Lecture: Software-Defined Networks

Presenter: Ben Ujcich

CS 461 / ECE 422

Fall 2019

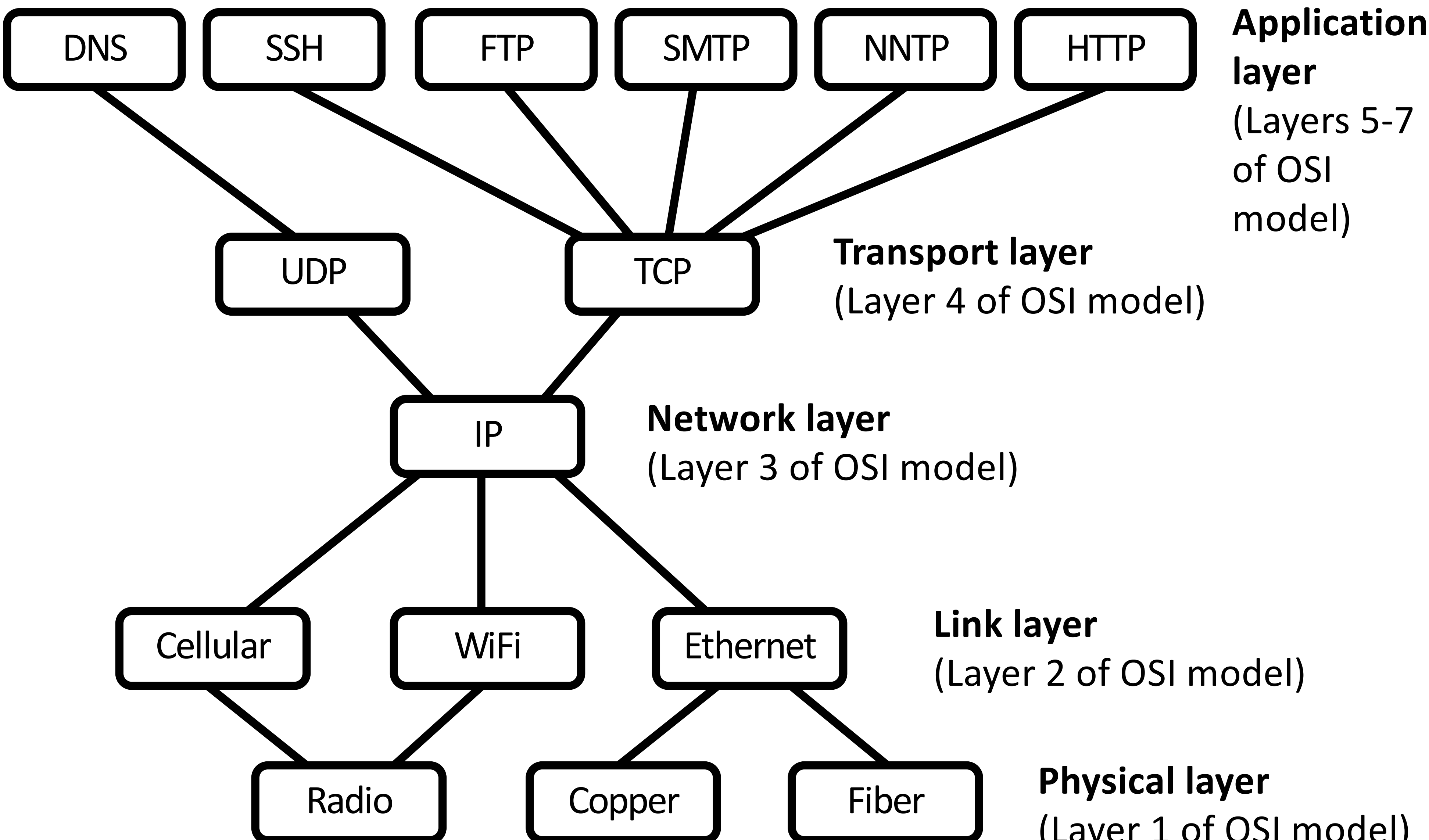
About Me

- PhD student in ECE department working with Adam Bates and Bill Sanders
- Broad area of research in security of systems and networks
- Specific area of research in security of software-defined networking (SDN) architectures

