PANIC: A High-Performance Programmable NIC for Multi-tenant Networks

OSDI 2020

Jiaxin Lin[1], Kiran Patel[2], Brent E. Stephens[2],
Anirudh Sivaraman[3] and Aditya Akella[1]

THE UNIVERSITY of WISCONSIN MADISON 1

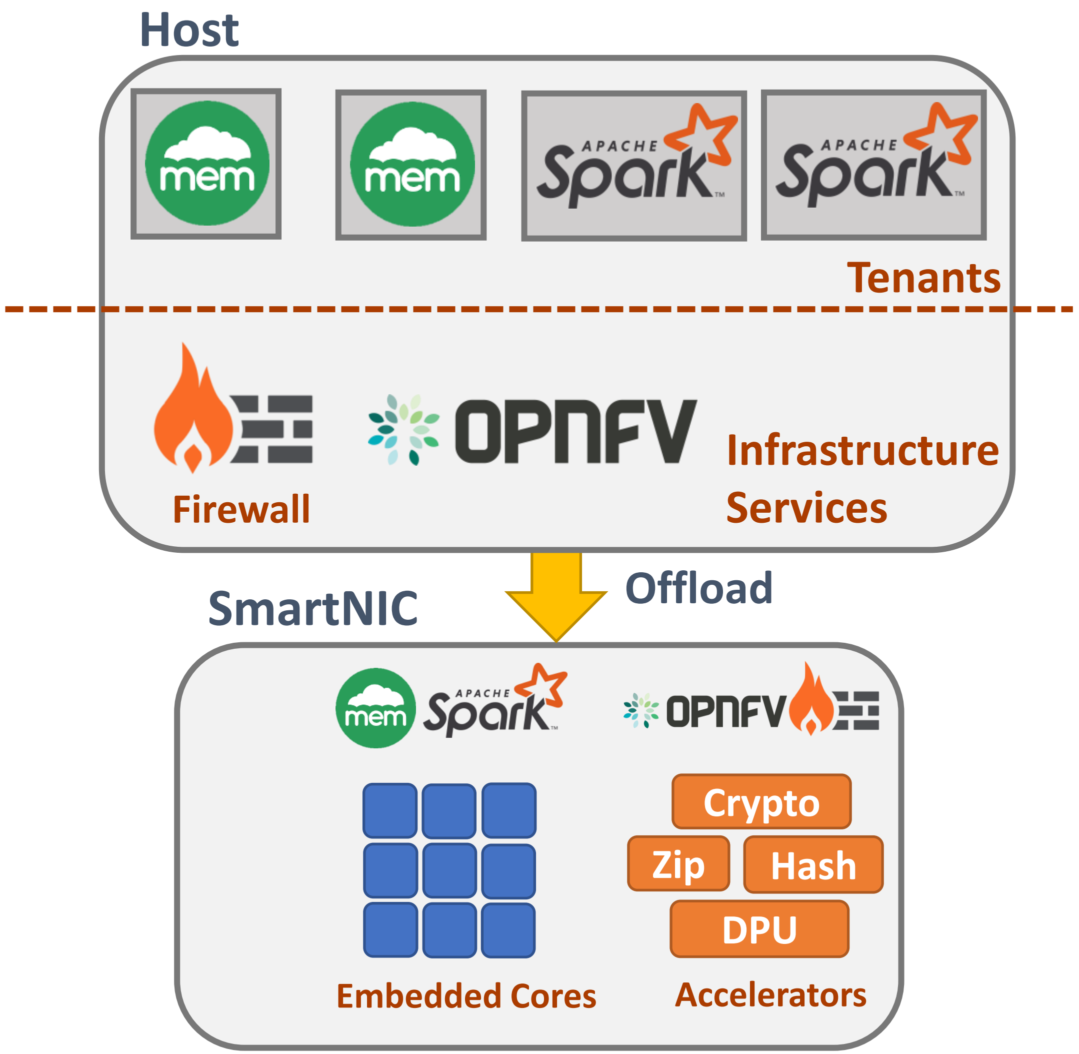
UIC UNIVERSITY OF ILLINOIS AT CHICAGO 2

NYU 3

SmartNIC and Multi Tenancy

- SmartNICs can help drive increasing network line-rates (100Gbps+) by offloading applications or cloud services

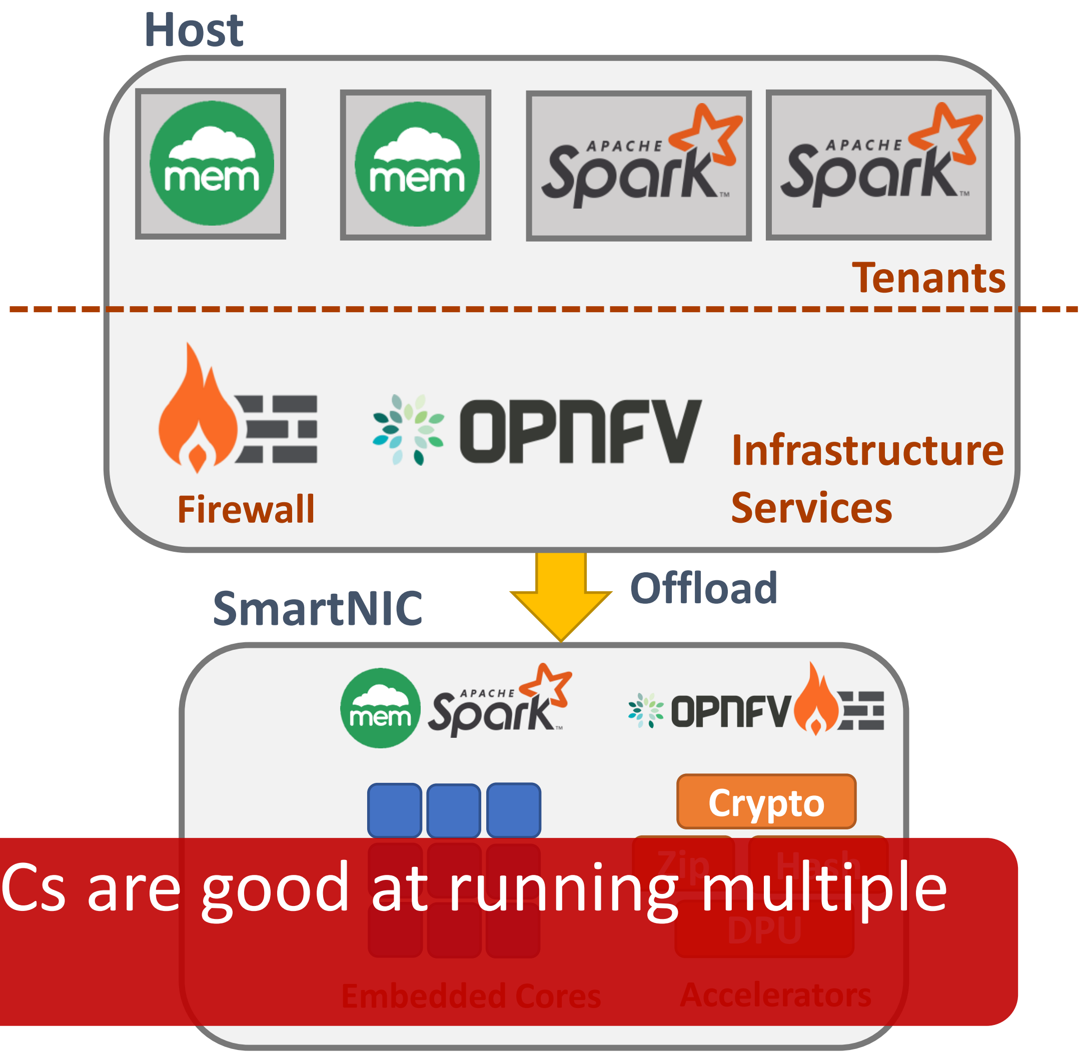
- In the multi-tenant environment, to get benefits from the SmartNIC, servers may want to run multiple offloads on the SmartNIC.



SmartNIC and Multi Tenancy

- SmartNICs can help drive increasing network line-rates (100Gbps+) by offloading applications or cloud services

- In the multi-tenant environment, to get benefits from the SmartNIC, servers may want to run multiple offloads on the SmartNIC.

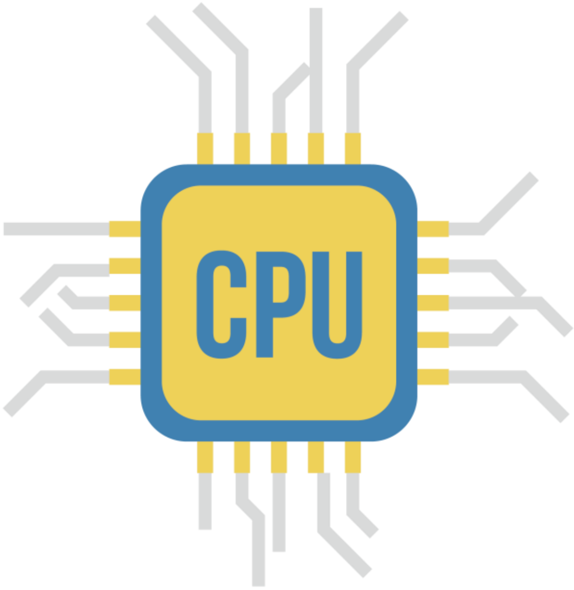


Problem: None of the current SmartNICs are good at running multiple tenants' offloads at the same time.

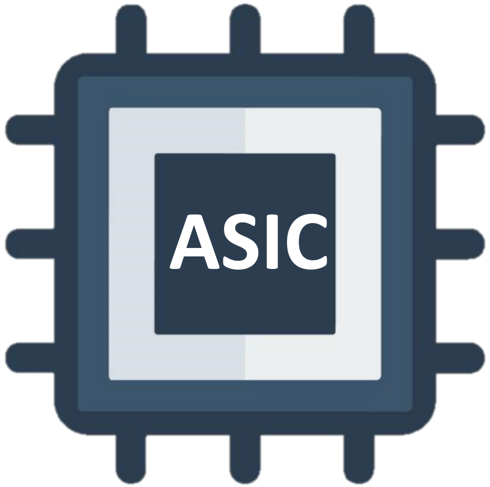
What are the requirements for a SmartNIC in a multi-tenant environment?

Requirements # 1 Generality

- **Generality**: Different tenants on the host may requires different types of offloads.
  - Both ASIC offload and CPU core should be supported
  - Offload may have below line rate/variable performance

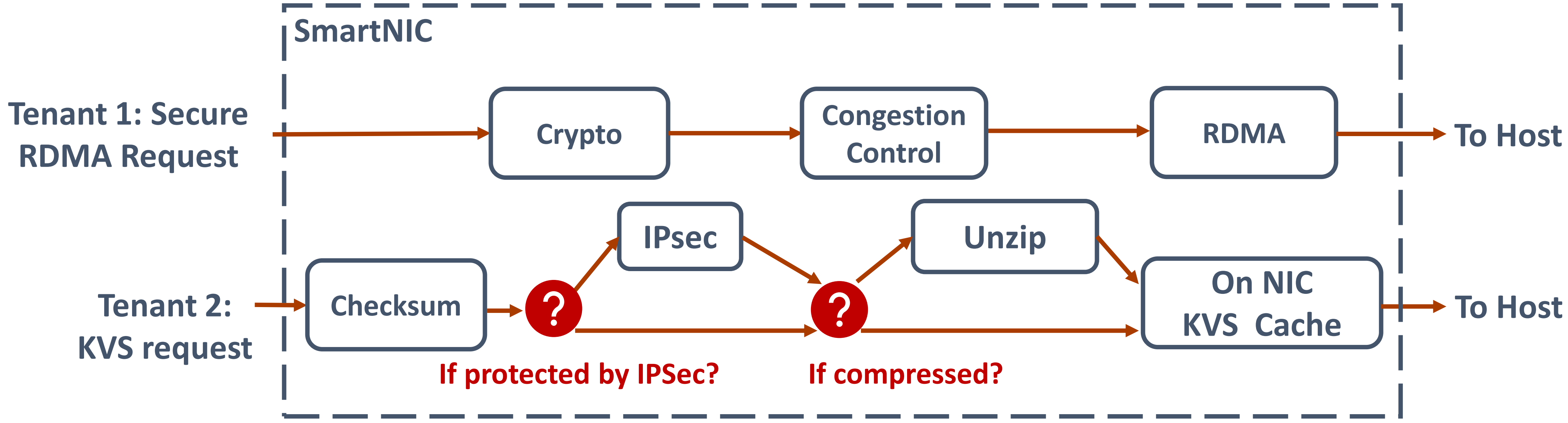


OR?



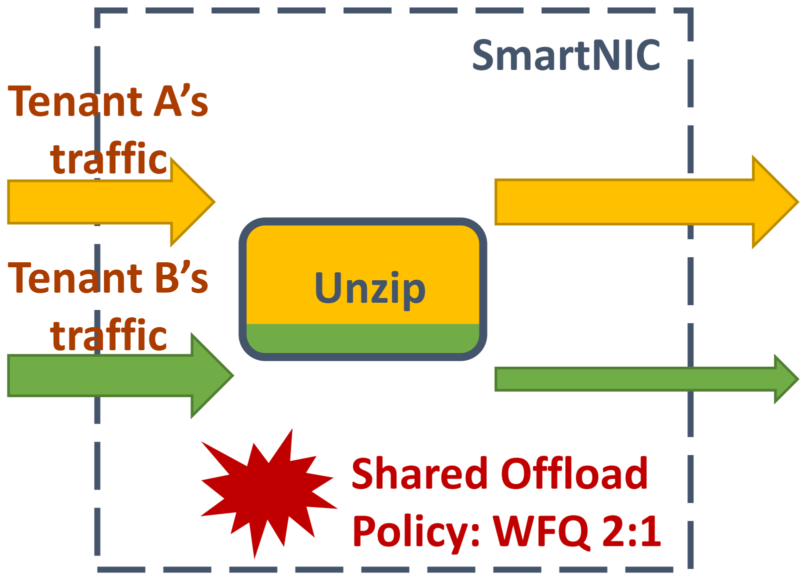
Requirements # 2 Flexible Chaining

- **Flexible Chaining:**
  - Different tenants will specify their own chains of offloads.
  - NIC should support sending packets through offloads in any order.



