TNS LAB
Trustworthy Networked System

UNIVERSITY OF SEOUL

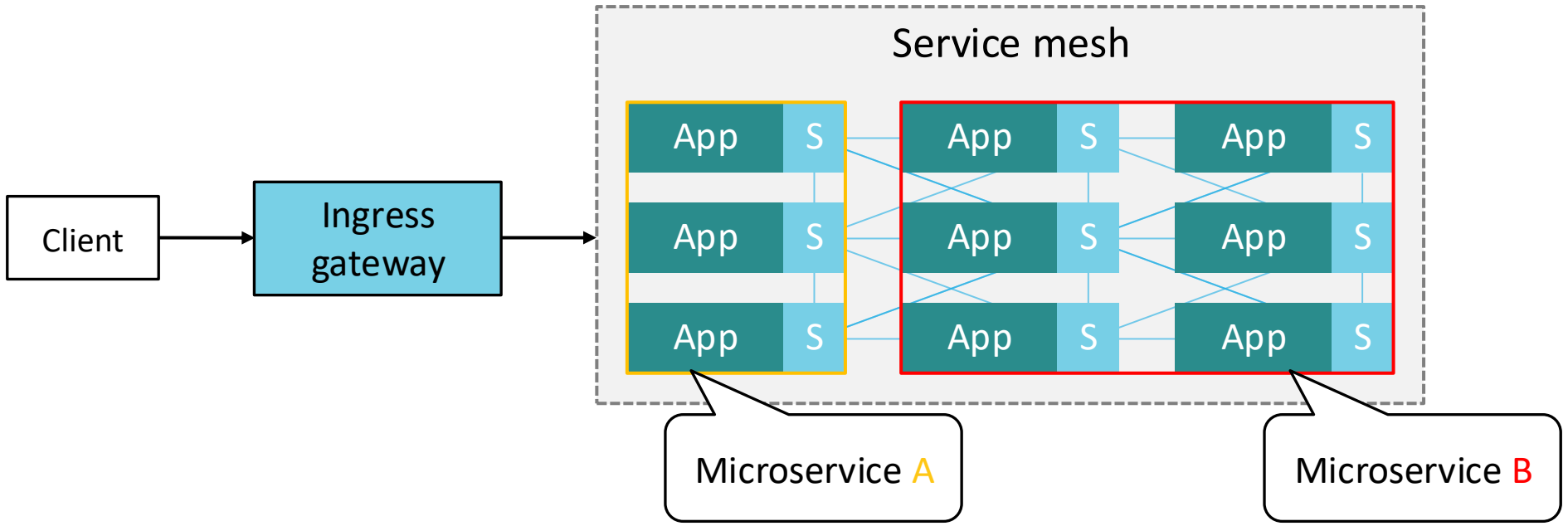
HybridMesh: A Hardware-software Hybrid Approach for Accelerating Service Mesh Ingress

Myoungsung You[1], Jaehyun Nam[2], Minjae Seo[3], Taejune Park[4], and Seungwon Shin[5]

University of Seoul[1], Dankook University[2], ETRI[3], Chonnam National University[4], KAIST[5]