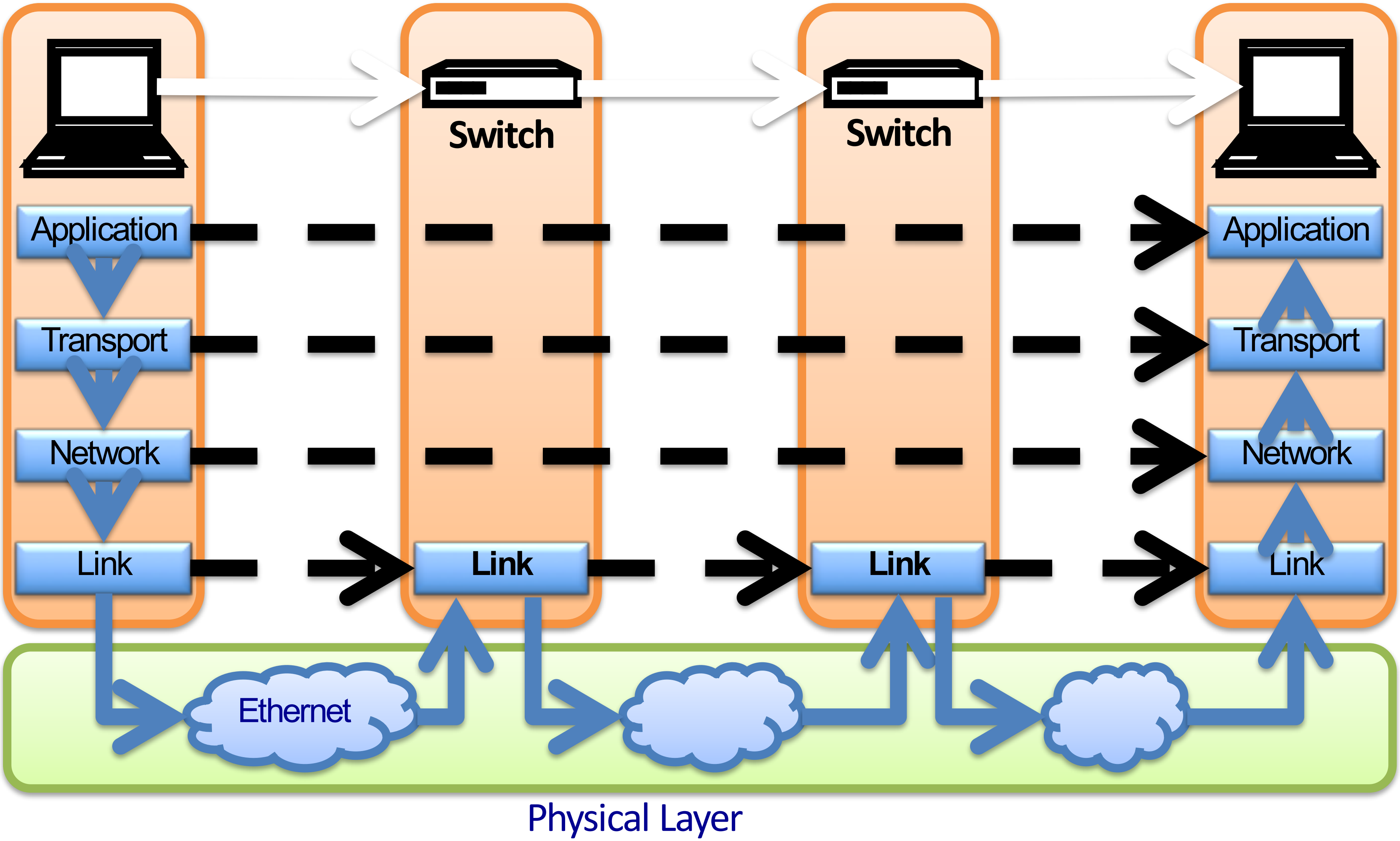
Goals for Today

- Understand the **challenges** and **limitations** of **traditional networking** architectures
- Understand the **software-defined networking (SDN)** architecture
- Understand the **security benefits** and **ramifications** of SDN
- Understand SDN **attacks** and SDN **defenses**