Requirements # 3 Isolation
4 Performance

- **Isolation:**
  - SmartNIC should provide performance isolation between competing tenants.
- **Performance:**
  - SmartNIC should provide high throughput for line-rate offloads.
  - SmartNIC should not incur additional latency for low latency offload.



Motivation: Build a programmable NIC that meets all these requirements!

Outline

Motivation

Existing NIC Limitation

Design & Evaluation

Existing NIC Design Overview

Pipeline of Offloads NIC

Chaining: ✗

Generality: ✗

Isolation: ✗

Performance: ✓

Manycore NIC

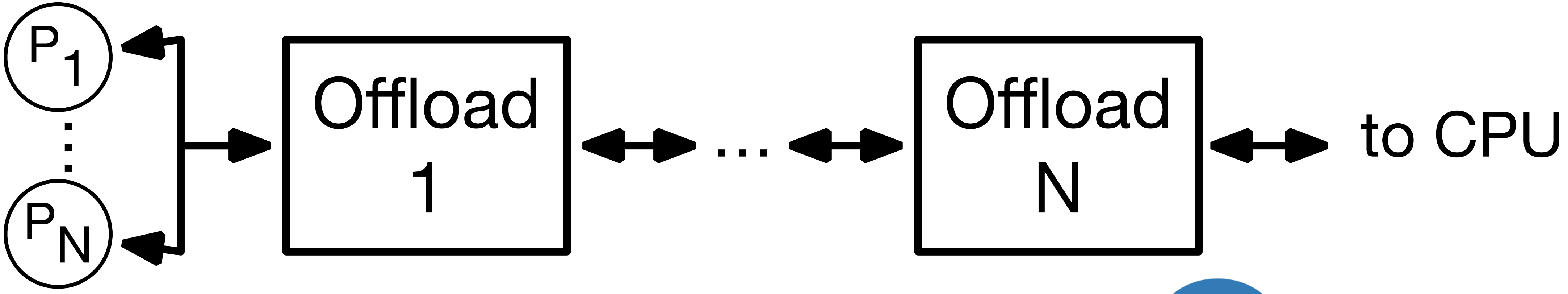
Chaining: ✓

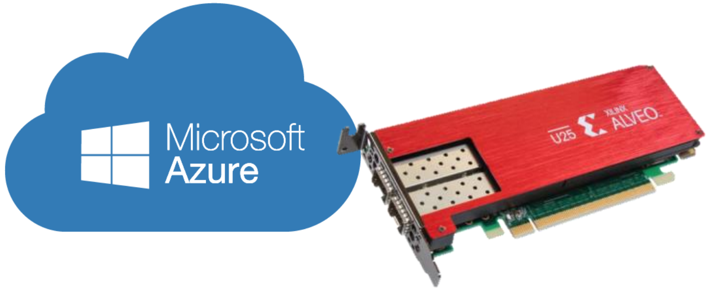
Generality: ✓

Isolation: ✗

Performance: ✗

Pipeline Design NIC





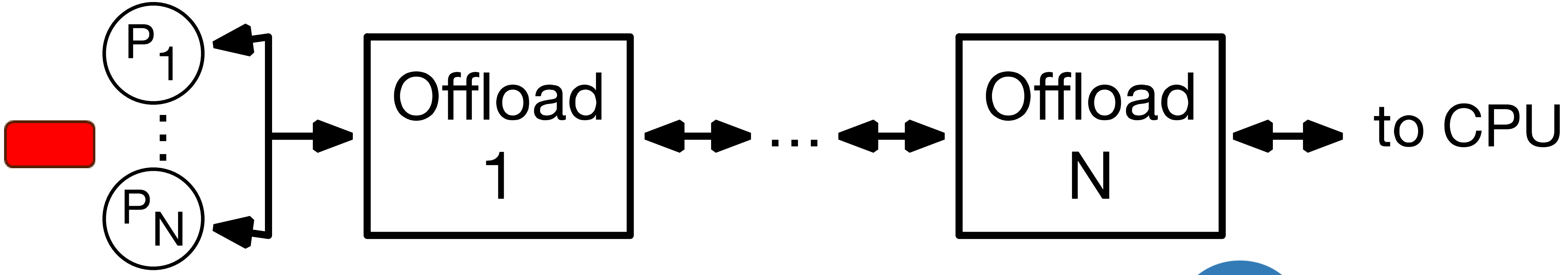
Chaining: ✗

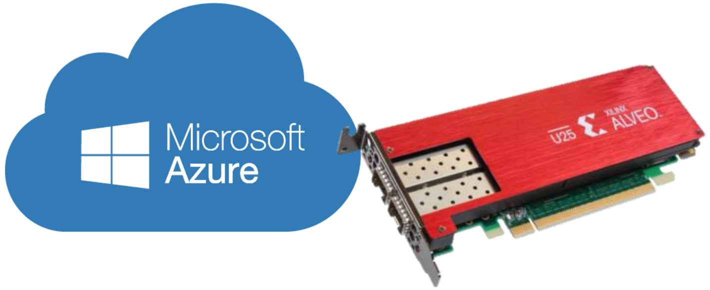
Generality: ✗

Performance: ✓

Isolation: ✗

Pipeline Design NIC





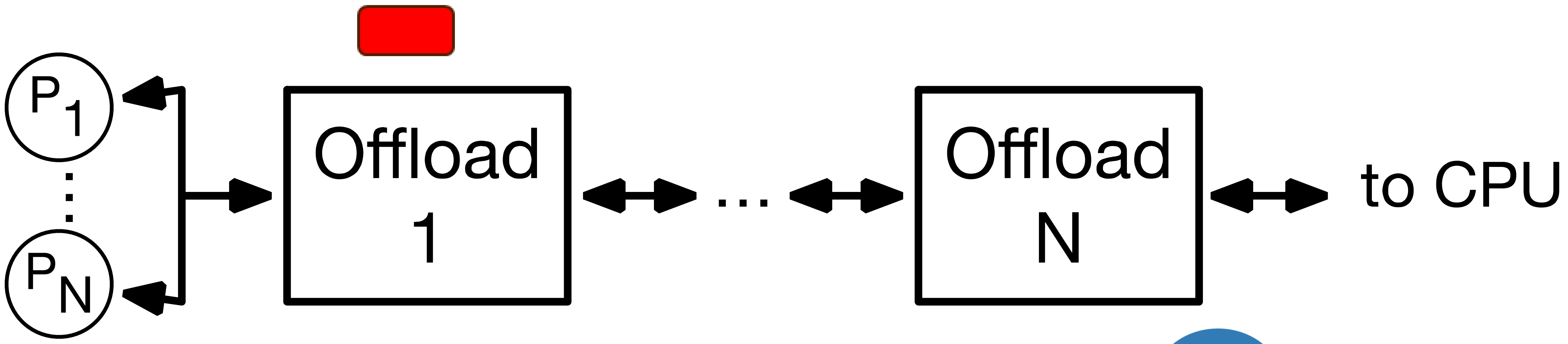
Chaining: ✗

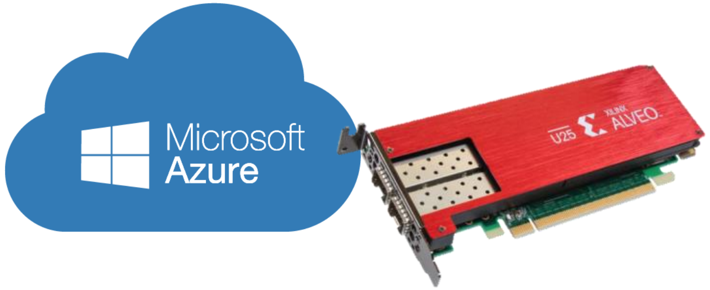
Generality: ✗

Performance: ✓

Isolation: ✗

Pipeline Design NIC





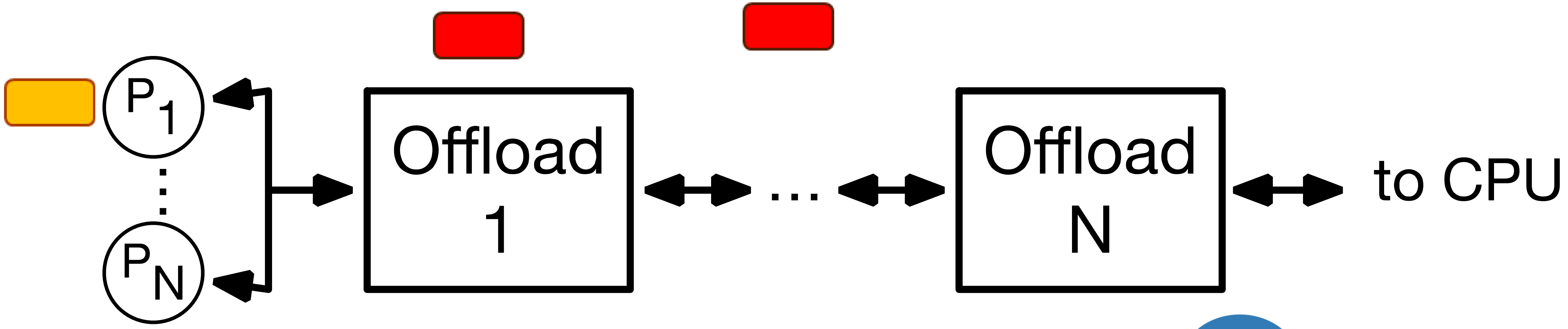
Chaining: X

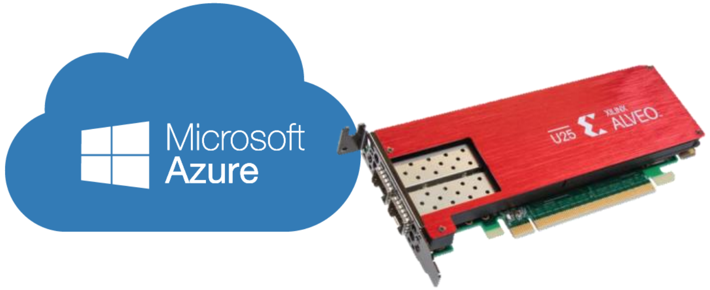
Generality: X

Performance: ✓

Isolation: X

Pipeline Design NIC





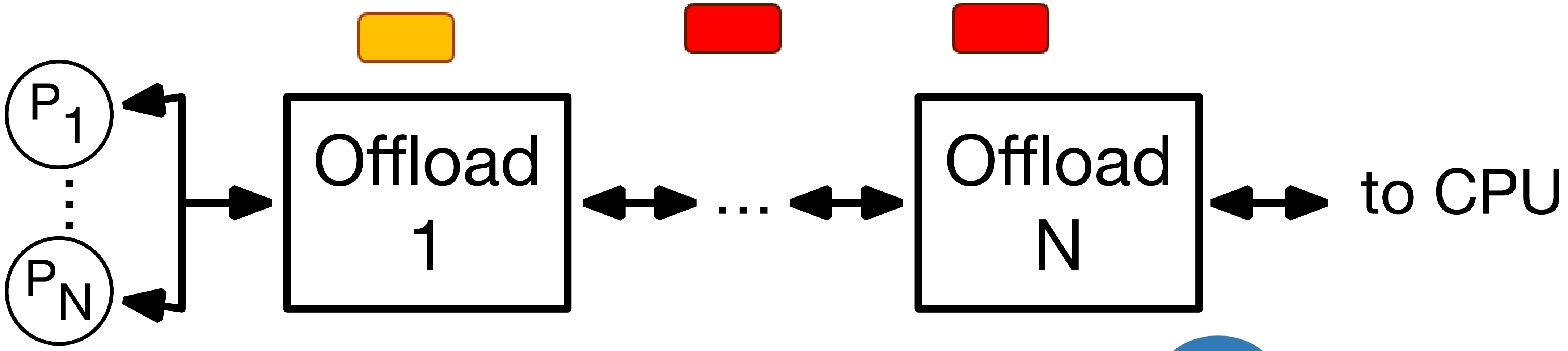
Chaining: ✗

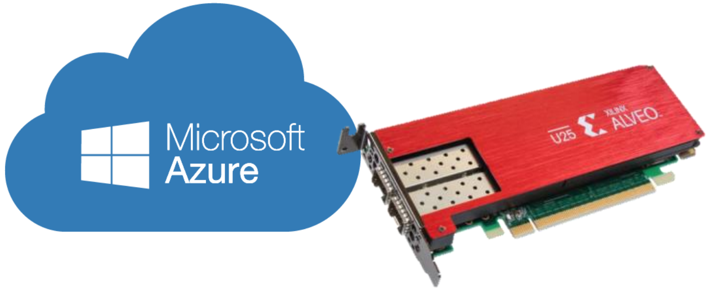
Generality: ✗

Performance: ✓

Isolation: ✗

Pipeline Design NIC





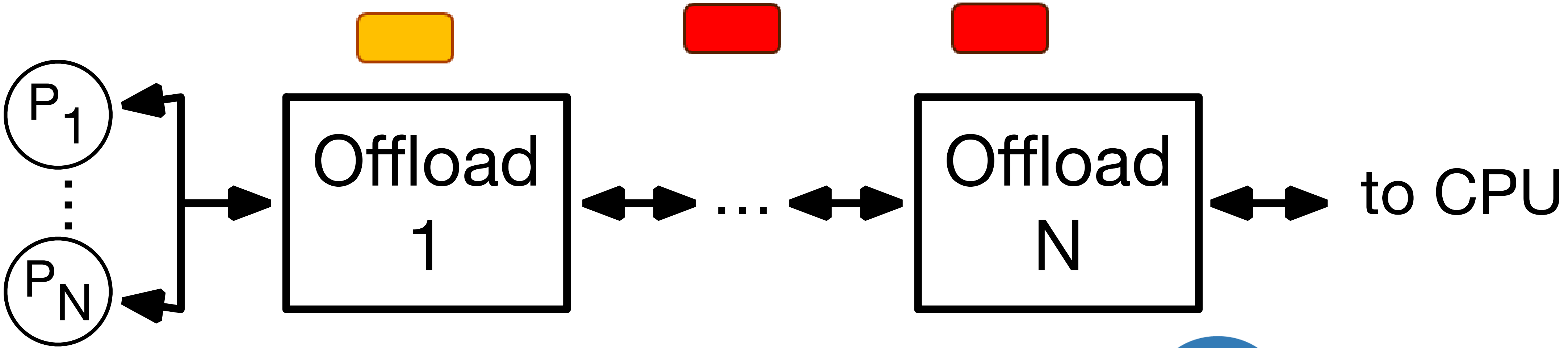
Chaining: ✗

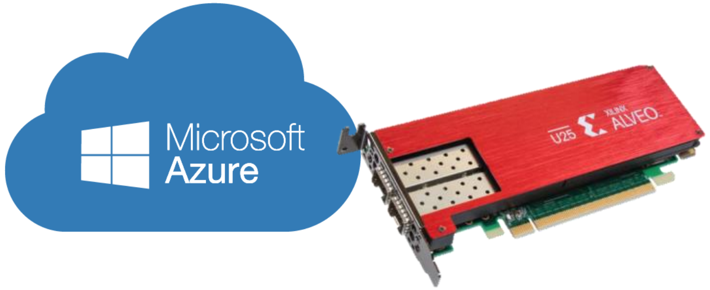
Generality: ✗

