Open-NFP Summer Webinar Series: Session 1: P4 for Custom Identification, Flow Tagging, Monitoring and Control

Nic Viljoen - Netronome

July 13, 2016

Open-NFP www.open-nfp.org

Support and grow reusable research in accelerating dataplane network functions processing

Reduce/eliminate the cost and technology barriers to research in this space

- Technologies:
  - P4, SDN, OpenFlow, Open vSwitch (OVS) offload
- Tools:
  - Discounted hardware, development tools, software, cloud access
- Community:
  - Website (www.open-nfp.org): learning & training materials, active Google group https://groups.google.com/d/forum/open-nfp, open project descriptions, code repository
- Learning/Education/Research support:
  - Summer seminar series, P4DevCon conference, Tutorials (P4 Developer Day), research proposal support for proposals to the NSF, state agencies

Summer seminar series to further progress to our objective. Present applied reusable research.

P4DevCon Attendees/Open-NFP Projects*

OpenNFP

Universities

- KU LEUVEN
- THE UNIVERSITY OF ARIZONA
- Carnegie Mellon University
- APL JOHNS HOPKINS APPLIED PHYSICS LABORATORY
- UCDAVIS UNIVERSITY OF CALIFORNIA
- UNIVERSITY OF CALIFORNIA UC RIVERSIDE
- Università della Svizzera italiana
- University of Massachusetts UMASS Lowell
- MIT
- UNIVERSITÀ DI PISA
- ICSI INTERNATIONAL COMPUTER SCIENCE INSTITUTE
- UQÀM
- USC University of Southern California
- BEIJING INSTITUTE OF TECHNOLOGY 1940
- 411 × 411 - en.wikipedia.org
- ISI Information Sciences Institute
- UNIVERSITY OF CAMBRIDGE
- UNICAMP
- TEL AVIV UNIVERSITY אוניברסיטת תל אביב
- WISCONSIN UNIVERSITY OF WISCONSIN–MADISON
- W UNIVERSITY of WASHINGTON
- Fraunhofer FOKUS
- TECHNISCHE UNIVERSITÄT DARMSTADT
- Penn
- THE UNIVERSITY OF TEXAS AT AUSTIN
- UMKC UNIVERSITY OF MISSOURI KANSAS CITY

Companies

- at&t
- Atomic Rules
- BAREFOOT NETWORKS
- Bell
- CISCO
- DELL
- ERICSSON
- f5
- HUAWEI
- JUNIPER NETWORKS
- Microsoft
- nicira
- noiro
- NTT
- ON.LAB
- PLUMgrid
- QOSMOS
- redhat
- tieto
- titanic systems

*This does not imply that these organizations endorse Open-NFP or Netronome

Session Agenda

Introduction

- Objectives

Analytics

- Why Analytics in the SmartNIC-Identification of potentially interesting traffic
- The Trusted Execution Environment

Demo

- Run through
- Simple Encap Test
- Decap and Monitoring/Analytics

SmartNIC architecture

- The MicroEngine
- The NFP Many Core Architecture

Summary

Introduction: Objectives

Understanding the need for SmartNIC-based monitoring

- Understand why the development of the next-generation of the carrier datacenter requires compute node-based monitoring and identification of potentially interesting traffic
- Understand why the SmartNIC is the correct place for this monitoring to be based
- Understand how to interleave with the data plane in a virtualized environment to ensure closed loop control
- Understand the motivation behind the Trusted Execution Environment - kernel extension, transparent hardware

Understanding programming options

- Understand the options (C, P4/C, eBPF)

The Code

- Understand the flow of a P4/C program - specifically the use of stateful C
- Understand how to use the finer details of NFP debugging-mailboxes, data watches and common issues
- Understand how to use existing primitives - such as timestamps

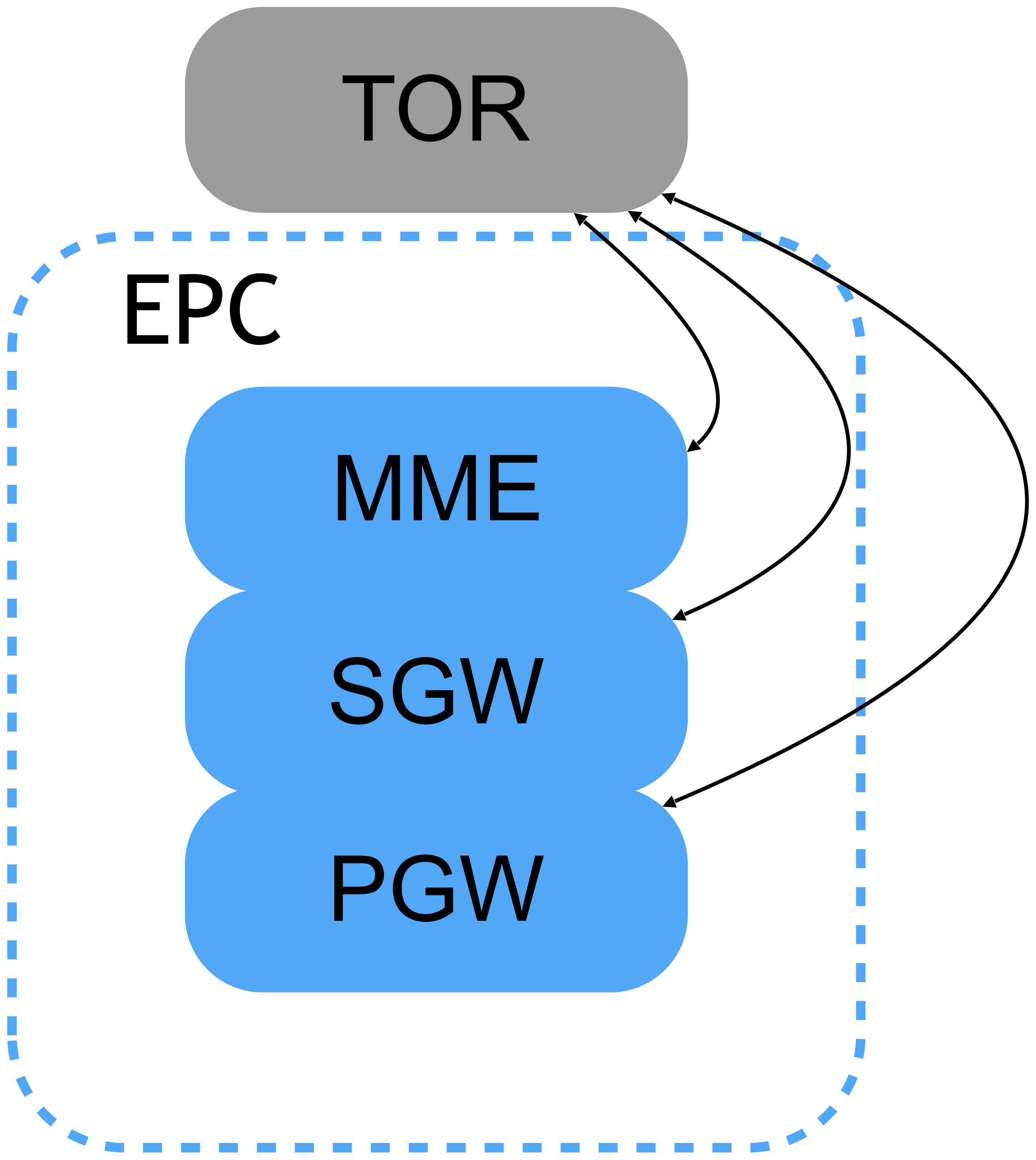
Understanding how the NFP architecture on the Agilio-CX enables high performing, fully programmable network offload