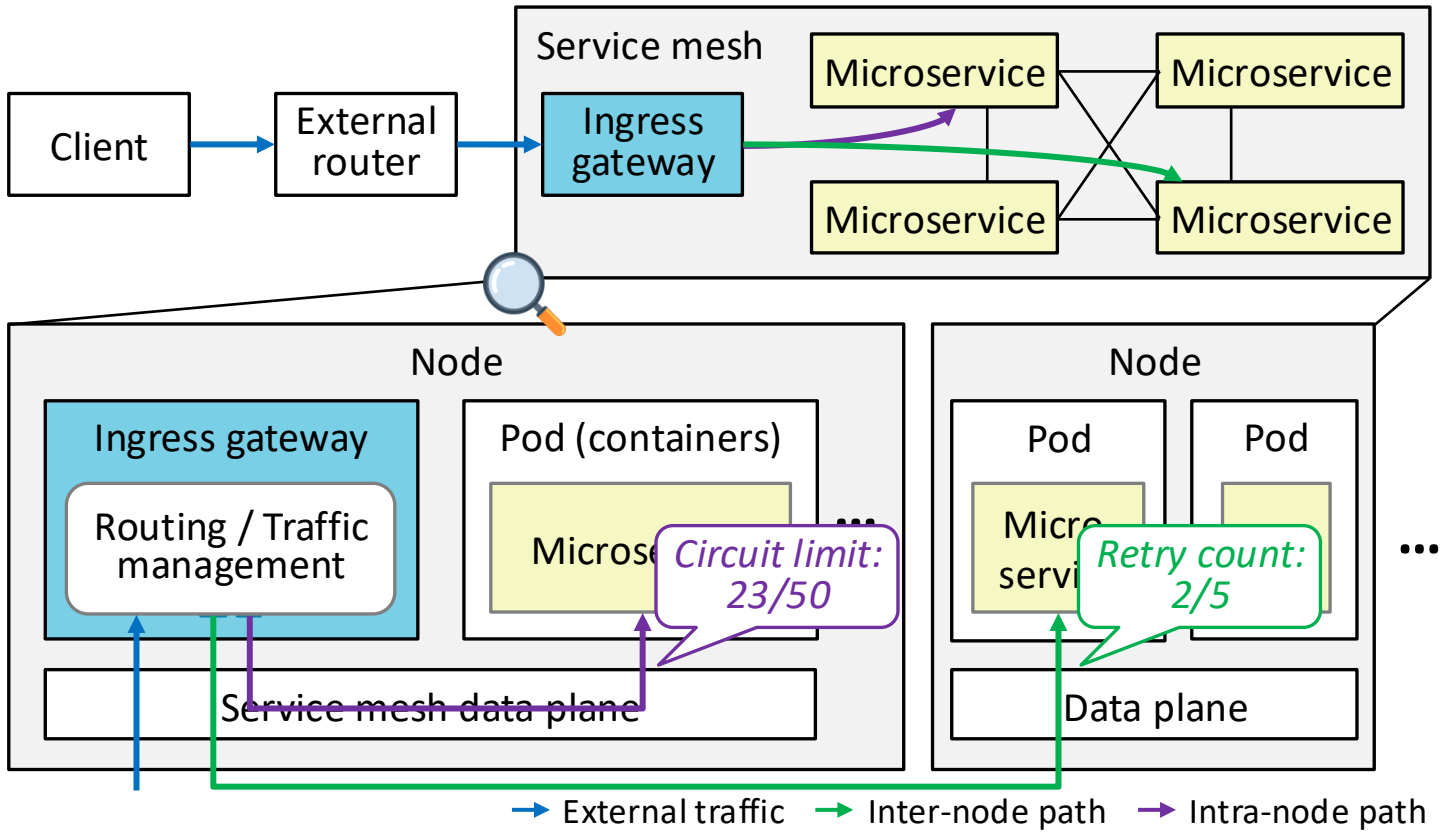
Service Mesh

- **Service mesh** has become the de facto standard for microservices, providing an infrastructure layer
- Consisting of a set of proxies: sidecars and an ingress gateway
- Providing inter-service communication functionalities (service discovery, load-balancing)



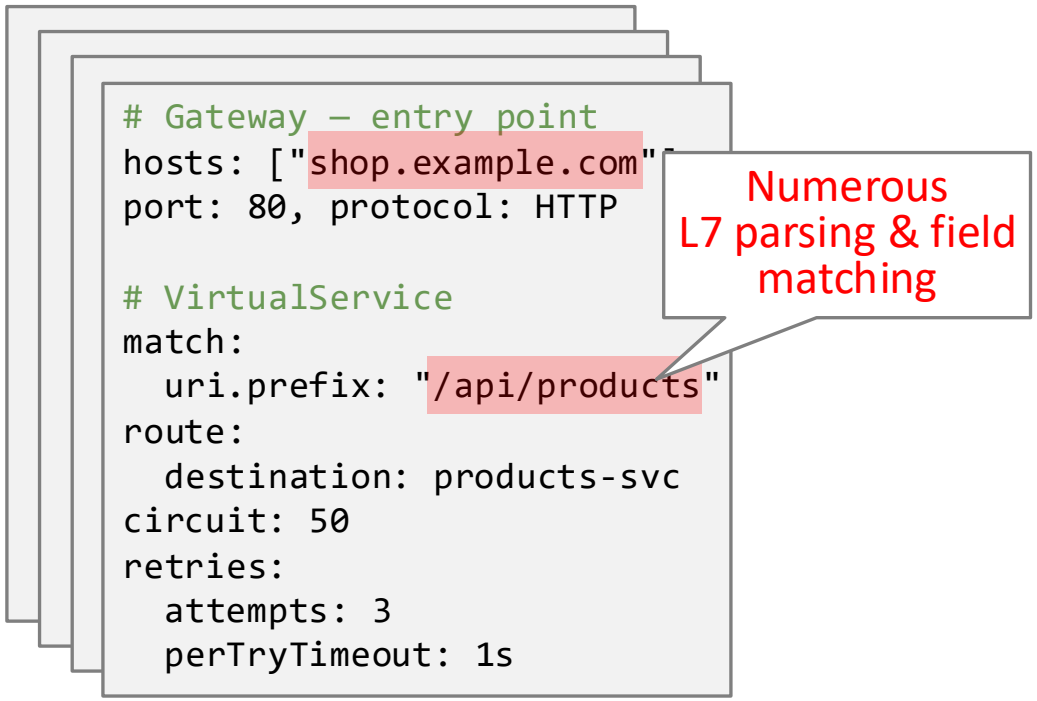
Ingress Gateway for Service Mesh

- The ingress gateway is the main entrance point for external traffic
  - Routing traffic to destination pods and providing traffic management features (e.g., circuit breaking)



Performance Challenges in Ingress Gateways

- The ingress gateway becomes a main bottleneck of the service mesh architecture
  - ***CPU-intensive traffic analysis*** due to L7 (e.g., HTTP REST API) traffic processing



