Reusing Network Services Logic to Improve Network Performance

Yotam Harchol
vmware Research

(This work was done while at the Hebrew University)

Joint work with Anat Bremler-Barr and David Hay

Appeared in ACM SIGCOMM 2016

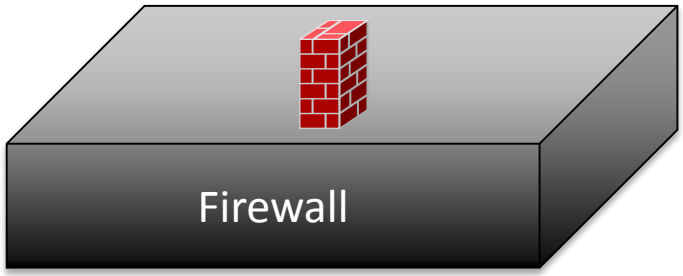
THE HEBREW UNIVERSITY OF JERUSALEM

IDC HERZLIYA

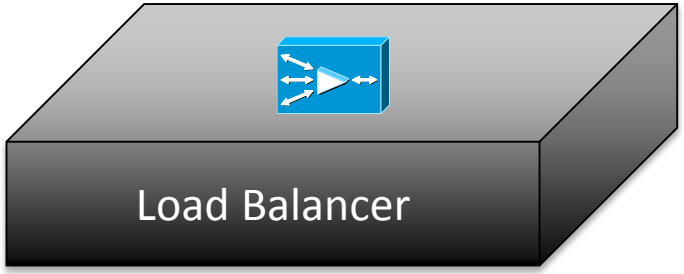
erc

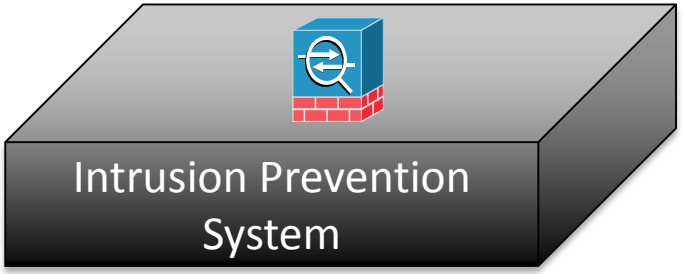
This research was supported by the European Research Council ERC Grant agreement no 259085, the Israeli Centers of Research Excellence (I-CORE) program (Center No. 4/11), and the Neptune Consortium.

Network Functions (Middleboxes)



- Monolithic **closed** black-boxes
  - ✗ High ***cost***
  - ✗ Limited ***provisioning*** and ***scalability***





Network Function Virtualization (NFV):

- ✓ Reduce ***cost*** (by moving to software)
- ✓ Improve ***provisioning*** and ***scalability*** (by virtualizing software NFs)

At the cost of:

- ✗ Reduced ***performance*** *(mainly **latency**)*

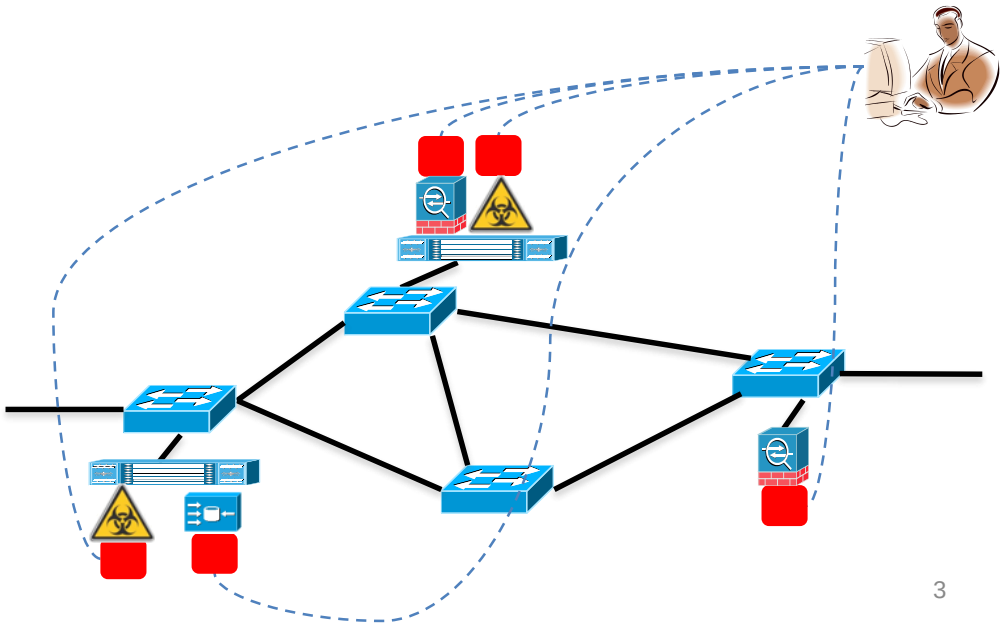
Network Functions (Middleboxes)

✗ High ***cost***

✗ Limited ***provisioning*** and ***scalability***

✗ Limited and separate ***management***

- Different vendors
- No standards
- Separate control plane



Network Functions (Middleboxes)

- Actually, many of these black-boxes are very modular

✗ High ***cost***

✗ Limited provisioning and scalability

✗ Limited and separate management

✗ Limited ***functionality*** and limited ***innovation*** (High entry barriers)

✗ Similar complex processing steps, ***no re-use***

OpenBox

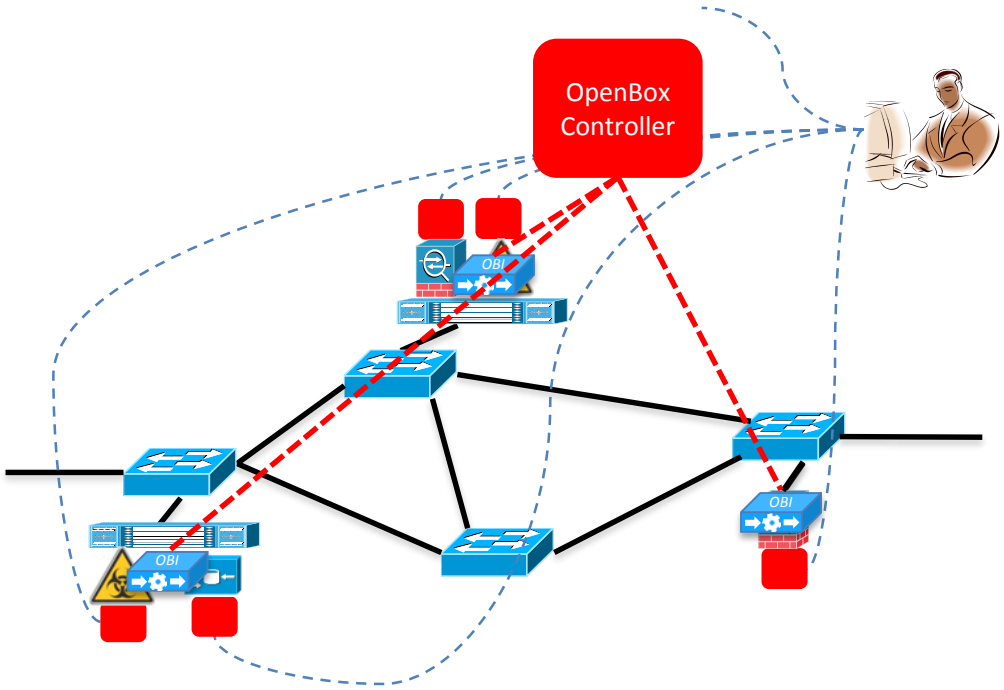
www.openboxproject.org

github.com/OpenBoxProject

- **OpenBox: A new software-defined framework for network functions**
- Decouples network function control from their data plane
- Unifies data plane of multiple network functions

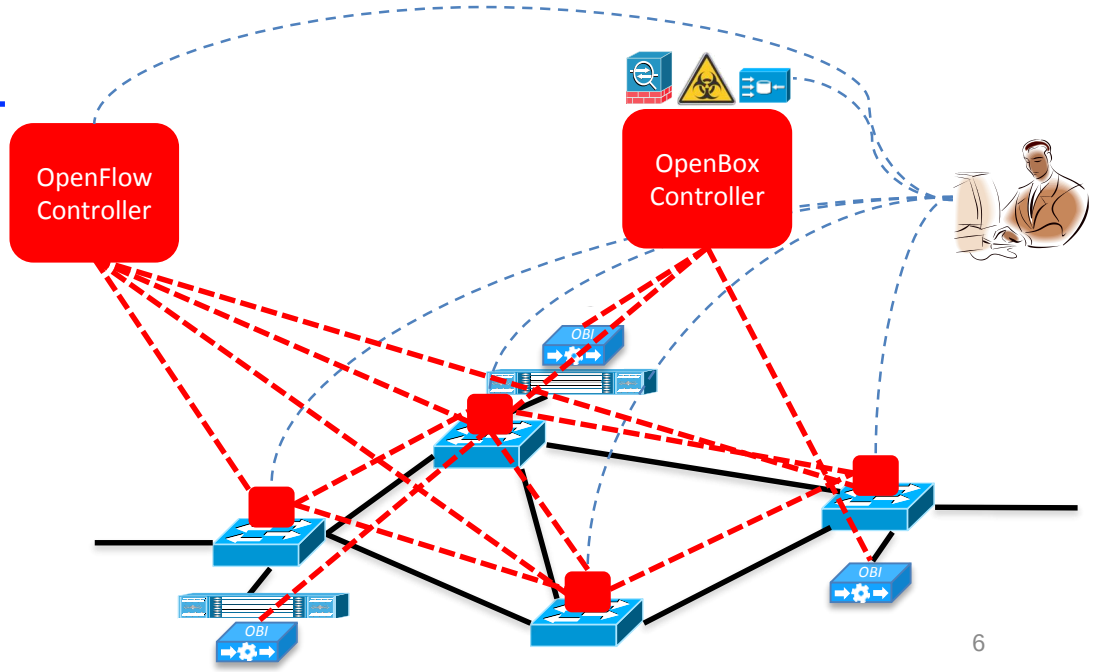
Benefits:

- ✓ Easier, unified control
- ✓ Better performance (improved latency)
- ✓ Scalability
- ✓ Flexible deployment
- ✓ Inter-tenant isolation
- ✓ Innovation



Software Defined Networking

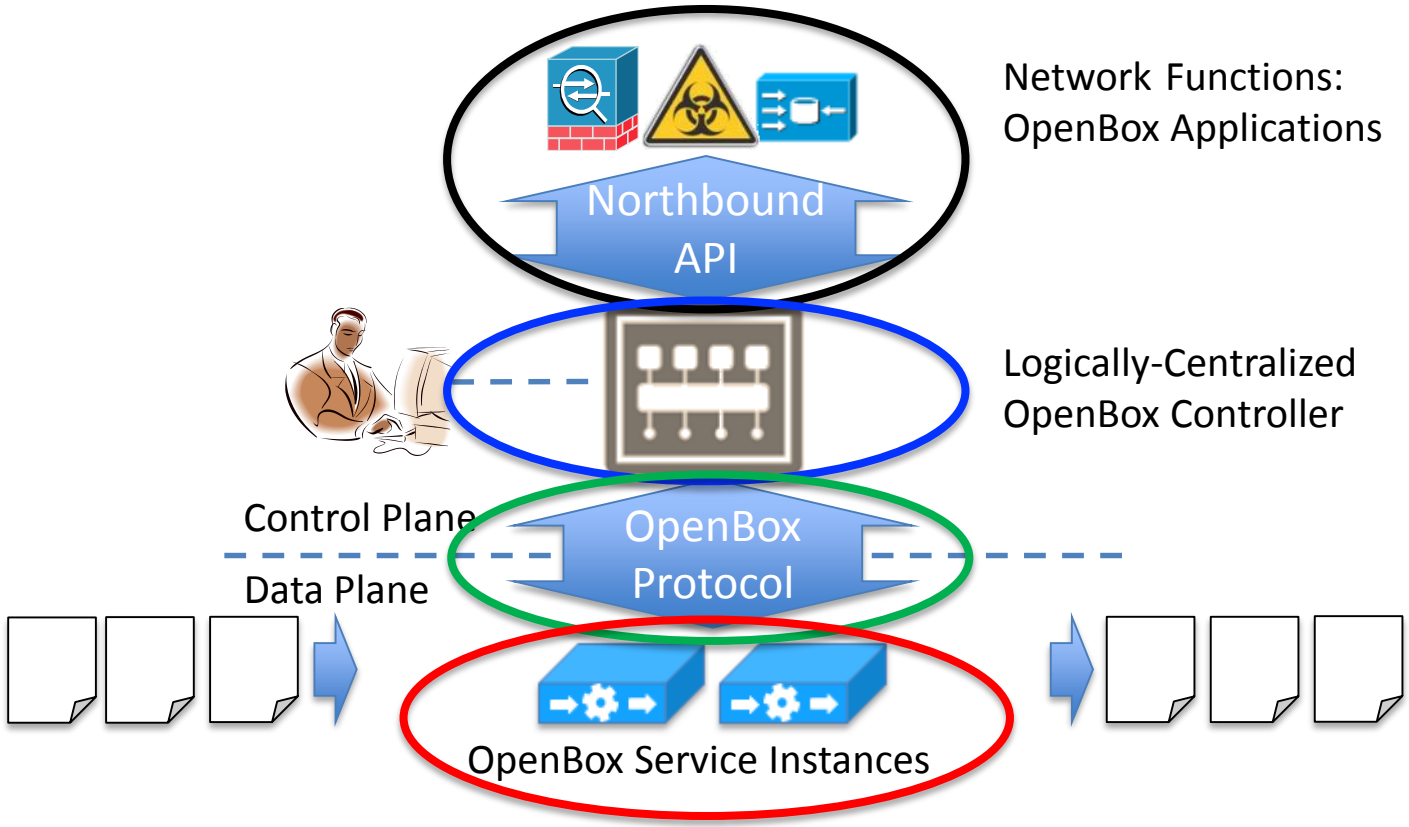
- High ***cost*** of ~~*middleboxes*~~ *switches*
- Limited ***provisioning*** and ***scalability*** of ~~*middleboxes*~~ *switches*
- Limited ***management*** of ~~*middleboxes*~~ *switches*
- Limited ***functionality*** and limited ***innovation***
- Complex ~~processing steps~~ *distributed algorithms*



40%-60% of the appliances in large-scale networks are middleboxes!

