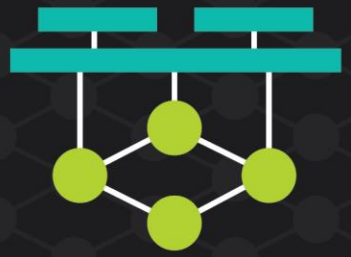
ons

EUROPE

OPEN NETWORKING //
Enabling Collaborative
Development & Innovation

Hosted By

THE LINUX FOUNDATION | LF NETWORKING



ons
EUROPE
OPEN NETWORKING //
Enabling Collaborative
Development & Innovation

Network Traffic Analytics through Accelerator and SmartNIC at Edge

Mrittika Ganguli, PE, Network Architect, Intel
Yuming Ma, Cloud Solution Architect, Intel
Rita Chattopadhyay, Data Analytics Architect, Intel
Priya Autee, Sr Engineer, Intel
Rahul Shah, Sr Engineer, Intel

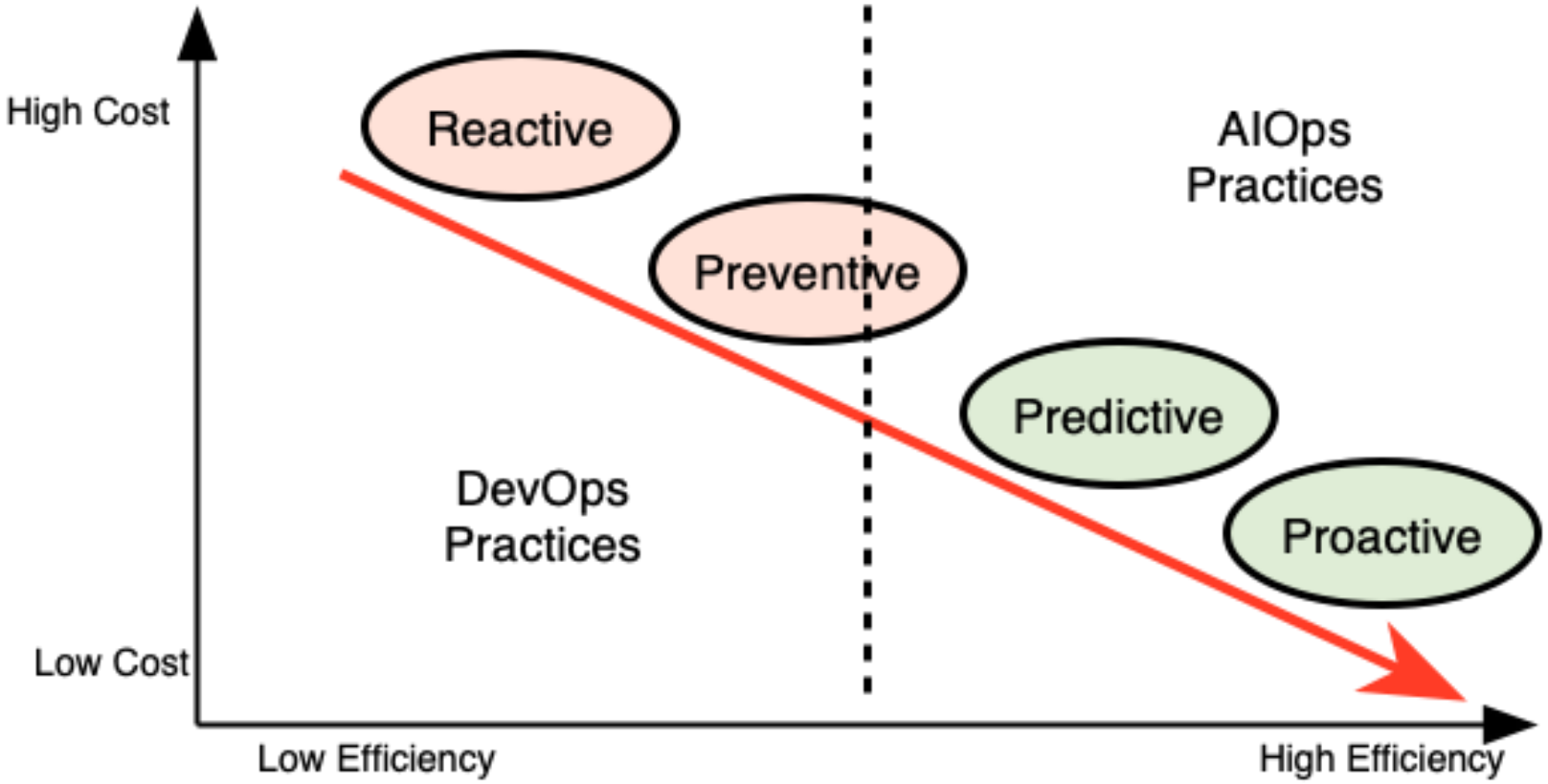
Hosted By
THE LINUX FOUNDATION | LF NETWORKING

Agenda

- Why Offload Data Center Analytics?
- Big Picture & Use Cases
- How to Do it? Algorithms and FPGA design
  - K-Means
  - Random Forest Analysis
- Host traffic flow and software stack configurations
- Data and Graphs
- What's Next?

ons EUROPE

Towards Intelligent Data Center Management



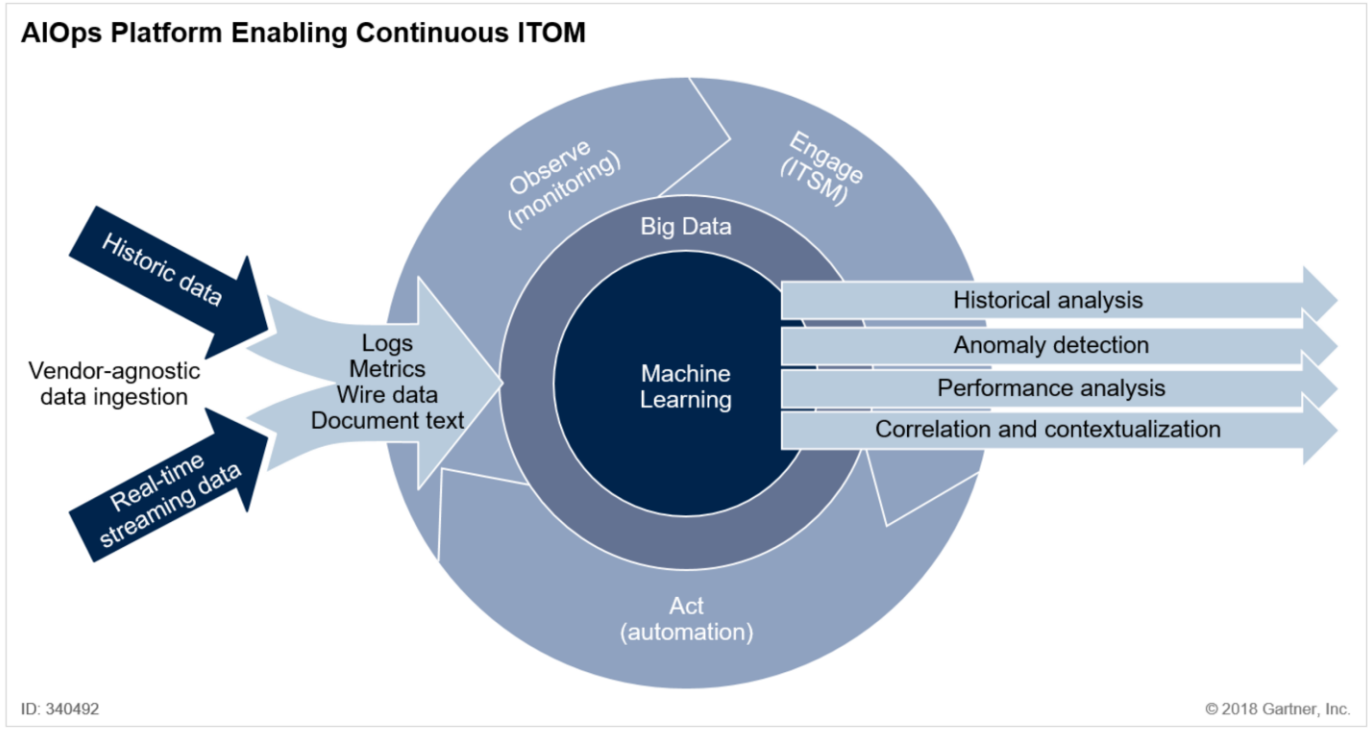
- Anomaly detection
- Capacity management
- Workload QoS
- Infrastructure optimization