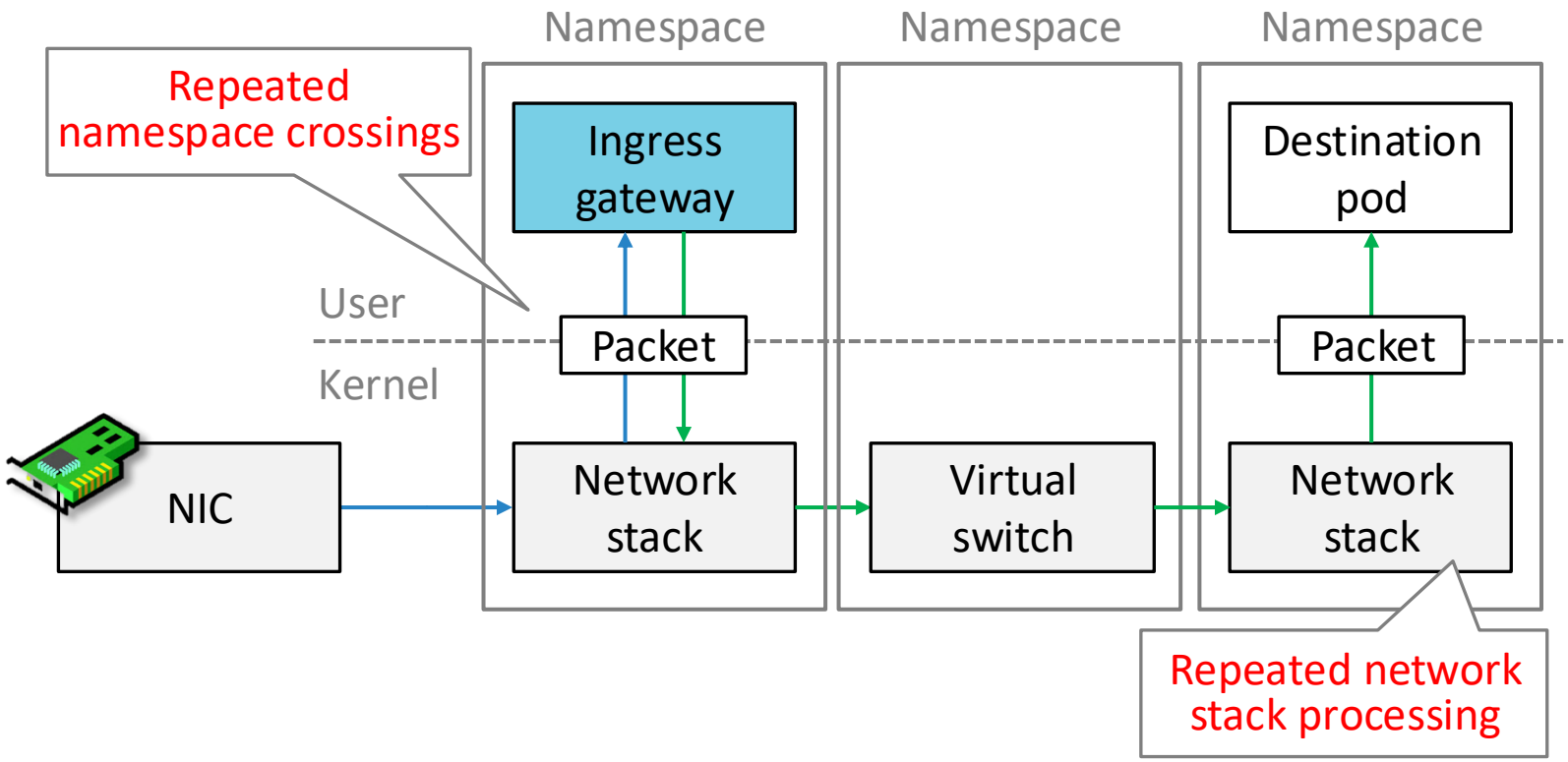

```
# Gateway – entry point
hosts: ["shop.example.com"]
port: 80, protocol: HTTP

# VirtualService
match:
  uri.prefix: "/api/products"
route:
  destination: products-svc
circuit: 50
retries:
  attempts: 3
  perTryTimeout: 1s
```

Ingress processing task	CPU	Type
Downstream processing	5.64%	Control
Traffic analysis/matching	**39.22%**	**Compute**
Traffic management	8.78%	Control
Upstream processing	5.96%	Control
Others	30.4%	Control

Performance Challenges in Ingress Gateways

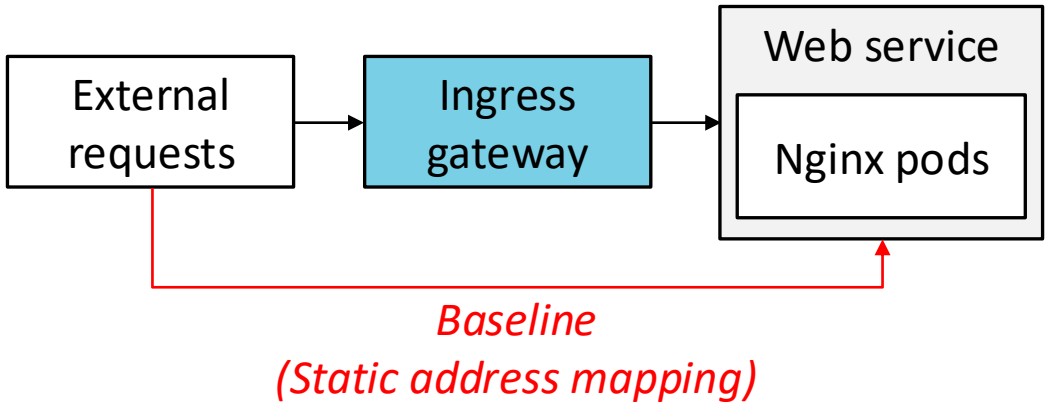
- The ingress gateway becomes a main bottleneck of the service mesh architecture
  - ***long and complex packet forwarding path*** between the gateway and containers

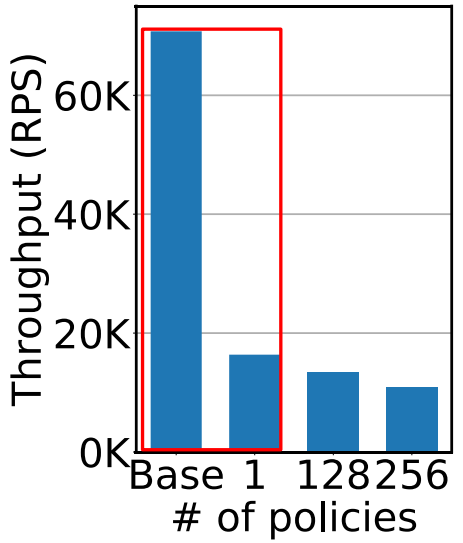


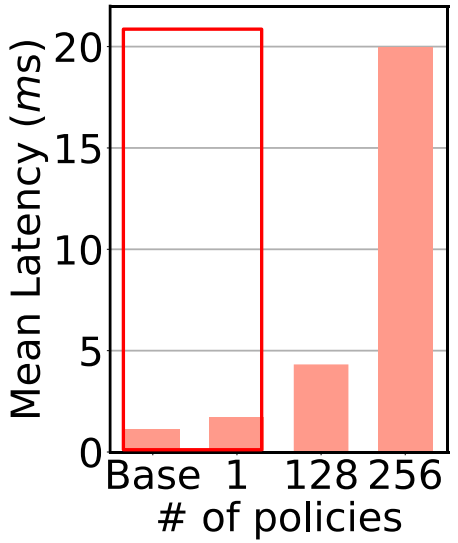
Performance Challenges in Ingress Gateways