[Sherry & Ratnasamy, '12]

The OpenBox Framework



Additionally:

- ✓ Isolation between NFs / multiple tenants
- ✓ Support for hardware accelerators
- ✓ Dynamically extend the protocol

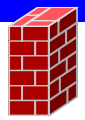
Observation:

Most network functions do very similar processing steps

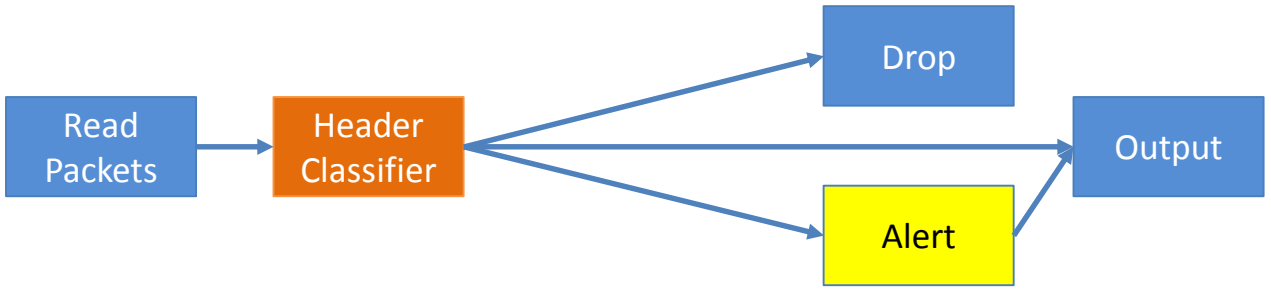
But there is no re-use…

The design the OpenBox framework is based on this observation

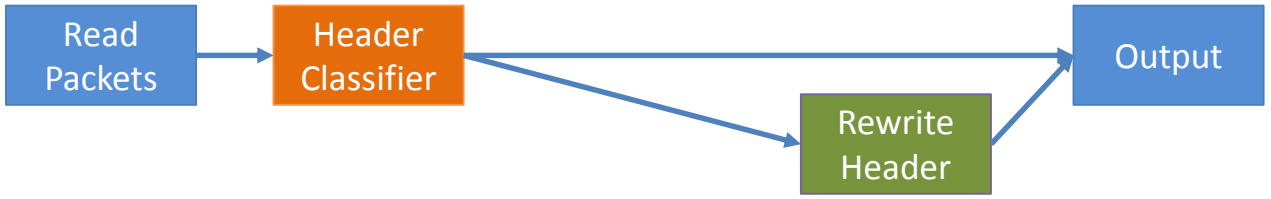
Network Function Decomposition



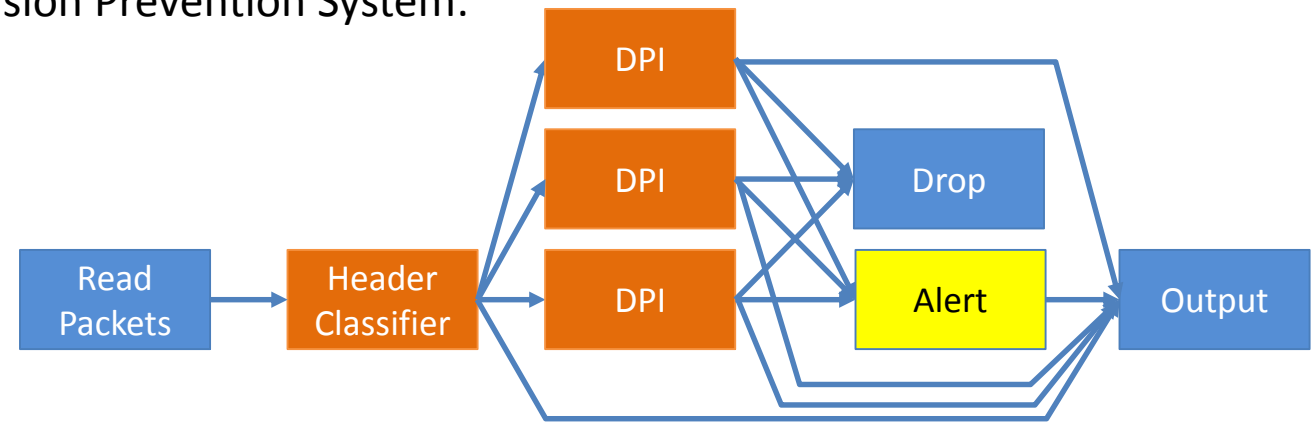
Firewall:



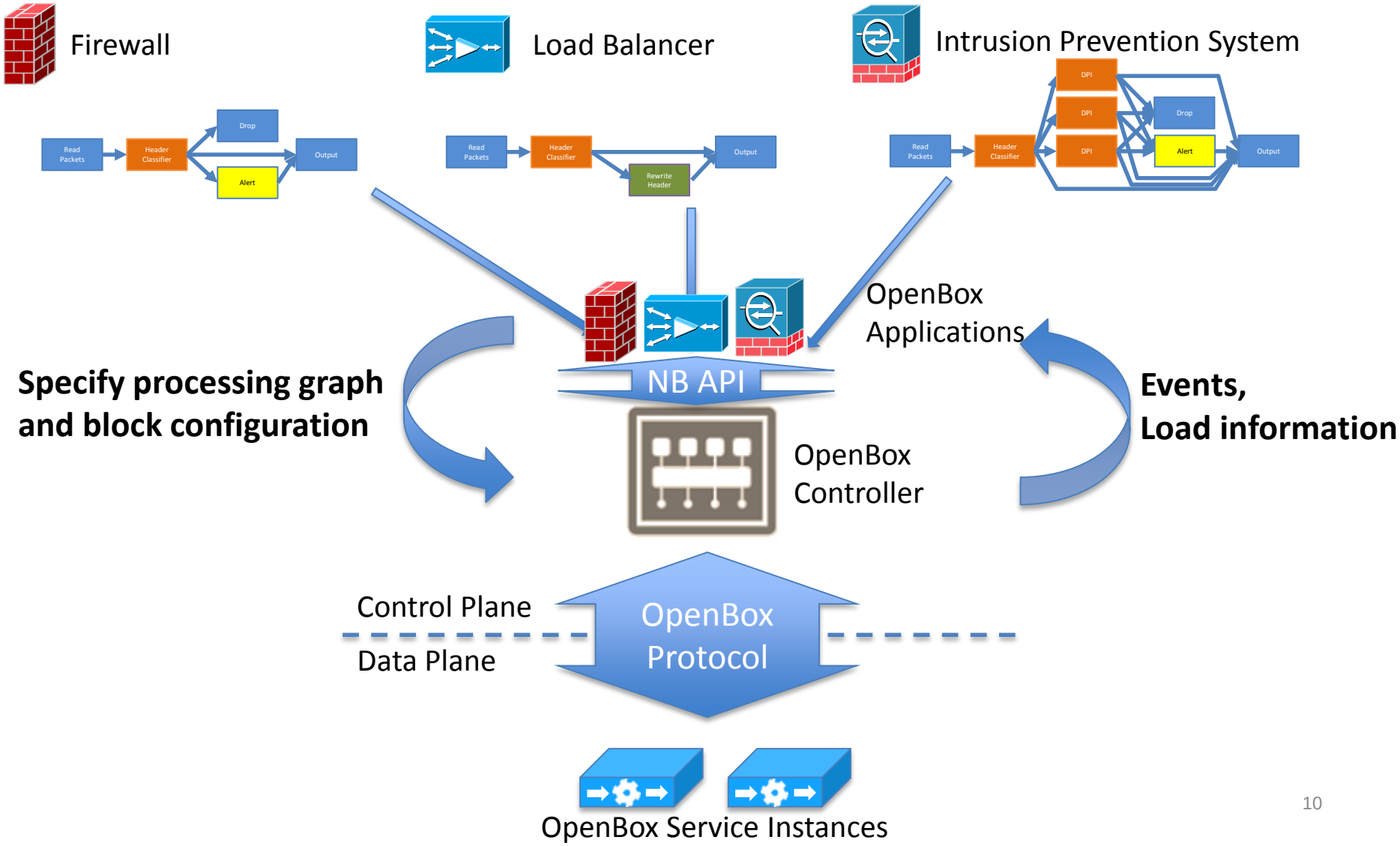
Load Balancer:



Intrusion Prevention System:



Northbound API

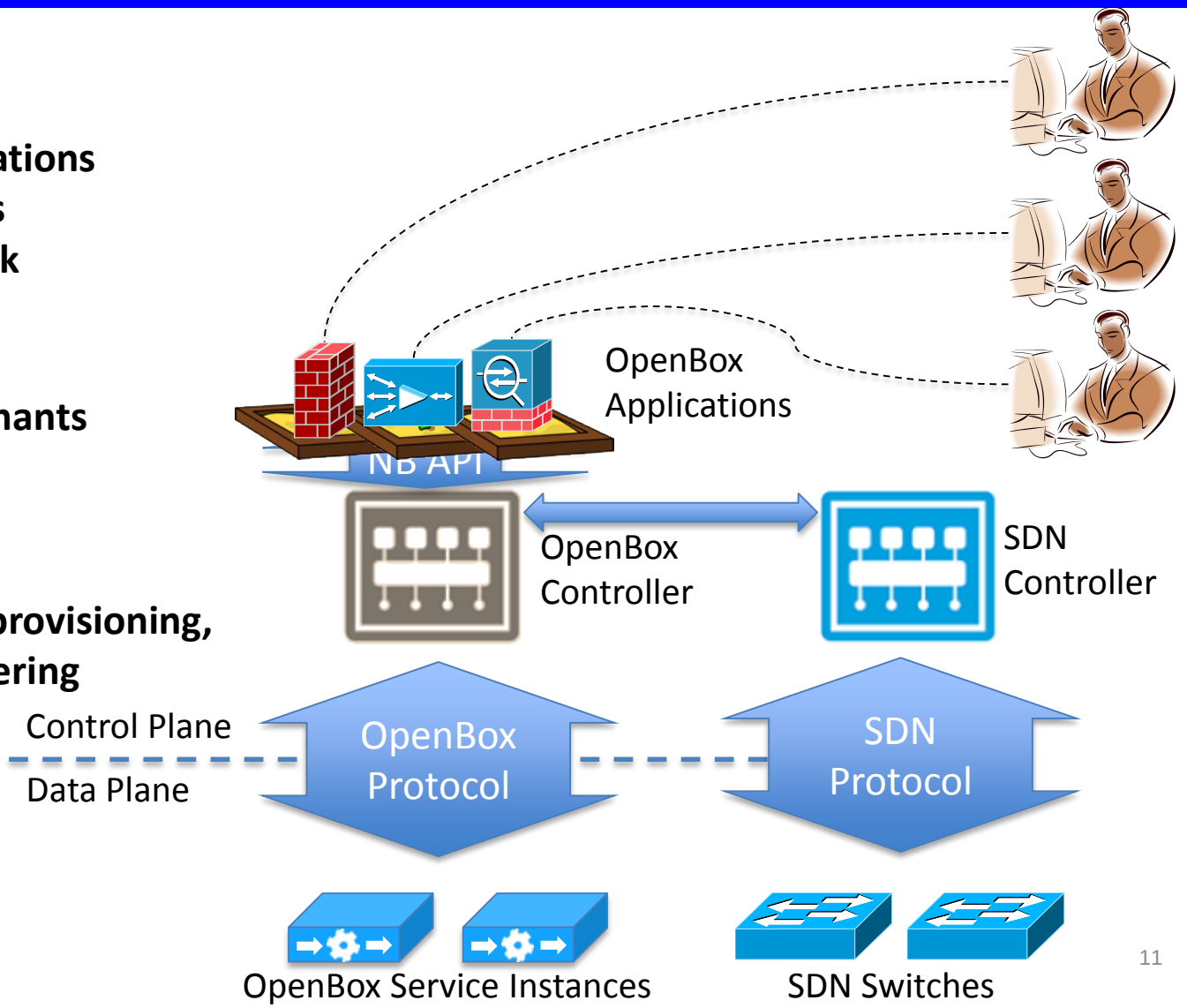


Logically-Centralized Controller

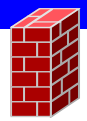
Multiple tenants
run **multiple applications**
for **multiple policies**
in **the same network**