- The Many Core architecture and its advantages
- How time multiplexed multithreading reduces latency and increases throughput

Disaggregation of the Network

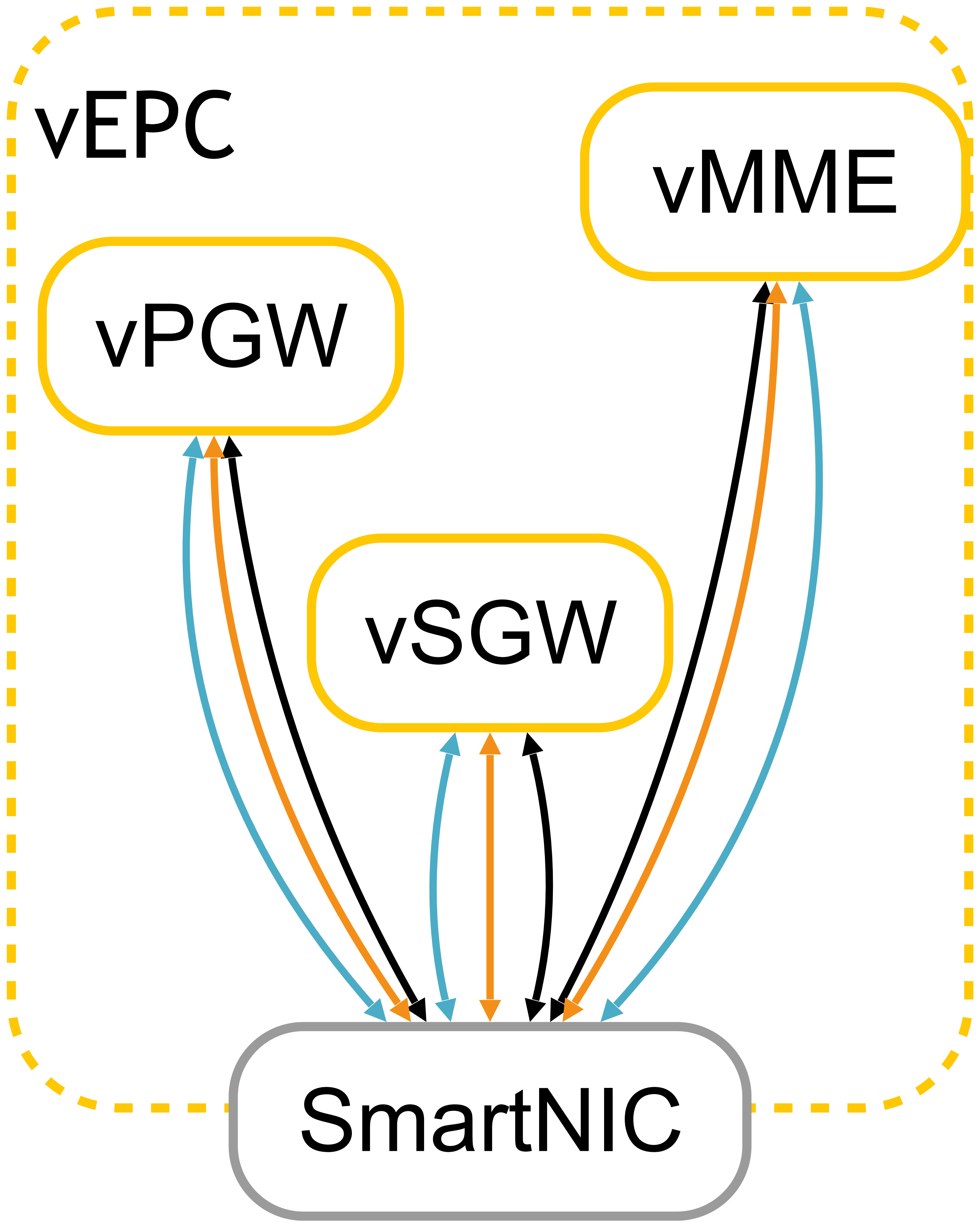
Disaggregation drives the requirement for compute-based monitoring

Rack-based Middleboxes



Inter VM traffic highly significant

COTS Server-based Whitebox-with service-based slicing



Why Place Analytics in the SmartNIC?

- Enables required depth of monitoring required for splitting-per VNF/VM based monitoring
- Not feasible with CPU as this level of monitoring will take away significant amount of cores from other processes
- There are likely spare computing cycles in the SmartNIC
- Low latency - the data being collected is already in the local cache
- Specialist monitoring equipment does not scale in disaggregated architectures:
- Either of these other options may require significant CAPEX

Three Stages of Monitoring

Identification of potentially interesting traffic

- Tagging/higher level control logic
- Broad monitoring

Deeper monitoring

- NFP: Trusted Execution Environment (P4/C) - **Our Focus**
- Host: DPI or other host based monitoring

