Causal Analysis for Software-Defined Networking Attacks

Benjamin E. Ujcich[1], Samuel Jero[2], Richard Skowyra[2], Adam Bates[3], William H. Sanders[4], and Hamed Okhravi[2]

[1] GEORGETOWN UNIVERSITY

[2] LINCOLN LABORATORY MASSACHUSETTS INSTITUTE OF TECHNOLOGY

[3] ILLINOIS

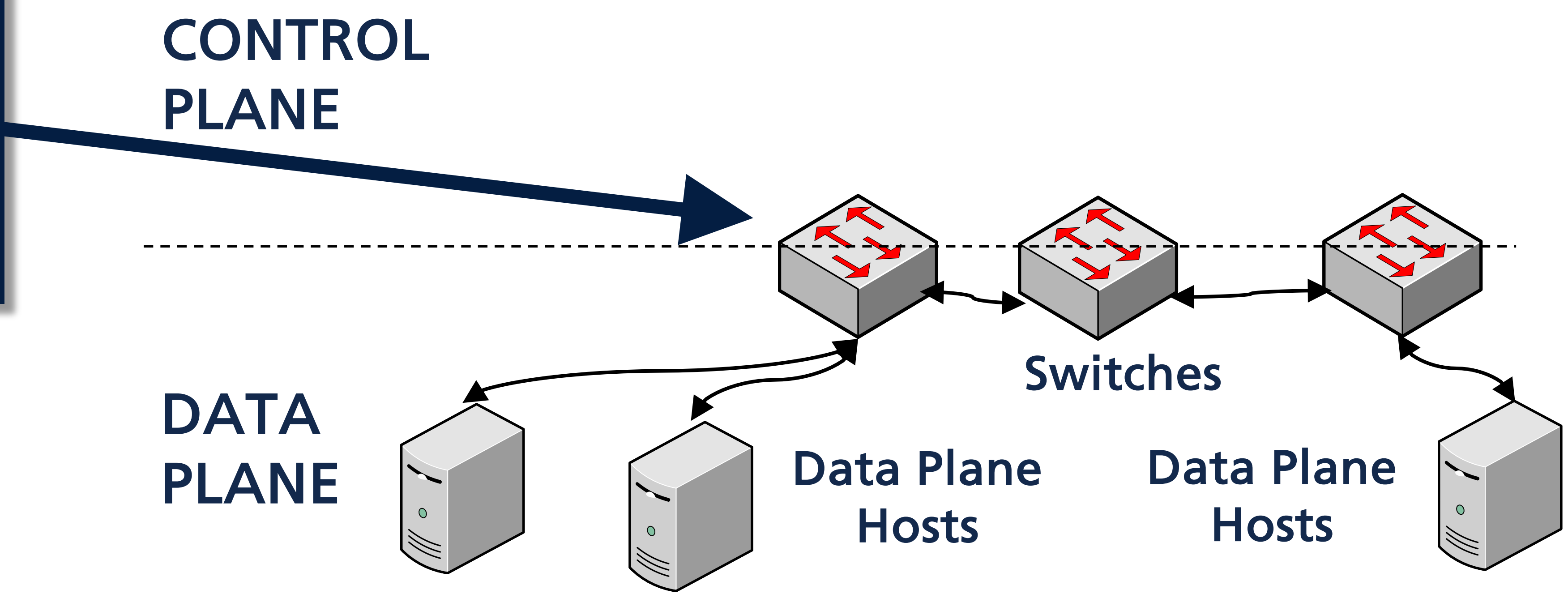
[4] Carnegie Mellon University

30th USENIX Security Symposium

August 11–13, 2021