Isolation between
applications and **tenants**
enforced by NB API

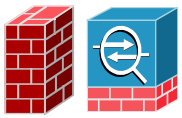
Network-wide view
Automatic **scaling, provisioning, placement,** and **steering**



Naïve Graph Merge



Firewall:



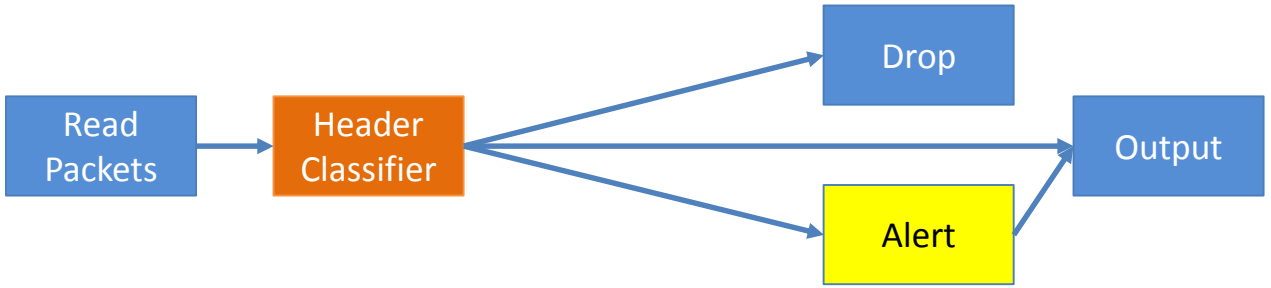
Concatenated Processing Graph:

Intrusion Prevention System:

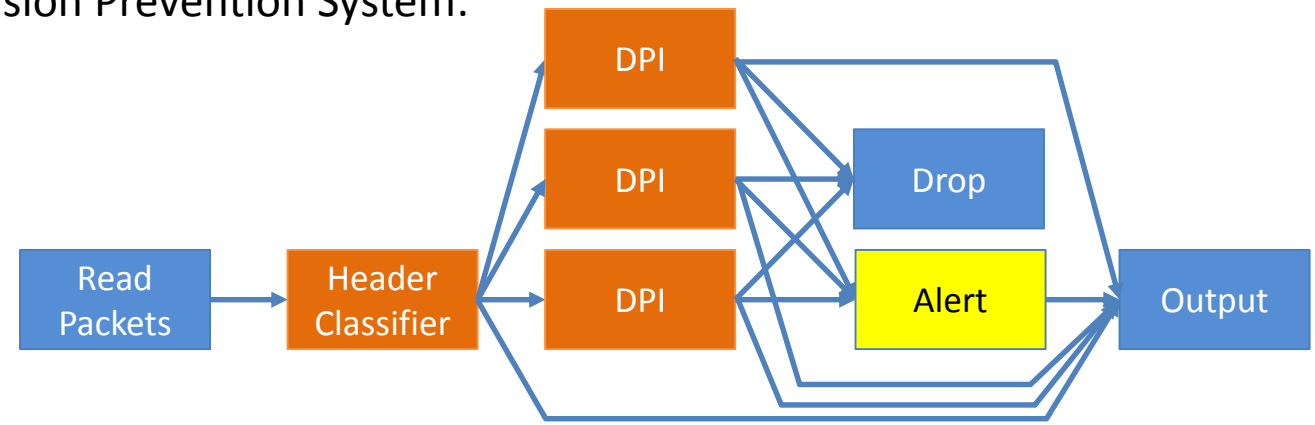
Graph Merge Algorithm

Input Graphs:

Firewall:



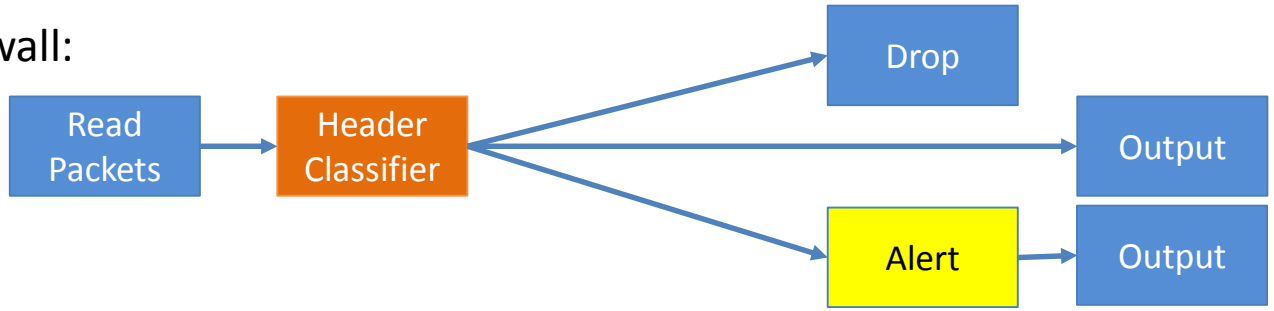
Intrusion Prevention System:



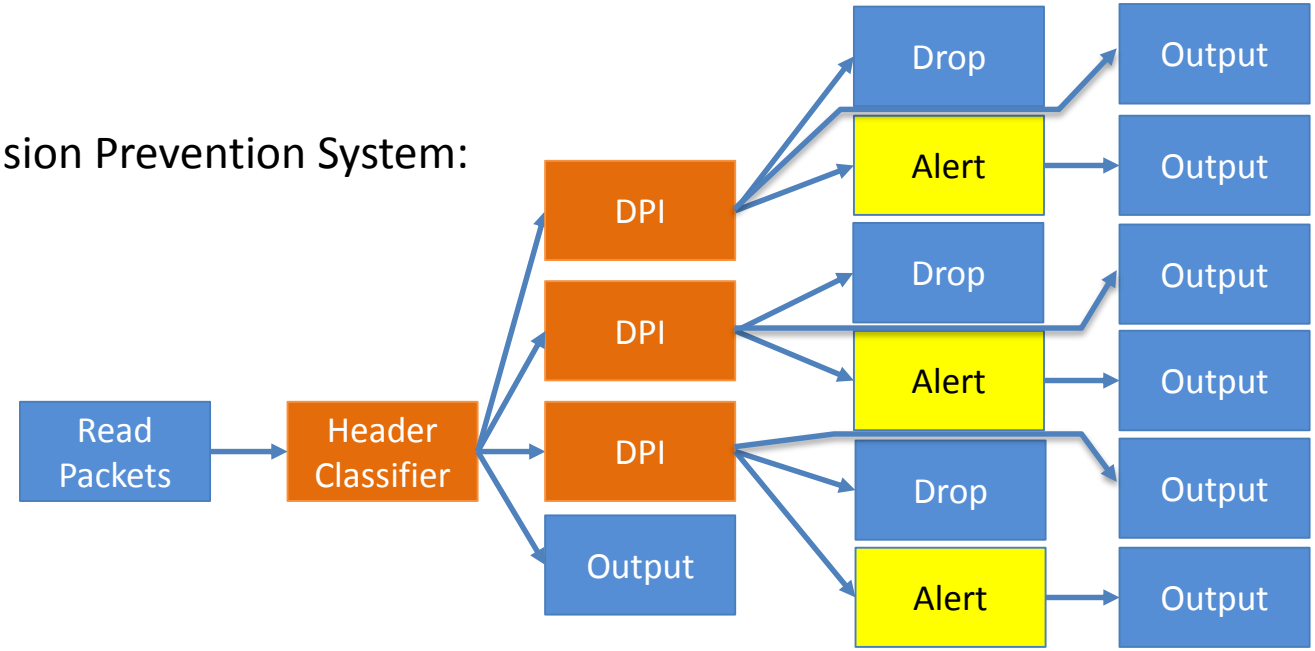
Graph Merge Algorithm

Step 1: Normalize graphs to trees

Firewall:

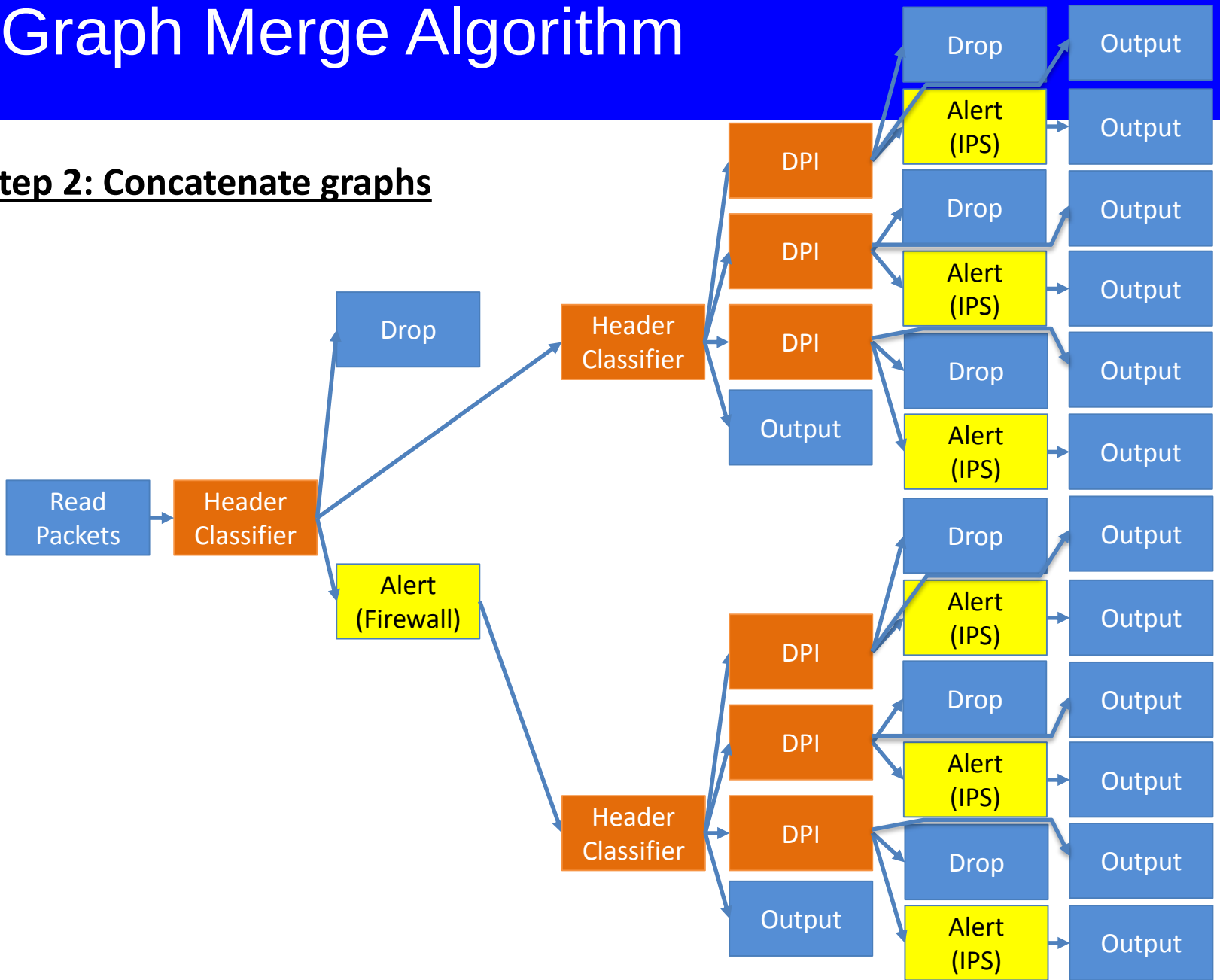


Intrusion Prevention System:



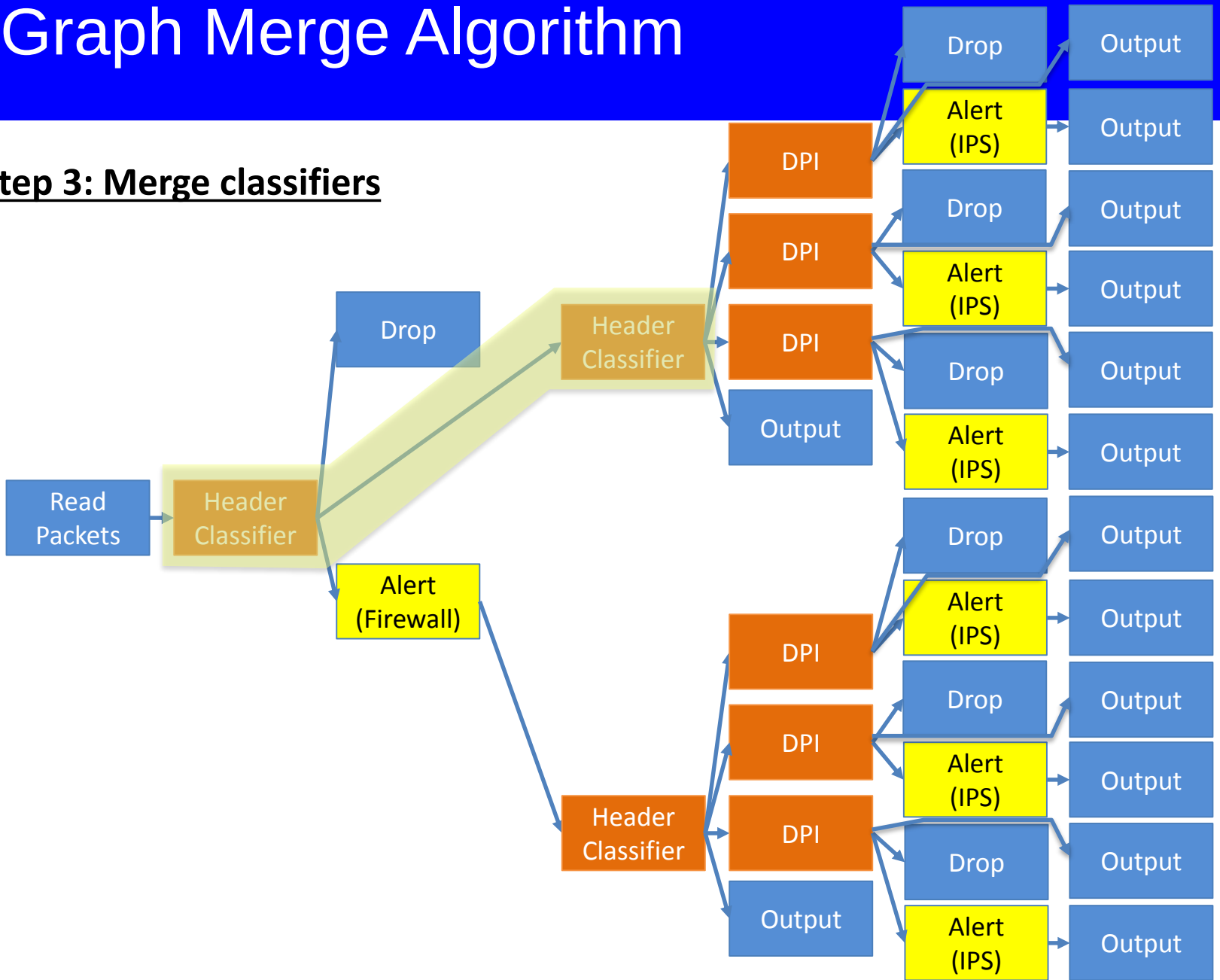
Graph Merge Algorithm

Step 2: Concatenate graphs



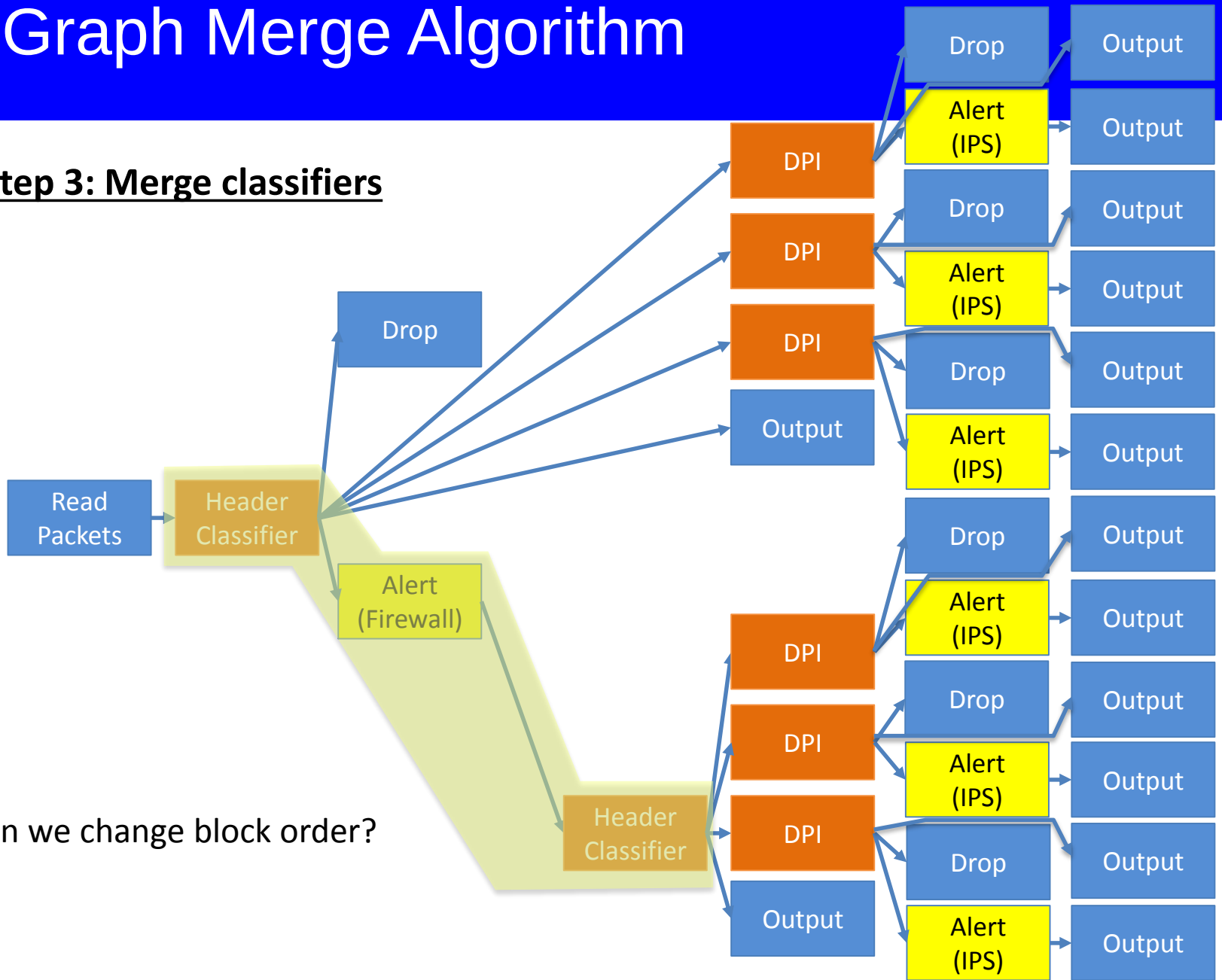
Graph Merge Algorithm

Step 3: Merge classifiers



Graph Merge Algorithm

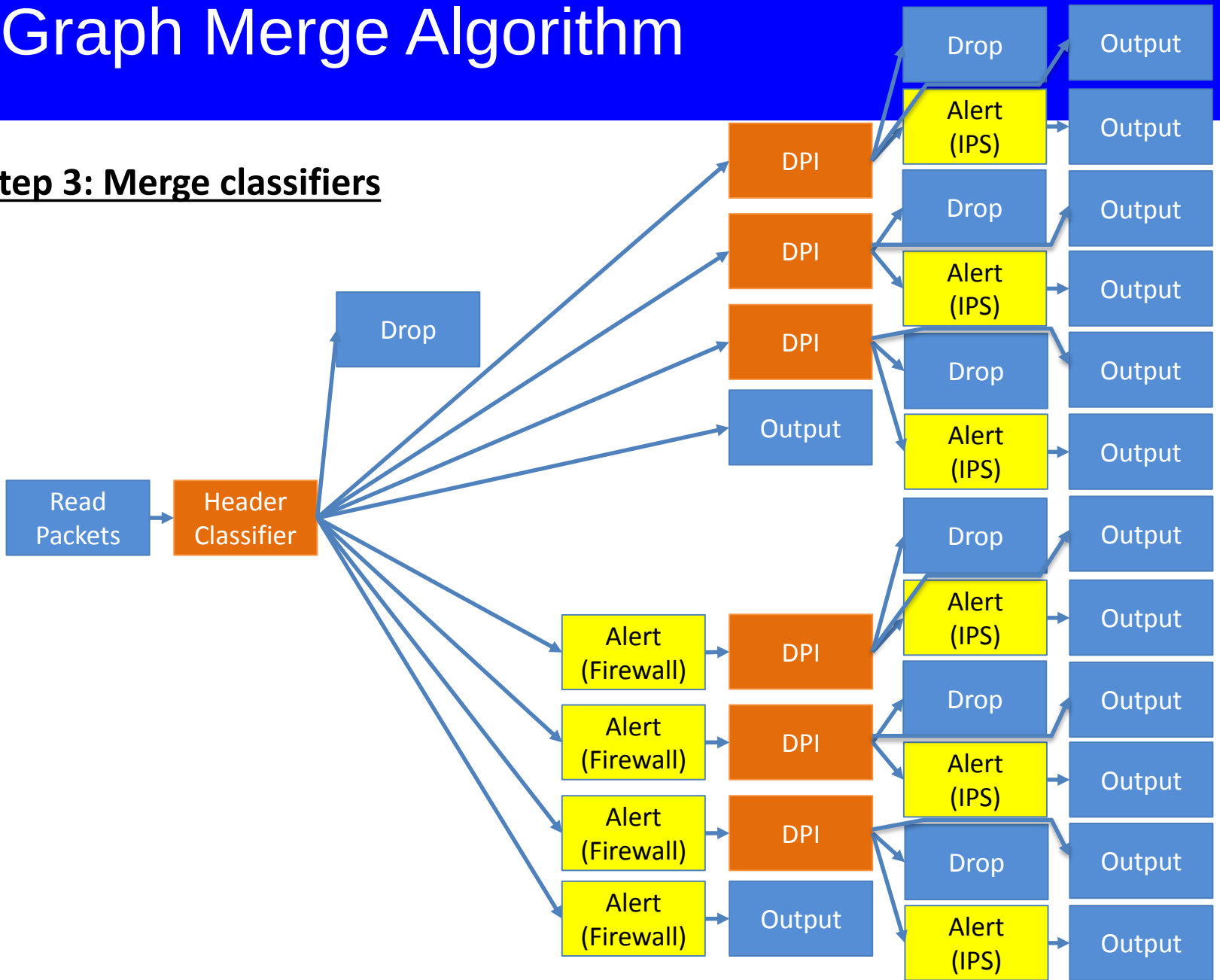
Step 3: Merge classifiers



Can we change block order?