Predictive reaction

- React to potential problems before they are noticeable to the end-user

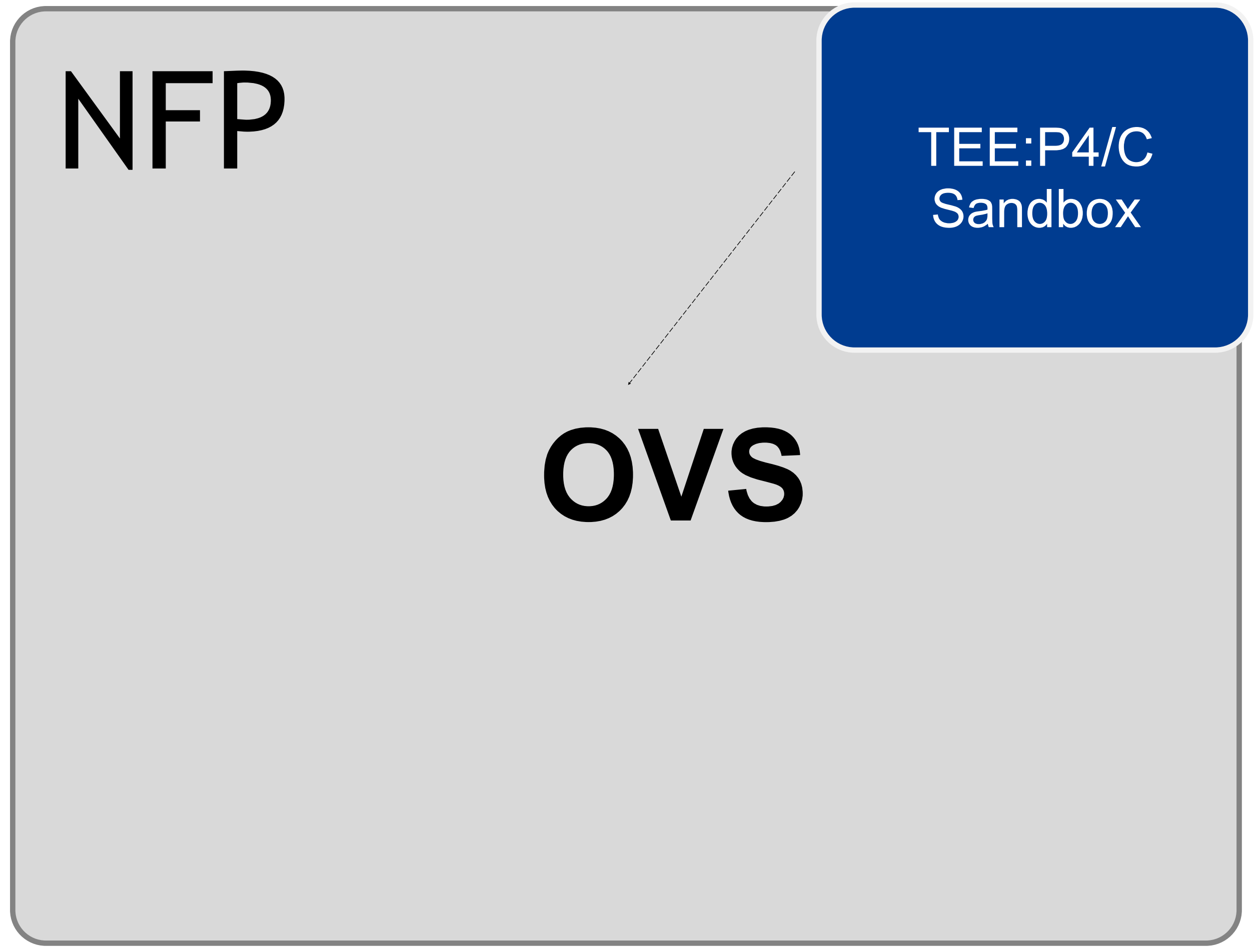
Deeper Monitoring

NFP or host-based

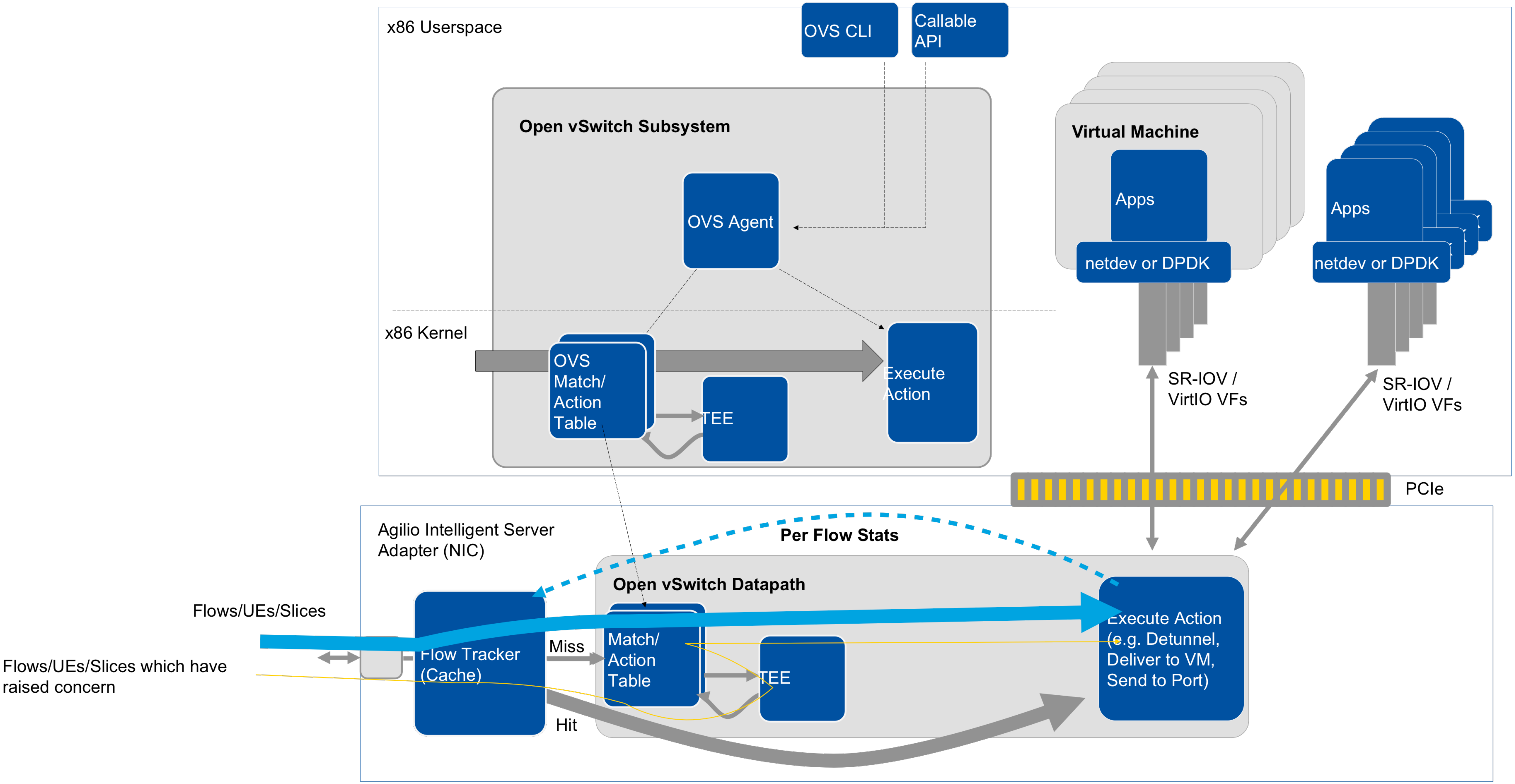
- NFP: Trusted Execution Environment (C, P4/C or eBPF)
  - Allowing a carrier to obtain a physical partition within the NFP which can be used as a TEE will allow carriers to enable custom tagging, deeper monitoring (regexp, frameworks such as INT and even rerouting to custom host modules)
  - **P4: Enables flexible encapsulation and custom tagging**
- Host: DPI or other host-based monitoring
  - Using APIs host based VNFs can be tied to the OVS environment