Performance: ✓

Isolation: ✗

Pipeline Design NIC





Chaining: X

Generality: X

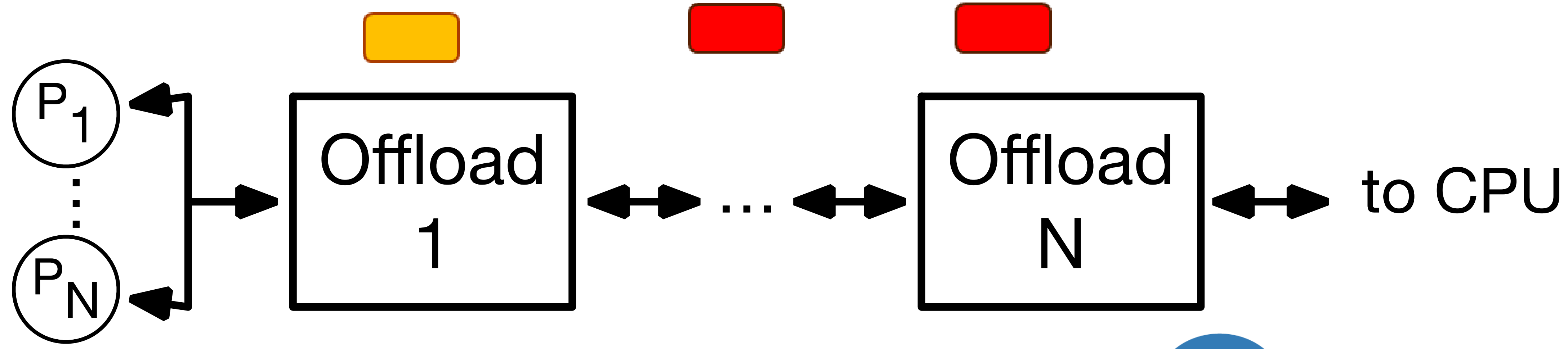
Performance: ✓

Isolation: X

Problems:

- Chaining: static chaining.

Pipeline Design NIC



Microsoft Azure

Chaining: ✗

Generality: ✗

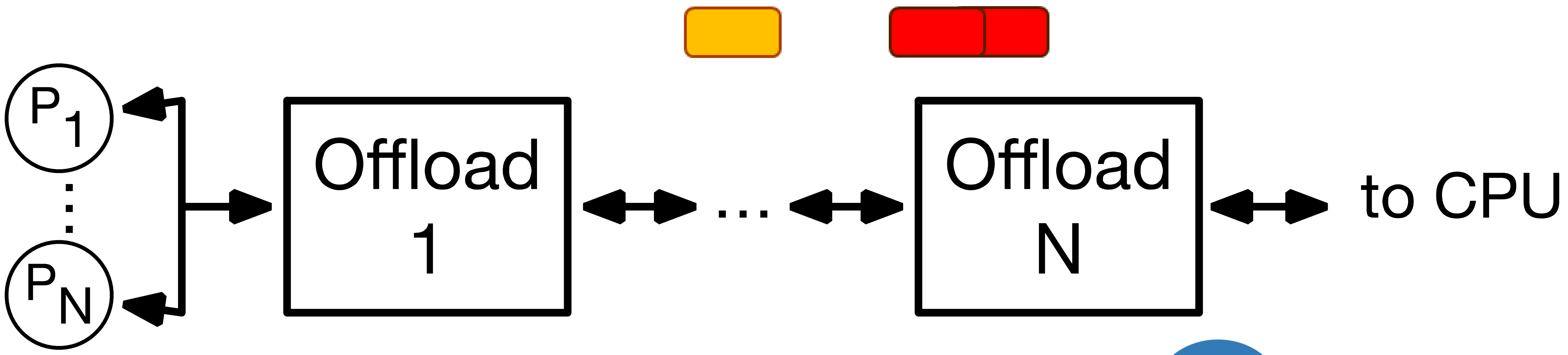
Performance: ✓

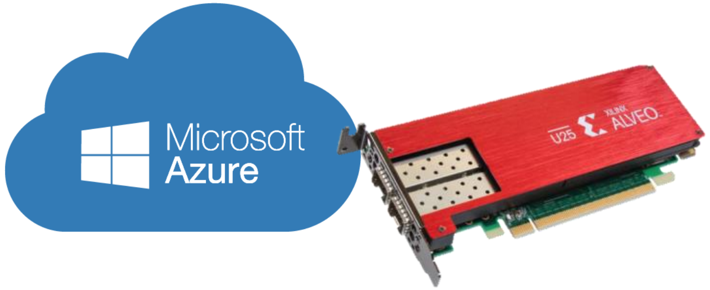
Isolation: ✗

Problems:

- Chaining: static chaining.
- Generality: non-line rate offload will cause Head-of-Line Blocking (HOL).
- Isolation: poor Isolation when HOL happens.

Pipeline Design NIC





Chaining: X

Generality: X

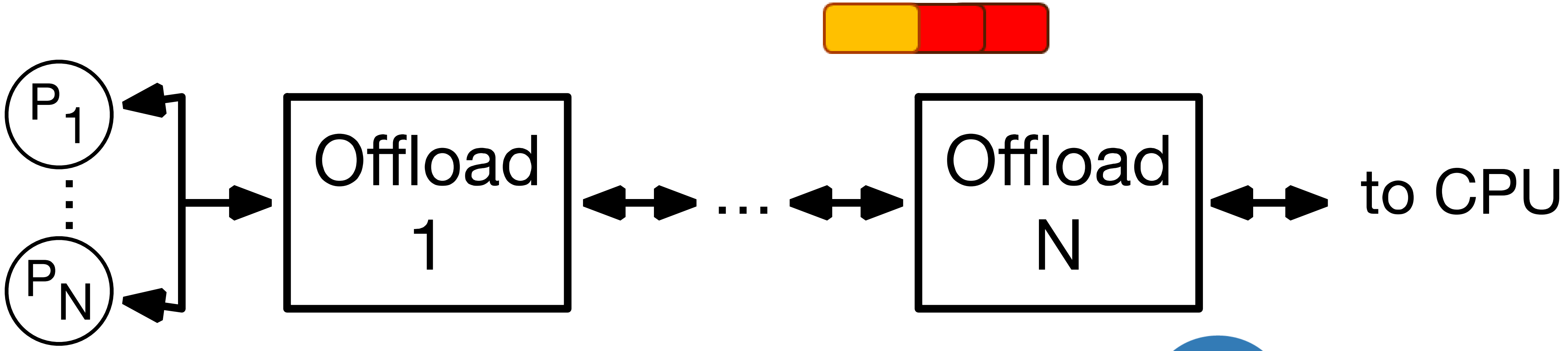
Performance: ✓

Isolation: X

Problems:

- Chaining: static chaining.
- Generality: non-line rate offload will cause Head-of-Line Blocking (HOL).
- Isolation: poor Isolation when HOL happens.

Pipeline Design NIC



Chaining: X

Generality: X

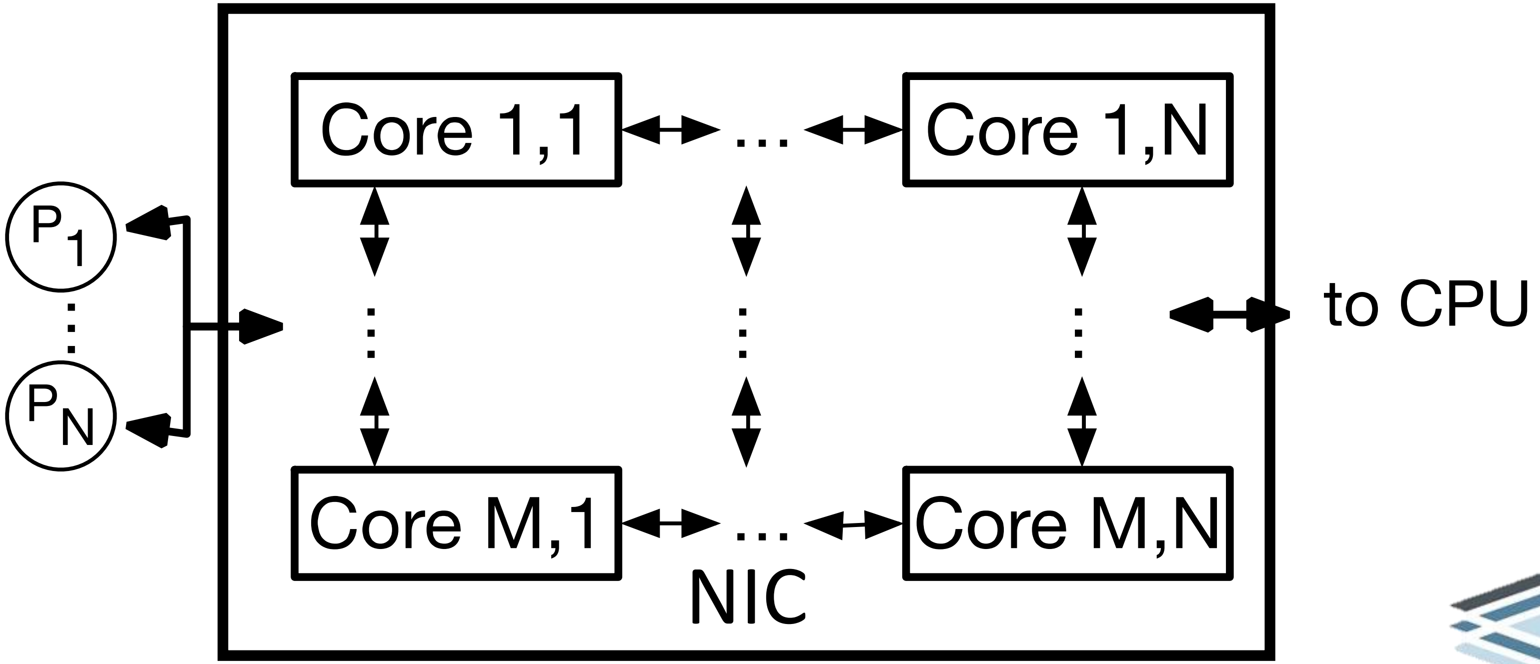
Performance: ✓

Isolation: X

Problems:

- Chaining: static chaining.
- Generality: non-line rate offload will cause Head-of-Line Blocking (HOL).
- Isolation: poor Isolation when HOL happens.

Manycore NIC



BlueField

Chaining: ✓

Generality: ✓

Performance: X

Isolation: X

Problems:

- Isolation: does not provide explicit supports for isolation.
- Performance: hard to saturate line-rate and high latency.

The solution is --- PANIC!

A new architecture for programmable NICs that supports full-line rate (> 100G) offload chaining and multi-tenant traffic isolation