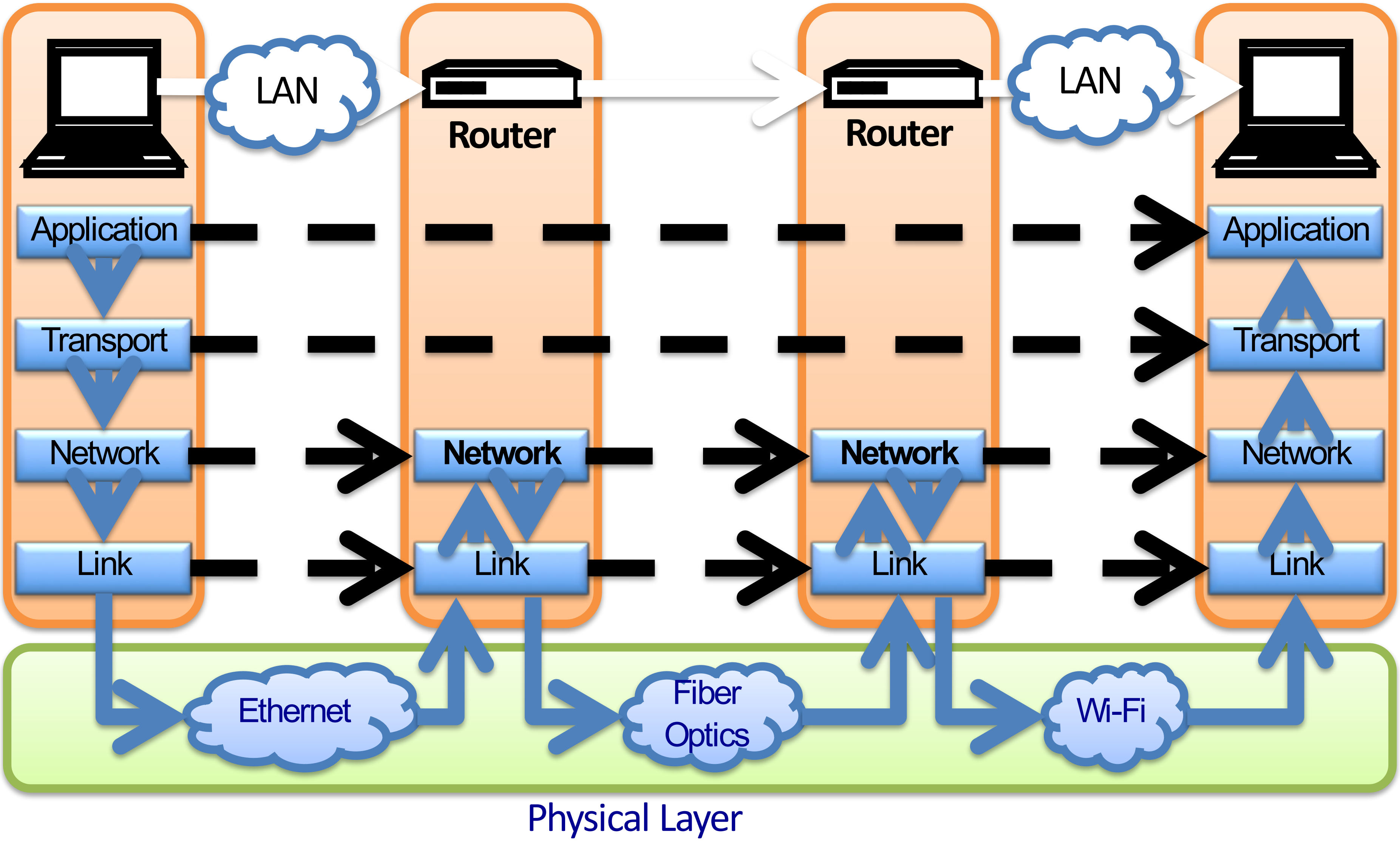
Recall: Network Stack