- The ingress gateway becomes a main bottleneck of the service mesh architecture
  - ***CPU-intensive traffic analysis*** due to L7 (e.g., HTTP/gRPC) traffic processing
  - ***Long and complex packet forwarding path*** between the gateway and containers

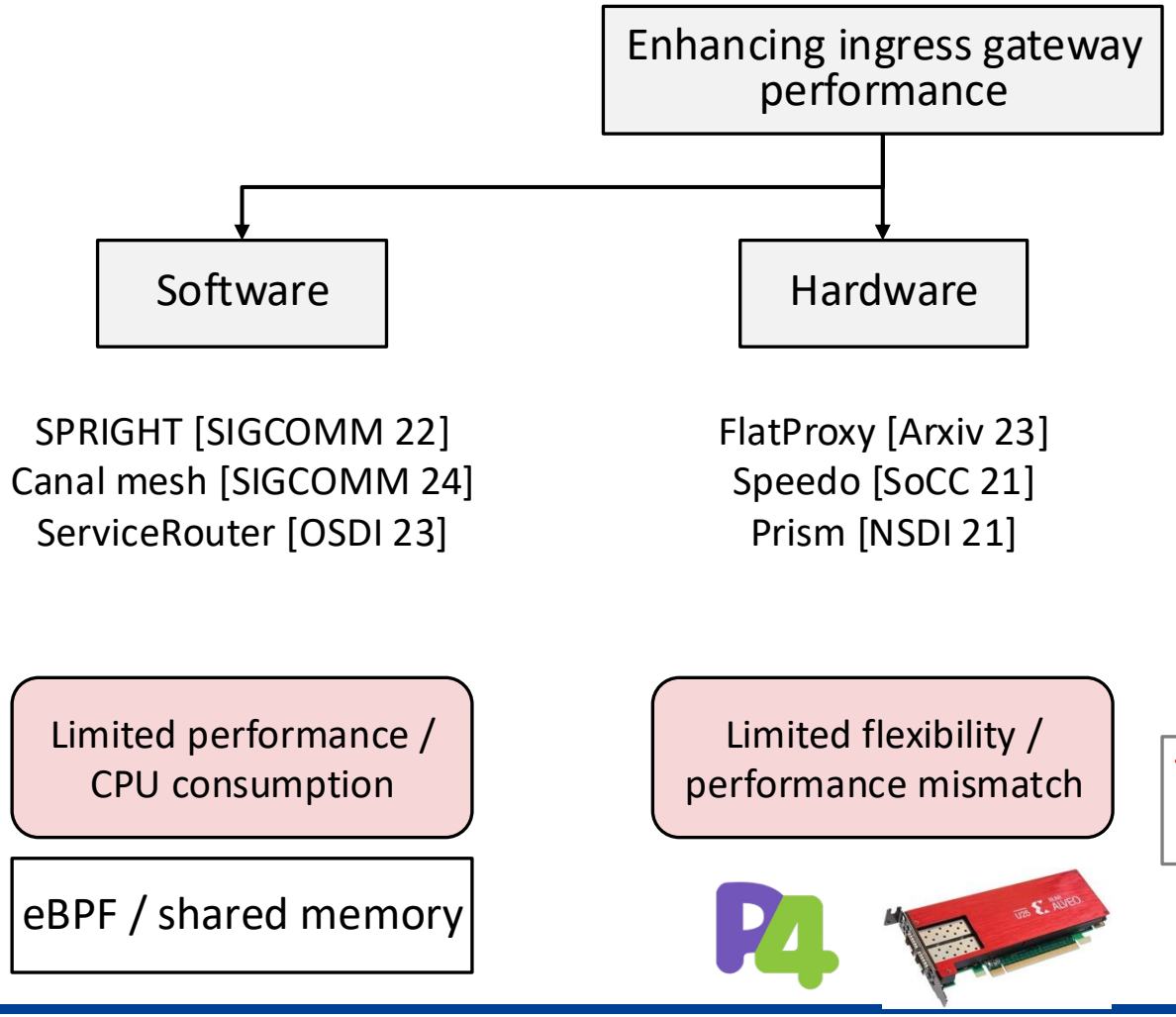
- → Resulting in up to a ***6.5×*** drop in throughput and a ***18×*** increase in latency