- Intelligent Data Center Management

ons EUROPE

Data Center Analytics at Edge & Cloud

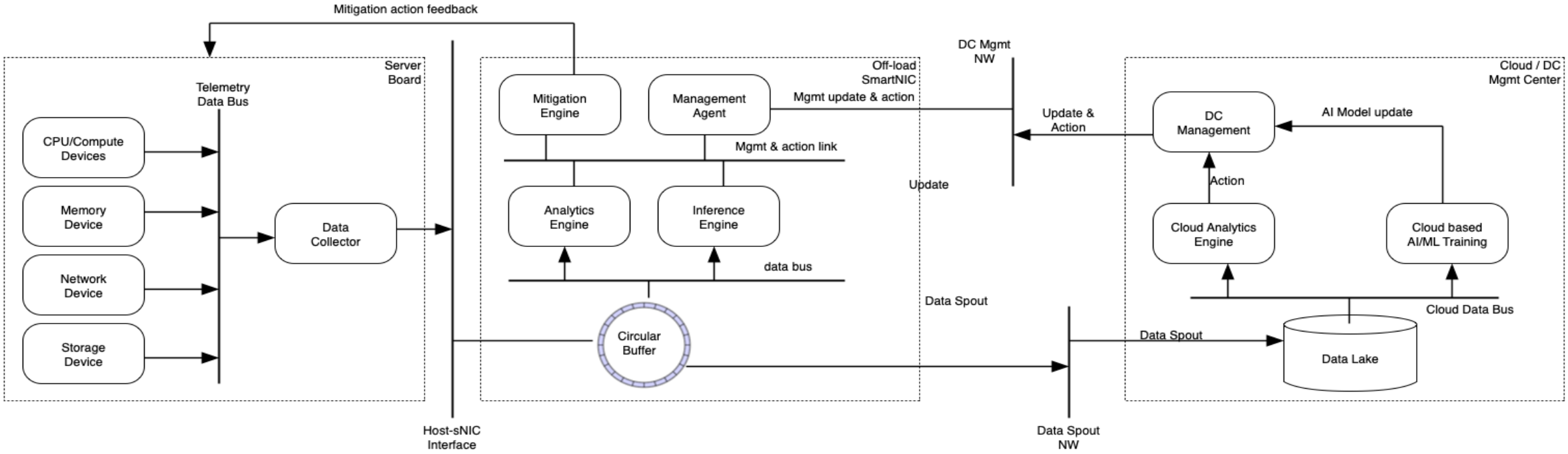
- DC Data: growing volume, complexity, variety, and velocity:
  - Infra telemetry data grows 2~3 fold/year
  - Deep data at silicon and device, middle layer data at platform/host, application data
  - High speed data rate and traffic
- DC Mgmt & Ops: real-time, global, auto, and proactive
  - Real-time with streaming data
  - Global perspective at cloud with historical data
  - Autonomous action for self-driving ops
  - ML with predictive analytics for proactive management
- Technology : accelerator and AI/ML
  - SmartNIC / Accelerator :
  - AL/ML for predictive analytics



Source: Gartner (November 2018)

Infrastructure Assets

Data Center Analytics Framework



- Telemetry data from silicon, device, host and application

Analytics and inference with incoming data at edge by SmartNIC/accelerator

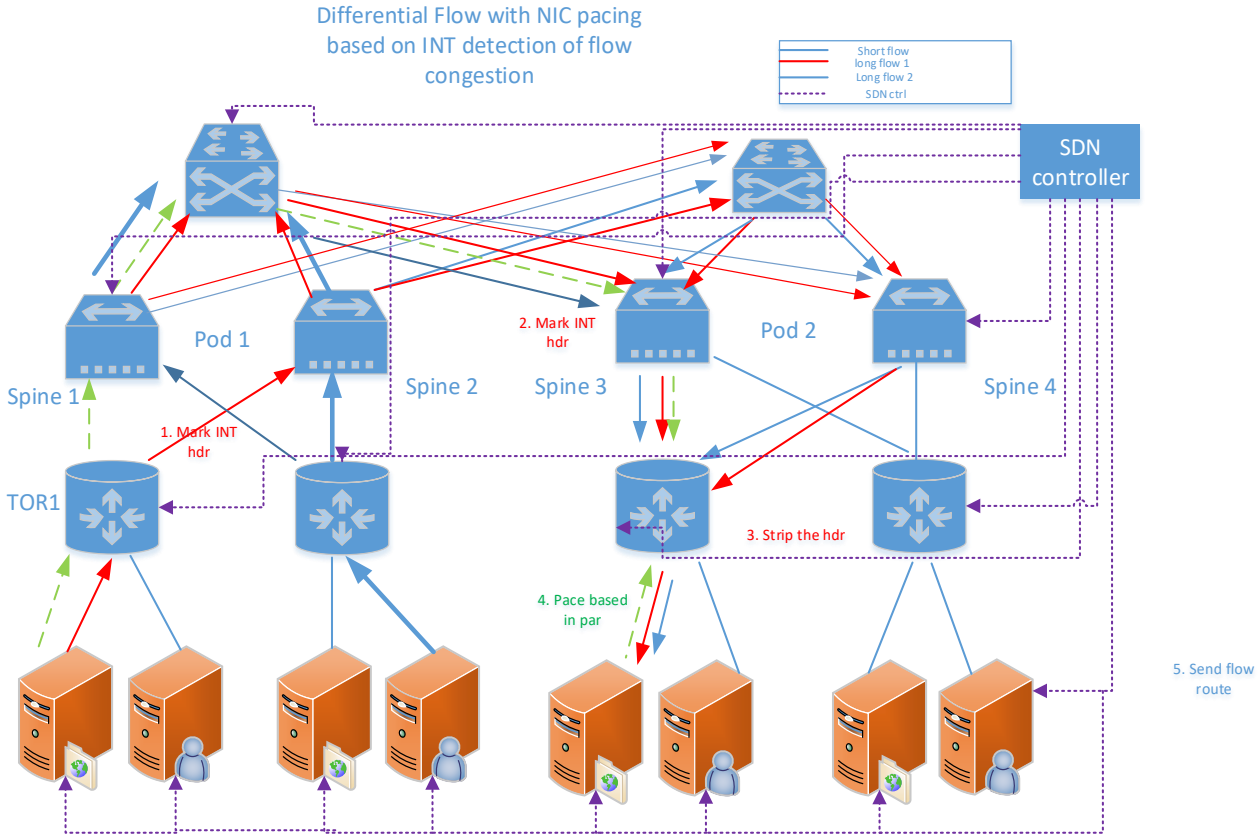
Cloud for global analytics and AI model. training

Network System Architecture Concepts

- **Scale** is changing all aspect of Network and application and DC efficiencies
  - Modern Network architecture changes in all aspects when addressing Scale!
- **Fine grained** network control at the Edge
- **Host Centric** – including Routing, Scheduling, Load-Balancing, Multi-Pathing, Security, Orchestration interaction, Application ⇔ Network interaction and host data plane
  - Provide platform hooks for emerging WL models e.g. uServices, FaaS, SFC etc.
- **Switch** - Coarse level traffic control, based on preset policy with escalation to NOS and Controller for non-real time events
- Extensive, **intelligent** and enlightened use of **Telemetry** from the Edge and Network switches to scalable collectors
  - Goal to provide the data as required for specific decision to be made
  - Connect to industry analytics
- Telemetry ⇨ Insight - **Closed loop Analytics** turning Data into Insights and feed it back to E2E entities to enable independent decision by entities, and wit localized impact