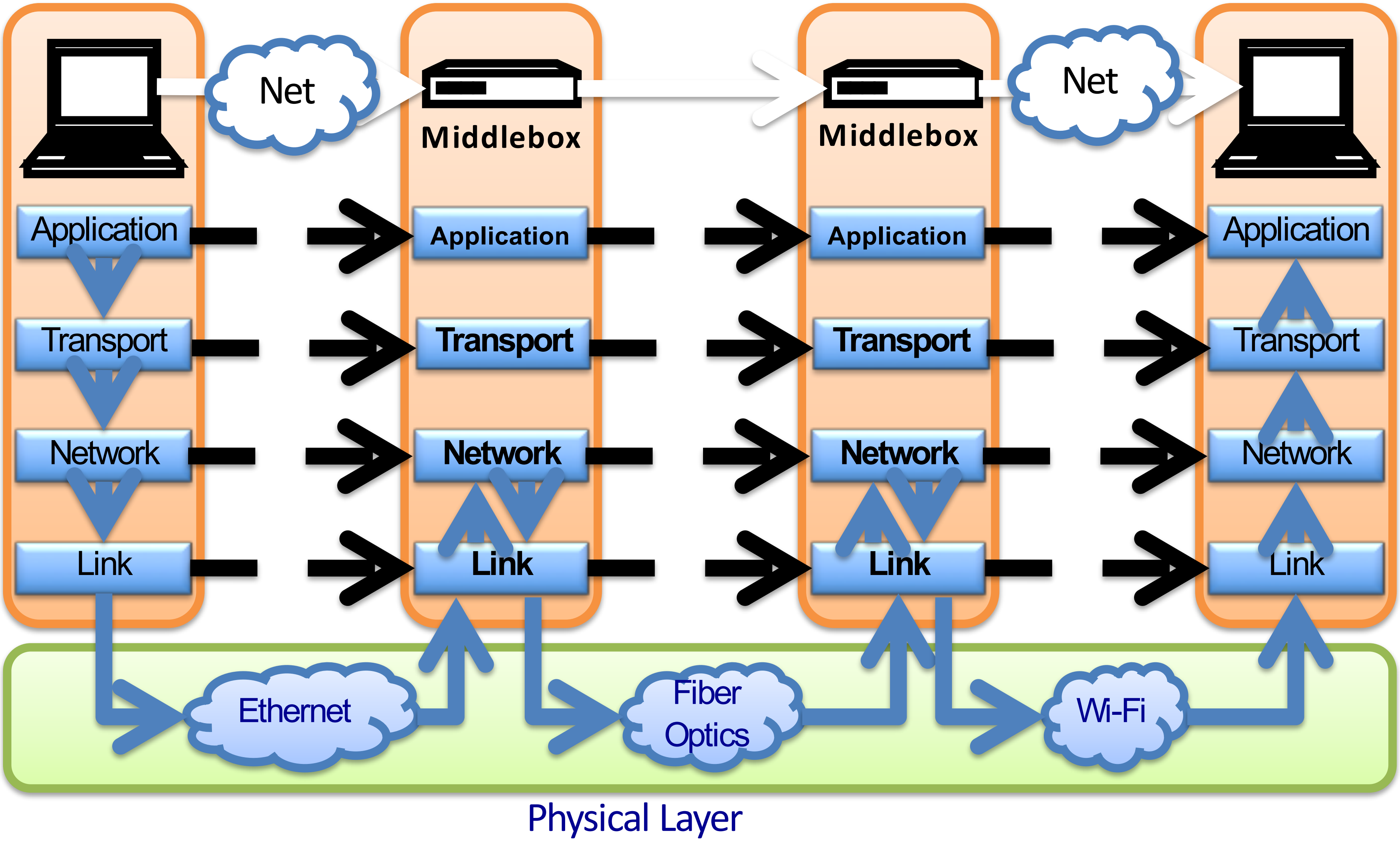
Recall: Local Area Networks (LANs)