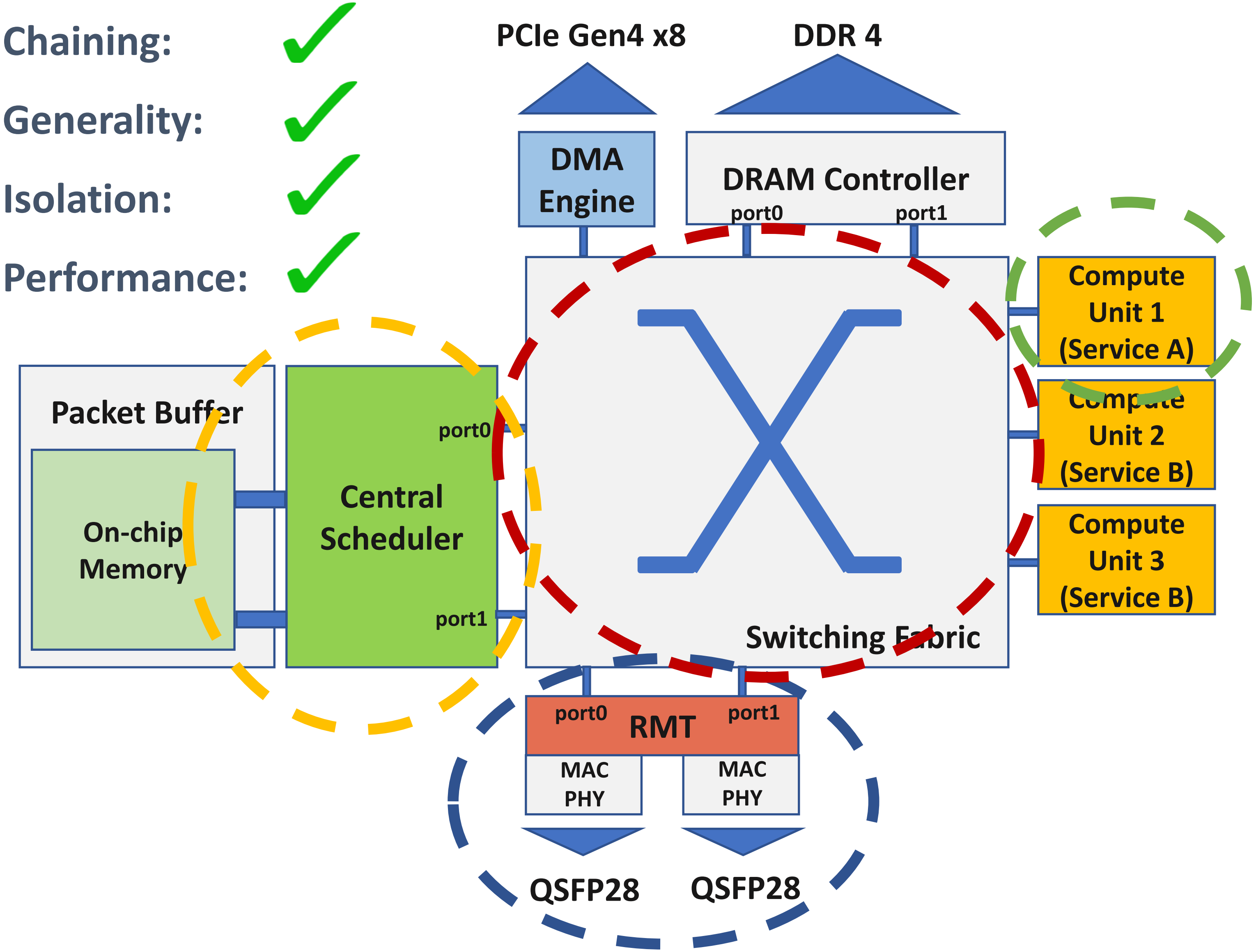
PANIC Design Overview

Chaining: ✓

Generality: ✓

Isolation: ✓

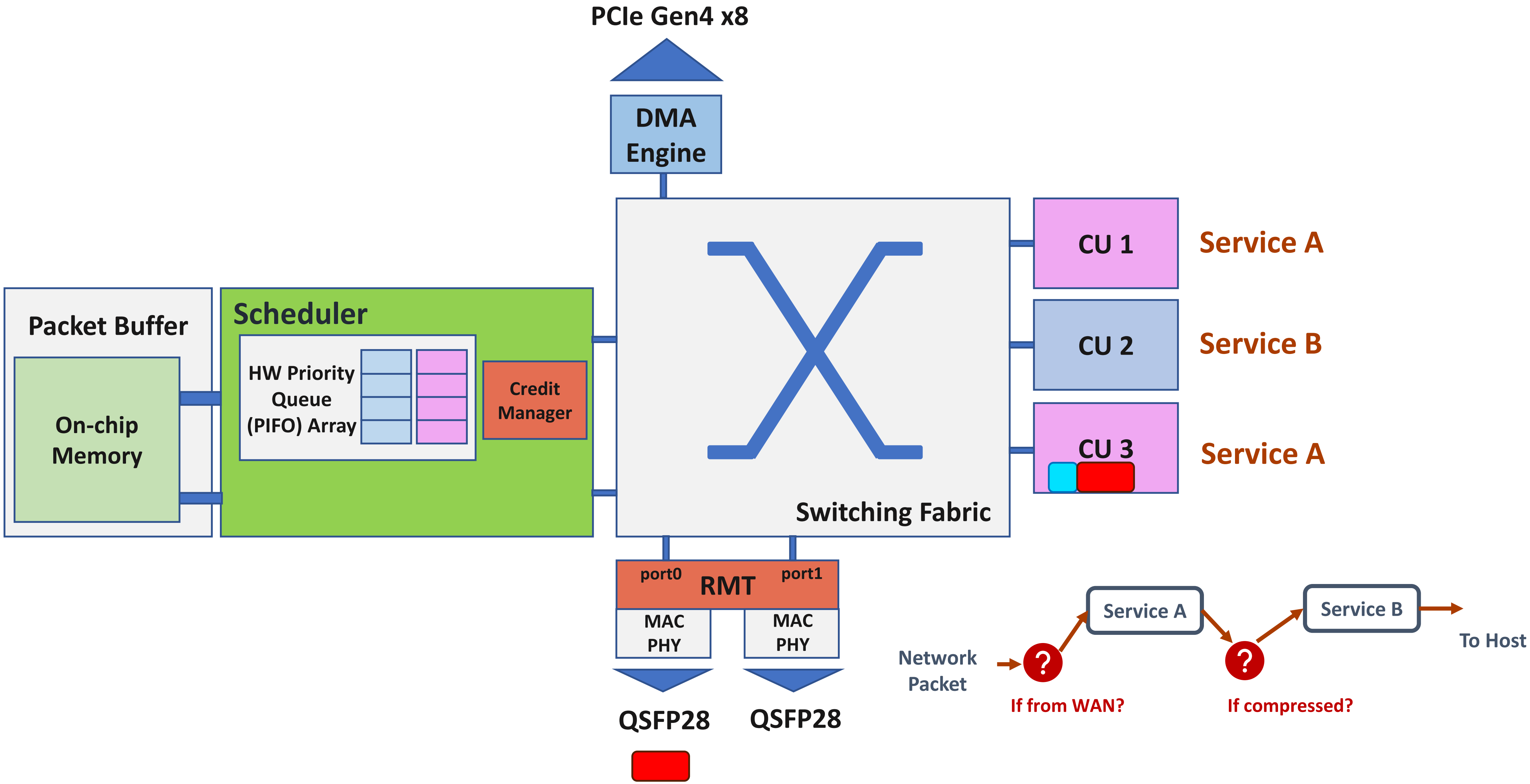
Performance: ✓



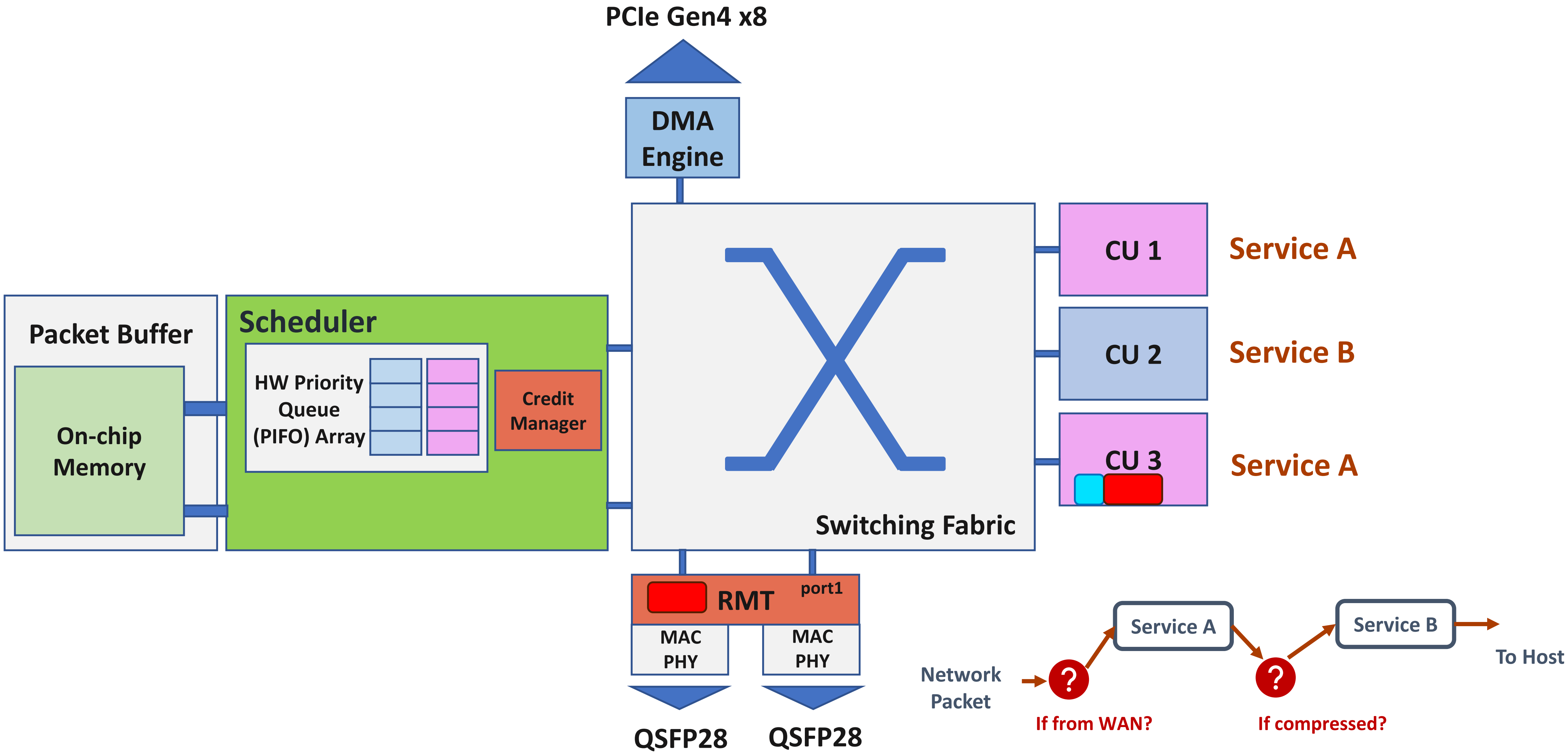
PANIC Components:

1. **Reconfigurable-Match-Action Pipeline**: Parse packets and determine offload chain
2. **Central Scheduler:** enforce isolation policies and schedule packets
3. **Independent Compute Unit:** Support hardware accelerator or CPU core
4. **High-throughput Switching Fabric:** Interconnects different hardware resources.