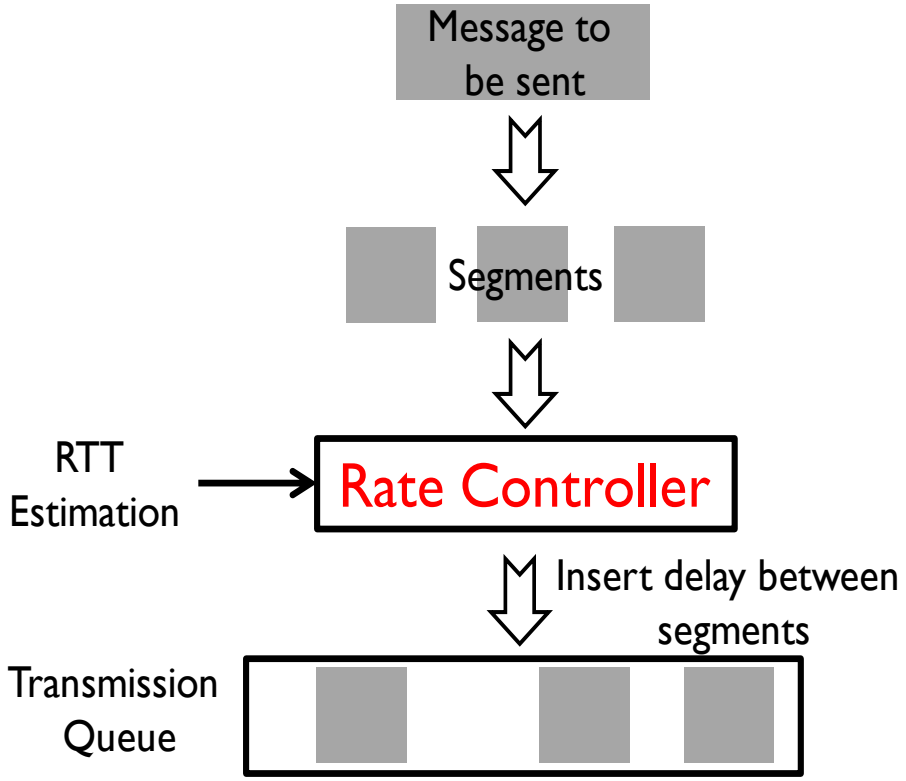
Usage: Congestion and Burst QOS with switch, NIC, CPU

- Preset policies
  - Allocating limited unsolicited budget to NICs (e.g. one RTT/NIC), solicited based on grants
  - Setting switch dynamic buffer allocation and priority per app types
  - Based on type of application – from orchestration, host and receiver (especially for RPC)
  - Mice/Elephant and solicited/unsolicited identification – NIC and/or switch
  - Route and LB ownership per app flows
- Telemetry
  - Telemetry streaming fed to ML (e.g. on Inference Engine or UNC) to analyze microburst and congestion and adjust switch buffer allocation and priority
  - Marking packets in INT headers to measure buffer usage (indicating burst)
- Pacing in NIC (multiple options)
  - Default: Linux HTB, FQ, QDISC
  - Timing wheel based scheduling – Carousel
  - Under evaluation
    - CPU SQM used for per flow QOS per queue
    - NIC queues scheduling for large number of flows OR
    - CPU core virtual queue based



Congestion control

- Algorithms to prevent that the sender overloads the network. Detection and Recovery. Congestion is detected with
  - an explicit congestion notification from a packet switch
  - packet loss: in wired networks, the main reason for packet loss is congested buffers.
- The mechanism is implemented at the sender. The sender has two parameters:
  - Congestion Window (cwnd)
  - Slow-start threshhold Value (ssthresh) Initial value is the advertised window size
- Congestion control works in two modes:
  - slow start (cwnd < ssthresh)
  - congestion avoidance (cwnd >= ssthresh)
  - Congestion avoidance phase is started if cwnd has reached the slow-start threshold value
- RTT based window adjustment:
  - Calculate the propagation delay and influence to RTT
  - Adjust cwnd based on delay

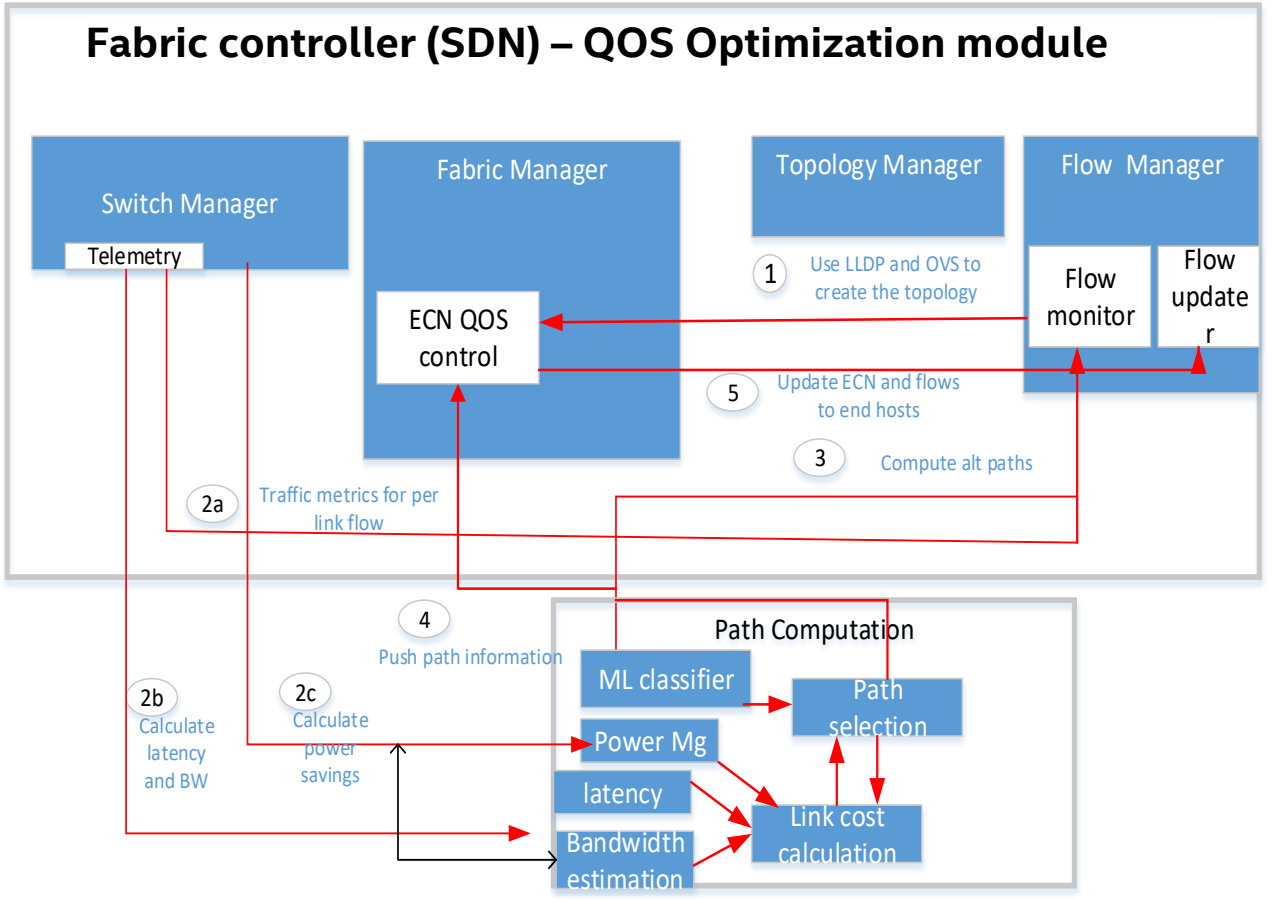


Target rate is determined by segment size and delay between segments

Ack: http://www.ece.virginia.edu/~mv/edu/ee136/Lectures/congestion-control/tcp-congestion-control.pdf