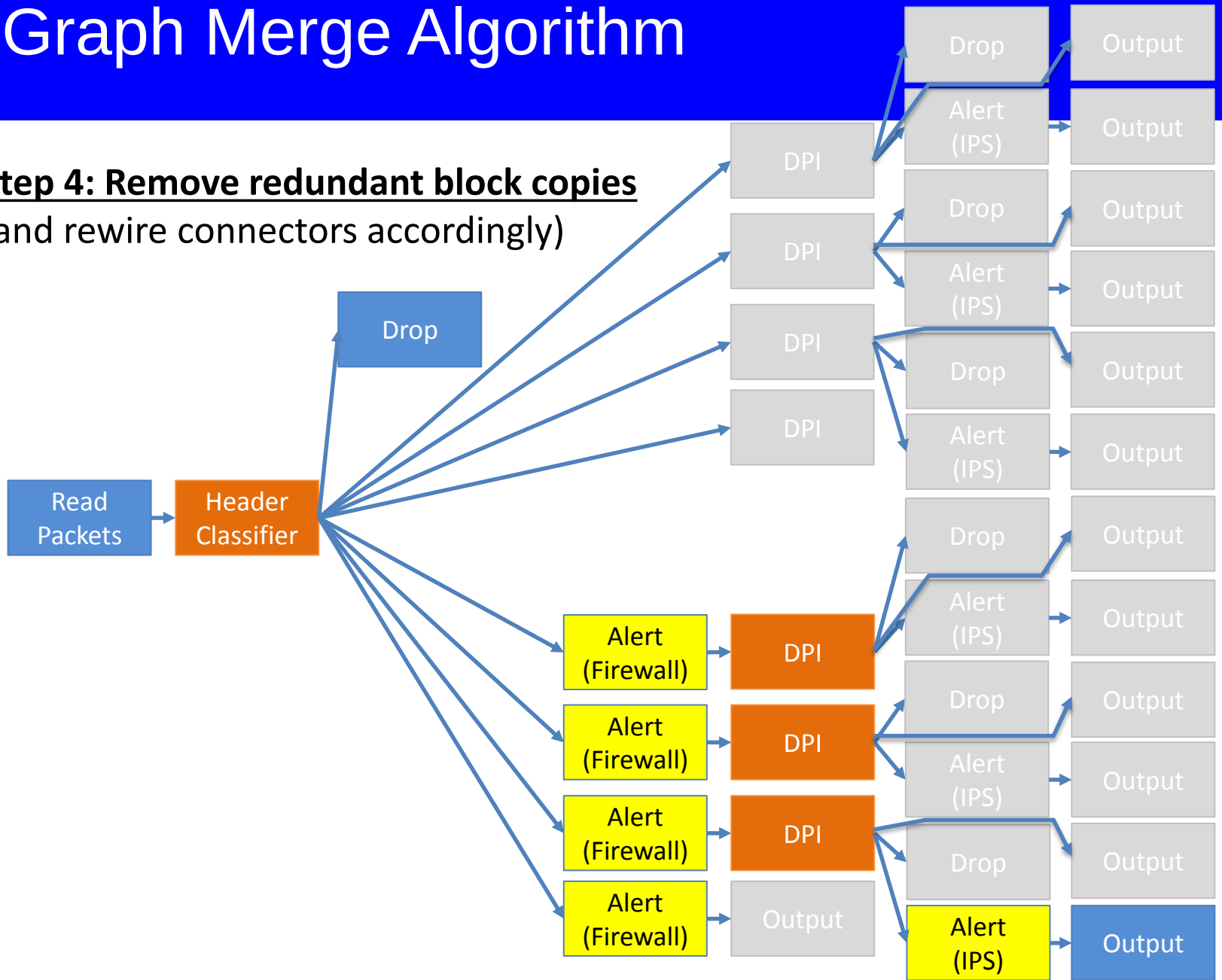
Graph Merge Algorithm

Step 3: Merge classifiers

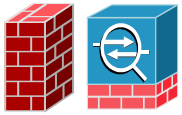


Graph Merge Algorithm

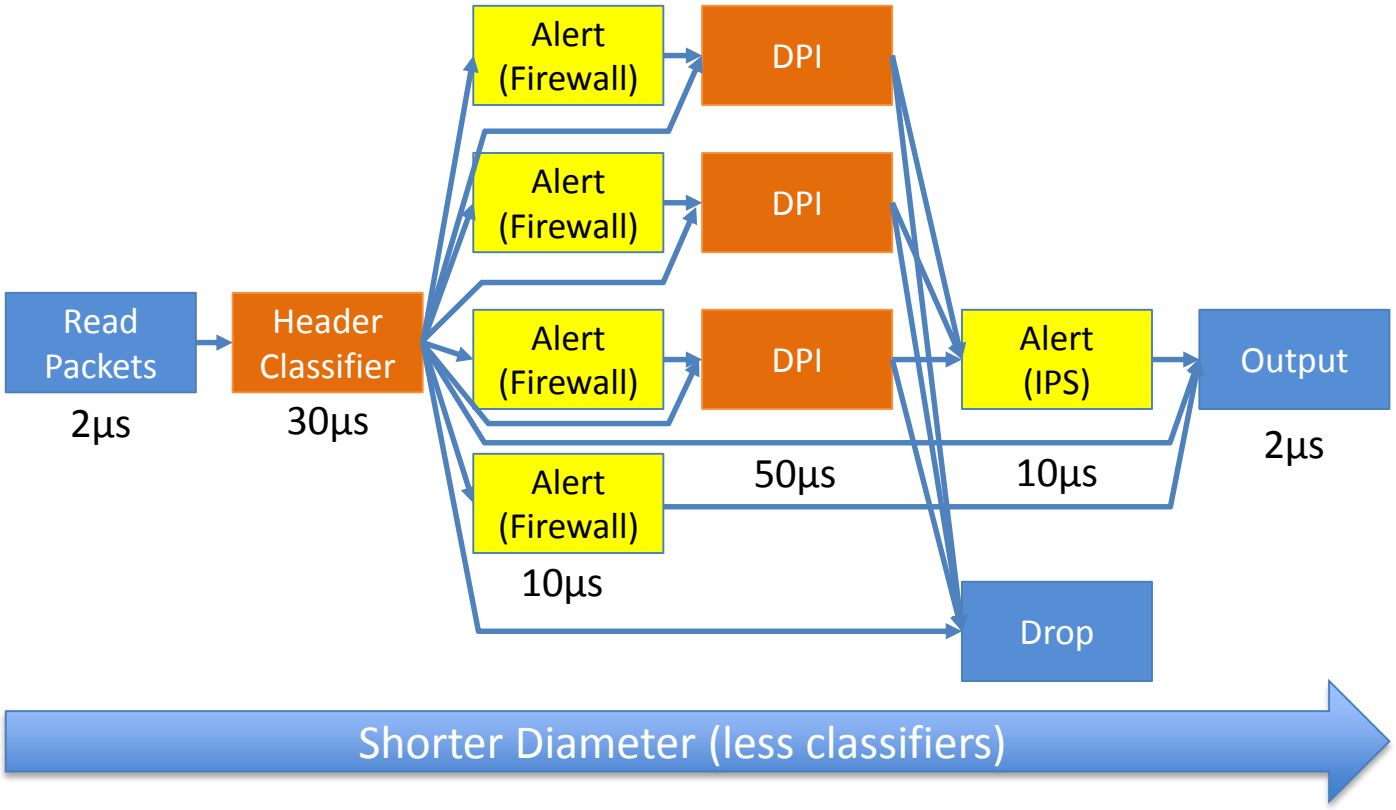
Step 4: Remove redundant block copies
(and rewire connectors accordingly)



Graph Merge Algorithm



Merged Processing Graph:



Total: 104µs (22% improvement)

When NOT to Merge?

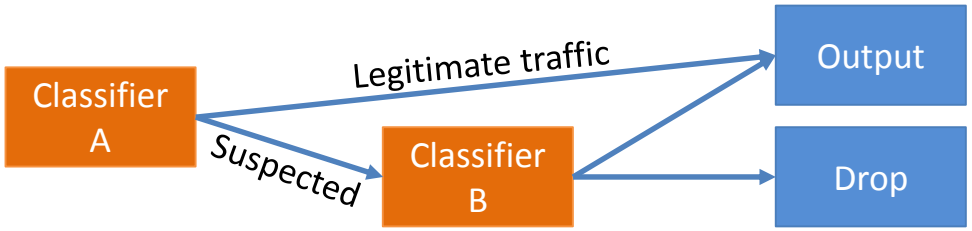
When cross product is too large:

- Two d-dimensional classifiers: A – n rules, B – m rules
- Classification is logarithmic with # of rules, exponential with dimension
- Serial classification time: $(\log n)^{d-1} + (\log m)^{d-1}$
- Cross product: $n \cdot m$ rules (worst case)
- Single classifier worst case time:

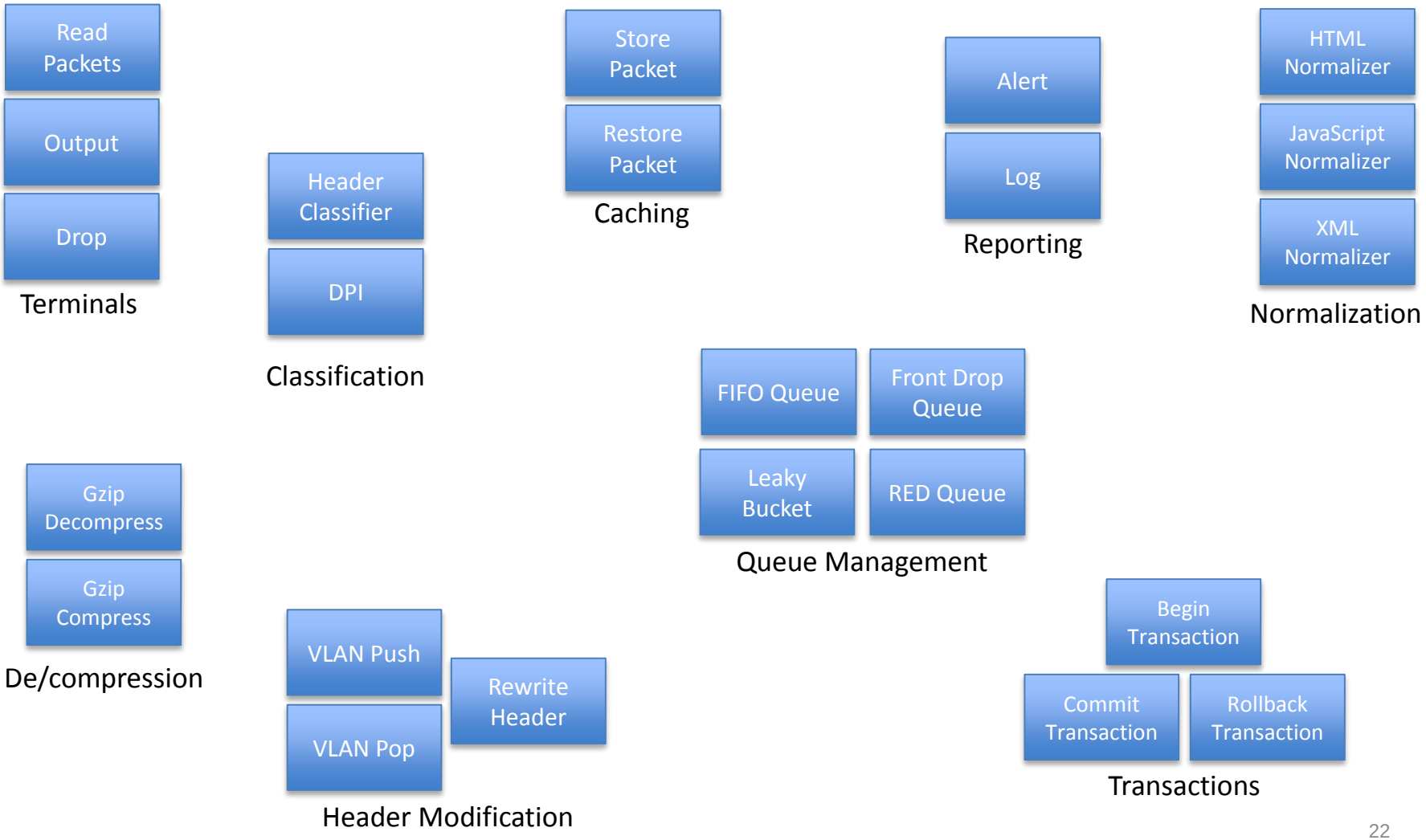
$$\log(n \cdot m)^{d-1} = (\log n)^{d-1} + (\log m)^{d-1} + \sum_{i=1}^{d-2} \binom{d-1}{i} \left((\log n)^{i} + (\log m)^{d-i-1}\right)$$