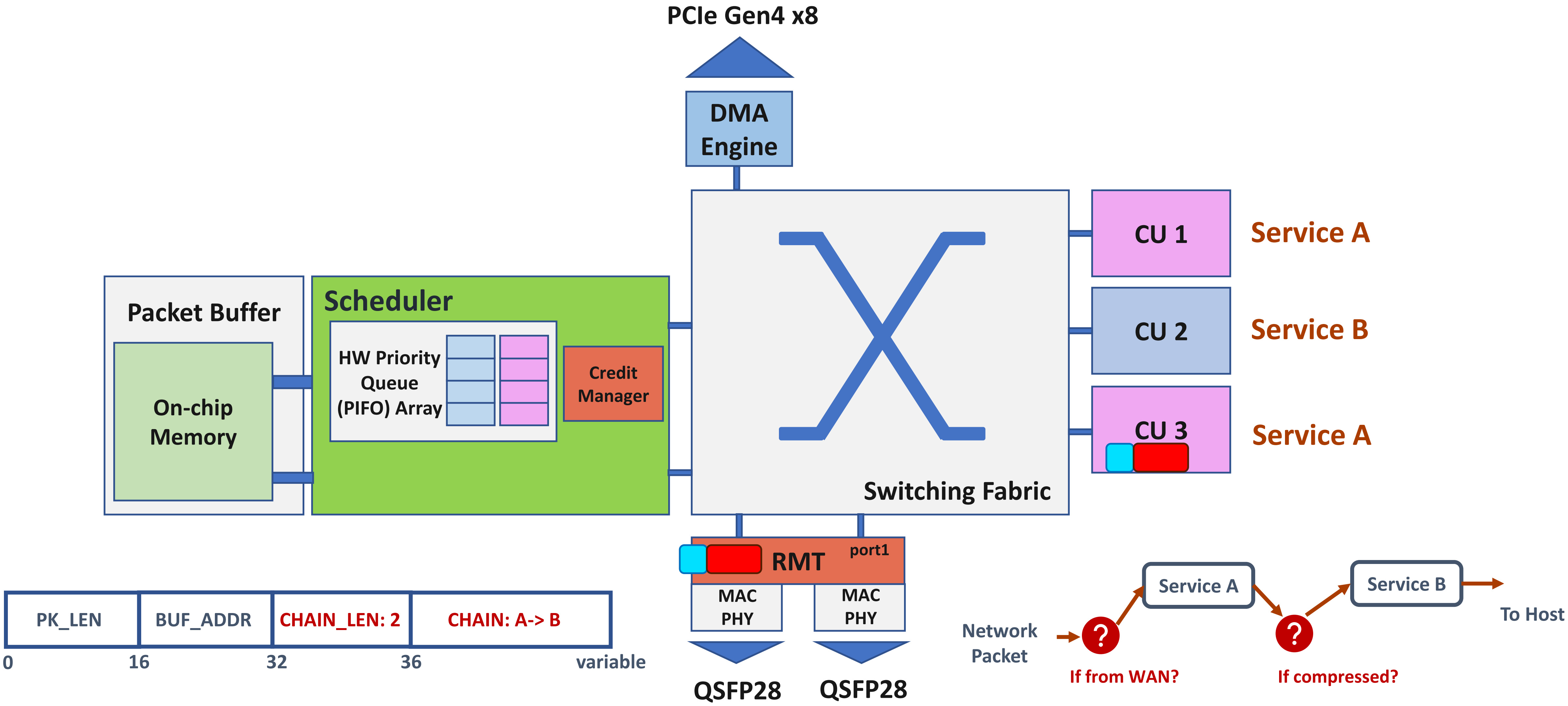
Life-Cycle of a Packet in PANIC



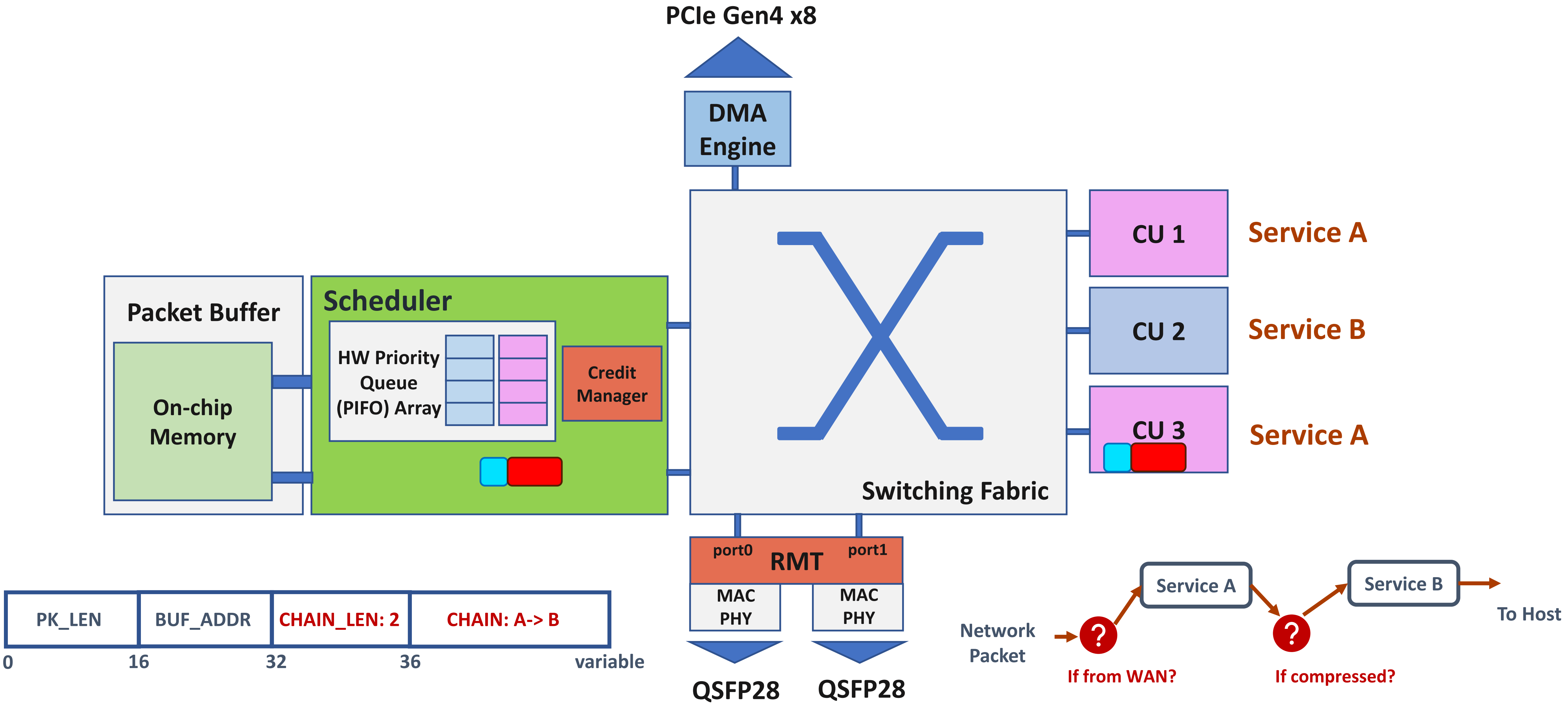
Life-Cycle of a Packet in PANIC



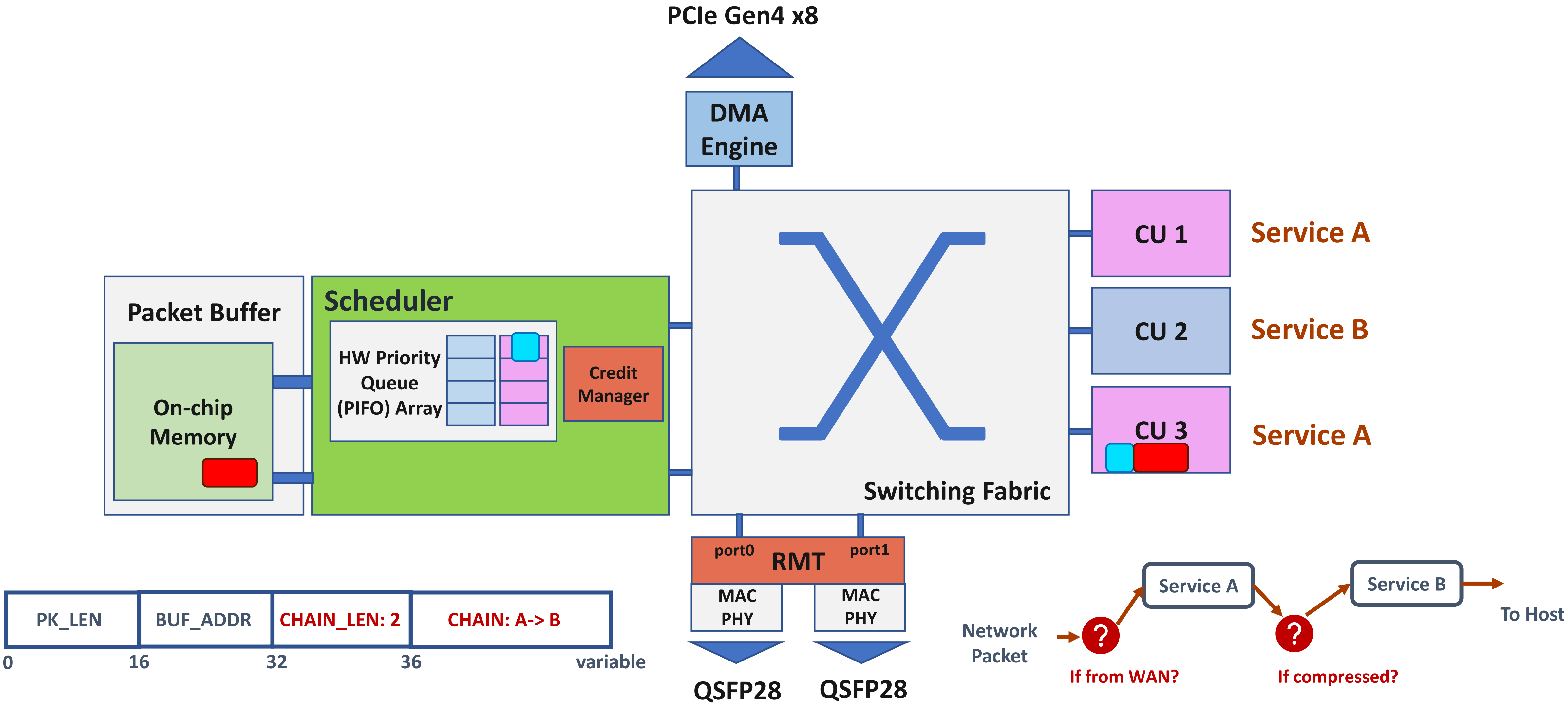
Life-Cycle of a Packet in PANIC



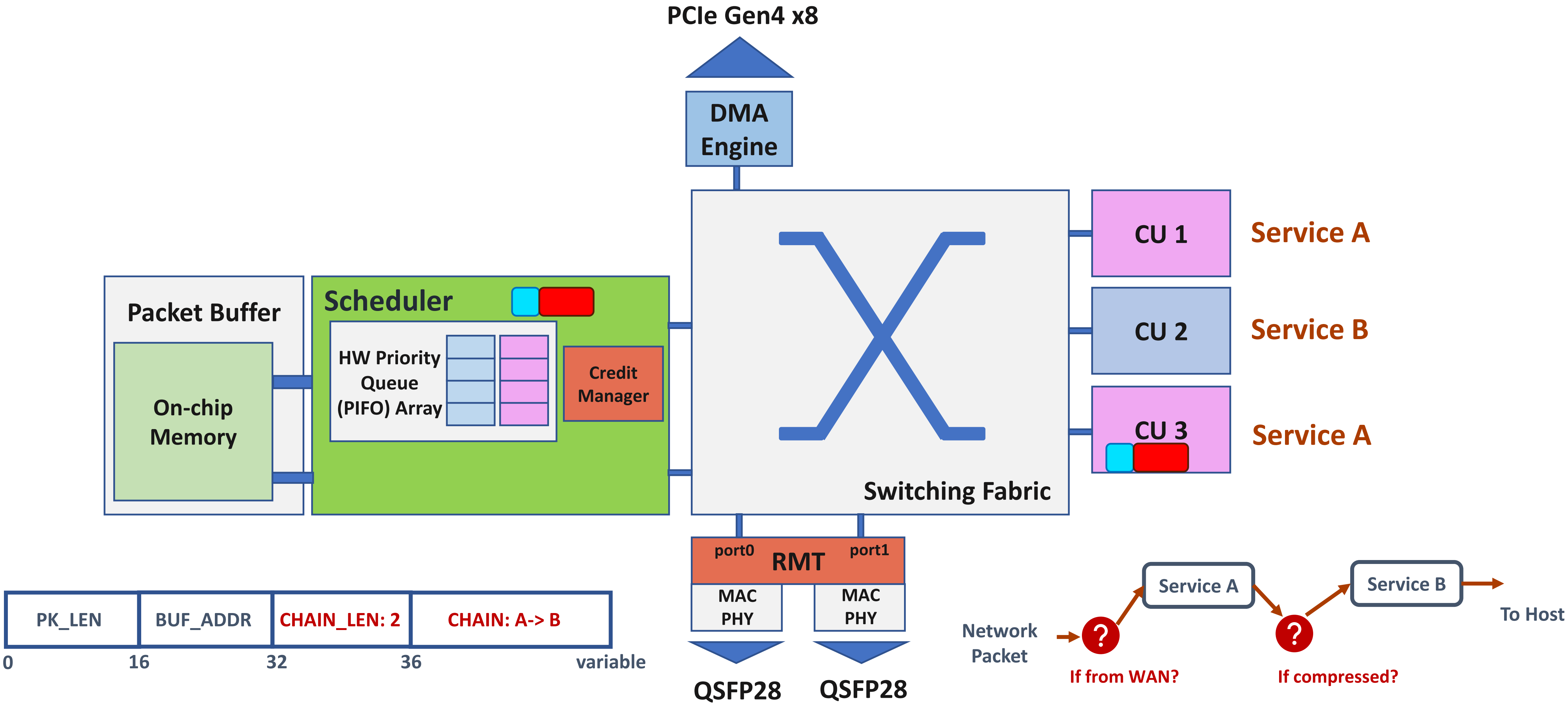
Life-Cycle of a Packet in PANIC



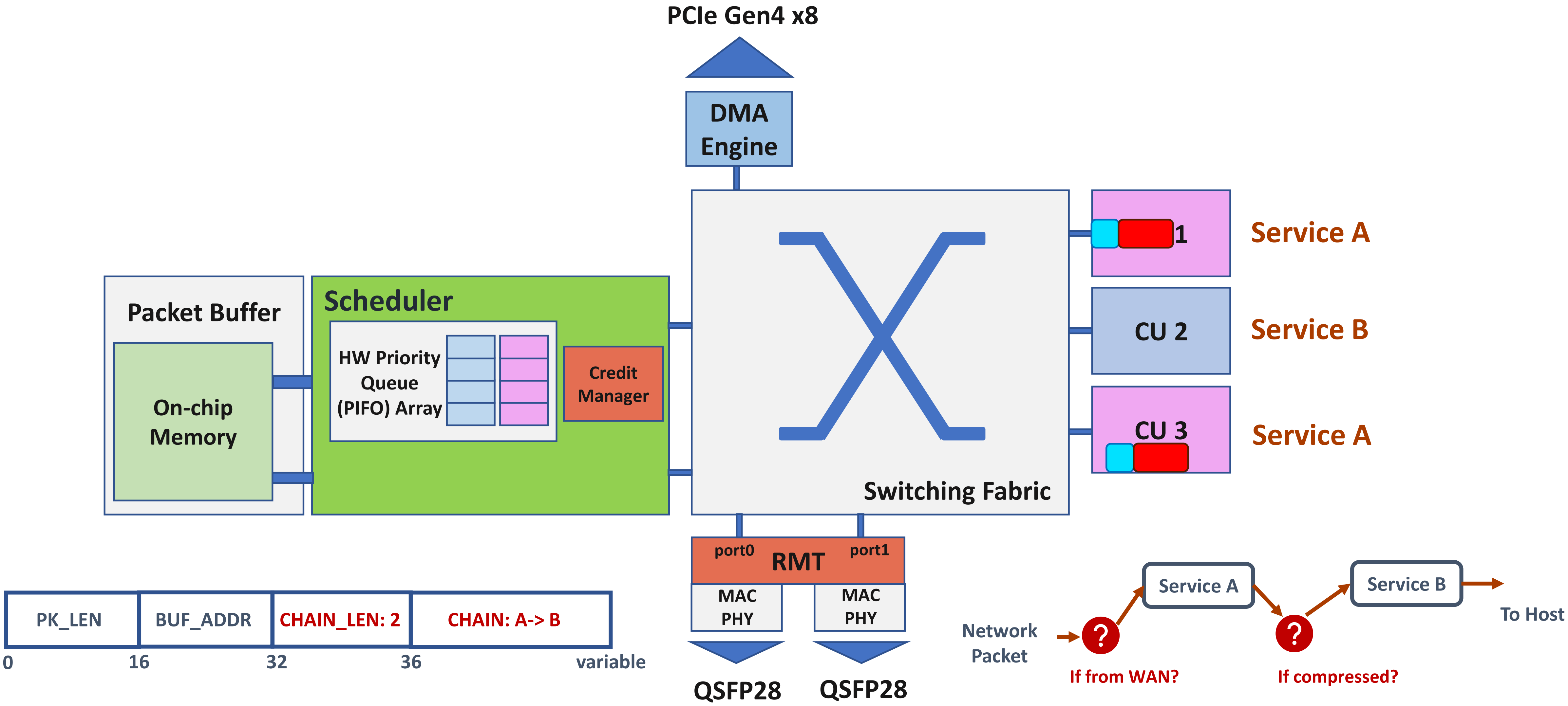
Life-Cycle of a Packet in PANIC



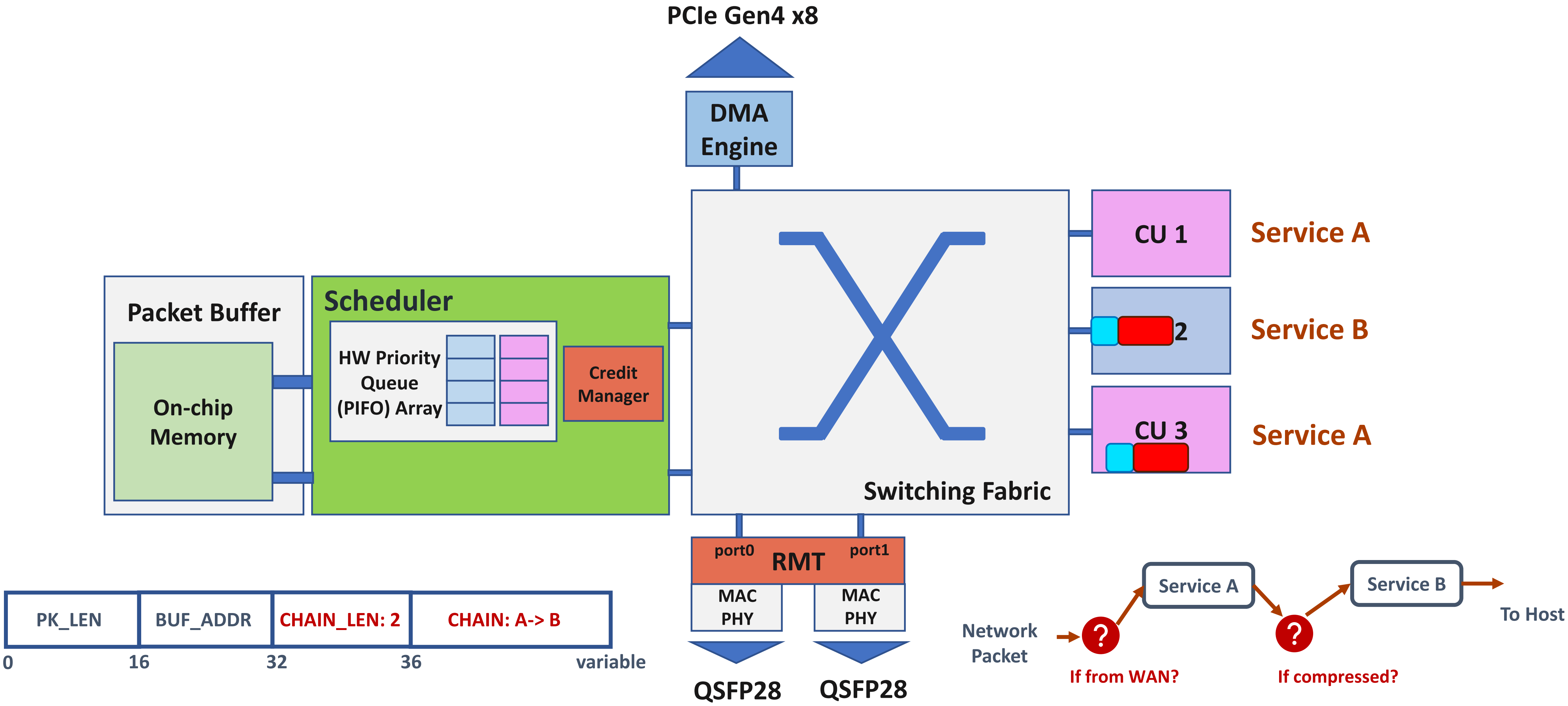
Life-Cycle of a Packet in PANIC



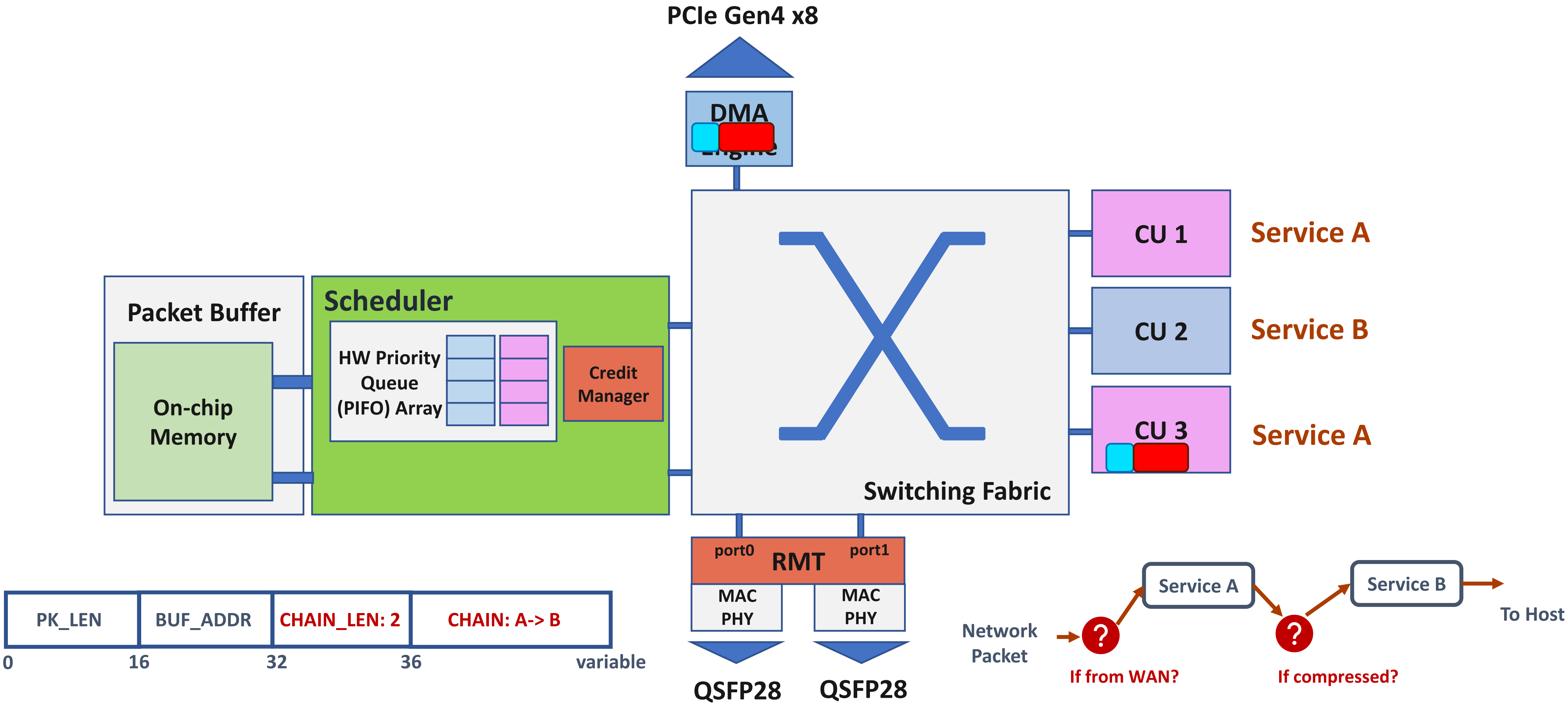
Life-Cycle of a Packet in PANIC



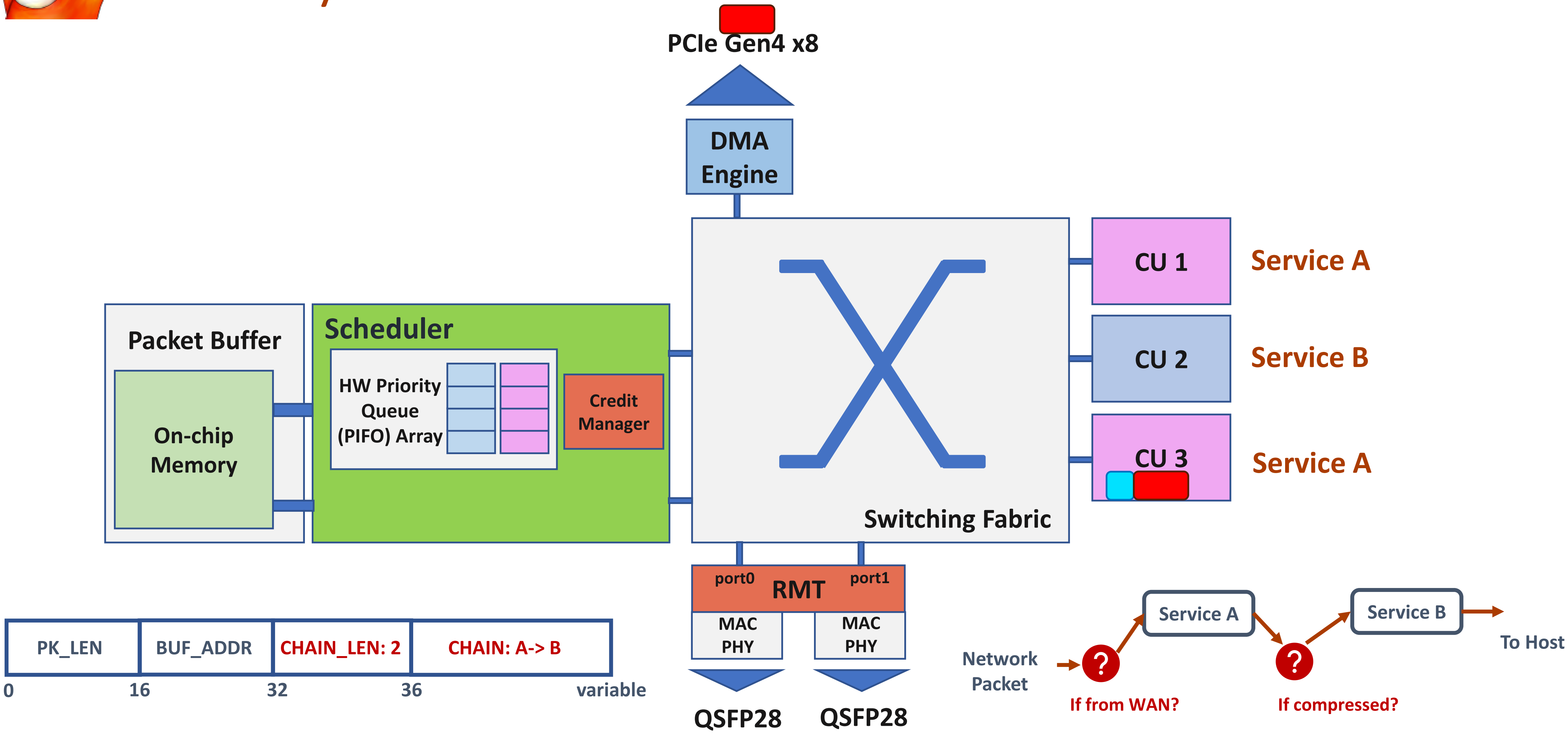
Life-Cycle of a Packet in PANIC



Life-Cycle of a Packet in PANIC



Life-Cycle of a Packet in PANIC



RMT Pipeline

- Reconfigurable Match Action Table
  - Programmers can specify the set of fields in a packet to match on.
  - Programmers compose the actions to modify packet fields.
- In PANIC, users program the match action tables:
  - **The service chain** for each tenant's traffic.
  - **The isolation policy and priority number** for each tenant's traffic.