ML based control At the infrastructure Edge

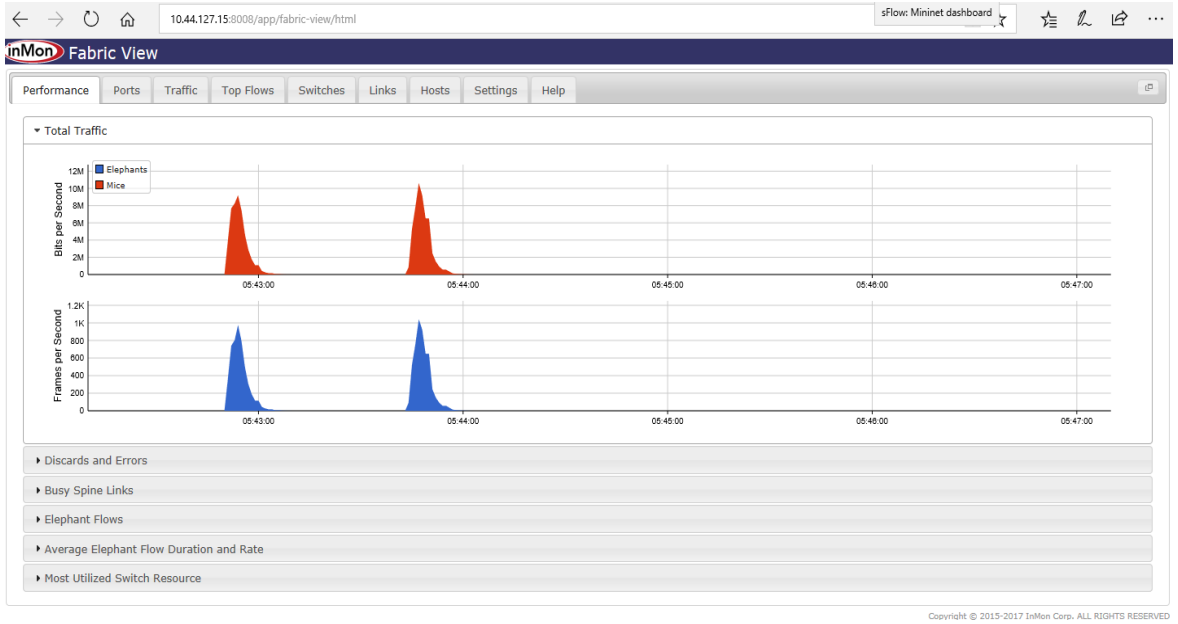
- Infrastructure edge – Infra that hosts VNFs, and other network services, application services,
- **ML based Configurable** Path computation with link status and QOS for 4 different objectives (power, latency, bandwidth) for congestion avoidance
- **Workload characterized and workload based flow management**
  - Packet level telemetry in addition to flow based Latency & perf analysis – to optimize performance
    - Enhanced end point capability discovery, more granularity of PB HW features
    - better orchestration, intelligent workload placement
    - End to end QOS



ons EUROPE

Elephant Flow detection and Packet drops

- **Calculations and packet tagging:**
- 1. Simple calculation on flow size: Flow Size = [b(t) – b(t-a)]/ a
- a = time interval set to 2 or 5 seconds. Flow size at time t and after an interval of a is calculated. If this flow size exceeds a threshold (eg 100 KB) it is marked as an elephant flow.
- Analytics based elephant flow-
- statistical data based
- K-means based
- Analysis of packet drops and factors related to it



Switch port	MAC src	MAC dst	Eth Type	VLAN	IP Sec	IP Dst	IP Port	TCP sport	TCP dport	Action
				PCB-001						

Host Workload Analysis model

1. Identify workload Type
   1. Uses Adaboost
      1. Supervised learning
      2. Ensemble method using weak classifiers.
      3. *f(metrics) -> workload_id*
      4. A workload id assigned to each workload

$$H(x) = \sum_{t=1}^{T} \beta_t \cdot h_t(x)$$

Where,
$H(x)$= Strong Classifier
$h_t(x)$= Weak or Basis Classifier or a feature

2. Identify workload phase.
   1. Current Phase Identifier
   2. Subsequent Phase Identifier
   3. Uses k-means clustering (Unsupervised learning)
   4. Allows us to use Naive-Bayesian analysis.

K-means based workload fingerprint

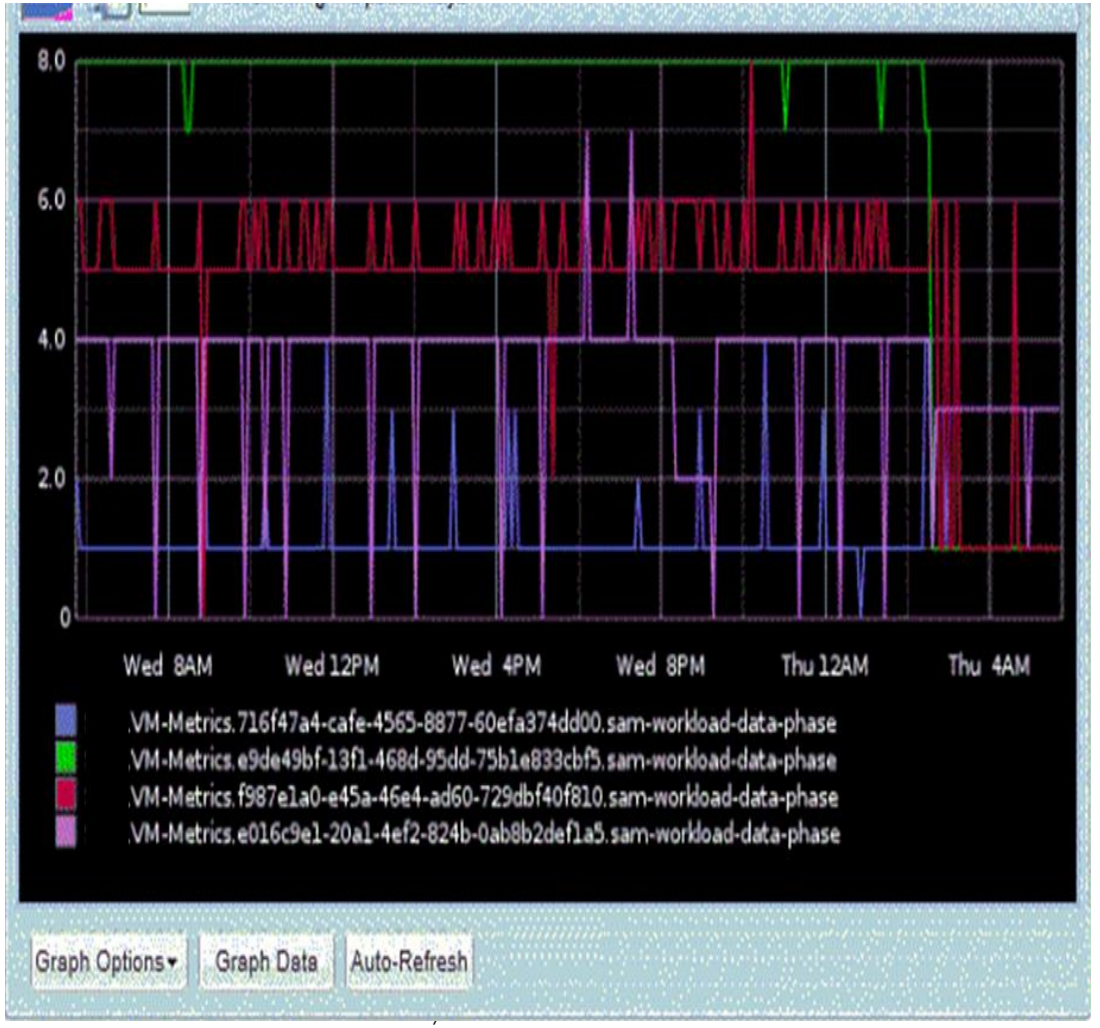
- **Fingerprint** identifies a specific pattern
  - Workload characteristics integrated through the sequence of known (or dynamically identified) phase are represented as a fingerprint.
  - Identification and detection of phases is a fundamental technique of exploiting program behavior.
- SPEC benchmarks, NFF-GO NAT, BNG and specialized workloads by usage are often used to measure the performance of a virtual infrastructure instance allocated to a CPU.
- KPI – IPC or GIPS that characterizes an active workload in a VNF.