Predictive reaction

- Whitelisting/Blacklisting
- Compute node resource allocation
- QoS/SLA changes

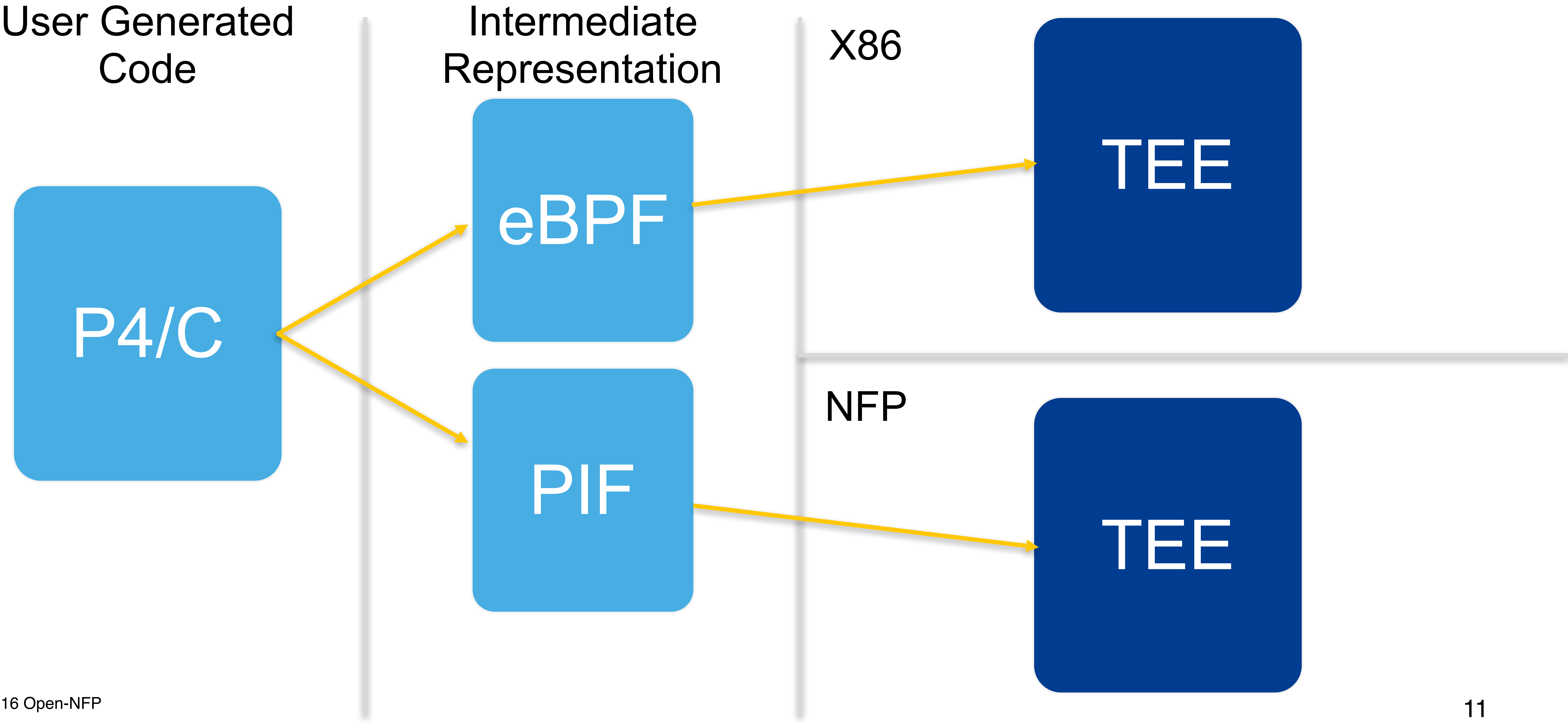


Dynamic Identification