- Action stage, RMT generates a PANIC descriptor for every packet

Traffic ID	Service Chain	<Isolation Policy, Priority (Weight)>
1	**A -> B**	**<Weighted Fair Queuing, 3>**
2	A -> C	<Weighted Fair Queuing, 2>
…	…	…

PK_LEN	BUF_ADDR	CHAIN_LEN: 2	CHAIN: A-> B	SCHE_METADATA: <WFQ, 3>

PANIC Scheduler:

Goal #1: Achieve high-performance chaining

Goal #2: Load-balance packets across parallel compute units in a service

Goal #3: Performance isolation across tenants

Goal #4: Buffer isolation across tenants

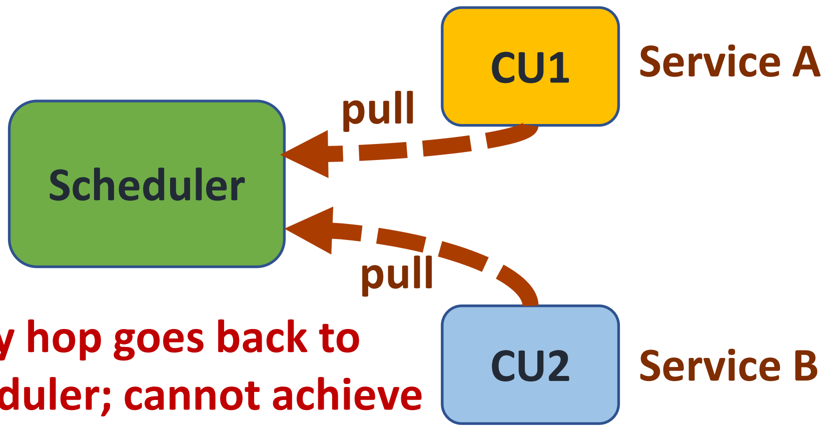
Problem: Chaining and Load Balancing

Goal #1: Achieve high-performance chaining

Goal #2: Load-balance packets across parallel compute units in a service.

- Pull-based scheduling

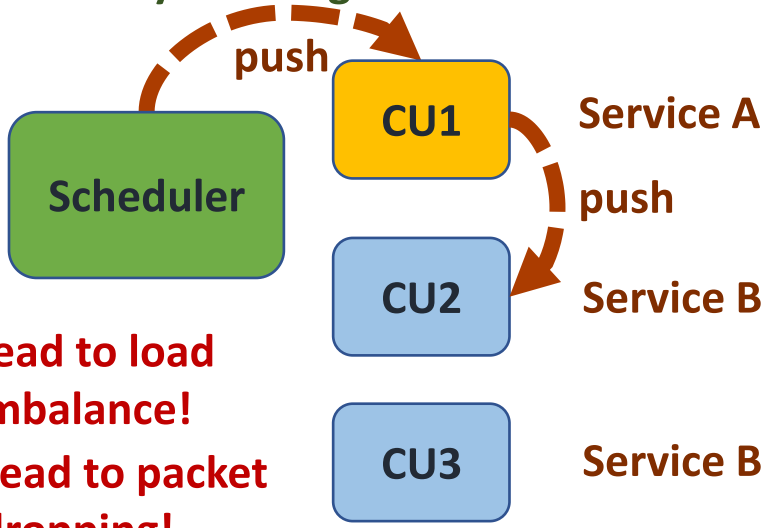
✓ Achieve Load balancing!



Every hop goes back to scheduler; cannot achieve high performance chaining!

- Push-based scheduling

✓ Low latency chaining!

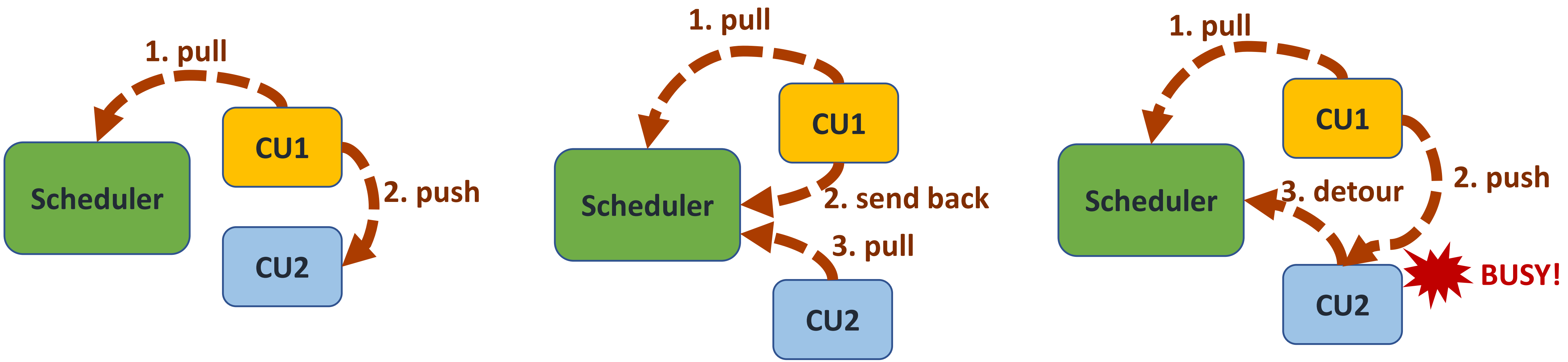


Lead to load imbalance!