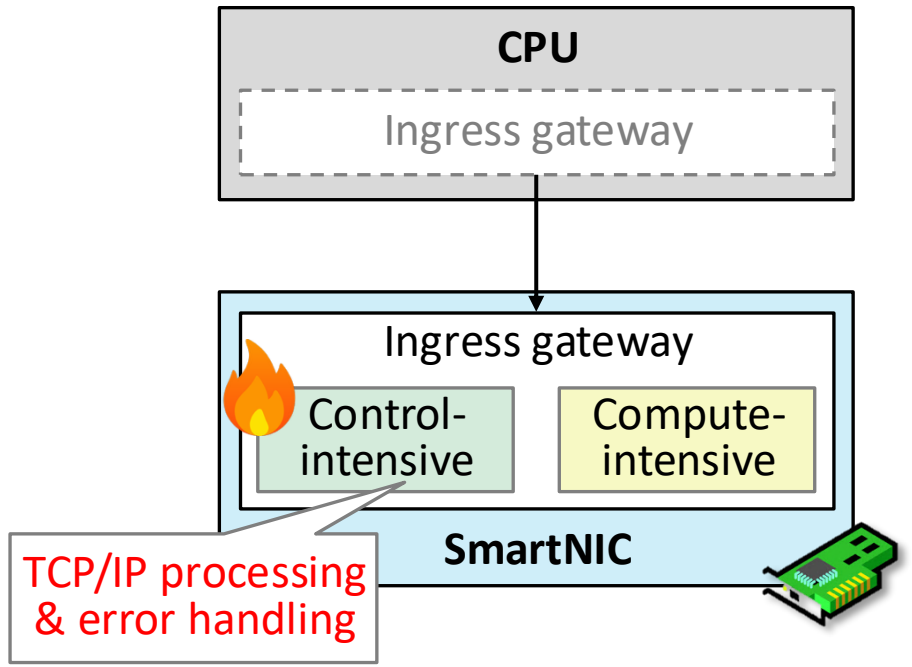




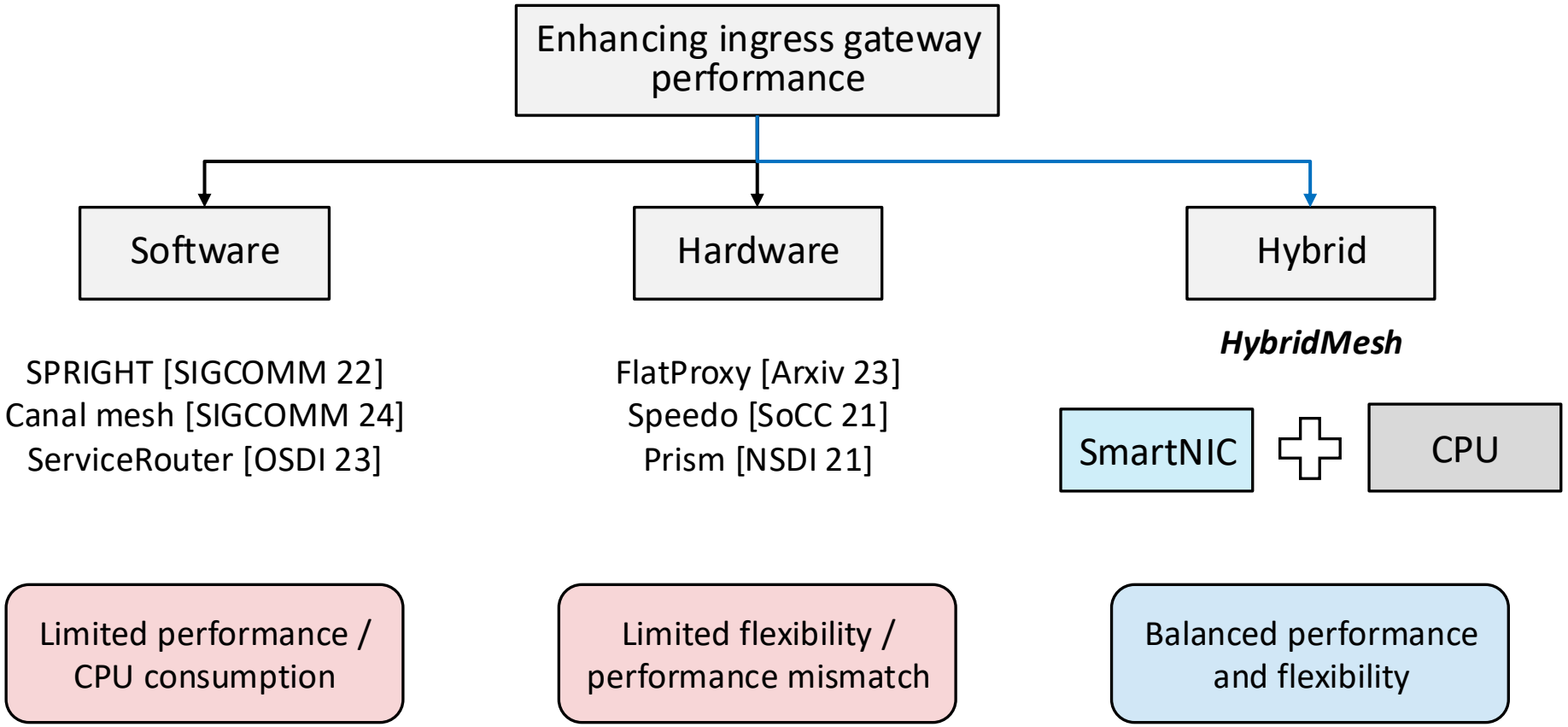


Design Space



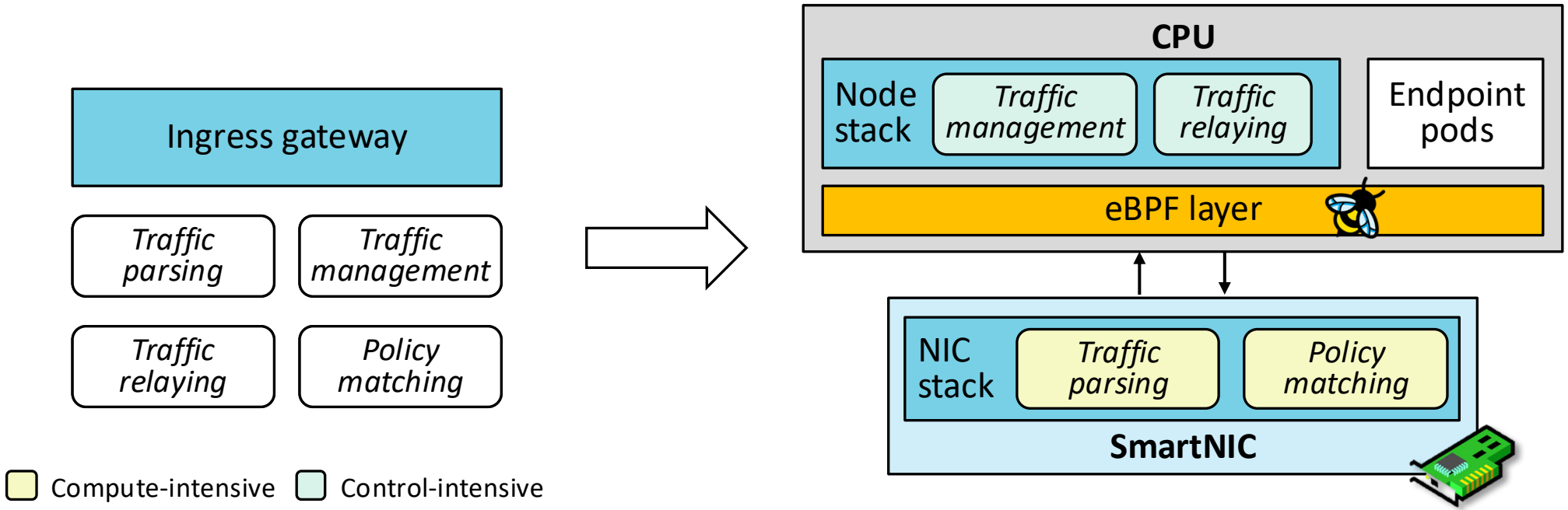


Design Space



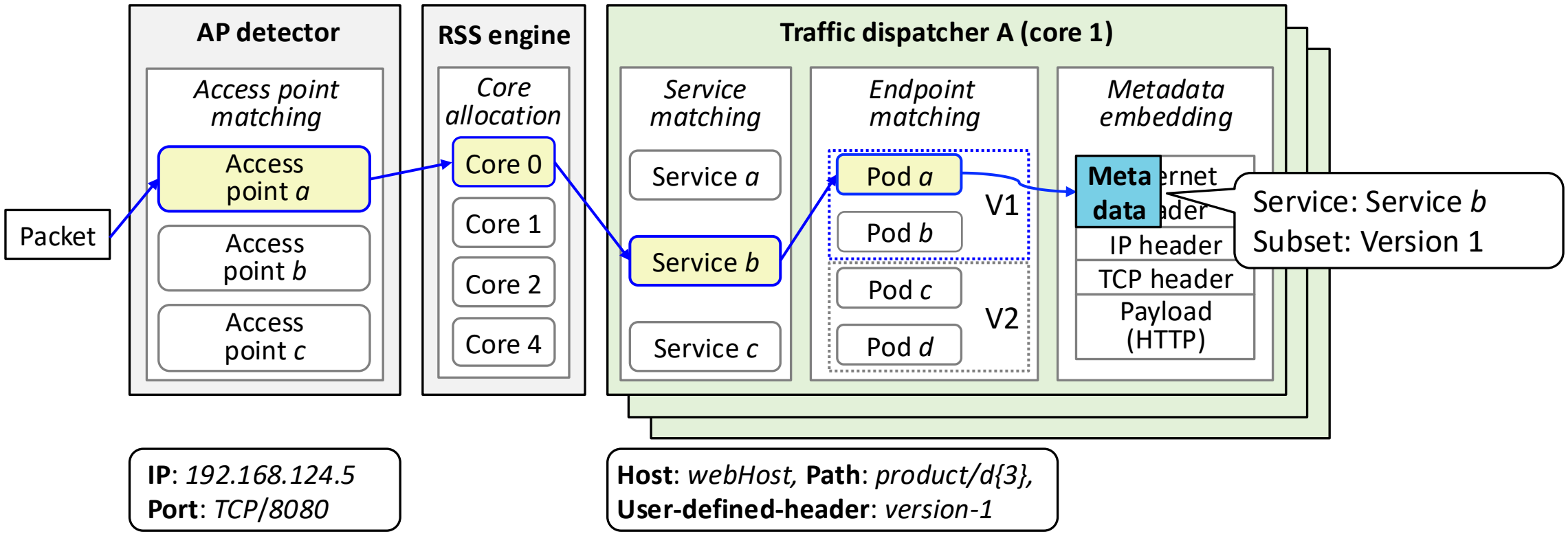
HybridMesh

- **Disaggregating an ingress gateway**: A hardware-software hybrid ingress gateway for service meshes
  - A ***SmartNIC*** for accelerating traffic analysis tasks
  - A lightweight CPU-side proxy for maintaining flexible traffic management features
  - ***eBPF***-based optimization for bridging these components seamlessly