$$GIPS = fn(F; T; C; I)$$

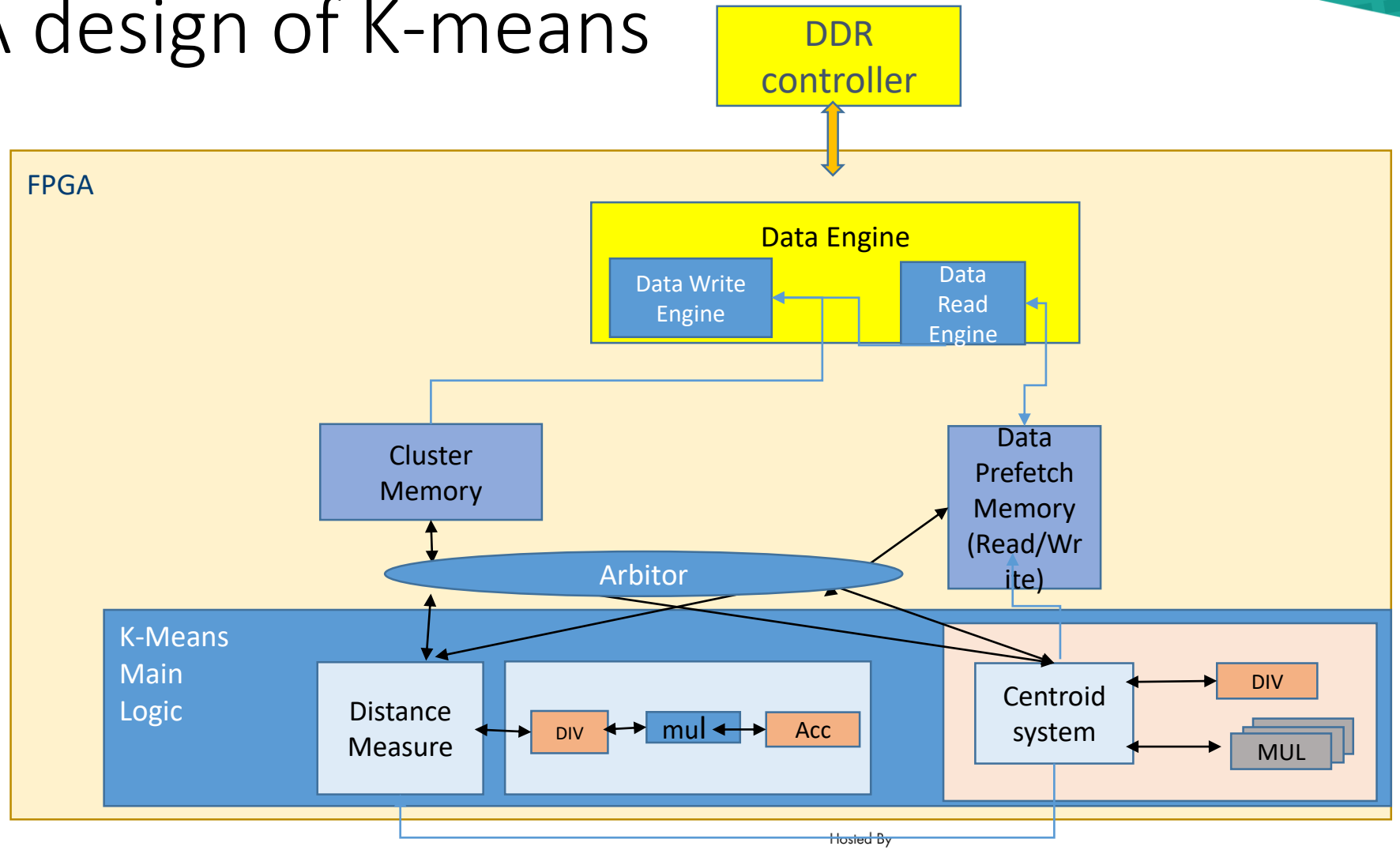
where,

F = *Frequency*, T = *Throughput*, C = *Cache Efficiency*, I = *Instruction Set Efficiency*.

- GIPS is measured, calculated and used as the basis for resource allocation and any remediation action required to guarantee the workload performance.
- Facilitates cloud orchestration and service framework to
  - proactively evaluate remedial actions through dynamic resource evaluation
  - sharing to comply with the SLOs.
  - Offload to meet the KPI