Virtual Event

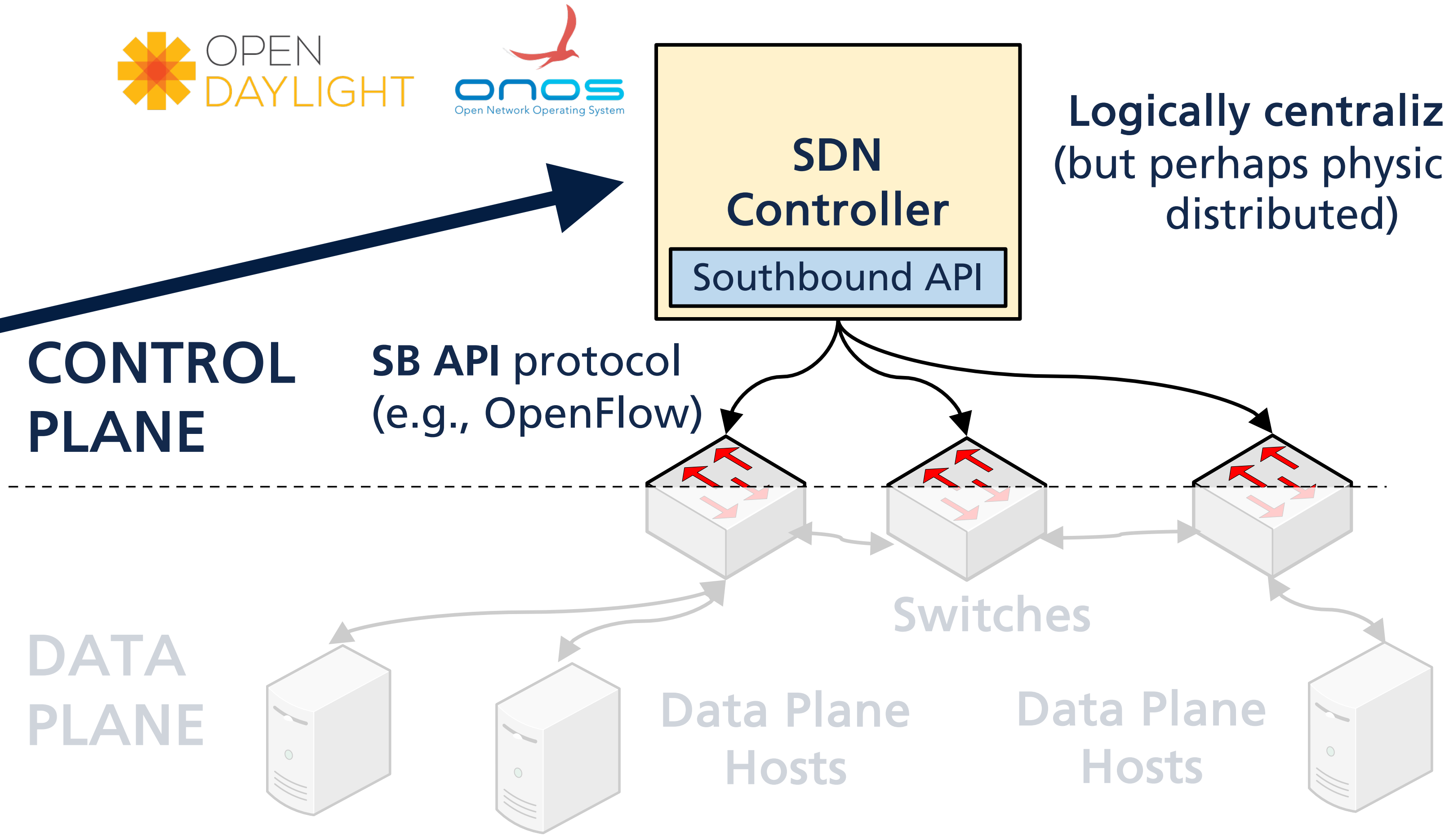
SDN: A Primer

Decoupling of traffic decision-making from traffic being forwarded



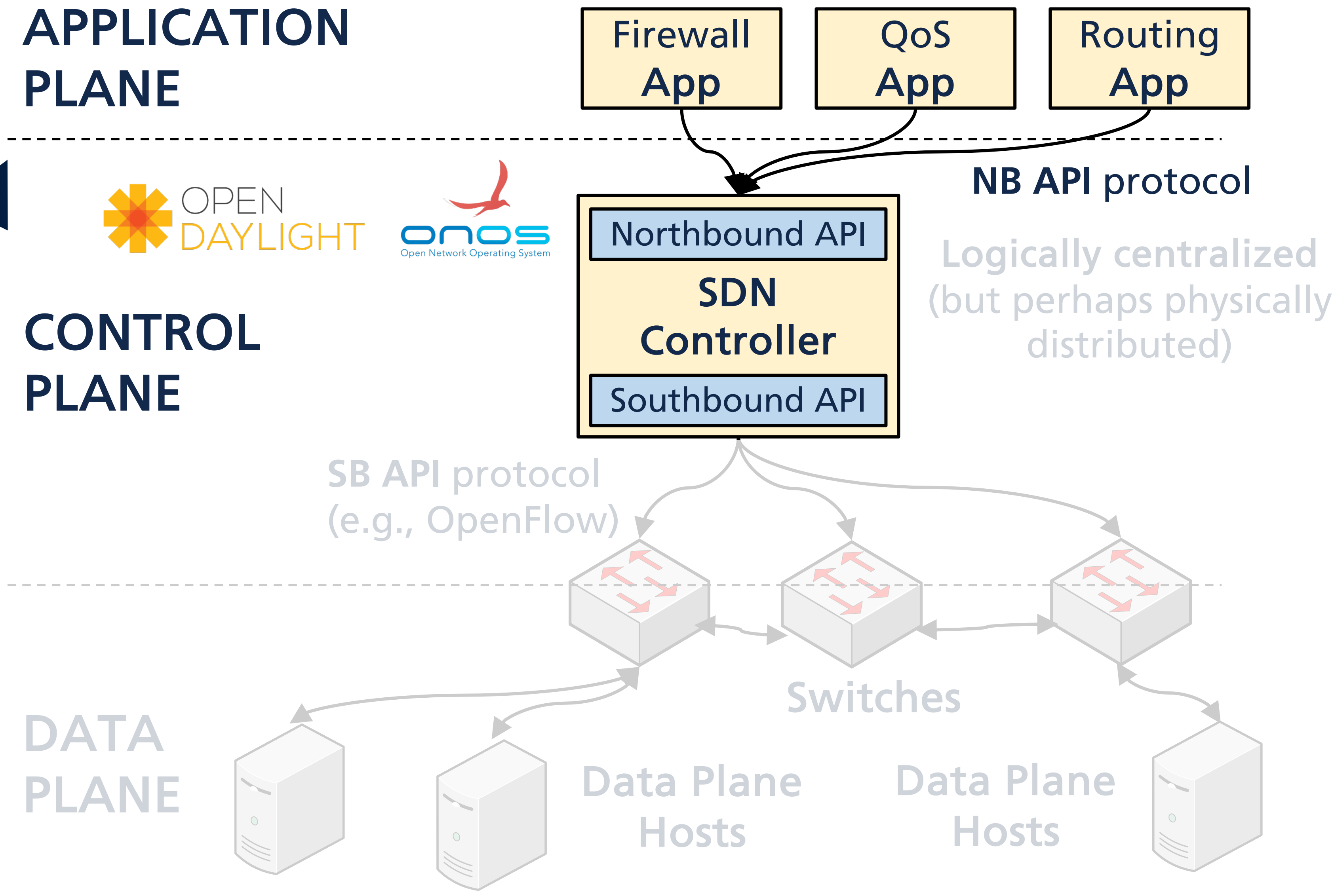
SDN: A Primer

Logically centralized control plane that determines behavior

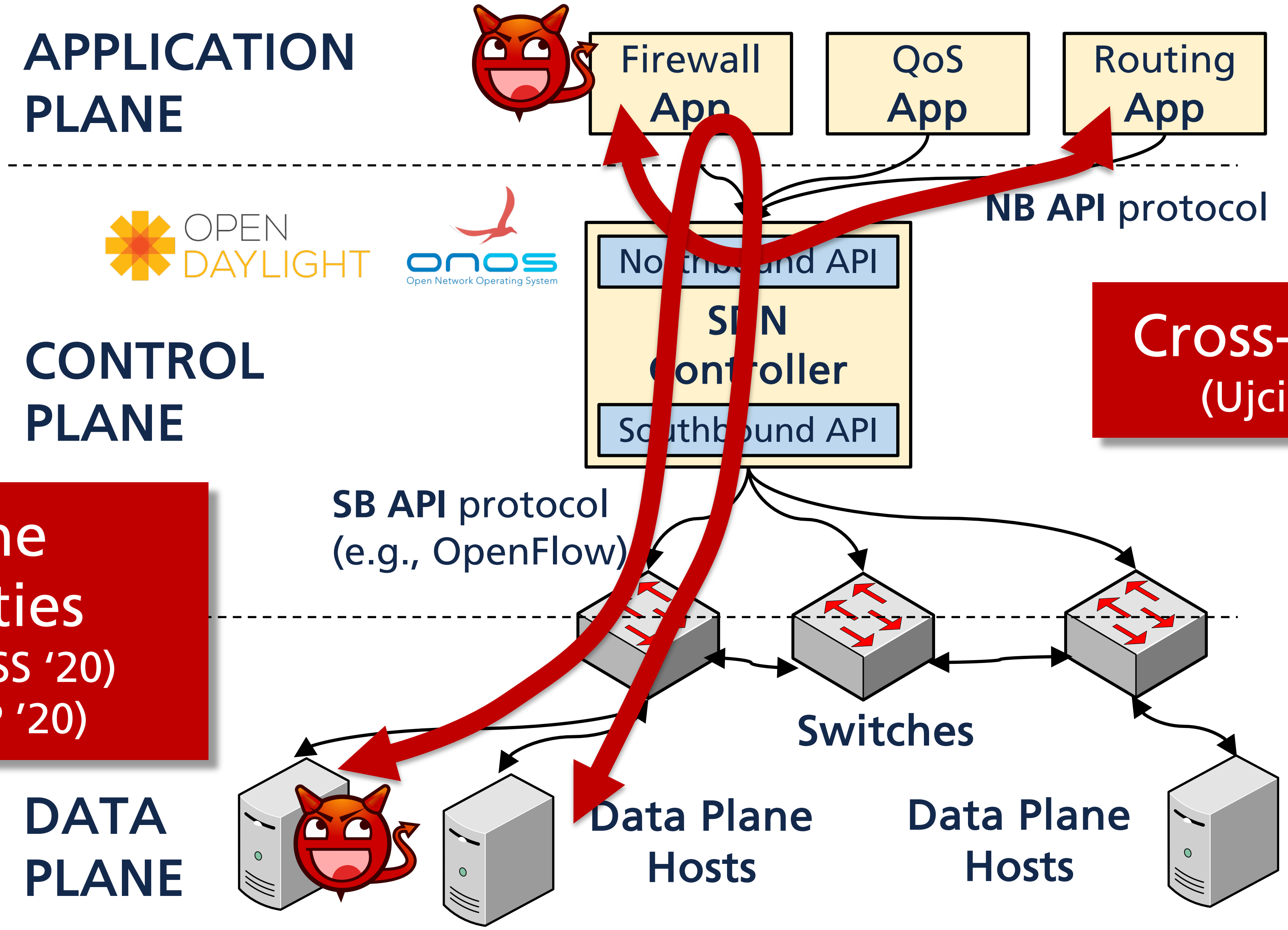


SDN: A Primer

Network services API for extensible network applications



SDN: A Security Target