$$> (\log n)^{d-1} + (\log m)^{d-1}$$

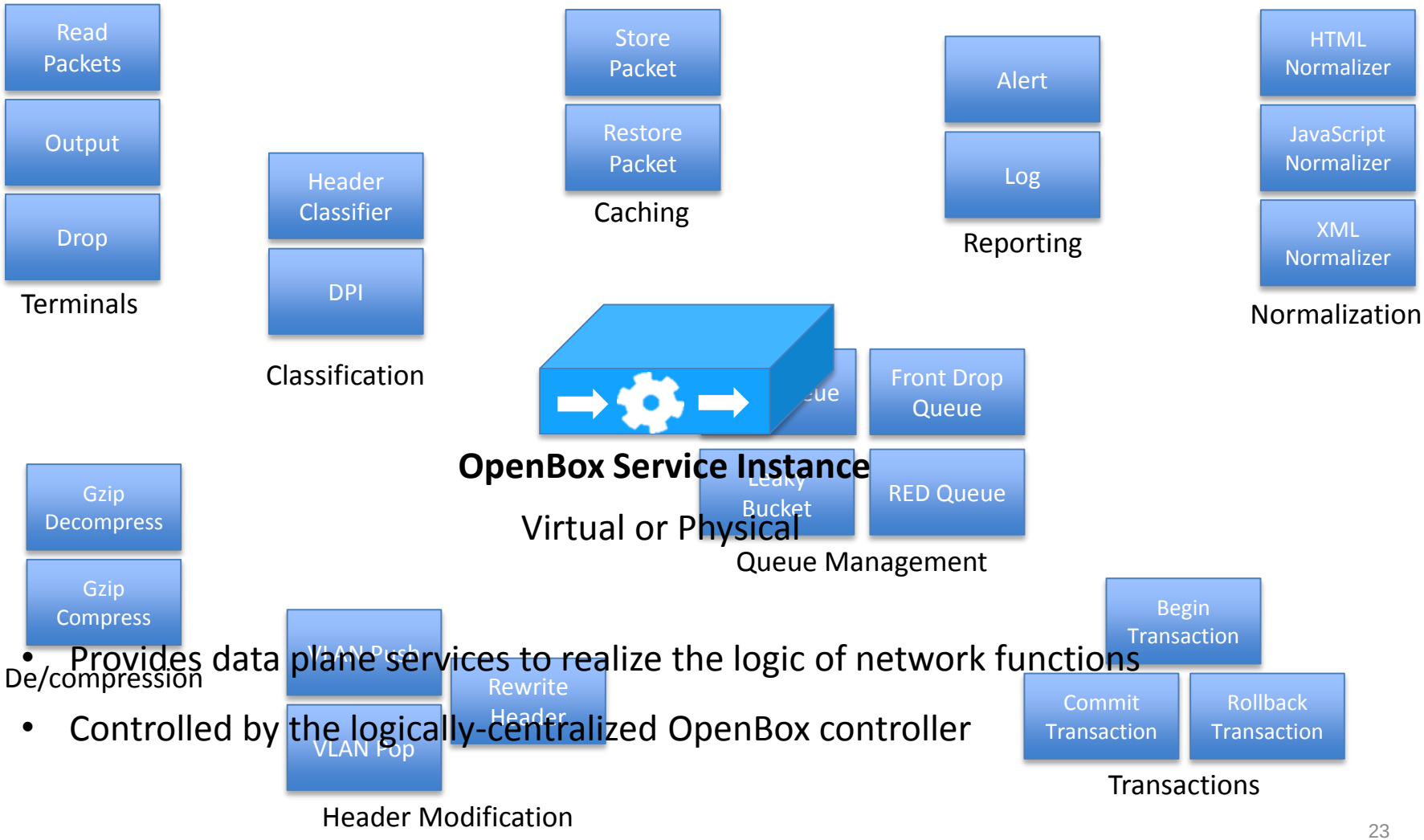
When most packets won't go through both classifiers:



OpenBox Data Plane Processing

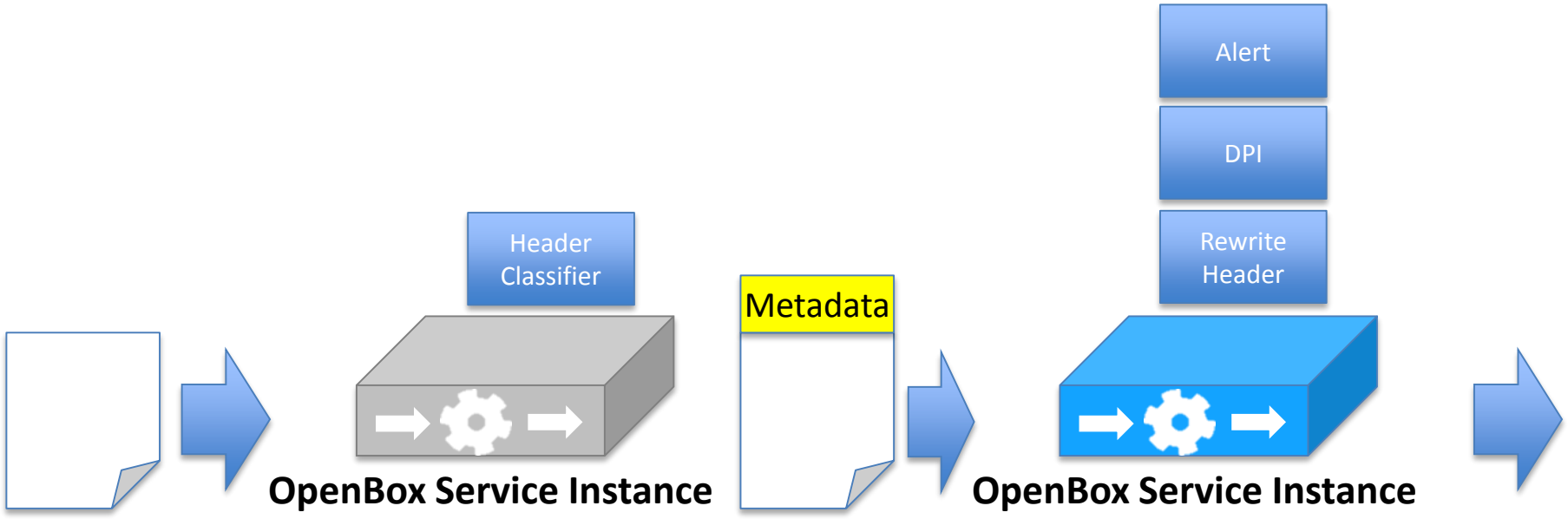


OpenBox Data Plane Processing



- Provides data plane services to realize the logic of network functions
- Controlled by the logically-centralized OpenBox controller

Distributed Data Plane



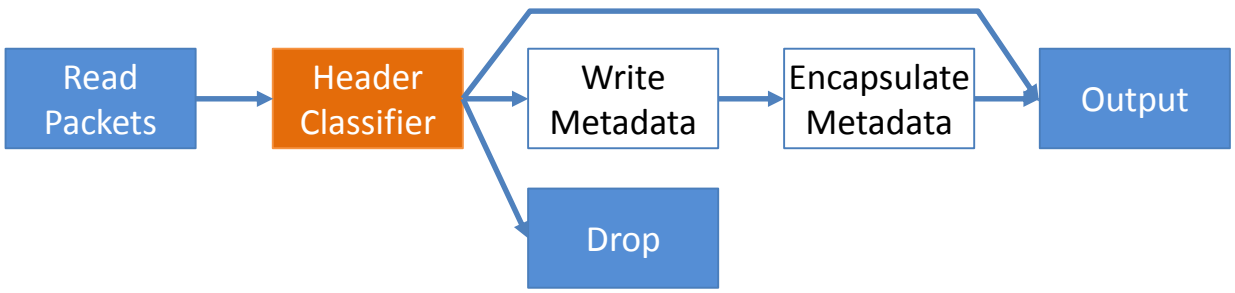
Hardware
(TCAM)

Software

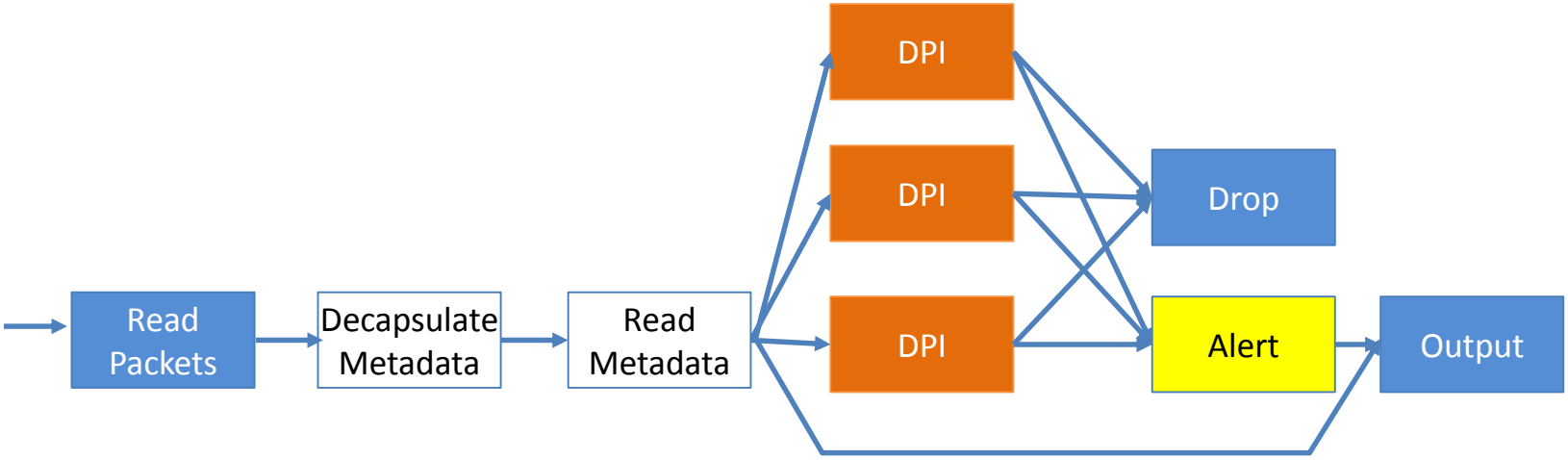
E.g., an OpenFlow switch
with encapsulation features
(e.g., NSH, Geneve, FlowTags)

Split Processing Graph

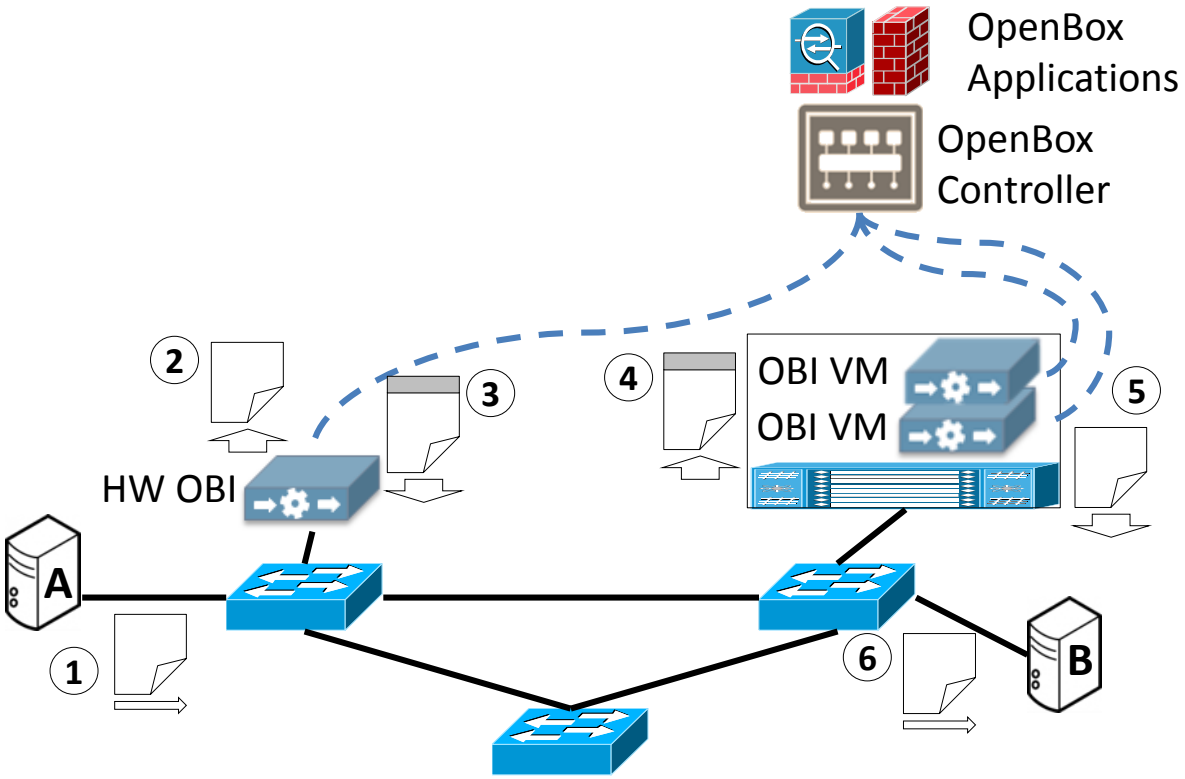
HW Instance:



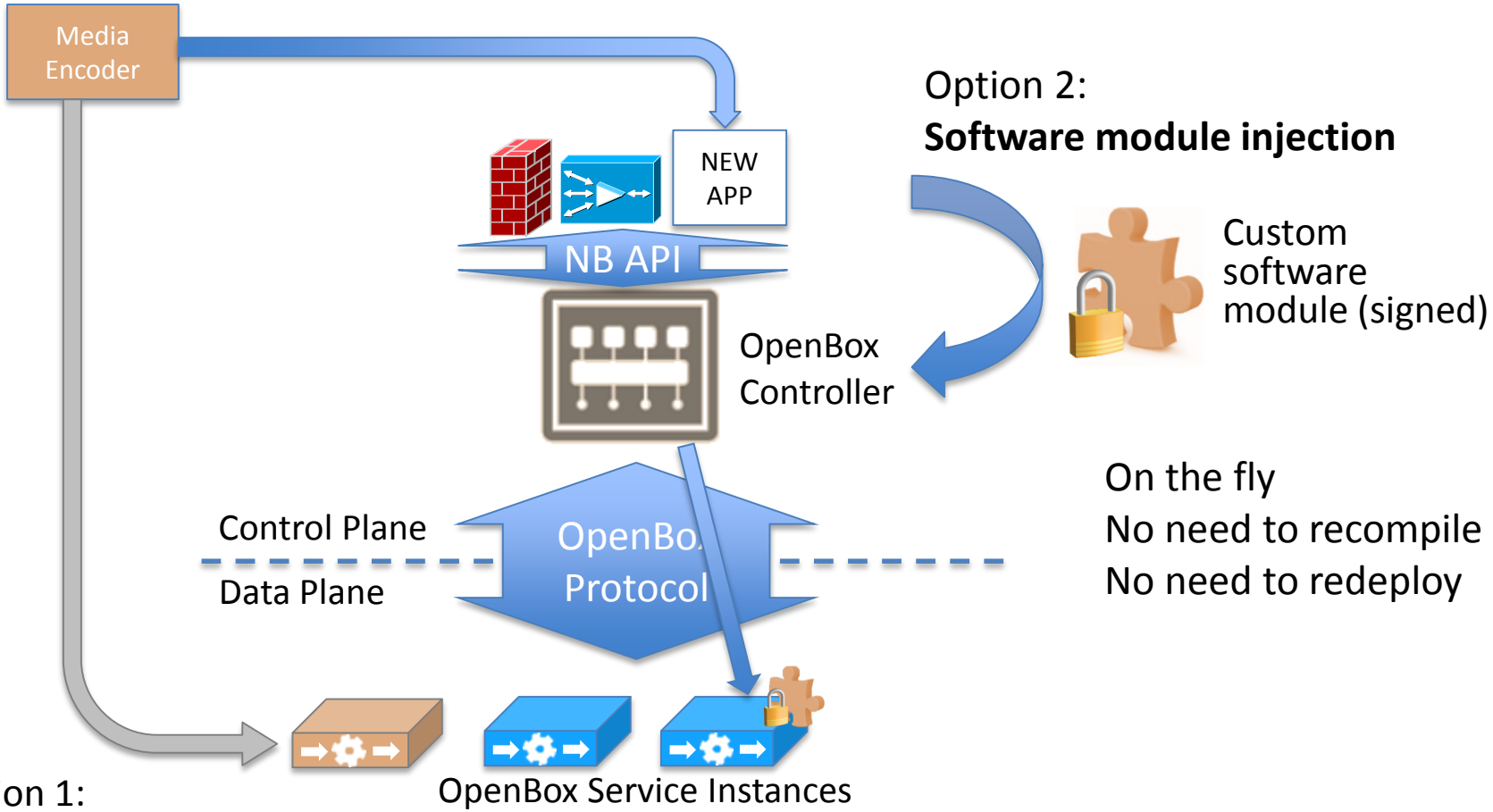
SW Instance:



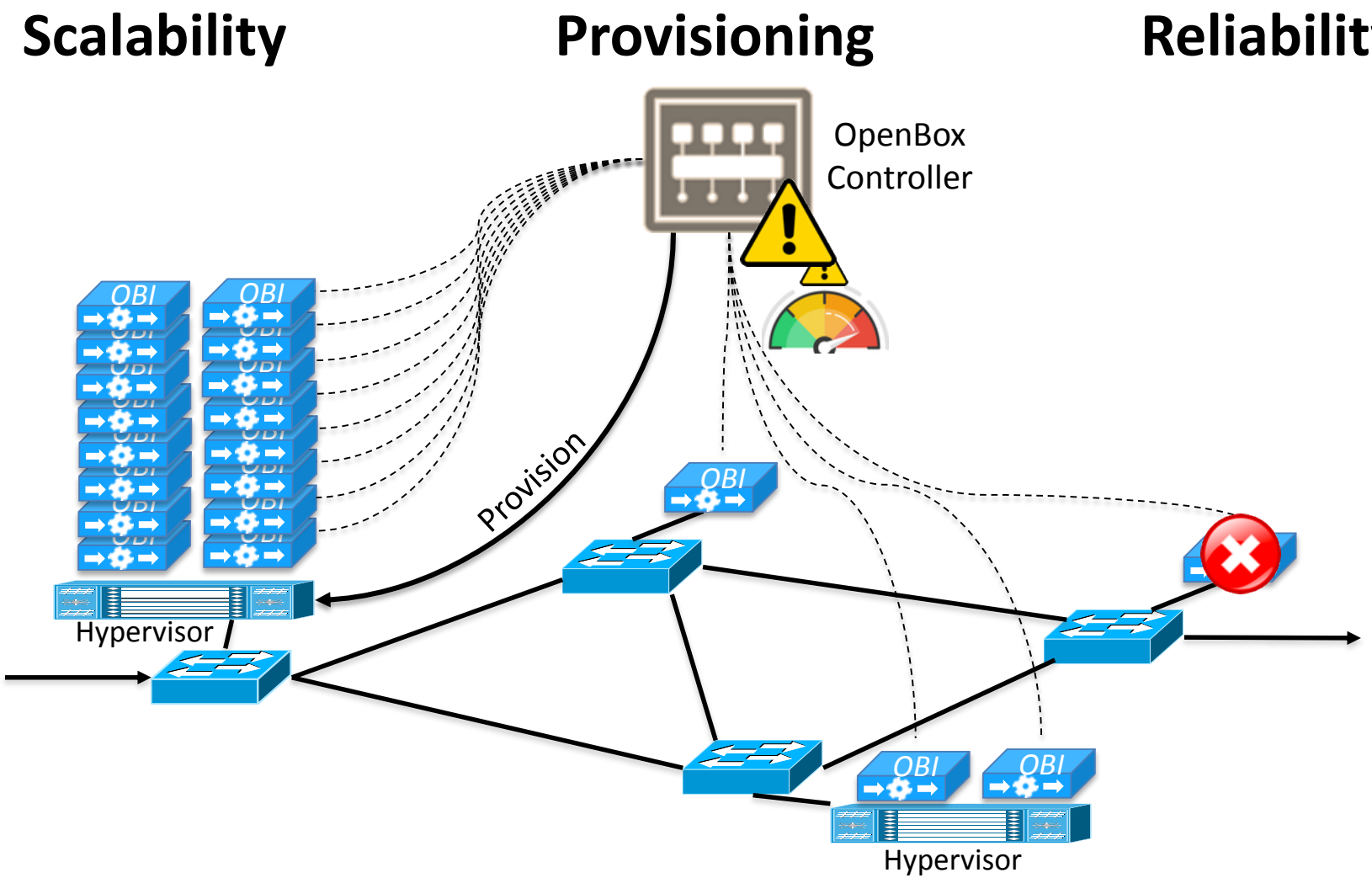
Distributed Data Plane



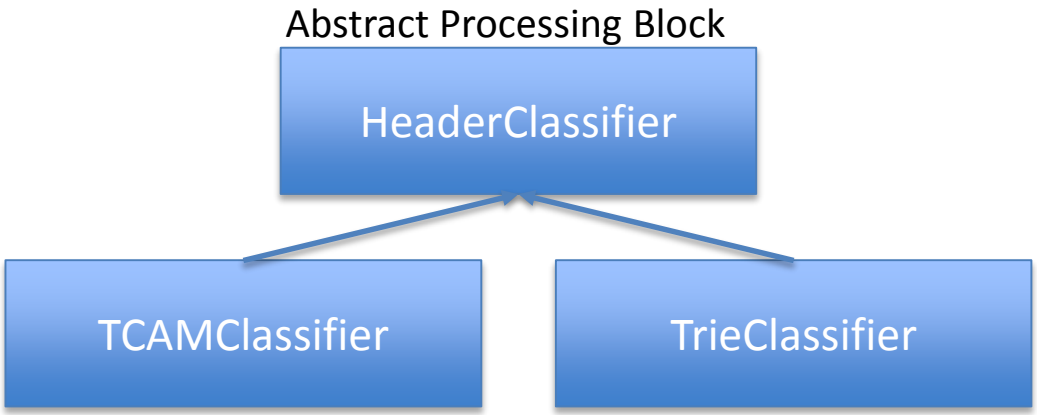
Extensible Data Plane

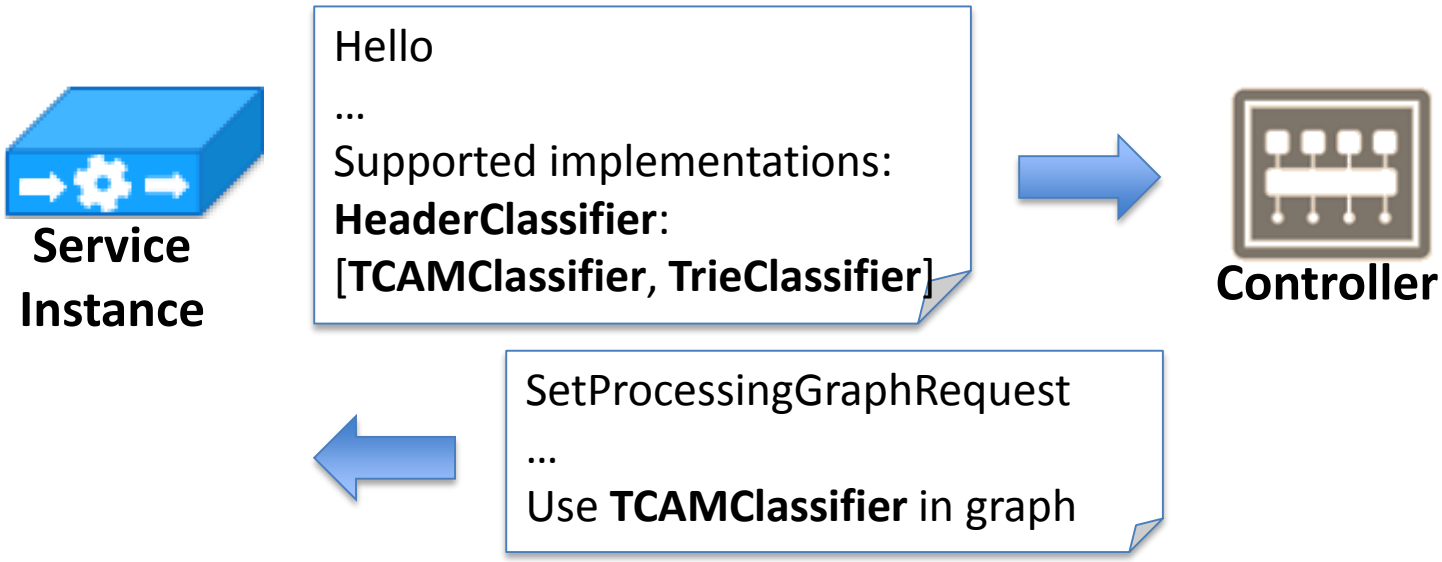


Scalable & Reliable Data Plane



OpenBox Protocol: Block Hierarchy



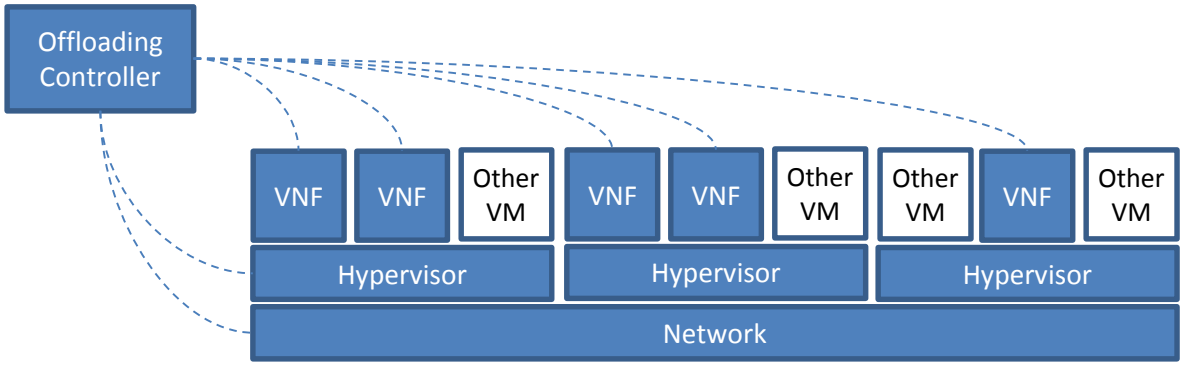


Future Work: Infrastructure Support

- Infrastructure can help VNFs
  - Provide high performance (e.g., hardware accelerators)
  - Reuse processing (e.g., packet switching, “outsourced” services)