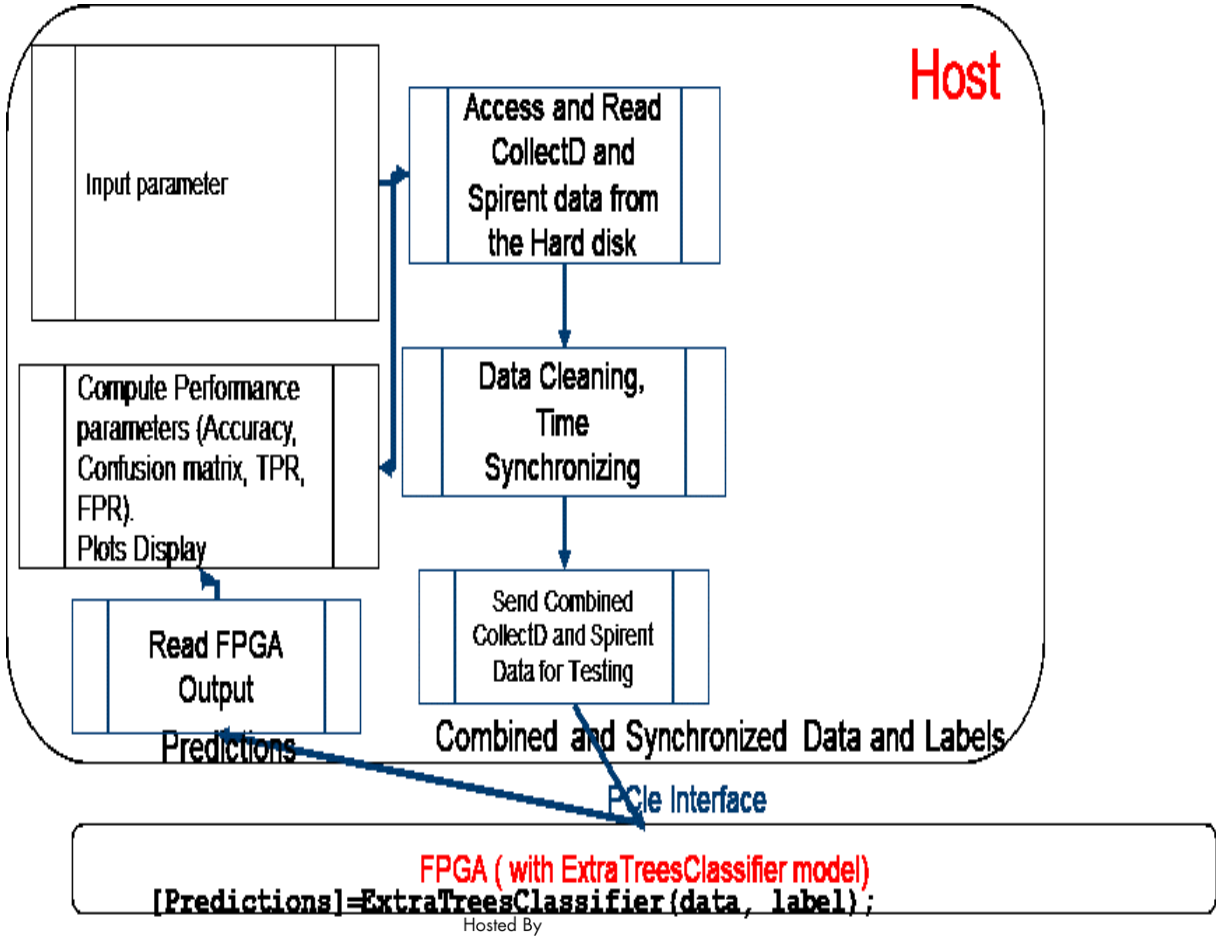


FPGA design of K-means

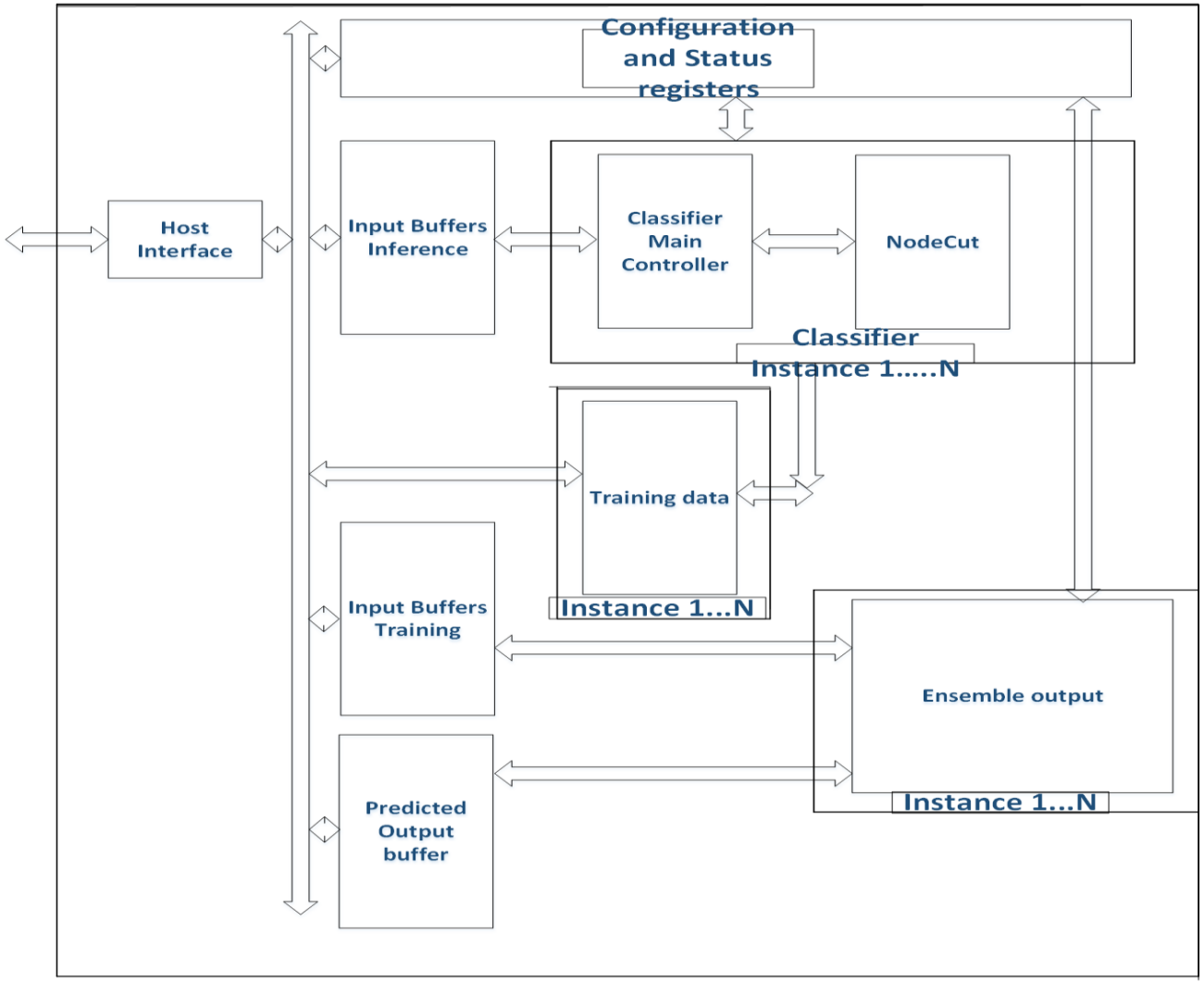


Efficient CPU Parameter Method for Packet Loss Detection

- Significant CPU and platform parameters, which can be monitored and passed through a pre-learned machine learning model to detect/predict network packet drops with very high confidence.
- Uses a feature ranking method to rank a large number telemetry data based on their correlation with network packet drop.
- Selecting just the top telemetry signals provided more than an order of magnitude (~15/400) reduction in the telemetry data.
  - Reduces the load on the devices generating the telemetry (e.g. server or switch),
  - Reduces the load on the network to transfer that data,
  - Prevents a Big Data problem (searching for the needle in the hay stack), and
  - Enables real time operation.