Cross-app poisoning
(Ujcich et al., CCS '18)

Cross-plane vulnerabilities
(Ujcich et al., NDSS '20)
(Xiao et al., S&P '20)

Help, My SDN Has Been Attacked!

What events happened in my network?

How do I know I have complete oversight?

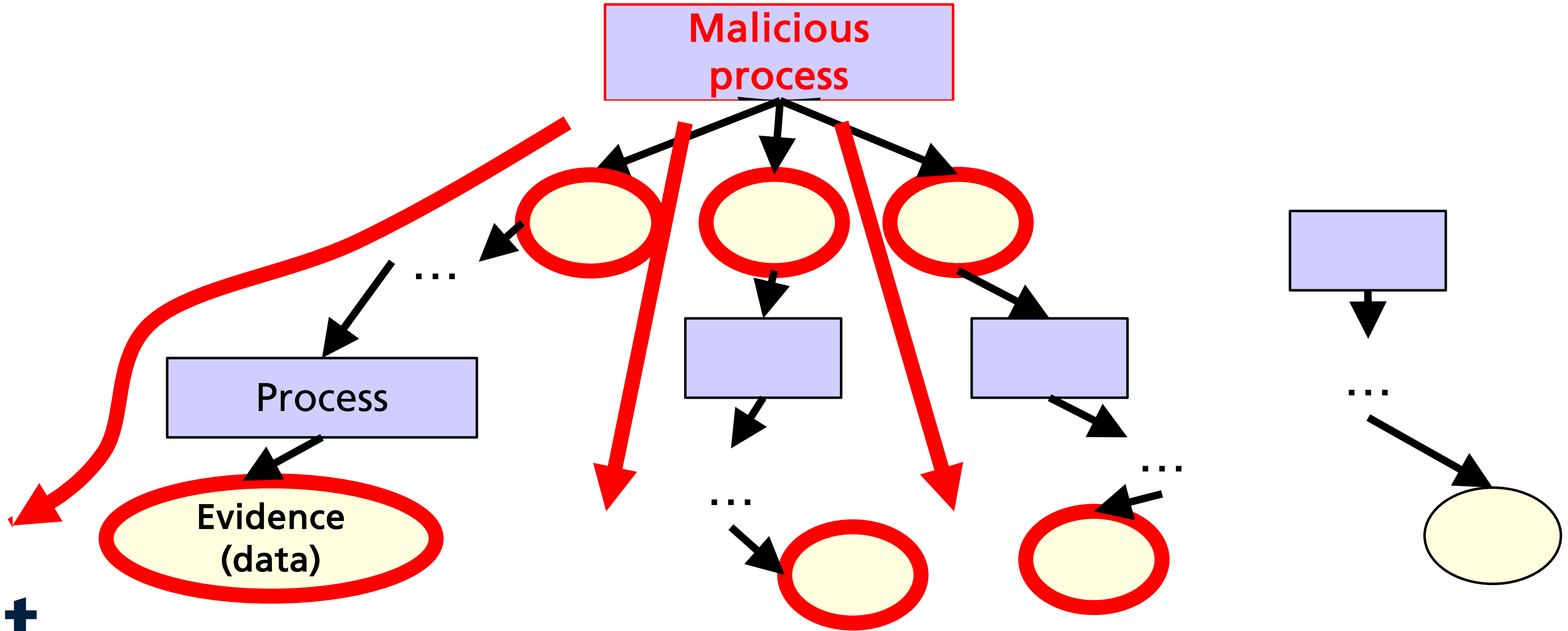
Can I accurately understand the attack?

What are the root causes of the attack?

What else did the attack affect?