Recall: Internet



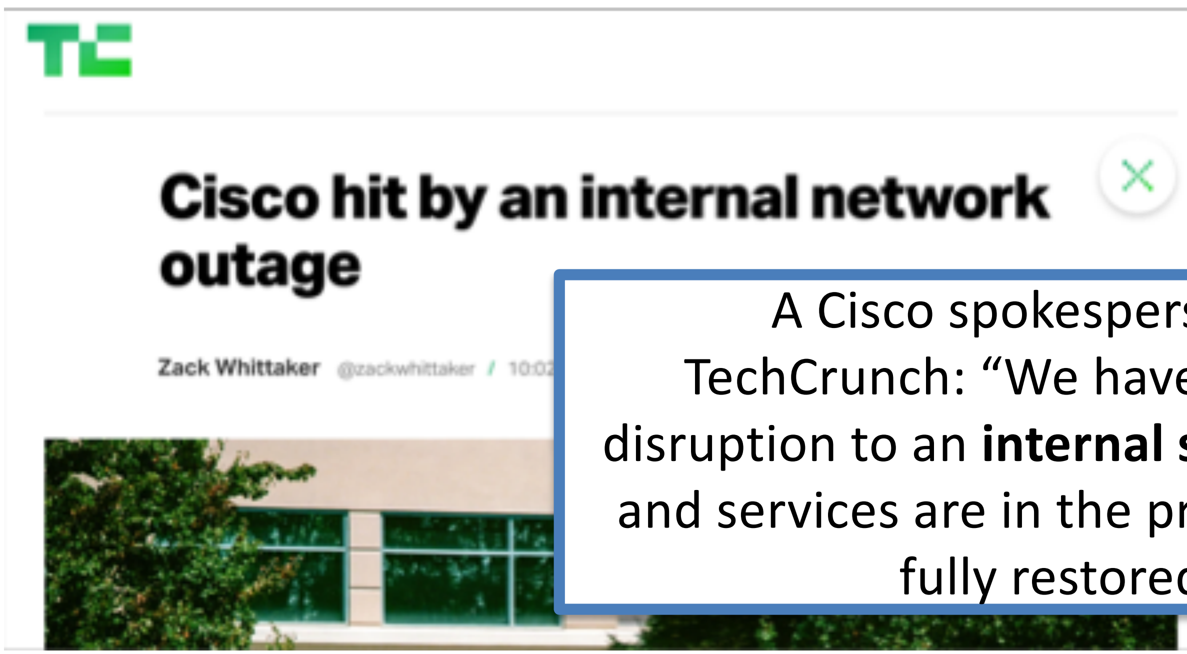
Higher Layers?



Challenges

Difficult to modify network configuration

- Getting the network configuration correct is challenging and error-prone
- Network outages can violate availability property