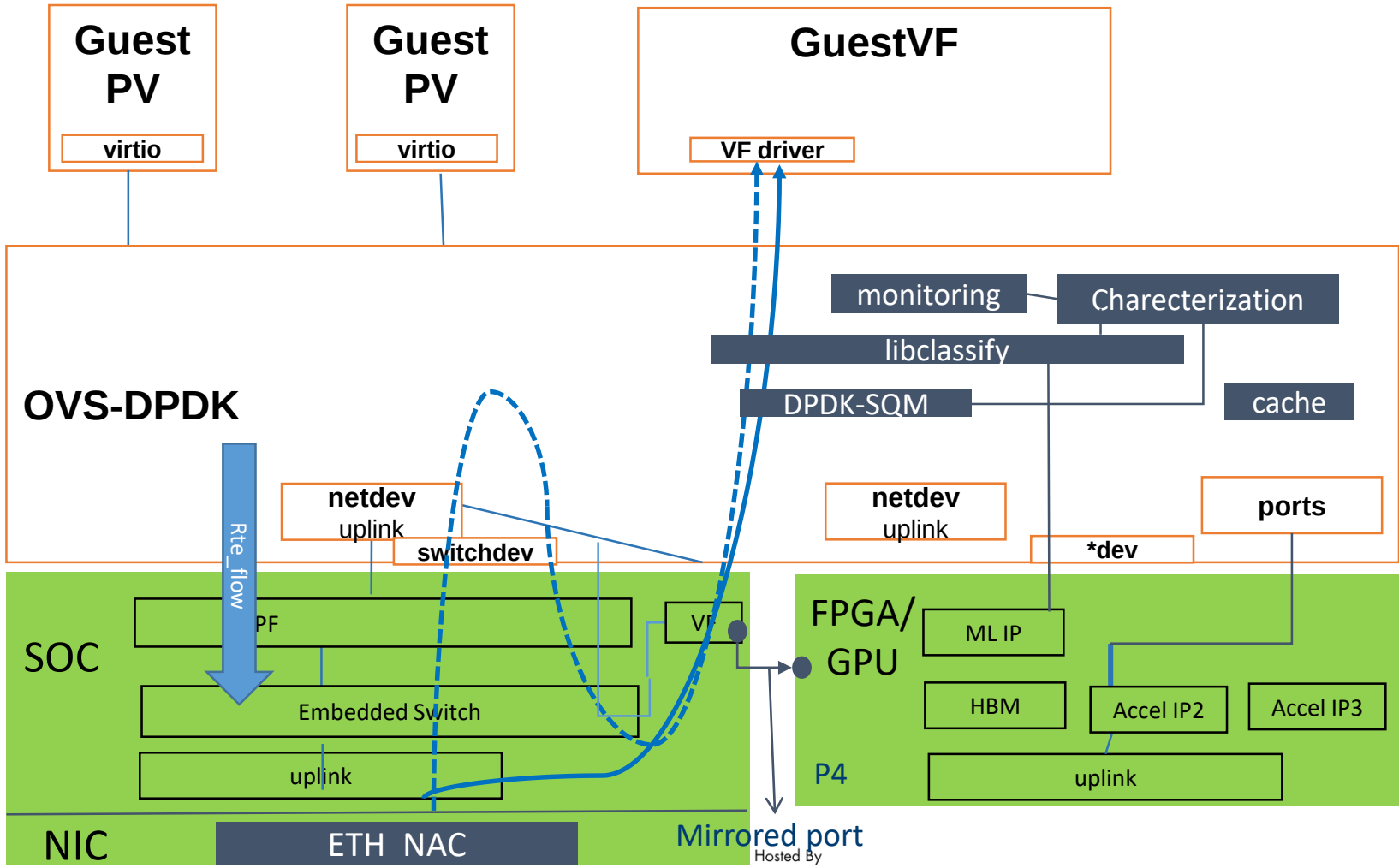


ML Network Packet Drop Architecture : Host-FPGA

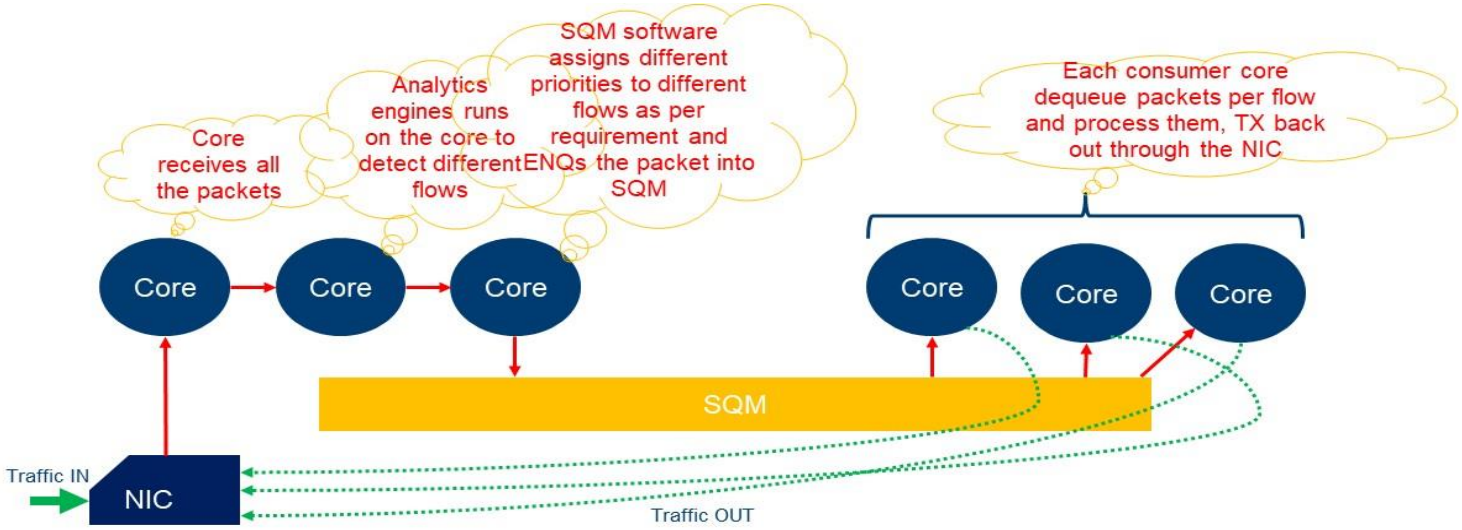


Monitoring, ML and Actions at the host

- Monitor per guest and per core metrics
- Traffic flows via mirrored port from SOC to FPGA/GPU
- Metrics collected and stored in HBM or SRAM
- ML IP activated
- Decisions stored back in memory
- Decisions read back by FW and traffic packets tagged as elephant flow or flow ids of elephant flows saved in memory
- QOS bits attached back to packet header
- Load balance large flows on cores using SQM
- Offload to IP
- Apply NIC pacing

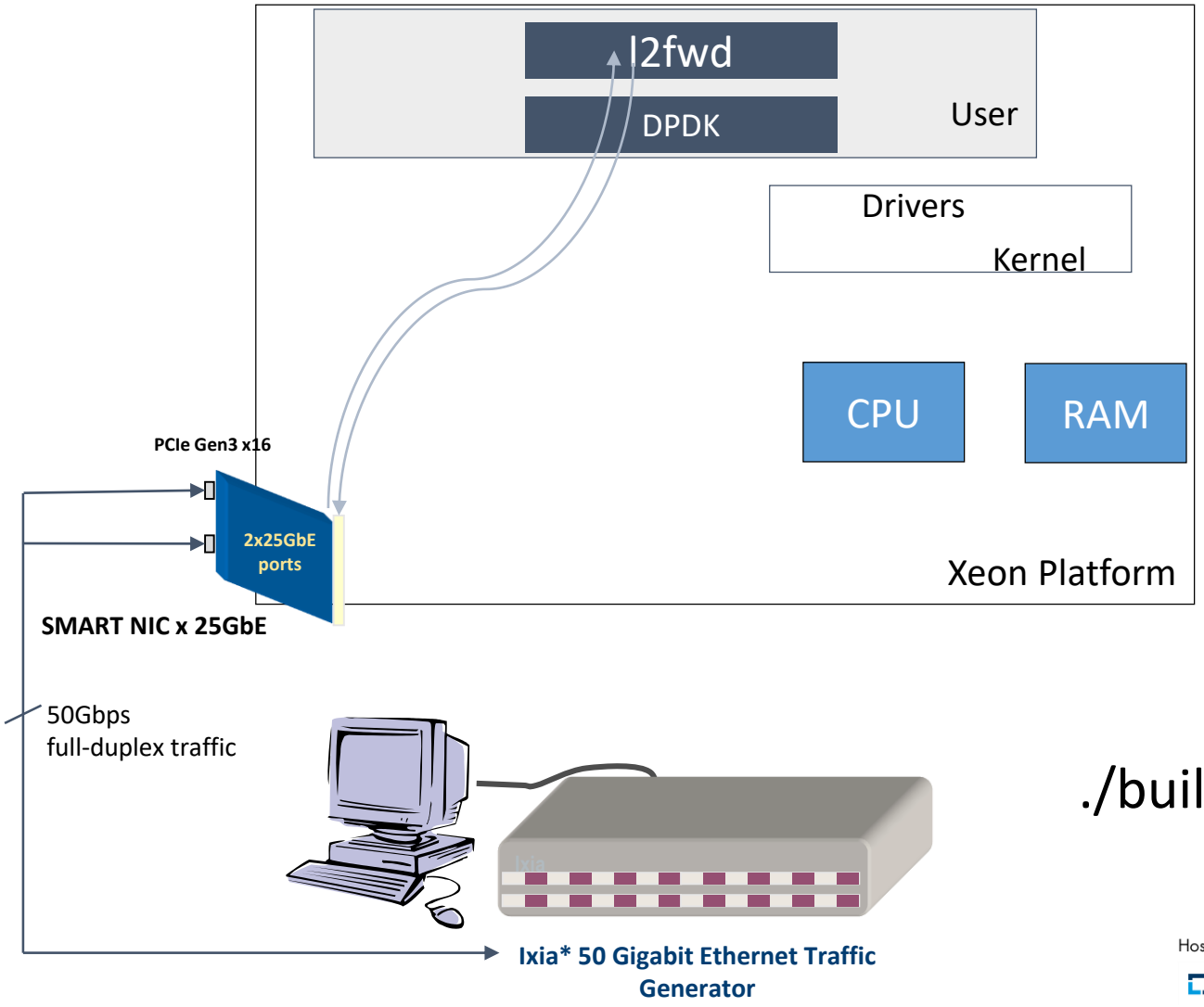


DPDK SQM for deterministic flow performance on CPU



- SQM's queue can schedule each flow on to separate core based on their flow id. It can provide load balancing when multiple flows are being received by the system
- Additionally SQM can also provide priority per Queue. So Elephant flows for instance can be assigned a higher priority vs mini flows per queue for more deterministic performance
- These flows can be detected before and software can assign appropriate priorities to the flows. This dynamic approach can help make the network more efficient using SQM