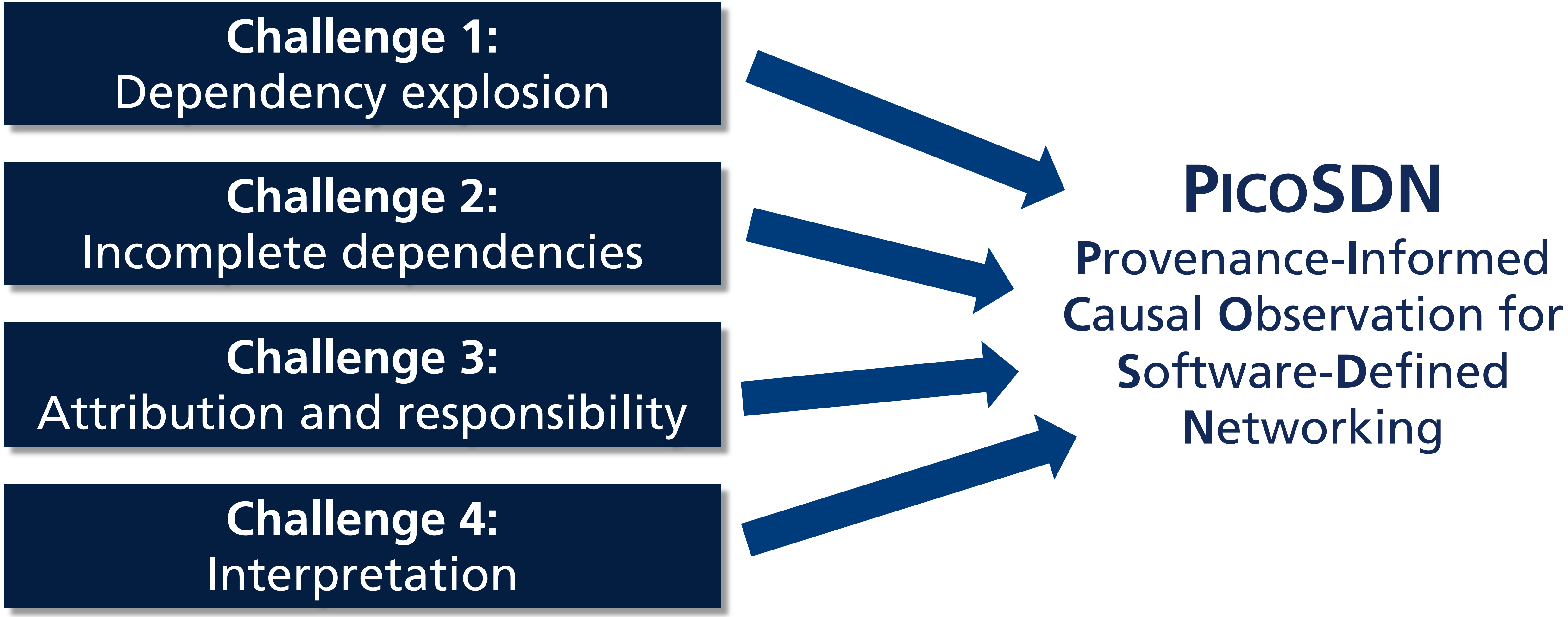
Data Provenance to the Rescue ☺

- Shows how data were **generated** and **used**
- Captures system **principals, processes,** and **data objects** in DAG
- Useful for **attack investigation** and **root cause analysis**



Forward Tracing
(What were the effects?)

What Makes This Challenging for SDN?



Challenge 1:
Dependency explosion

Challenge 2:
Incomplete dependencies

Challenge 3:
Attribution and responsibility

Challenge 4:
Interpretation

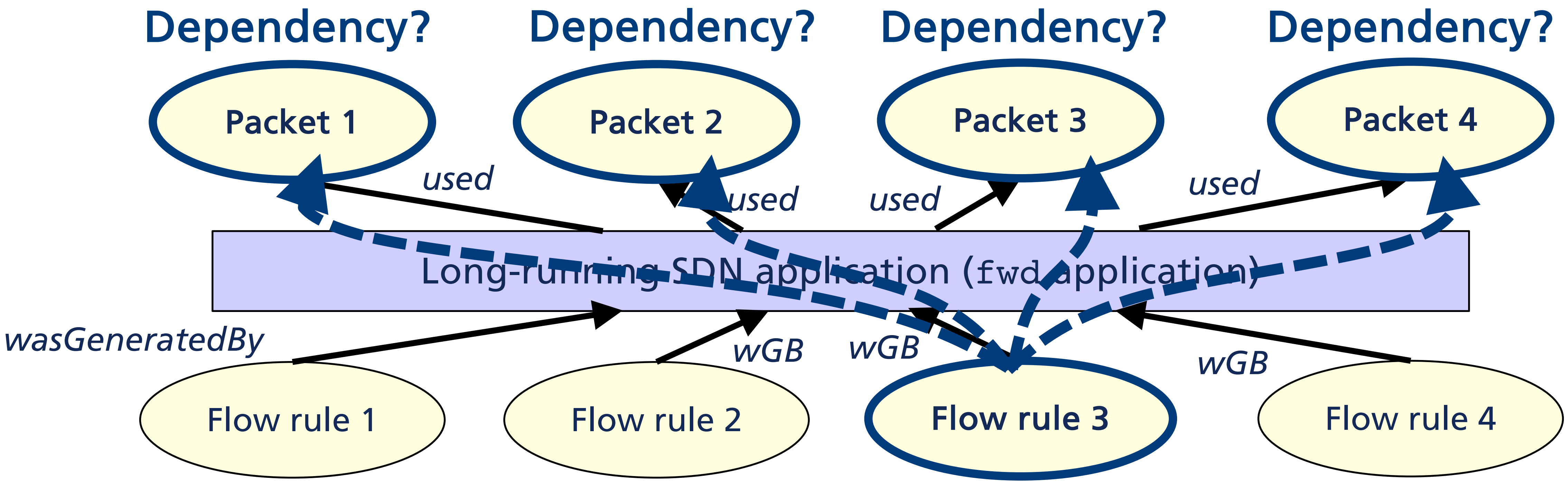
PicoSDN

Provenance-**I**nformed
Causal **O**bservation for
Software-**D**efined
Networking