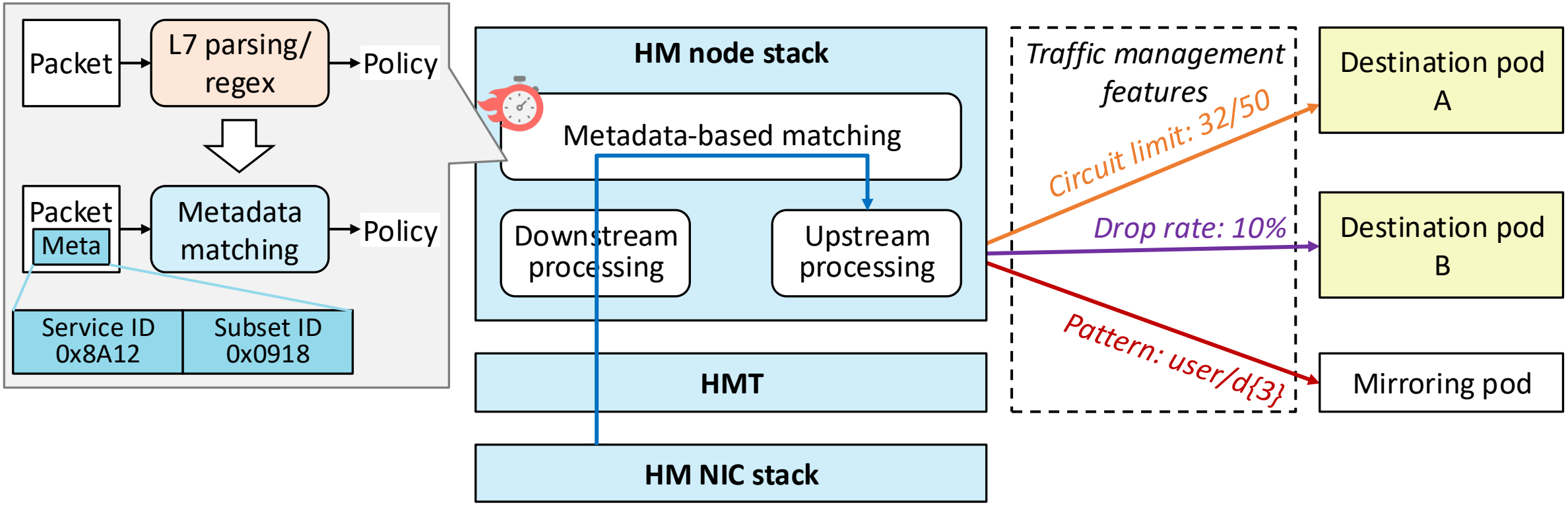
Accelerating Traffic Analysis Tasks with the NIC stack

- The *HM* NIC stack determines destinations of incoming packets (services, subsets, or pods)
  - The processing results are embedded into packet as ***metadata***



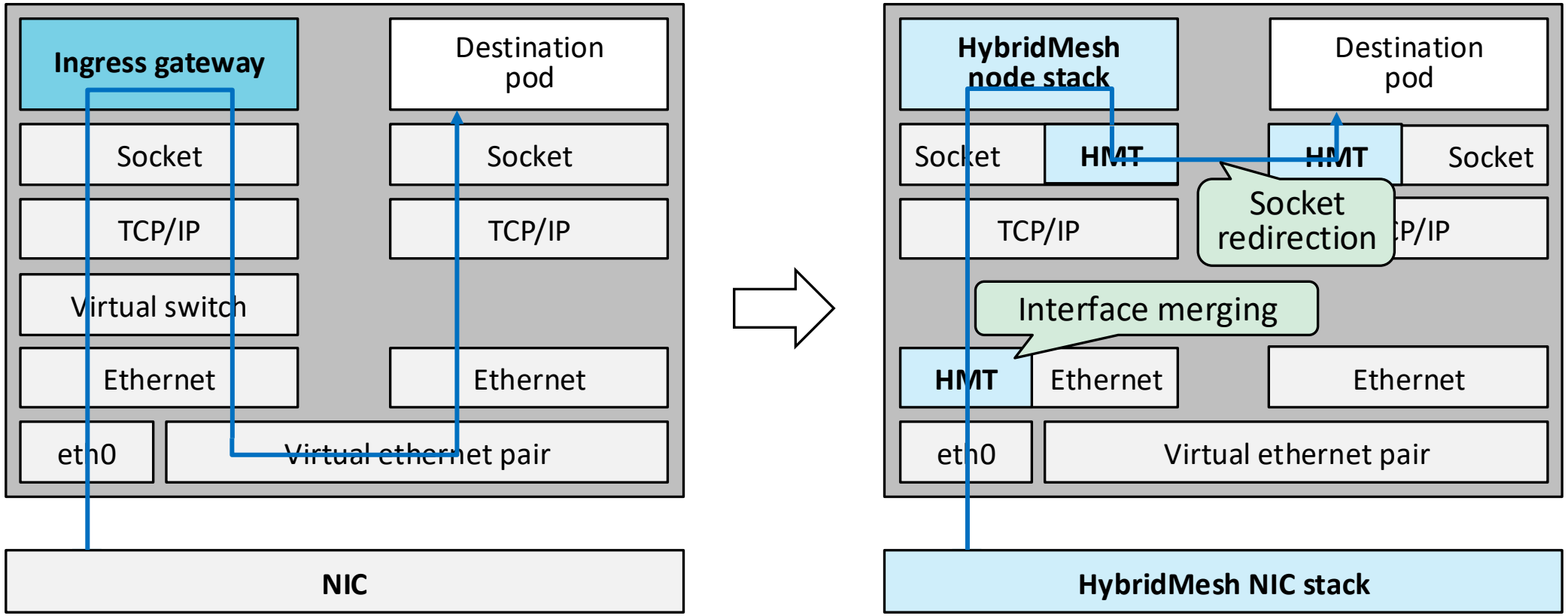
Maintaining Flexible Traffic Management

- The node stack performs various traffic management features based on ***metadata*** within each packet
  - Maintaining the same features as software alternatives while ***avoiding CPU-intensive tasks***