P4/C

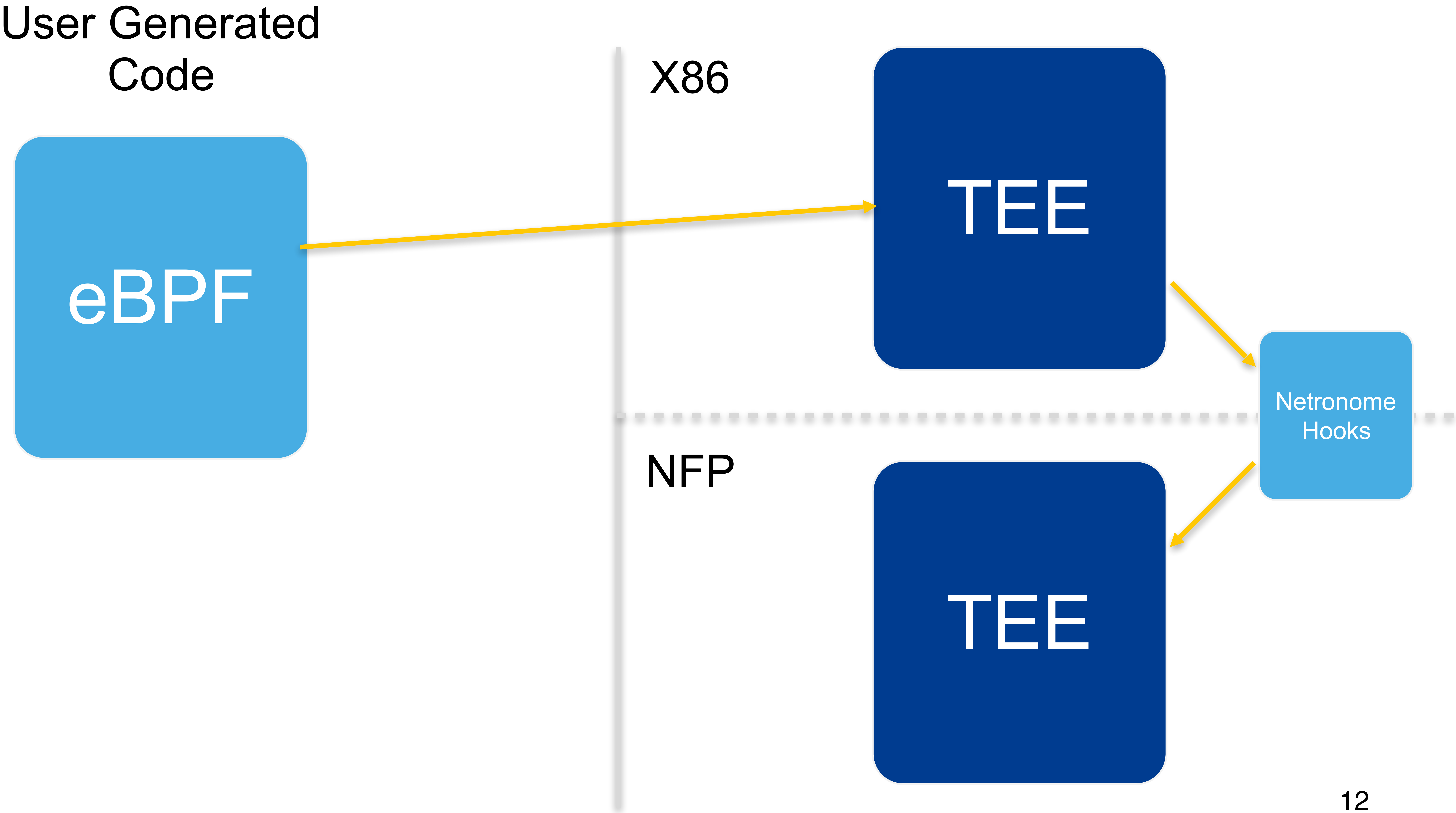
Compile Time



©2016 Open-NFP

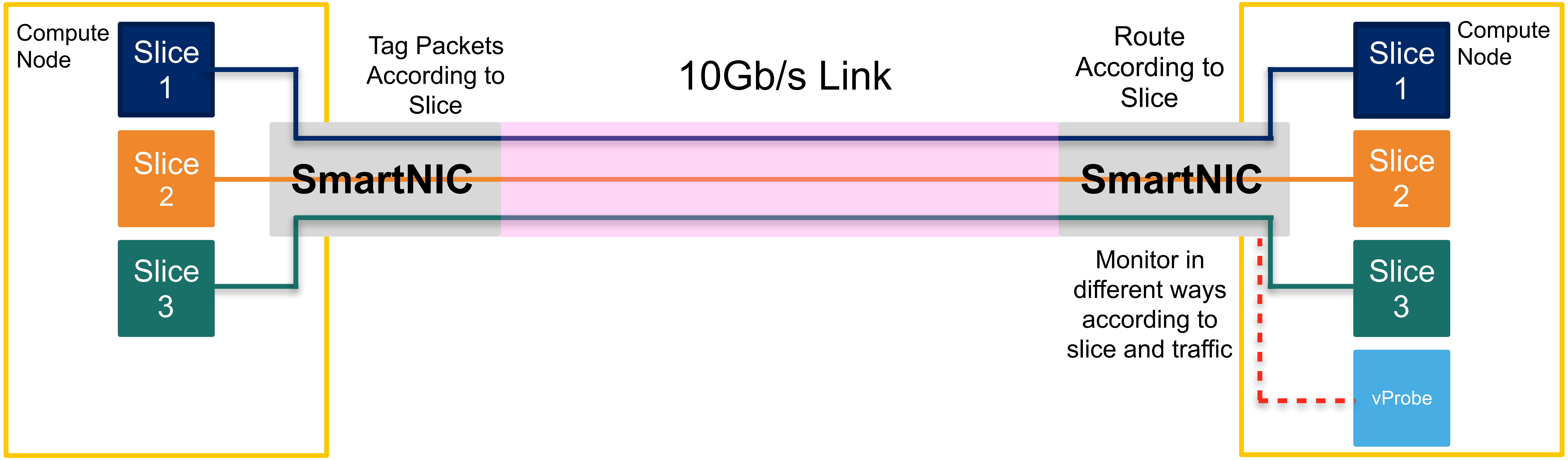
eBPF

Run Time

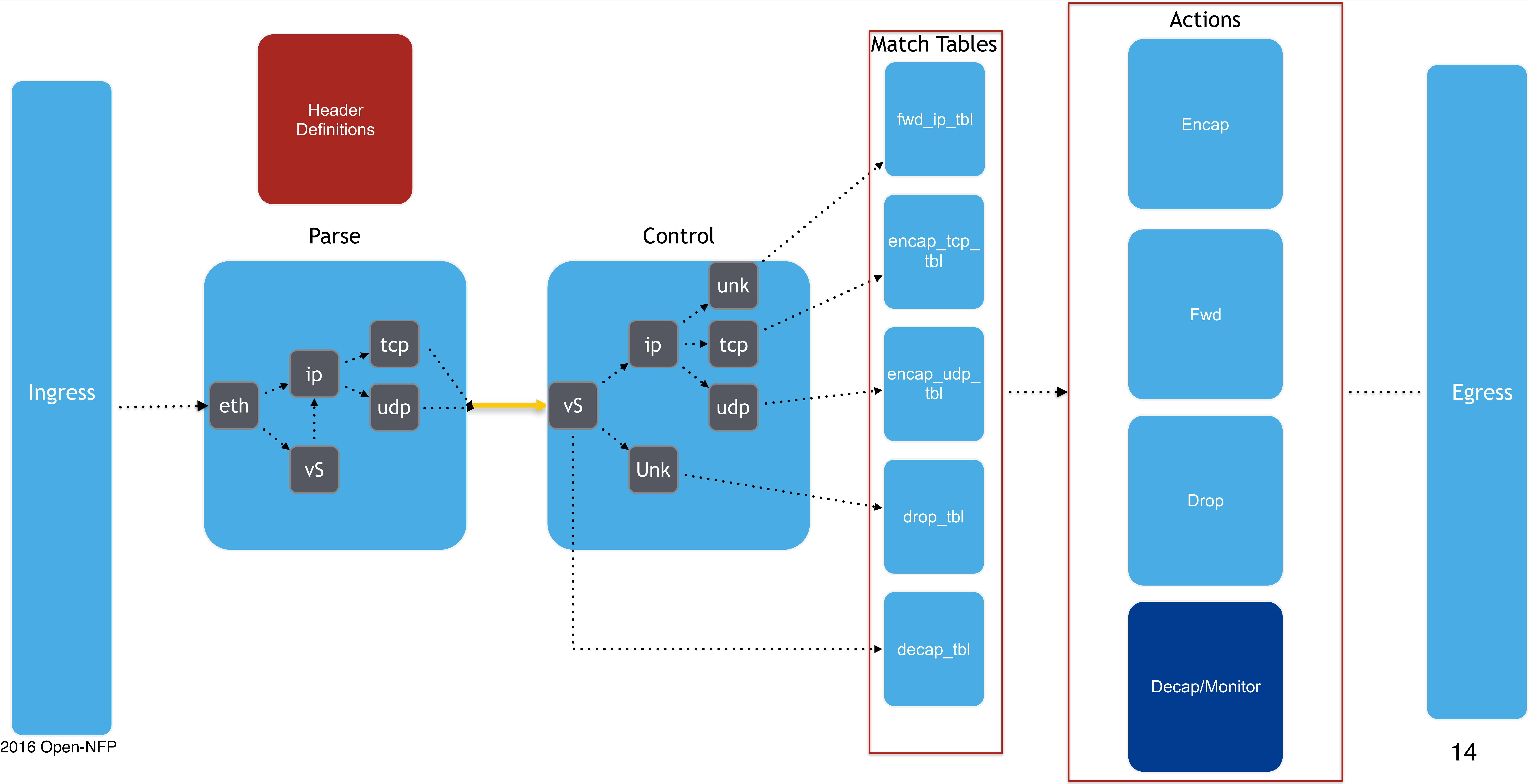