PicoSDN Challenges and Solutions

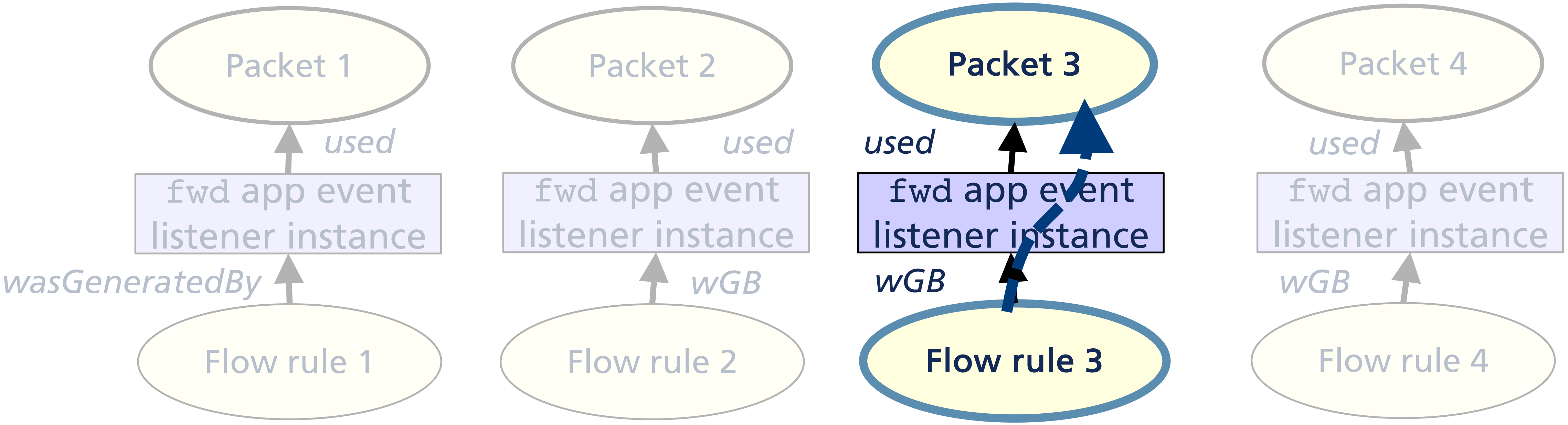
Challenge 1: Dependency explosion

Discovery: Long-running apps produce false data and process dependencies



PicoSDN Challenges and Solutions

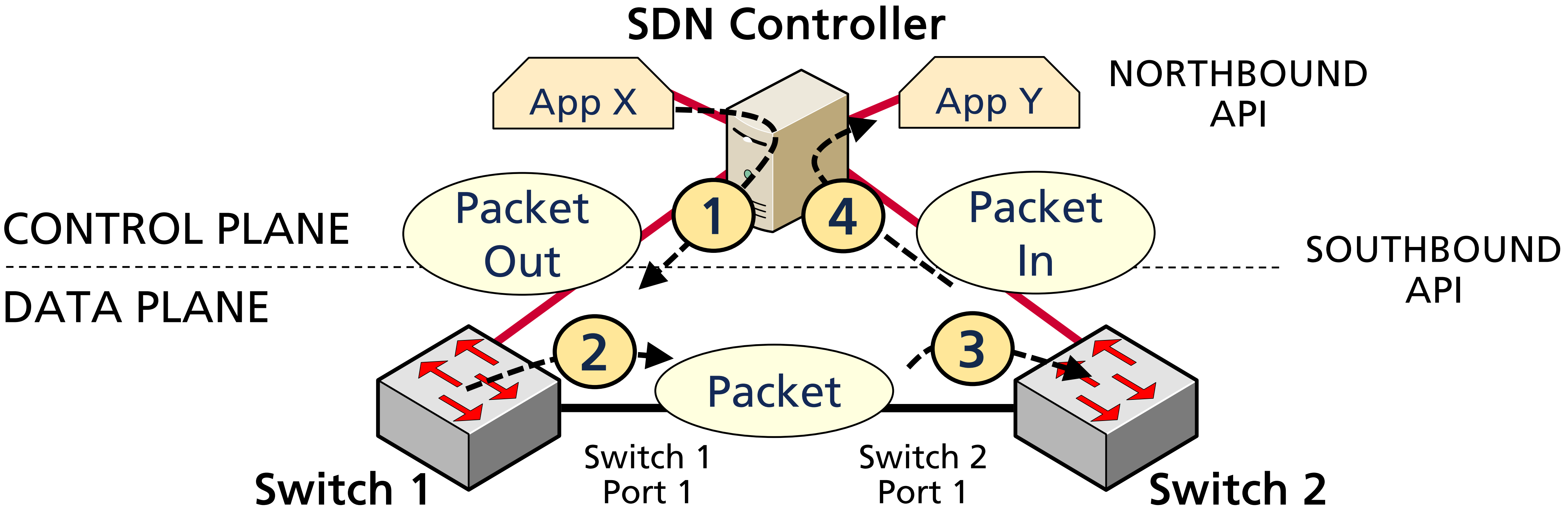
Solution: Mitigate with events as short processes (*execution partitioning*) and control plane objects as data (*data partitioning*)


Packet 1
Packet 2
Packet 3
Packet 4
used
used
used
used
fwd app event listener instance
fwd app event listener instance
fwd app event listener instance
fwd app event listener instance
wasGeneratedBy
wGB
wGB
wGB
Flow rule 1
Flow rule 2
Flow rule 3
Flow rule 4

PicoSDN Challenges and Solutions

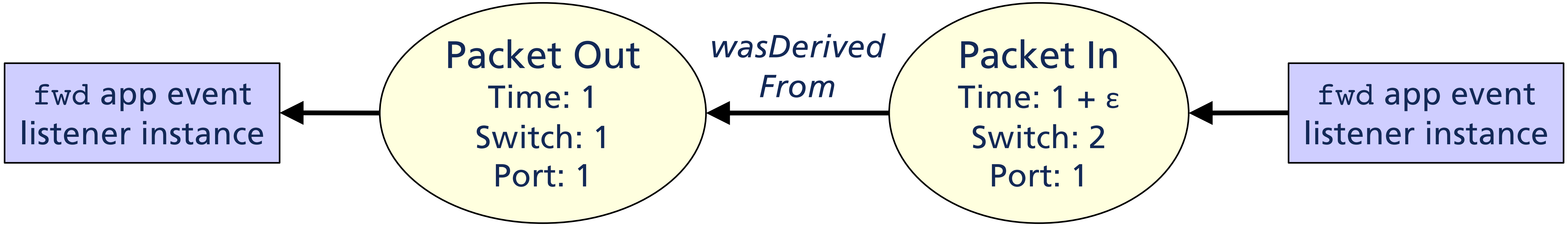
Challenge 2: Incomplete dependencies

Discovery: Control plane can trigger other control plane activities via the data plane