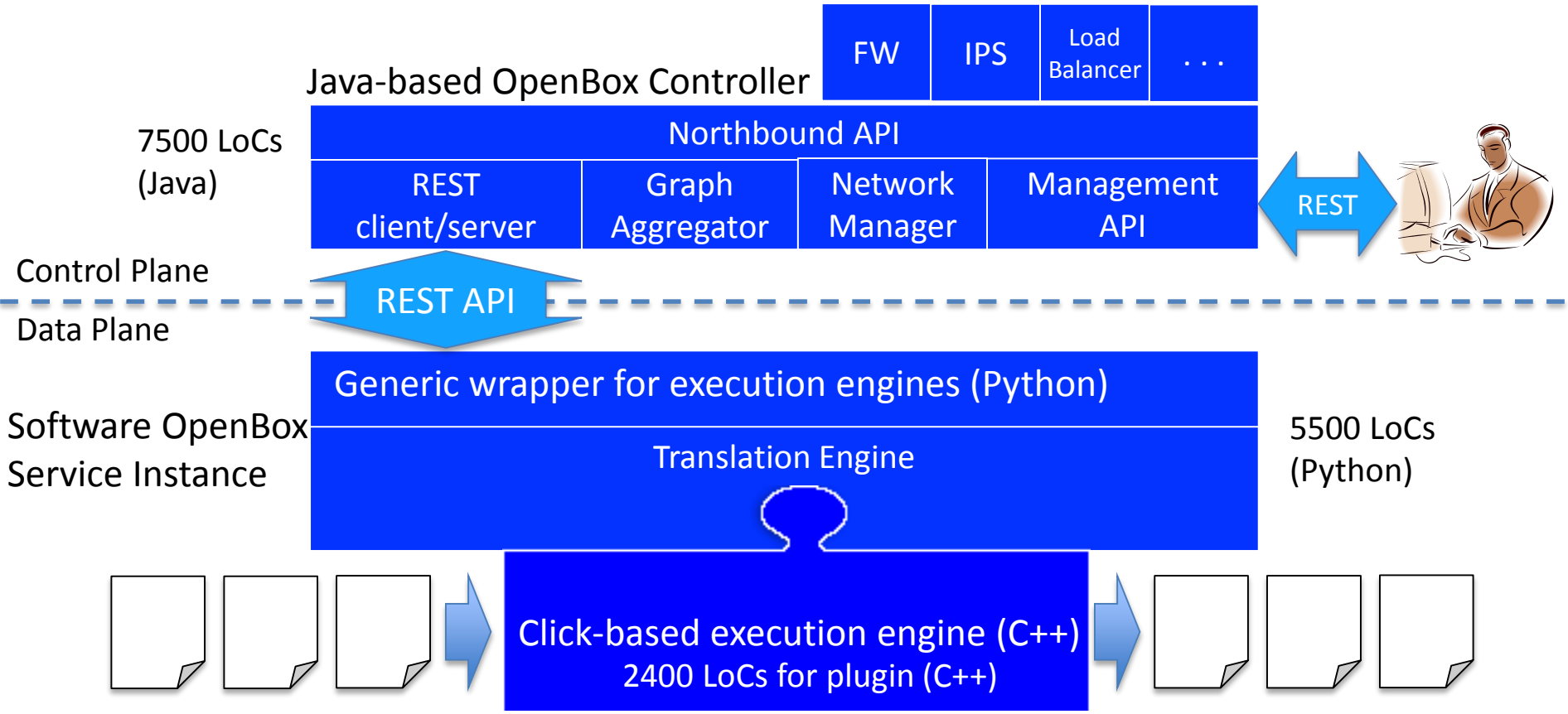
- Challenge: Design a system, define a protocol to offload processing from VNFs to infrastructure

- Gradual solution, easier to adopt for existing VNFs



Implementation

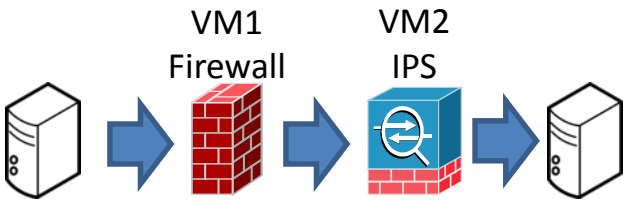
github.com/OpenBoxProject



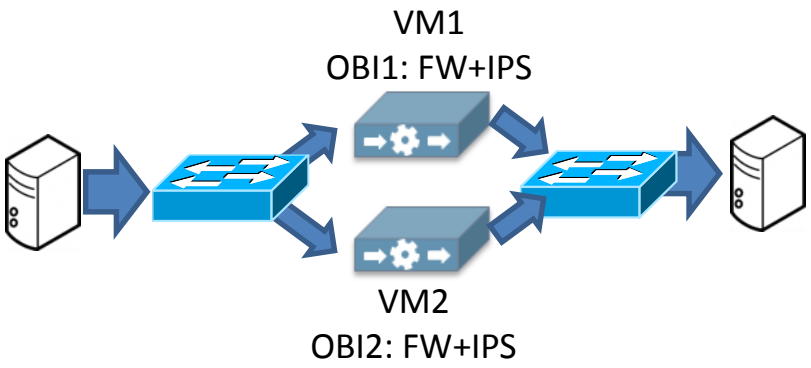
(Plug here other execution engines. E.g., ClickNP [SIGCOMM '16])

Performance Improvement

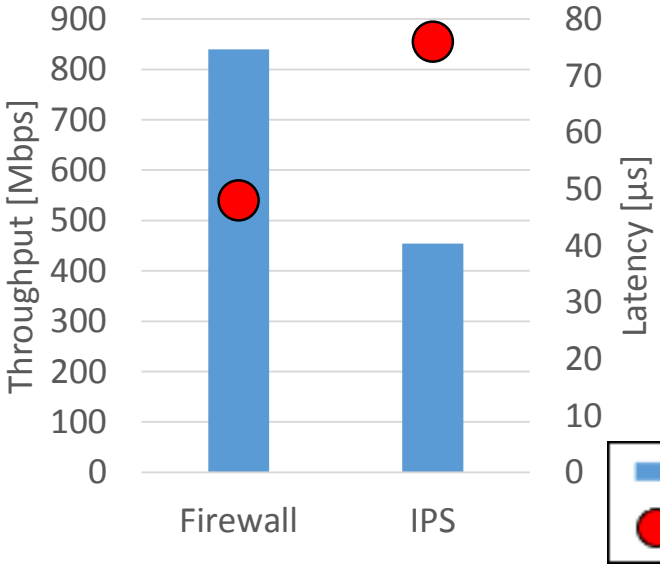
Without OpenBox



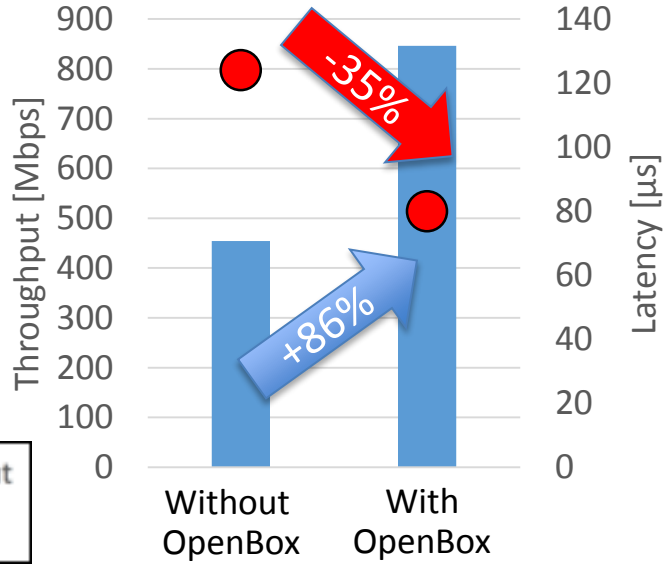
With OpenBox



Standalone VM



NF Pipeline

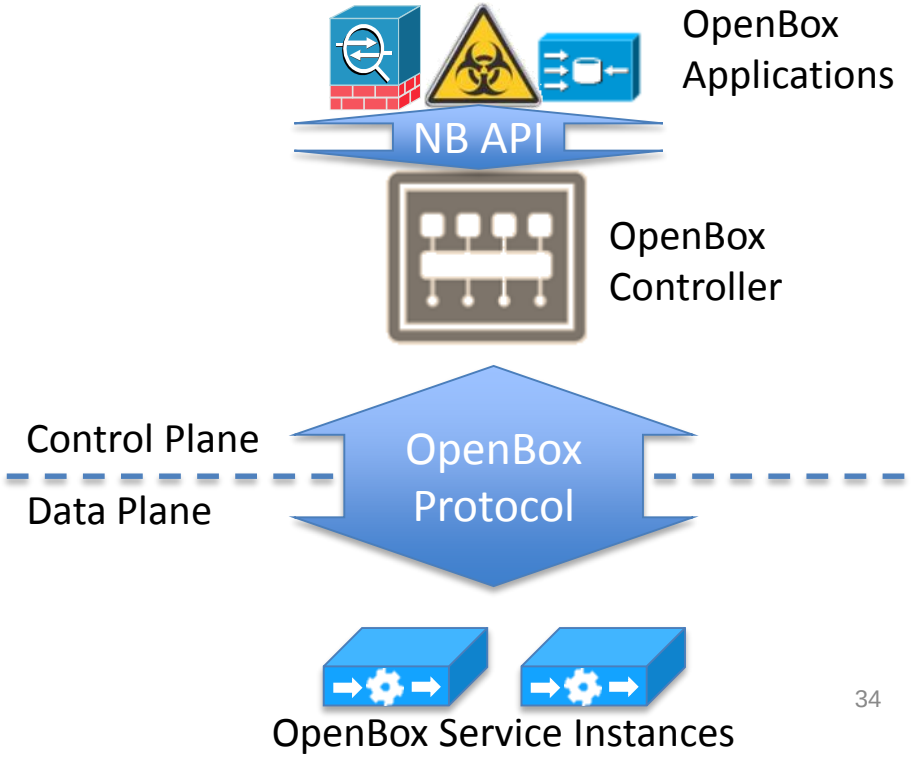


Related Work

- Orthogonal to OpenBox:
  - NF traffic steering (e.g., SIMPLE [SIGCOMM ’14])
  - NF orchestration (e.g., Stratos, OpenMano, OpenStack)
  - Runtime platforms (e.g., xOMB [ANCS ‘12], ClickNP [SIGCOMM ‘16])
- Similar Motivation:
  - CoMb [NSDI ‘12] – focuses on resource sharing and placement
  - E2 [SOSP ‘15] – composition framework for virtual NFs
  - Slick [SOSR ’15] – focuses on the placement of data plane units
- Only OpenBox provides:
  - Core processing decomposition and reuse
  - Standardization and full decoupling of NF control and data planes

Conclusions

- Network functions are currently a real challenge in large scale networks
- By decoupling the data plane processing of NFs their control logic we:
  - Reduce costs
  - Enhance performance
  - Improve scalability
  - Increase reliability
  - Provide inter-tenant isolation
  - Allow easier innovation
- There is still work to do…



Questions?

THANK YOU!

Play with OpenBox on a Mininet VM:
github.com/OpenBoxProject/openbox-mininet