Lead to packet dropping!

PANIC Scheduler: Hybrid Push Pull Scheduling

- **Hybrid Push Pull scheduling:**
  - Compute units can either **pull** packet from the scheduler or **accept the pushed** packet from other units.
  - According to CUs' load, switches between push pull scheduling.
    - **During Low Load:** the packet is pushed to all the units in a chain.
    - **During High Load:** the packet is sent back to the scheduler, until it can be pulled by an idle CU.
    - **Detour Routing** : In push scheduling, if the downstream is busy due to a burst.

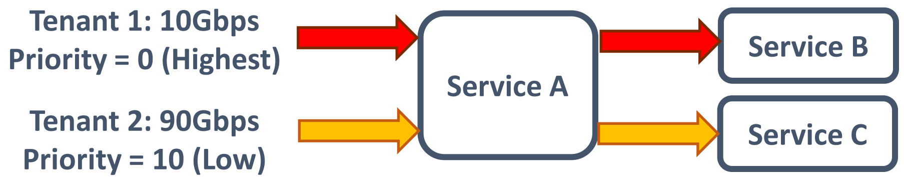


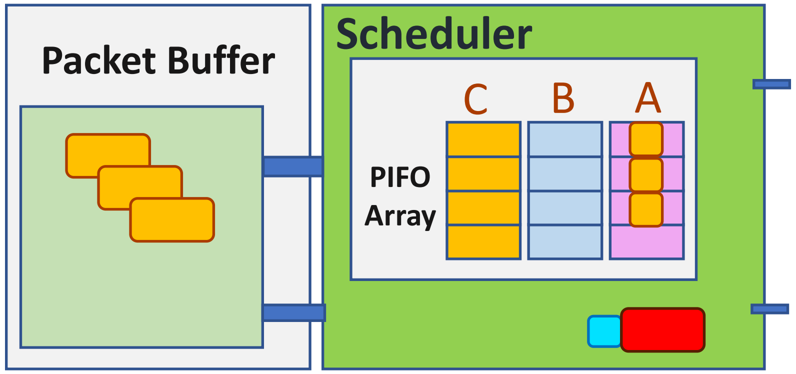
PANIC Scheduler: Performance Isolation

Goal #3: Priority scheduling and performance isolation

- **PIFO array for performance Isolation:**
  - PIFO (*PUSH IN, FIRST OUT Queue*) runs like the hardware priority queue. One service has one logic PIFO queue.
  - Packet descriptors is sorted according to the packet rank in per-service PIFO.

Isolation Policy:
Strict Priority



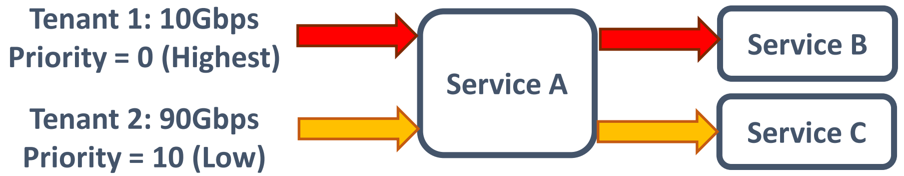


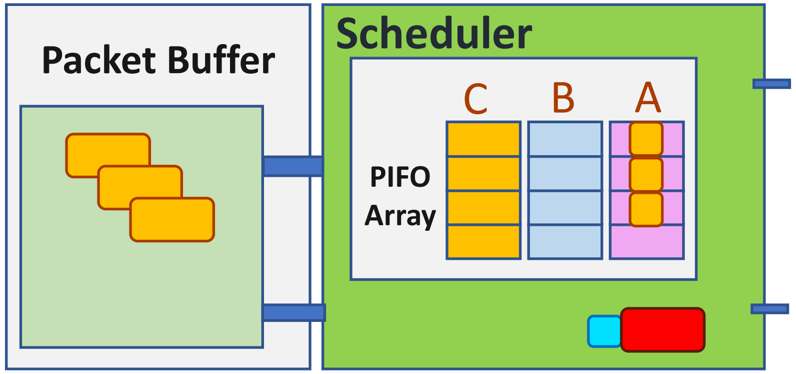
PANIC Scheduler: Performance Isolation

Goal #3: Priority scheduling and performance isolation

- **PIFO array for performance Isolation:**
  - PIFO (*PUSH IN, FIRST OUT Queue*) runs like the hardware priority queue. One service has one logic PIFO queue.
  - Packet descriptors is sorted according to the packet rank in per-service PIFO.

Isolation Policy:
Strict Priority





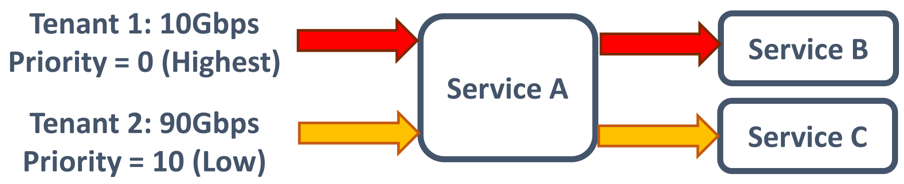
PK_LEN	BUF_ADDR	CHAIN_LEN: 2	CHAIN: A-> B	SCHE_METADATA: <Strict, High>

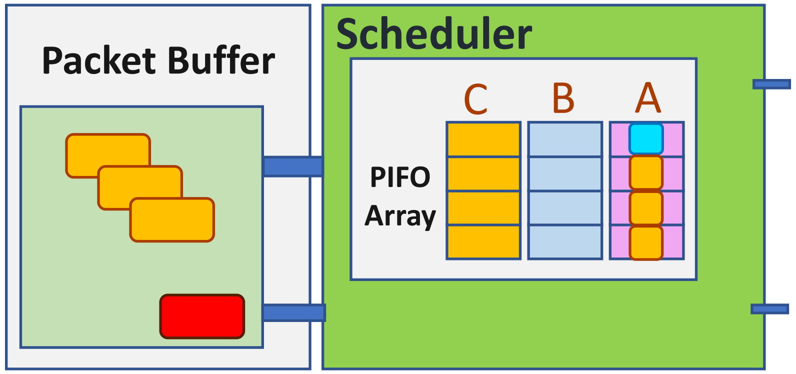
PANIC Scheduler: Performance Isolation

Goal #3: Priority scheduling and performance isolation

- **PIFO array for performance Isolation:**
  - PIFO (*PUSH IN, FIRST OUT Queue*) runs like the hardware priority queue. One service has one logic PIFO queue.
  - Packet descriptors is sorted according to the packet rank in per-service PIFO.
  - Support different isolation policy
    - (WFQ, LSTF, Rate Limiting…)

Isolation Policy: Strict Priority





PK_LEN	BUF_ADDR	CHAIN_LEN: 2	CHAIN: A-> B	SCHE_METADATA: <Strict, High>

Problem:
Buffer Isolation

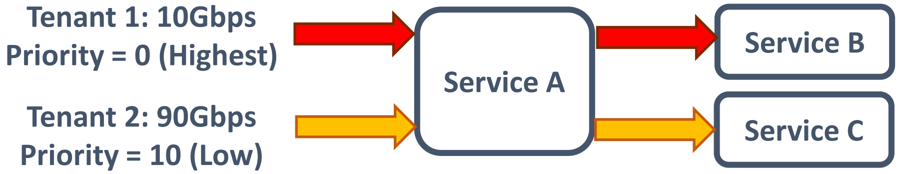
Goal #4: Ensure buffer isolation across tenants

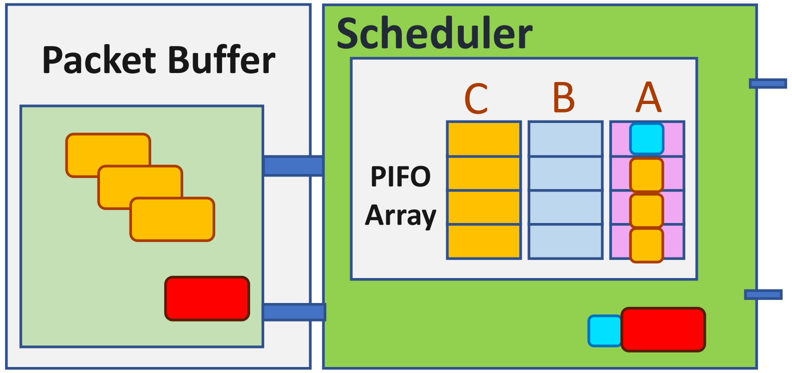
- **Naïve Packet dropping method:** drop the newest income packet when the buffer is full
  - **No Isolation!** A high-volume low-priority flow can lead to packet loss for a high priority flow.

PANIC Scheduler: Prioritized Dropping

- **Prioritized Dropping:** drop the lowest rank packet when the buffer is almost full.
  - Extend PIFO's interface to allow it to support *PUSH IN, FIRST OUT, **REMOVE LAST***
  - **Isolation!** PANIC can ensure the high priority packet enters buffer and receive service.

Isolation Policy:
Strict Priority

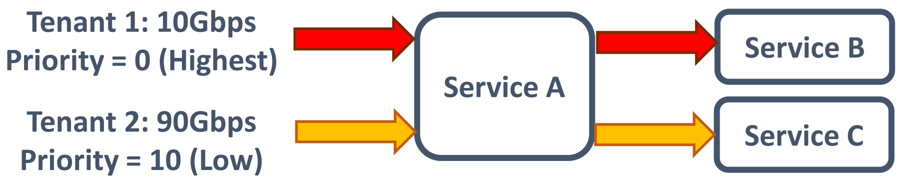


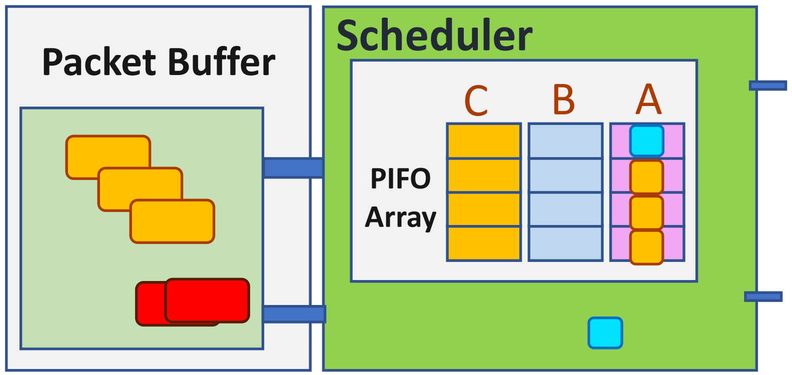


PANIC Scheduler: Prioritized Dropping

- **Prioritized Dropping:** drop the lowest rank packet when the buffer is almost full.
  - Extend PIFO's interface to allow it to support *PUSH IN, FIRST OUT,* ***REMOVE LAST***
  - **Isolation!** PANIC can ensure the high priority packet enters buffer and receive service.

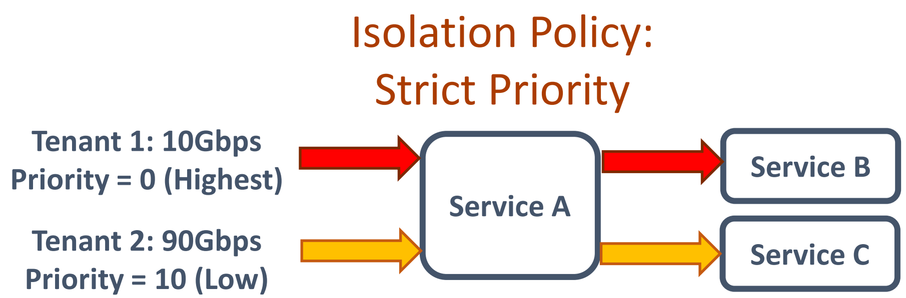
Isolation Policy:
Strict Priority

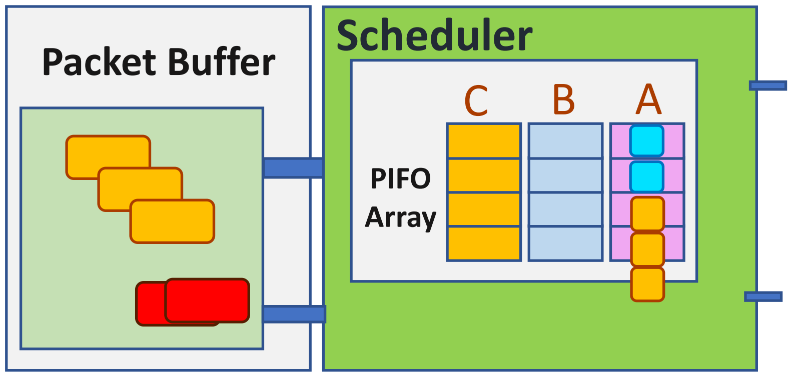




PANIC Scheduler: Prioritized Dropping

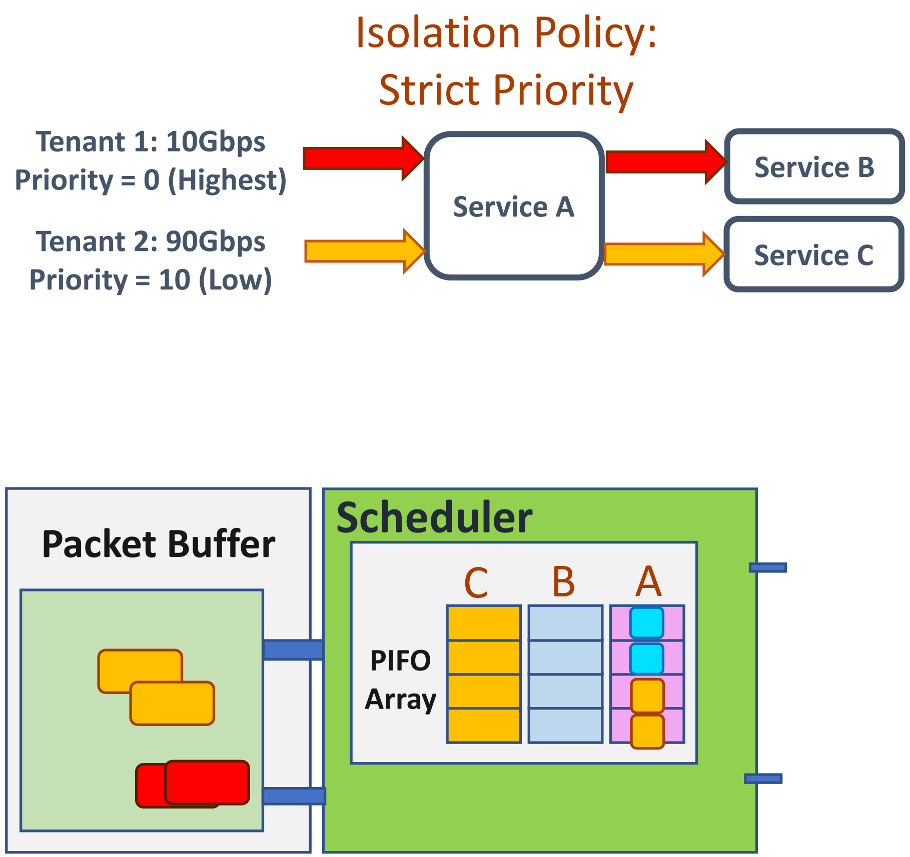
- **Prioritized Dropping:** drop the lowest rank packet when the buffer is almost full.
  - Extend PIFO's interface to allow it to support *PUSH IN, FIRST OUT,* ***REMOVE LAST***
  - **Isolation!** PANIC can ensure the high priority packet enters buffer and receive service.