PicoSDN Challenges and Solutions

Solution: Mitigate by modeling the data plane

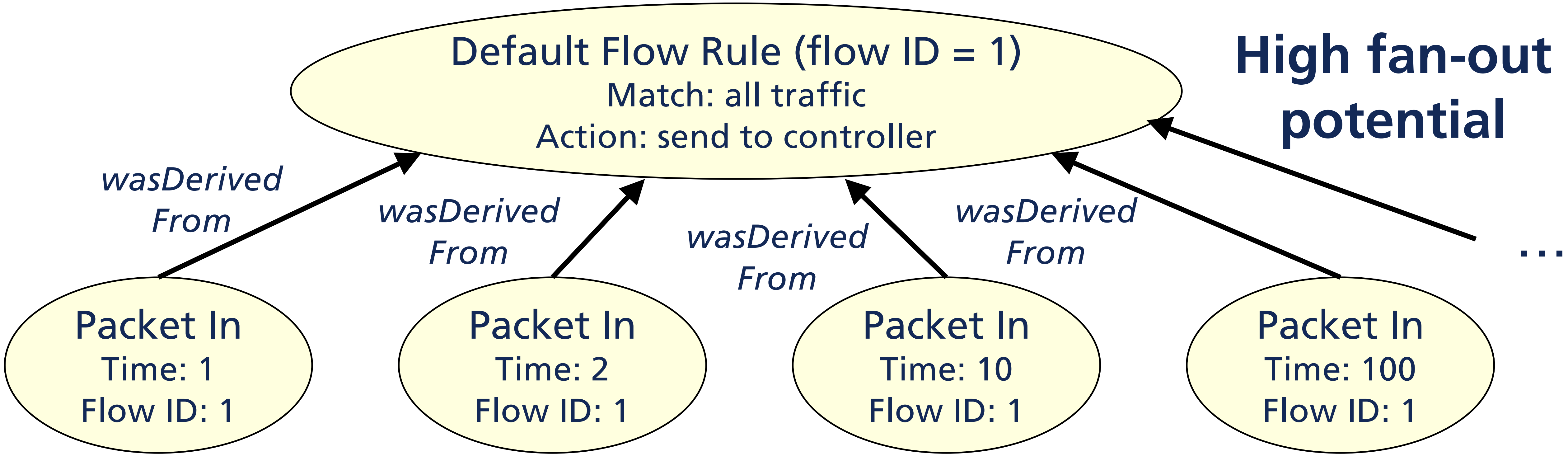


Data plane model based on:
happens-before relations, packet timestamps (within threshold), match fields, and network topology

PicoSDN Challenges and Solutions

Challenge 3: Attribution and responsibility

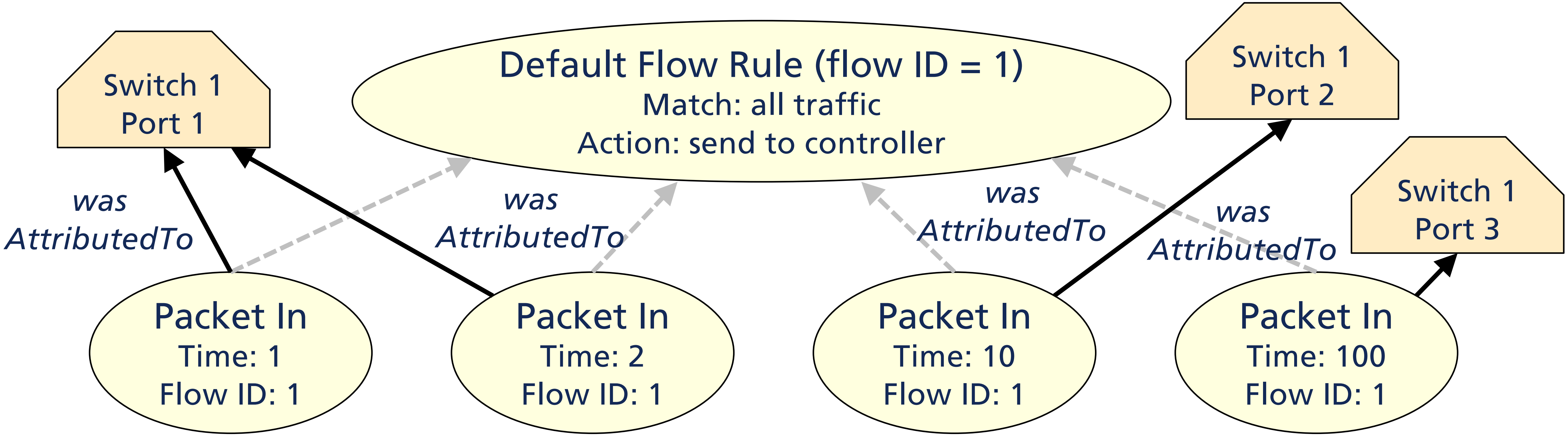
Discovery: Default flow rules create a data dependency explosion