DEMO

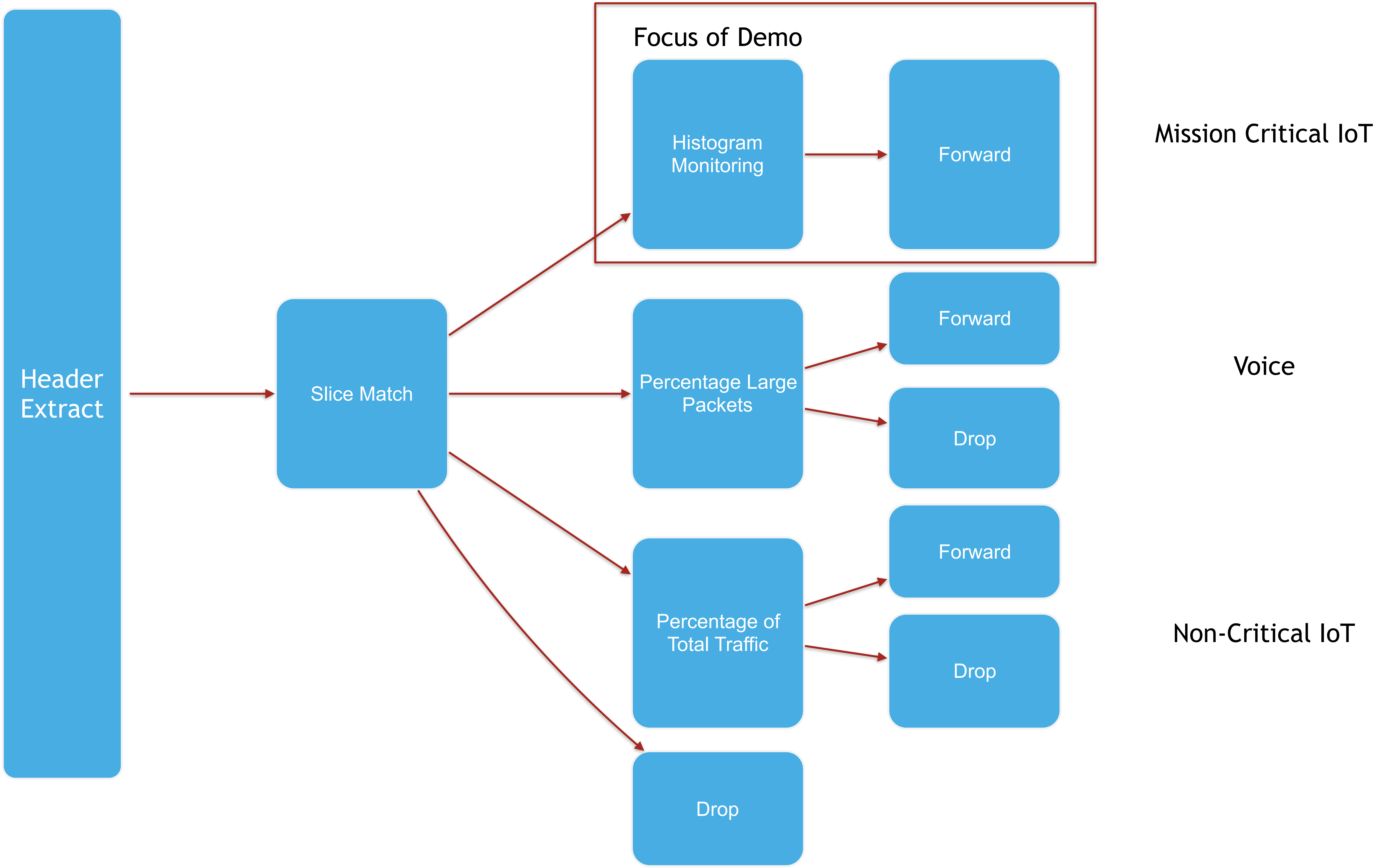
- Run through
- Simple Encap - custom identification and tagging
- Decap and monitoring/control