Device Under Test (DUT)

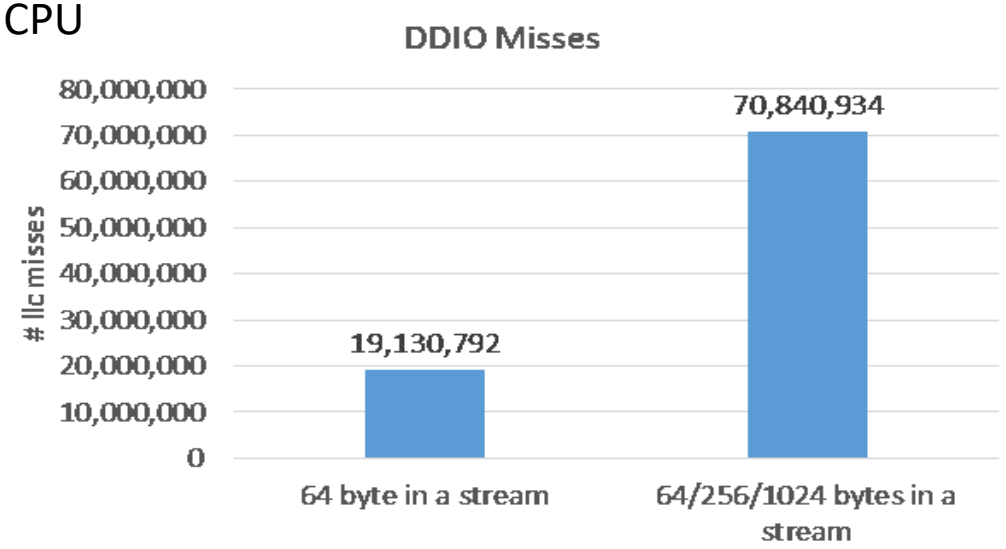


./build/l2fwd -c f -n 6 -- -p 0x3

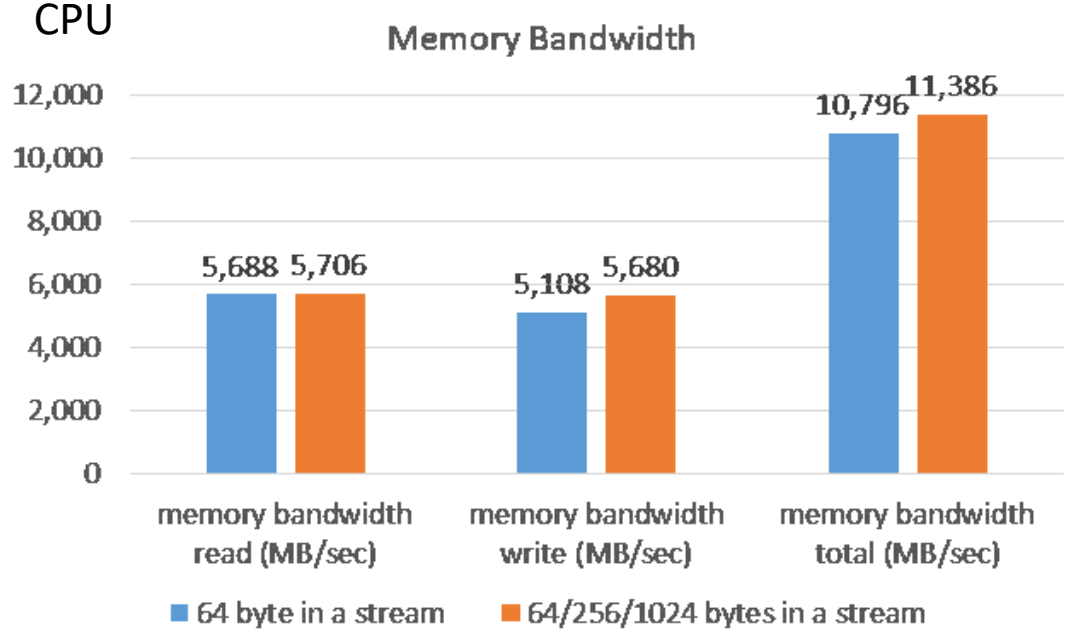
ons EUROPE

Sample Performance Data

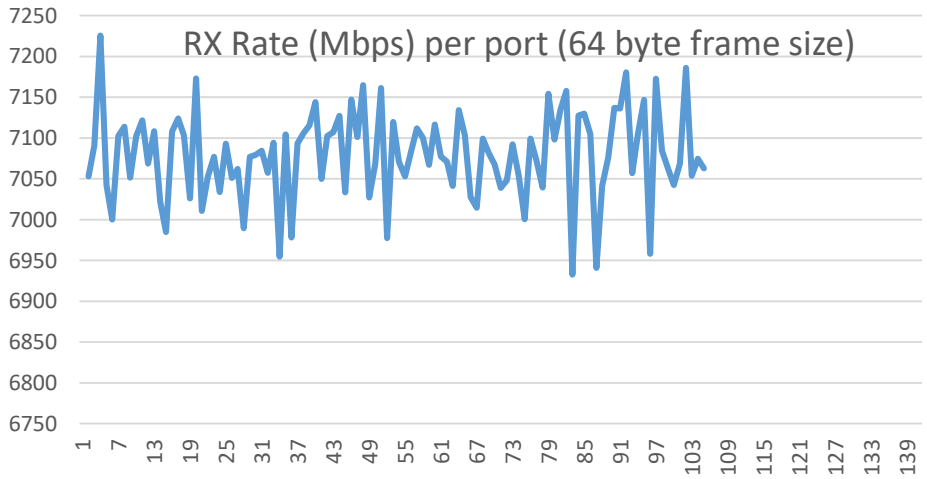
CPU



CPU



NIC



Future Work

- More at NIC :
  - What interfaces and performance of ports do we need for specific performance, e.g. OVS mirrored port has lower performance.
  - Plumbing at kernel and user mode for this usage
  - Speed of analytics at the FPGA and other algorithms
- More on Analytics & Use Cases :
  - Deep data from silicon and device
  - Global & historical data analytics, training & inference at cloud
- More on Solution :
  - Analytics, training, inference and mitigation combing edge and cloud

Next Steps

- We are looking to collaborate on the software stack for the interfaces and libraries to be exposed in the community,
- We are looking to map customer POCs by plugging in these solutions in a POC environment and refine the solution
- Email: Mrittika Ganguli (Mrittika.Ganguli@intel.com) or Yuming Ma (Yuming1.Ma@Intel.com)
- Join us for the next series of POCs
- Thank You!

References

- Analytic Technique for Optimal Workload Scheduling in Data-center using Phase Detection
  - M Ganguli, R Khanna et al, ICEAC 2015
- Autonomic Characterization of Workloads using Workload Fingerprinting
  - M Ganguli , R Khanna et al, CCEM 2014