High fan-out potential

PICOSDN Challenges and Solutions

Solution: Mitigate by assigning agency to a switch port



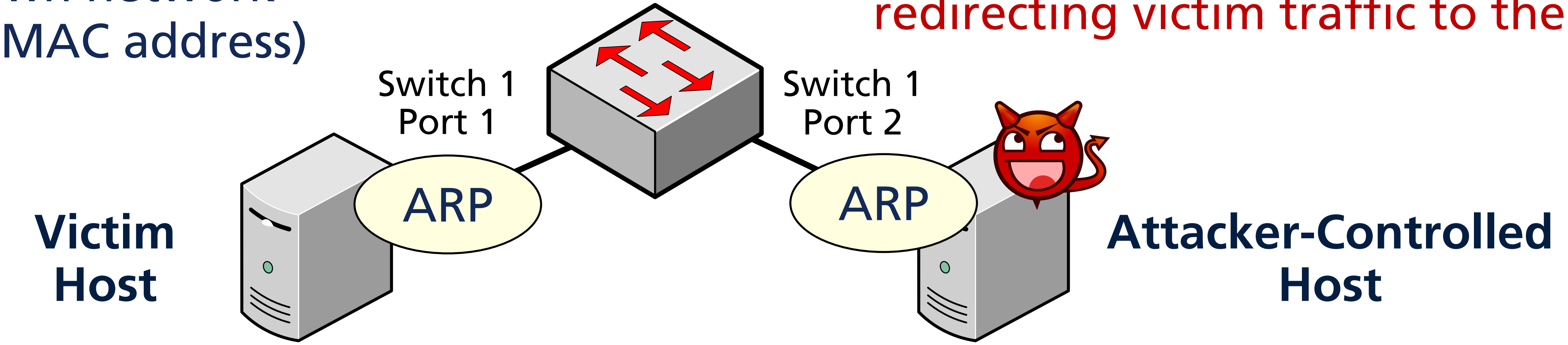
PicoSDN Challenges and Solutions

Challenge 3: Attribution and responsibility

Discovery: Host identifiers can be easily spoofed

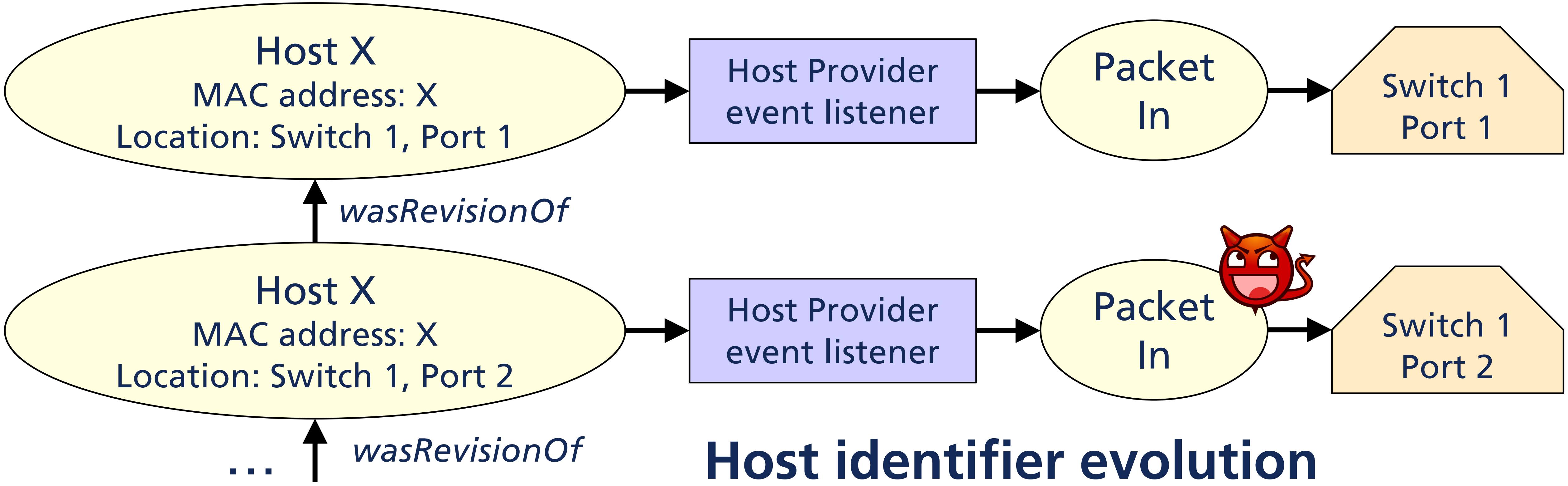
Victim sends packets with its own network identifier (MAC address)

Attacker spoofs victim's network identifier to trick SDN controller into redirecting victim traffic to the attacker