DEMO-P4 Program Pipeline

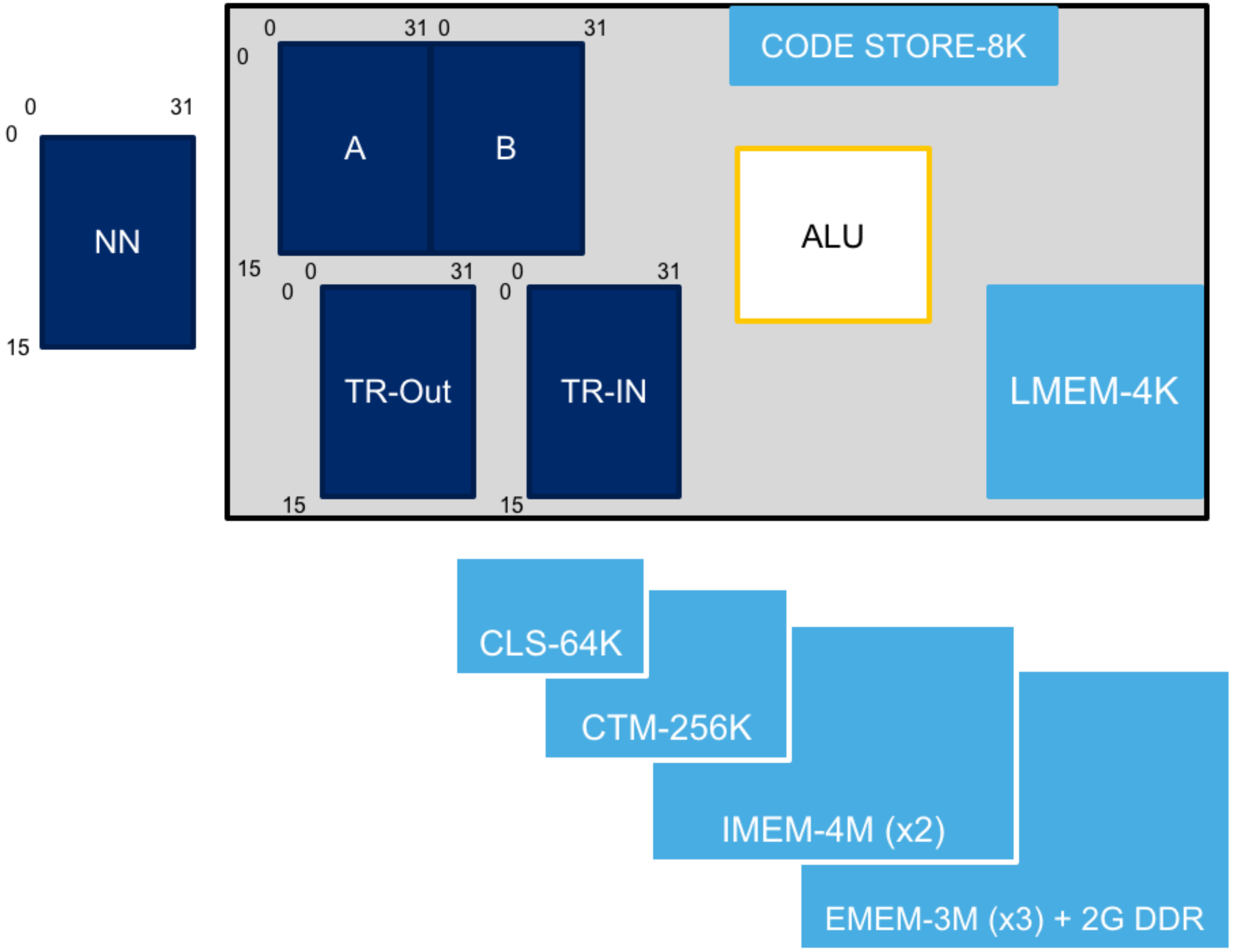


DEMO-C Action

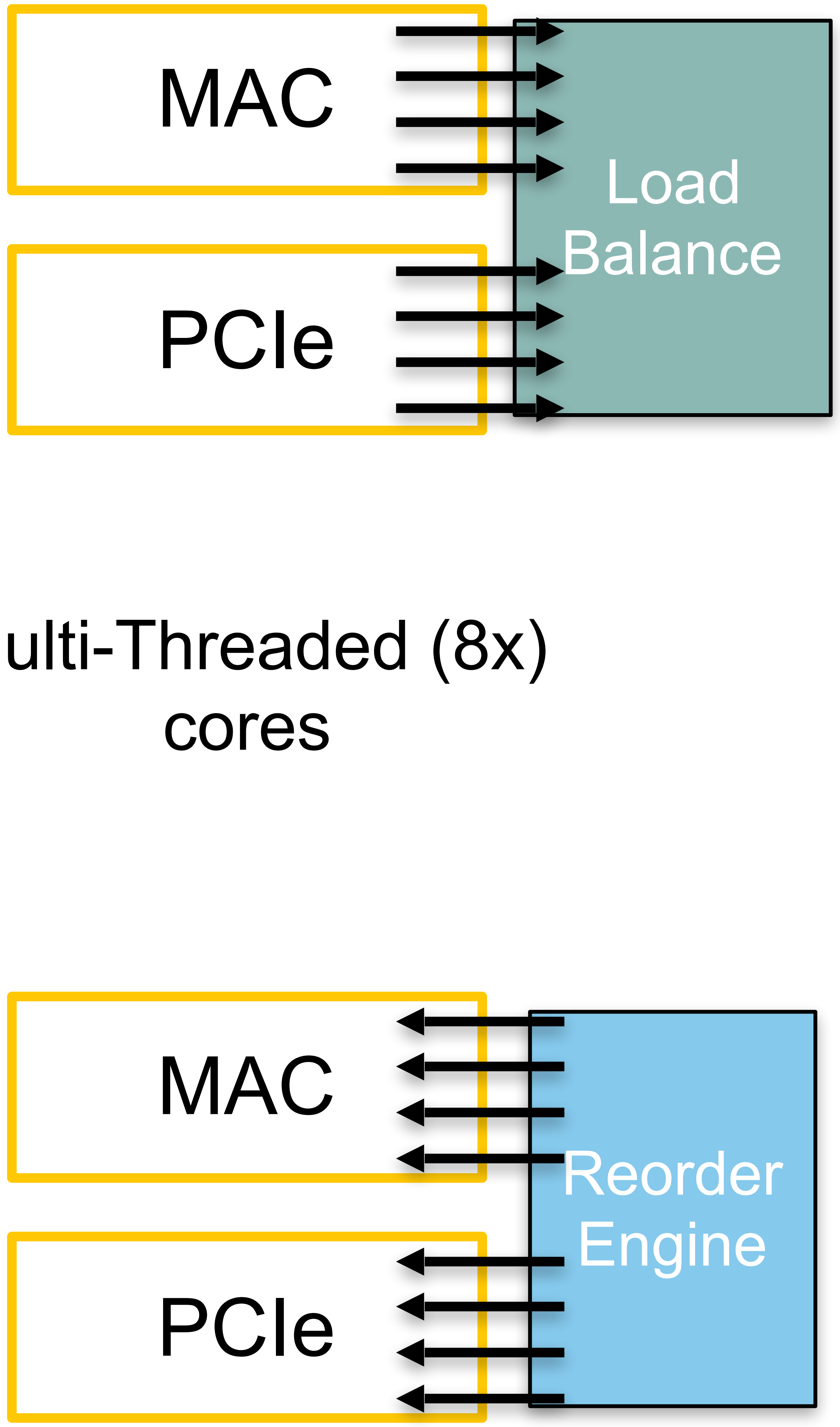


The Microengine (ME)

- These Values are **Per Thread-8x per ME**
- A/B Registers interact
- xwrite (TR-Out) writes through CPP bus
- xread (TR-In) reads through CPP bus
- Nearest Neighbour (NN) Registers can assist
- Code Store Can be extended to 16K