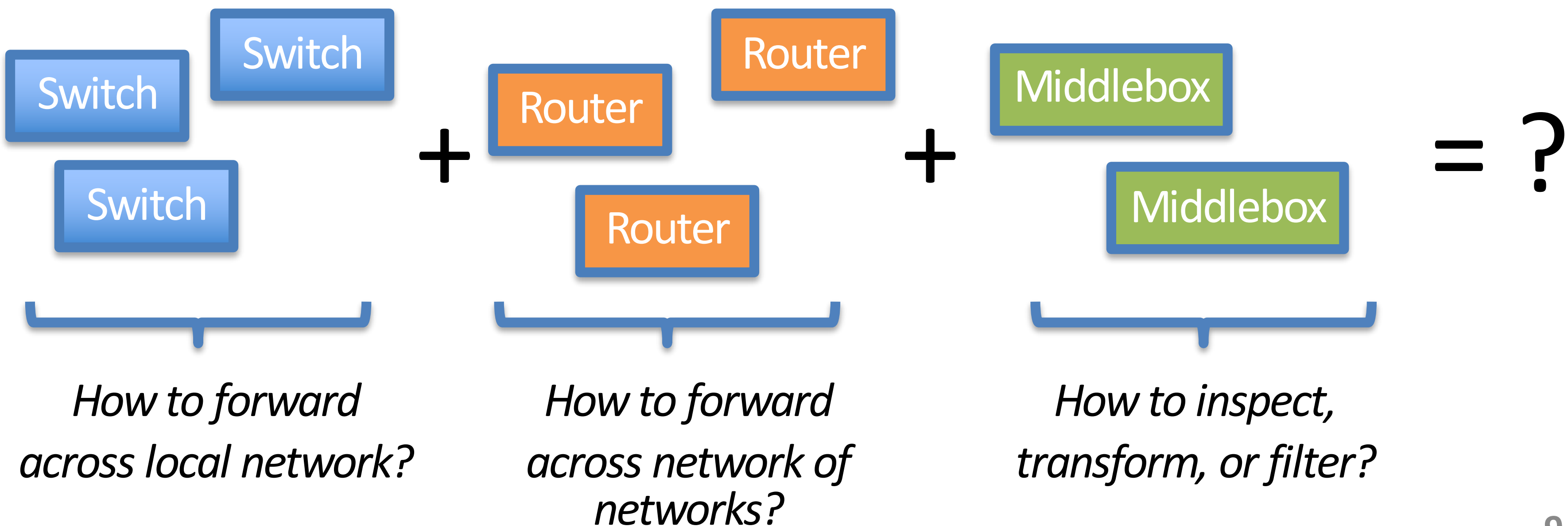


A Cisco spokesperson told TechCrunch: “We have traced our disruption to an **internal system change**, and services are in the process of being fully restored.”

Challenges

Difficult to extend / coordinate network functionality and concerns

- Creates cross-layer, cross-protocol problems
- May be limited by physical configuration



Challenges

Difficult to separate control and data plane

- The data plane forwards packets (or frames or datagrams)
- The control plane decides behavior of how packets should be forwarded
- Control and data plane often coupled → *switches care about L2, routers about L3, etc.*
- Individual switches and routers maintain their own views of network state → *network state is distributed*

Solution: Programmable Networks

- **Insights:**
  1. Decouple control and data planes → *more flexible forwarding across layers of network stack*
  2. Centralize control plane → *coordinated network state and greater insight into decision-making*
  3. Turn decision-making into a set of software processes → *an extensible control plane for an application domain of interest*
- Programmable networks realized as software-defined networking or SDN

SDN Data Plane Design

Traditional Networks

- **Switches** implement forwarding tables based on **MAC addresses**

$L2_{src}$	$L2_{dst}$	Action

- **Routers** implement forwarding tables based on **IP addresses / prefixes**

$L3_{src}$	$L3_{dst}$	Action

SDN

- Generic forwarding devices ("switches") implement forwarding tables based on **Layers 2–4 header fields**

$L2_s$	$L2_d$	$L3_s$	$L3_d$	…	Action

Commonly realized with OpenFlow protocol

OpenFlow

SDN Data Plane Design

Traditional Networks

- **Switches** implement forwarding tables based on MAC addresses

Takeaway: Network stack layers in SDN can be mixed

- **Routers** implement forwarding tables based on **IP addresses / ranges**

L3src	L3dst	Action

SDN

- Generic forwarding devices implement forwarding tables based on combination of **Layers 2–4 attributes**