PicoSDN Challenges and Solutions

Solution: Track how hosts' identifiers change over time

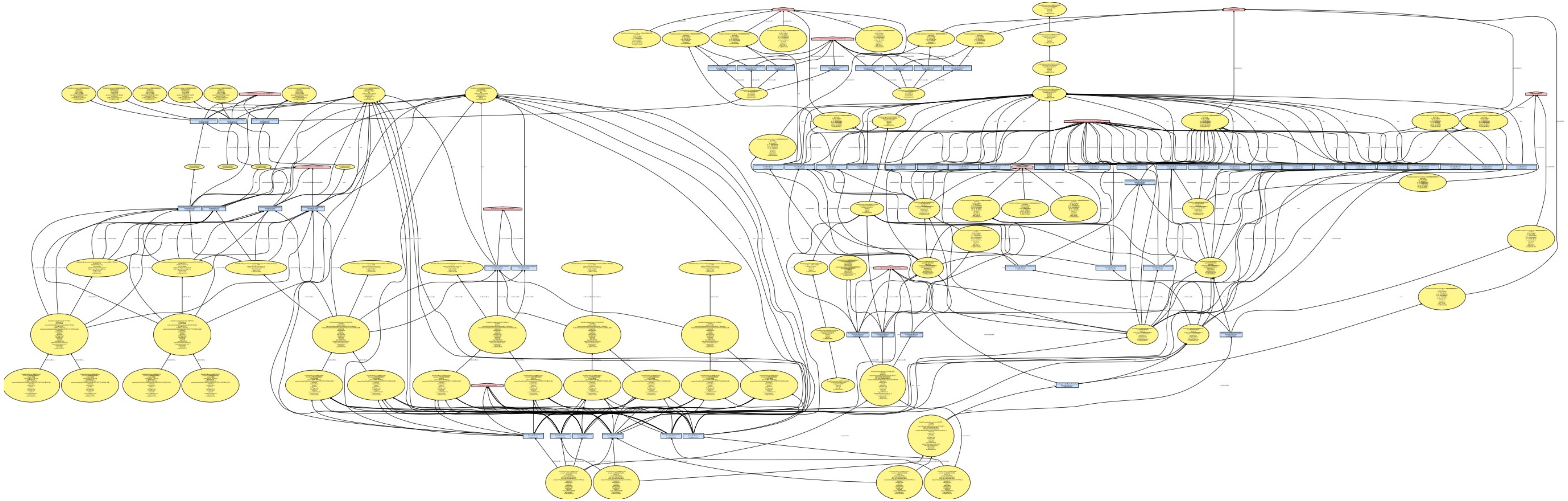


Host identifier evolution

PicoSDN Challenges and Solutions

Challenge 4: Interpretation

Discovery: Provenance is not useful unless we can understand it

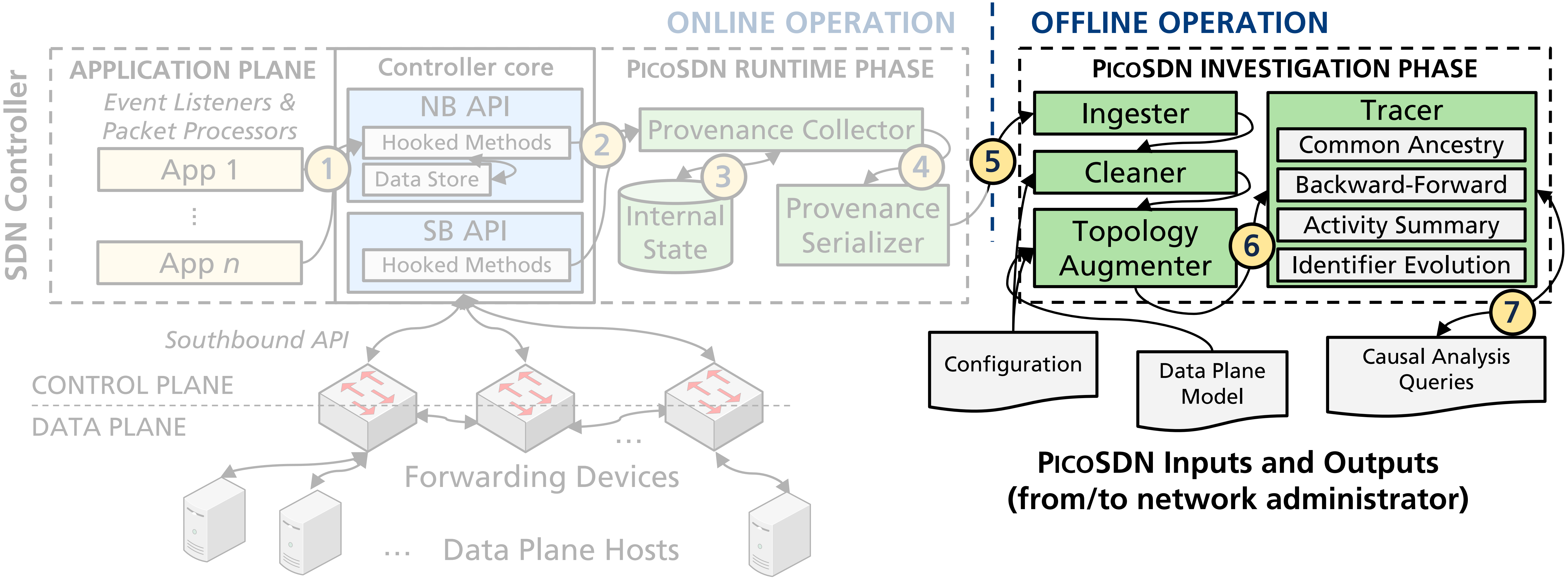


PicoSDN Challenges and Solutions

Solution: Provide practical tools to summarize, analyze, and trace network activities

1. **Common ancestry:** Given several nodes, what nodes are the common ancestors?
2. **Backward-forward:** Given a path between evidence (node) and root, what does the ancestry look like at each stage?
3. **Activity summary:** How do data plane packets impact flow rules?
4. **Identifier evolution:** How do hosts change identity?

PicoSDN Architecture

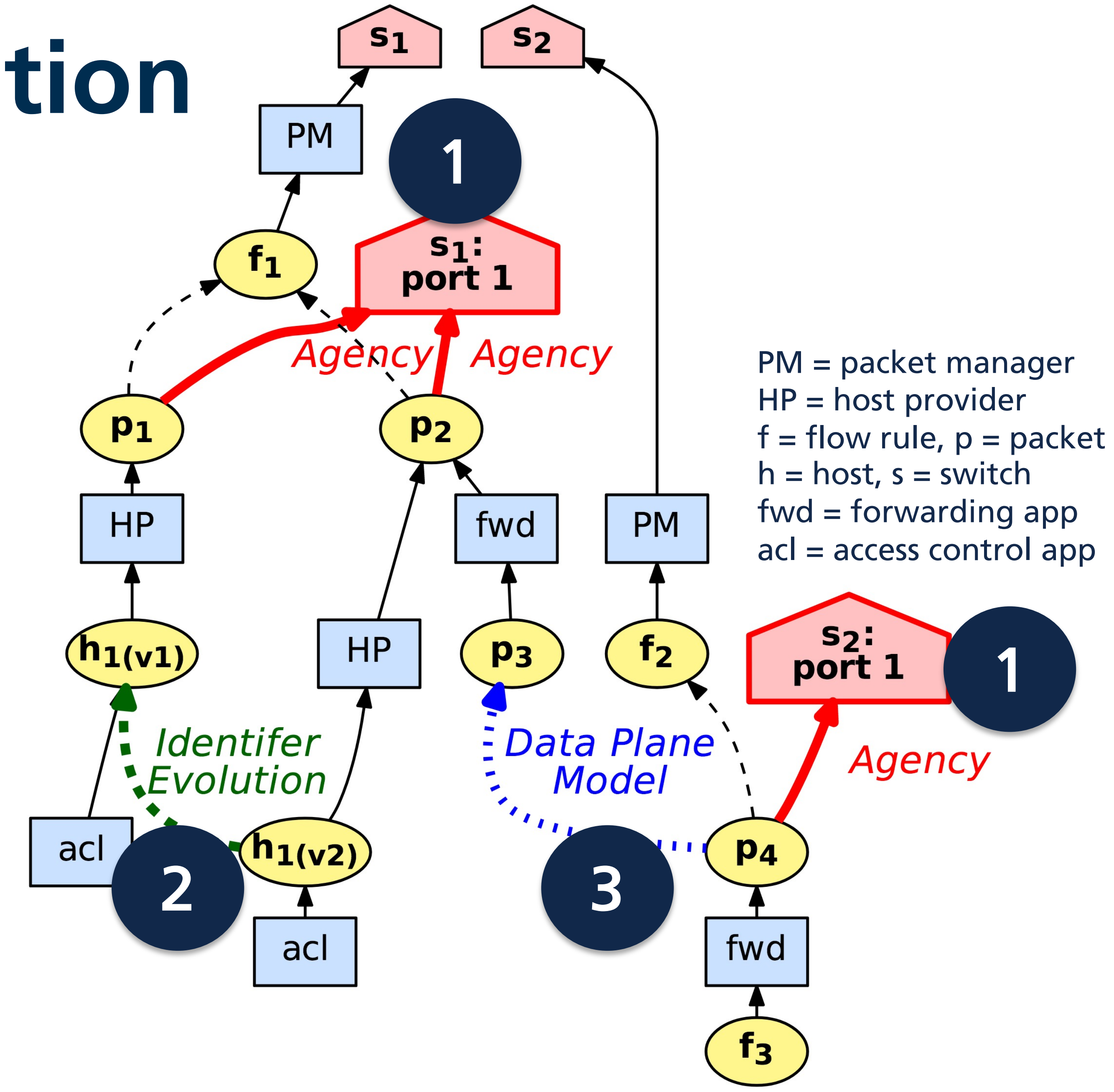


PicoSDN Inputs and Outputs (from/to network administrator)

PicoSDN Security Evaluation

Example: Cross-Plane Event-Based Vulnerabilities

1. Switch ports as agents
2. Host identifier evolution (i.e., spoofing)
3. Data plane model based on reactive forwarding



PM = packet manager
HP = host provider
f = flow rule, p = packet
h = host, s = switch
fwd = forwarding app
acl = access control app

Conclusions

- Considered **causal analysis challenges** in **SDN attacks**
- Design takeaways
  - **Dependency explosion** mitigated by control plane **control plane objects** (data) and **events** (execution)
  - **Incomplete dependencies** mitigated by **data plane model**
  - **Attribution** and **responsibility** are **challenging**
- Designed **PicoSDN** and implemented on ONOS SDN controller

Thanks!

Thank you for your time!

Benjamin E. Ujcich
E-mail: bu31@georgetown.edu
Web: https://benujcich.georgetown.domains/

This work was supported in part by NSF Grant No. CNS-1750024. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.