SmartNICs: Many-Core Approach (CMT)-NFP



MAC

PCIe

Load Balance

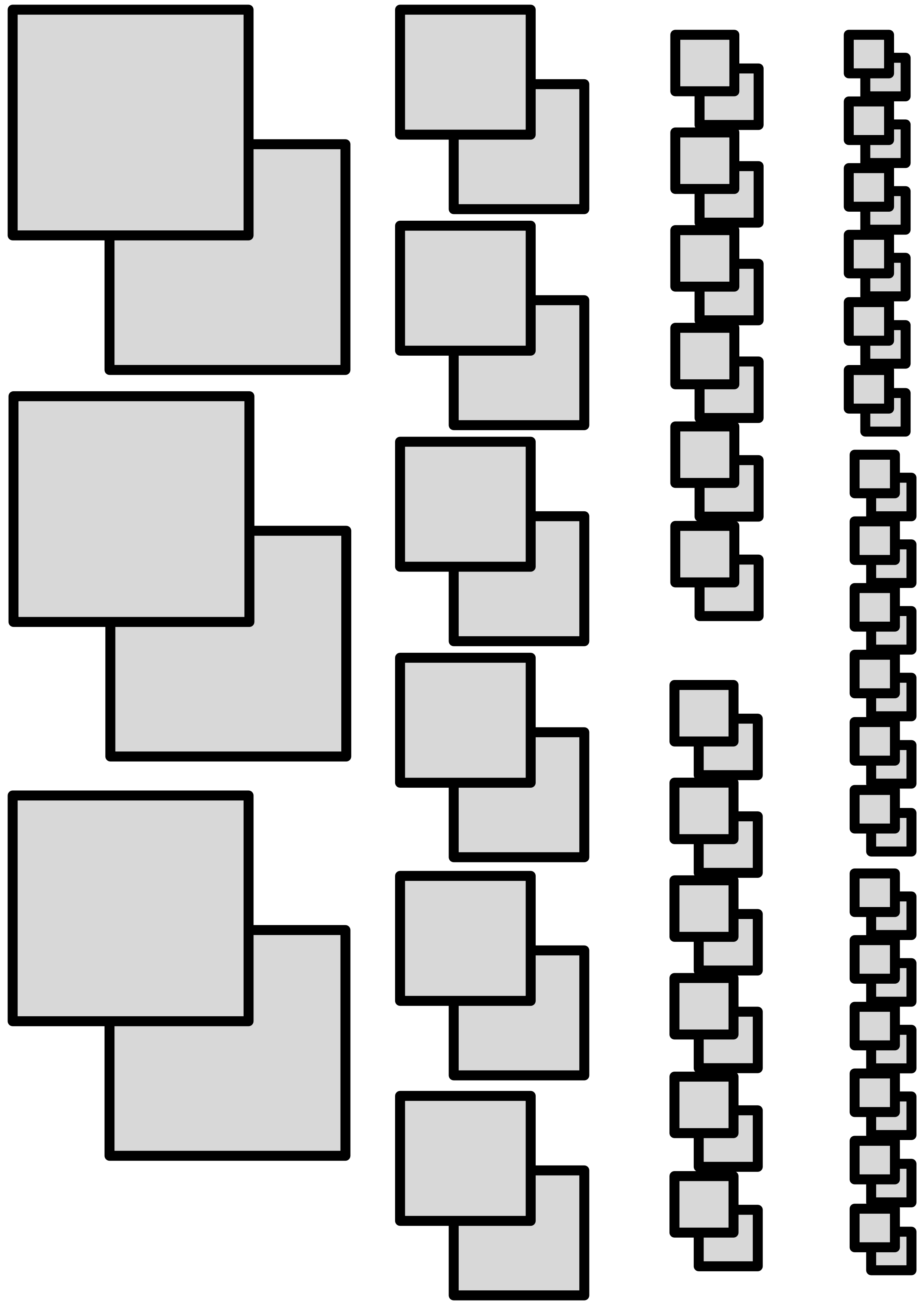
MAC

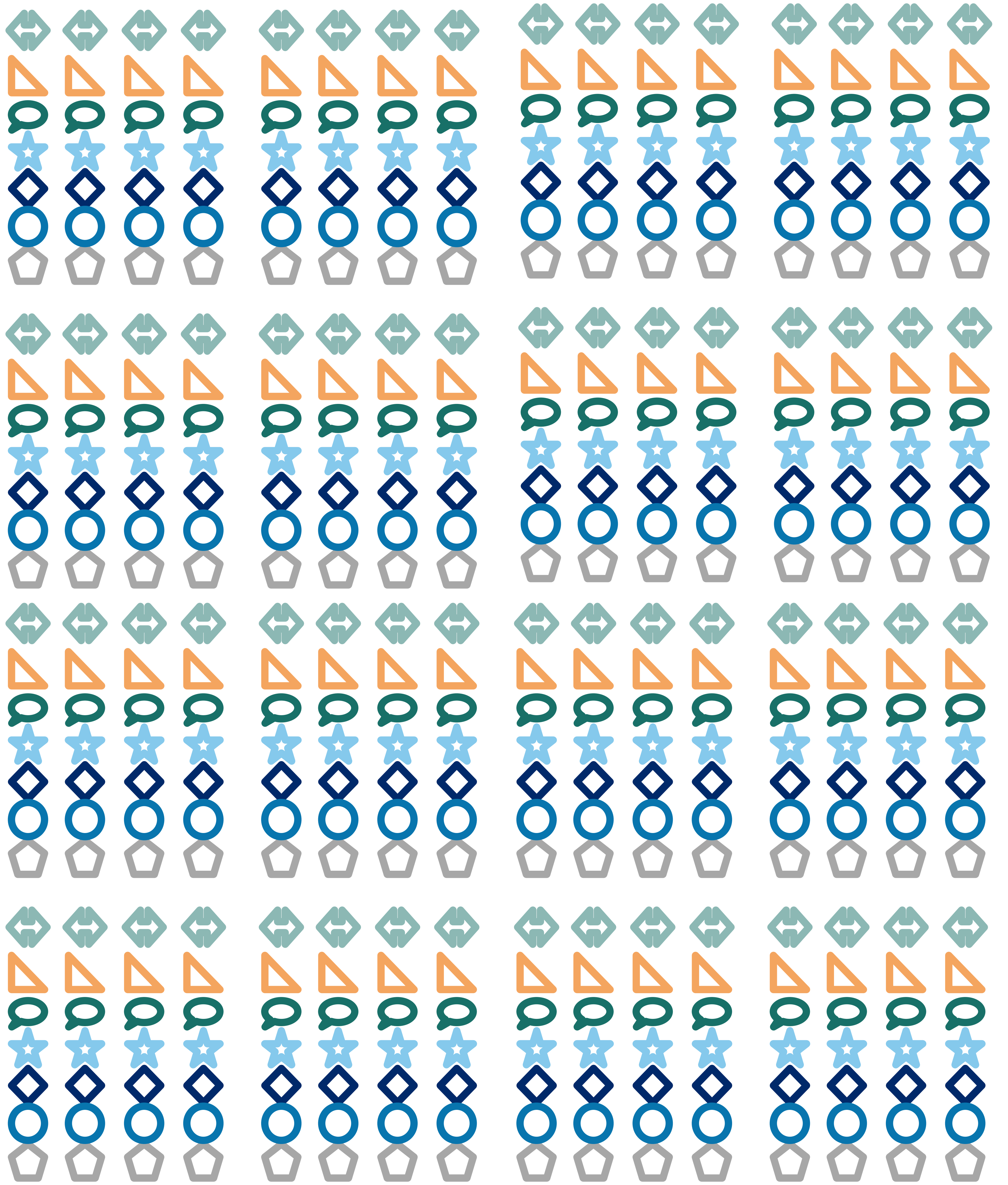
PCIe

Reorder Engine

Multi-Threaded (8x) cores

Hierarchical Transactional Memories





Reorder Engine is Key Feature due to the use of ~1000 simultaneous threads

Many Multi-Threaded NPUs with specialized hardware offload

SMP vs CMT: Definitions

SMP (Symmetric MultiProcessing)

- Architecture where two or more identical single threaded cores are connected to a single shared coherent main memory
- Homogenous cores work independently but share system bus and memory

CMT (Chip MultiThreaded) :

- Architecture employing hierarchical transactional memory with highly multithreaded cores
- Homogenous cores work independently and have access to a group of memories - The higher level the memory the more cores have access to it

SMP vs CMT for Network Processing

Characteristics of Networking Workloads

- **Highly parallel processing requirements** - packets are processed independently of other packets
- Due to an increase in virtualization and the introduction of techniques such as slicing, **branching within code is significantly increasing**
- **High throughput, low latency** (especially as mission critical IoT applications start to be hosted)

SMP (Symmetric MultiProcessing):

- Deep Pipeline-Processors used within this architecture tend to be heavier and single threaded (e.g MIPS64), leading to deeper processing pipelines, this **leads to more missed cycles per code branch**
- Due to flat, coherent memory architecture (2/3 layers), lots of memory requests and cache locking, **increasing latency**
- Due to the single thread per core there are more cycles wasted as well as less cycles available - **decreased throughput**

CMT (Chip MultiThreaded) :

- Shallower pipeline - processors are more lightweight but multithreaded (e.g MicroEngines) therefore have **very little dependency on branching**
- The hierarchical memory structure (5/6 layers) of non-coherent memory allows for the avoidance of cache locking **decreasing latency**
- Due to the multithreaded cores very few cycles are wasted-**significantly increasing throughput**

Modern CMT Processors have up to **20X as many cycles available** as current SMP architectures

Summary

Custom tagging/encapsulation is easily defined within P4

- This allows fast innovation and reconfiguration of systems-‘fail fast model’
- Allows one-tuple based flow control

Using the combination of P4/C allows this to be paired with custom monitoring rules

- Adding statefulness to P4 is important in the aim to use it within NICs-allows monitoring and control to be easily implemented

The NFP is able to offload complex software-defined processes due to its 500+ concurrent threads and transactional memory

- The Many Core architecture allows significantly more parallelization than would otherwise be possible
- This allows low cost, high performance software-defined networking in the data plane

Using spare cycles in the NFP we are able to offload some of the heavy duty work that monitoring applications have to do

- This solves the problem of probing virtualized environments where physical probes do not scale and CPU resource is scarce
- This enables fine grained real time data plane analytics at the compute node
- Using the TEE as a space for customer innovation allows speed of movement and vendor independence due to kernel analogues

QUESTIONS?

Nic Viljoen
nick.viljoen@netronome.com

THANK YOU