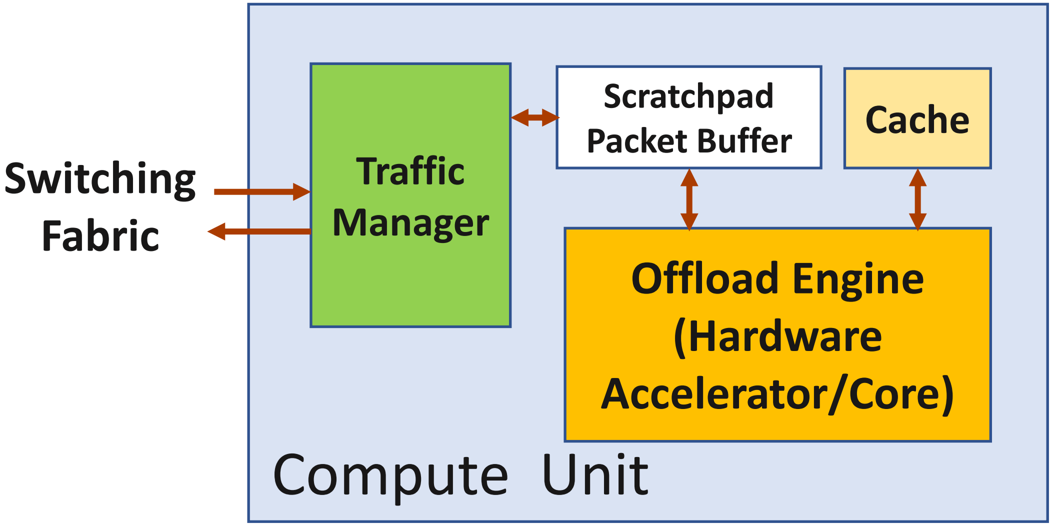
PANIC Scheduler: Prioritized Dropping

- **Prioritized Dropping:** drop the lowest rank packet when the buffer is almost full.
  - Extend PIFO's interface to allow it to support *PUSH IN, FIRST OUT,* ***REMOVE LAST***
  - **Isolation!** PANIC can ensure the high priority packet enters buffer and receive service.



Compute Unit

- Compute Unit can either be a CPU core or a hardware accelerator, variable/non-line rate
- Traffic Manager handles communication with scheduler
  - Offloads Engine can be designed without needing to understand other PANIC components.

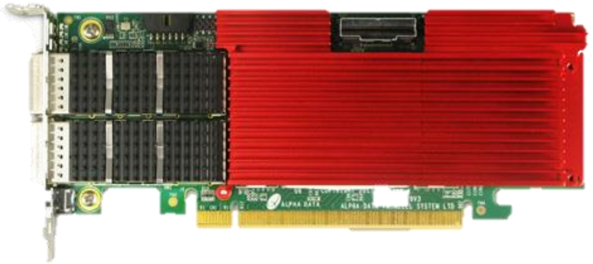


Switching Fabric

- Providing offload chaining for an arbitrary chain.
- The switching fabric is non-blocking and high-throughput
  - Each interconnect port should send and receive at full line-rate (> 100Gbps)
    - A crossbar with a bit width of 512 bits at 250 MHz frequency.
    - Scalability? NoC topology is selected according to the CU number.

PANIC Implementation

- 100G FPGA prototype in ADM-PCIE-9V3 accelerator
- ~6K lines of Verilog code
- Prototype Components:
  - A lightweight RMT pipeline
  - 8 * 8 full connected crossbar (512 bit width @ 250MHz)
  - Dual-port central scheduler (512 bit width @ 250MHz)
    - PIFO block @ 125MHz
  - Compute Units
    - **AES-256-CTR** encryption unit (24Gbps @ 250Mhz)
    - **SHA-3-512** hash unit (32Gbps @ 150Mhz)
    - **An RISC-V core** unit (5-stage pipeline @ 250MHz)



Open Source

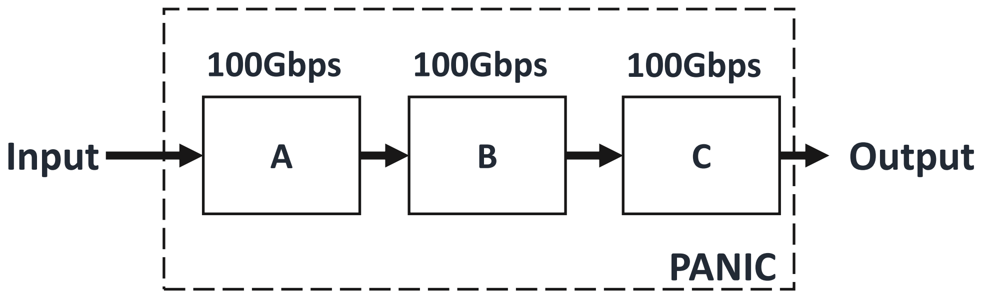
PANIC Evaluation

- Can PANIC achieve high throughput and low latency under different chaining models ?
- Can PANIC isolate traffic using different isolation policies?
- What is the hardware resource consumption of PANIC?

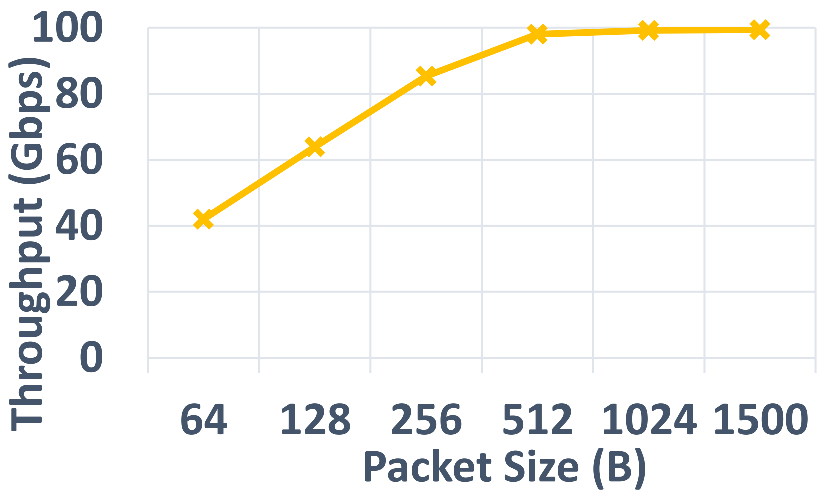
PANIC Evaluation

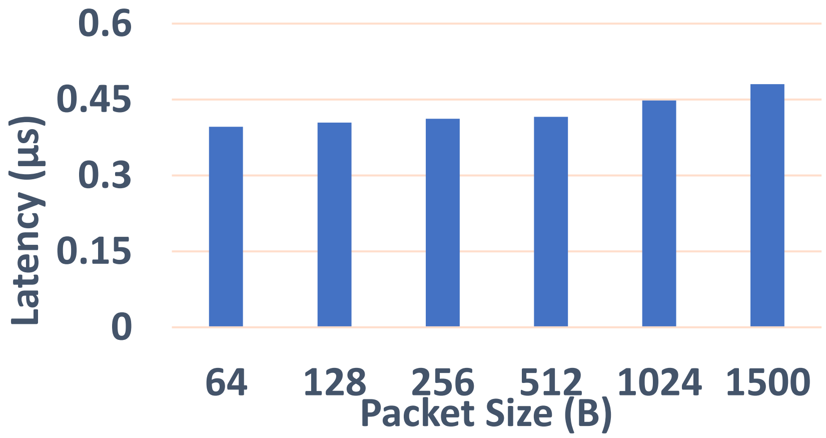
Can PANIC achieve high throughput and low latency under different chaining models ?

Chaining Model 1: Pipelined Chain



Setup:
- Use delay unit emulate real compute unit.
- Single line-rate delay unit for each service.
- Each delay unit initially has 8 credits



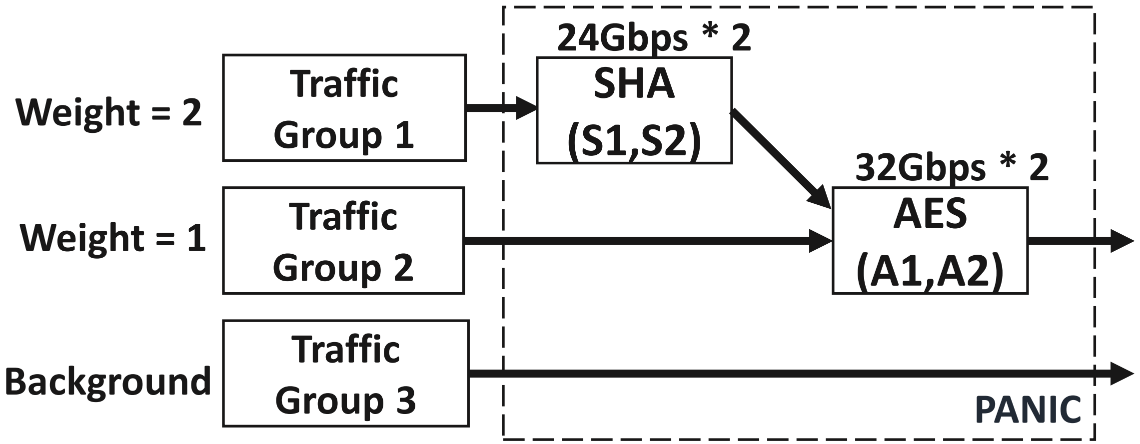


PANIC achieves 100G throughput and low latency chaining!

PANIC Evaluation

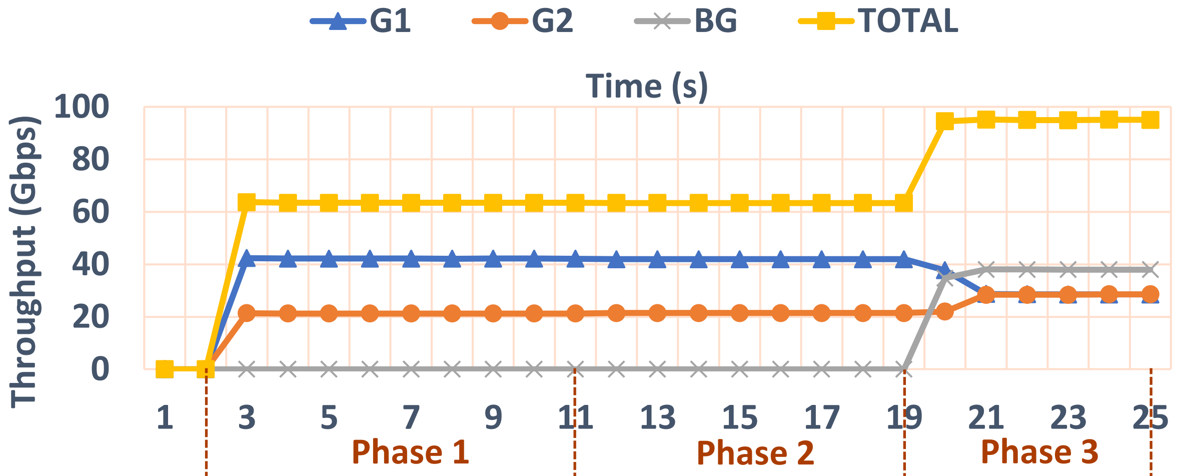
Can PANIC isolate traffic?

End-to-End experiment



Setup:

- The sender server generate network traffic using different traffic pattern.
- 2 SHA engines and 2 AES engines are attached.
- Isolation Policy: weighted fair queuing.



Traffic Pattern

	Phase 1	Phase 2	Phase 3
G1	70Gbps	50Gbps	30Gbps
G2	30Gbps	50Gbps	30Gbps
BG	0Gbps	0Gbps	40Gbps

PANIC can isolate traffic using different isolation policy!

PANIC Evaluation

What is the hardware resource consumption of PANIC?

- Resource Usage in ADM-PCIE-9V3
  - Most of the on-chip logic resources in PANIC is occupied by crossbar and PIFO.
  - In total, PANIC only cost ~11% LUT resource and ~9% BRAM resource.

Module	Setting	LUT (%)	BRAM (%)
Crossbar	8 ports	5.5	0.00
Scheduler (PIFO)	PIFO size = 256	5.18 (4.9)	0.07 (0.01)
Packet Buffer	256 KB	0.16	8.94
Simple RMT	/	0.27	0.00

PANIC can easily fit on any middle-end FPGA without utilization or timing issues.

Conclusion

- PANIC is a full line-rate programmable NIC design that overcomes current NICs limitation in multi-tenant environments.

Thank you!