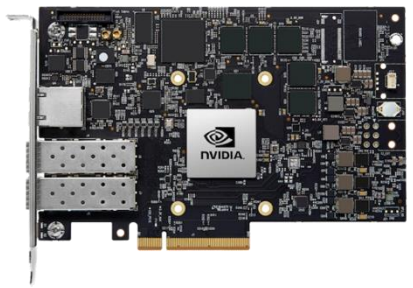
Bridging the System Components with eBPF

- The *HybridMesh tunnel (HMT)* adjust the location of metadata (L2 to L7)
- HMT optimizes the packet forwarding path between the node stack and containers



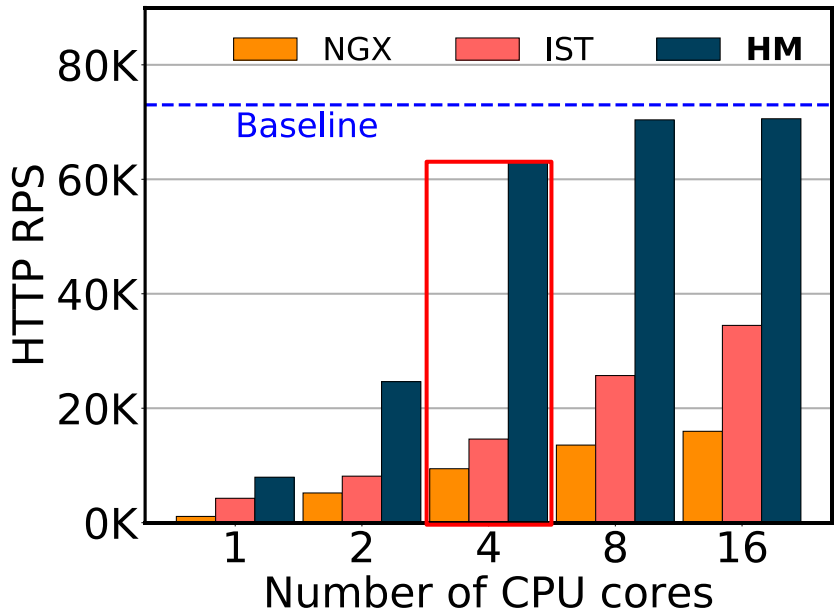
Implementation

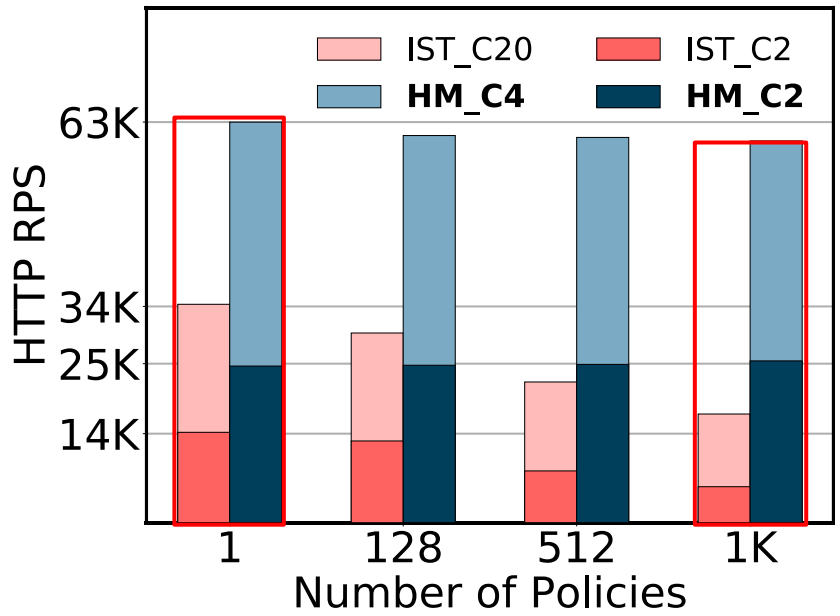
- **HybridMesh NIC stack**
  - Implemented within NVIDIA BlueField 2 SmartNIC by using DOCA SDK and DPDK
- **HybridMesh node stack**
  - Implemented with C languages by extending HAProxy with its Filter API
- **HybridMesh transport**
  - Implemented with three types of eBPF programs: XDP, TC and SK_MSG



Performance Evaluation

- End-to-end ingress traffic processing performance (8 Nginx pods within the same k8s service)
  - Two Intel Xeon Silver CPUs (20 cores each)
- HybridMesh with ***4 cores*** outperforms Istio with ***20 cores*** by up to ***4.4×***





Cost Effectiveness and Power Efficiency

- **CPU-only**: Istio ingress with 20 CPU cores (+ ConnectX5 NIC)
- **SmartNIC-only**: fully offloaded HybridMesh, running both the NIC and the Node stack using on-NIC cores.
- **SmartNIC-hybrid**: the original HybridMesh setting (+ 4 cores)

Hardware	Normalized		Throughput	Cost-effectiveness	Power-efficiency
	Initial price	Power			
CPU-only	$1,000	100W	28K RPS	1.0	1.0
SmartNIC-only	$825	58.5W	25K RPS	1.06	1.49
SmartNIC-hybrid	$921	73.8W	60K RPS	***2.3***	***2.87***

Summary

- **Software-based ingress gateways** become a main performance bottleneck in service meshes
  - Up to **6.5×** throughput drop and **18×** latency increase
  - High CPU utilization, preventing efficient resource allocation for microservices

- **HybridMesh: a novel hardware-software hybrid ingress gateway**
  - Offloading CPU-intensive tasks (traffic analysis) to SmartNICs
  - Retaining traffic management features on the CPU-side proxy
  - Bridging these components with eBPF (HybridMesh Tunnel)

4.4× higher throughput · **2.3×** cost-effectiveness · **2.8×** power-efficiency

Questions?

famous@uos.ac.kr