$L2_s$	$L2_d$	$L3_s$	$L3_d$	...	Action

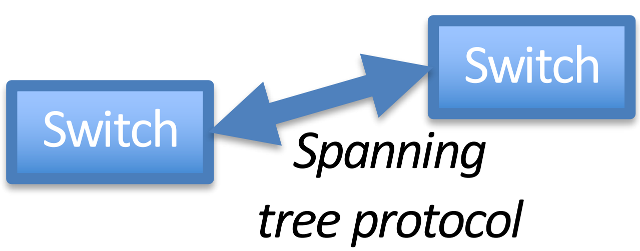
Commonly realized with OpenFlow protocol

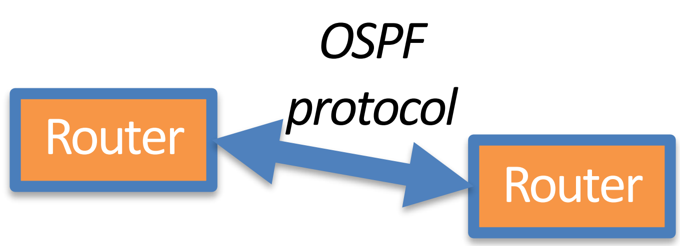
OpenFlow

SDN Control Plane Design

Traditional Networks

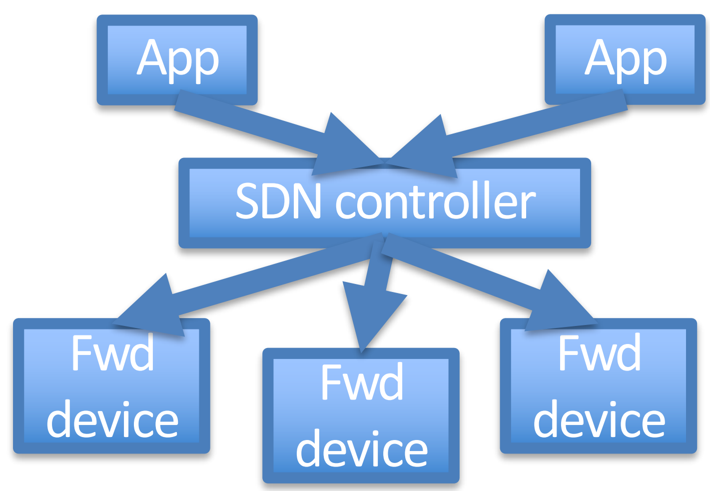
- **Distributed protocols** (e.g., STP, OSPF, BGP, etc.)
- Decision-making resides **on each device**





SDN

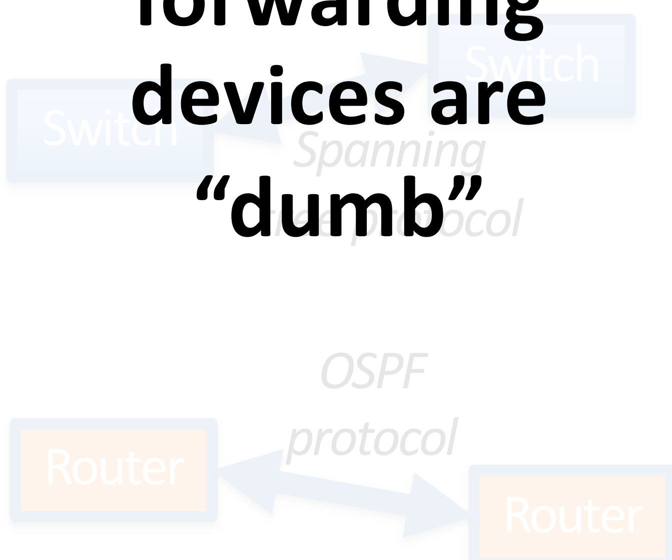
- **Centralized protocols** implemented in software programs (or apps)
- Decision-making resides **in SDN controller** (OpenFlow as configuration protocol)



SDN Control Plane Design

Traditional Networks

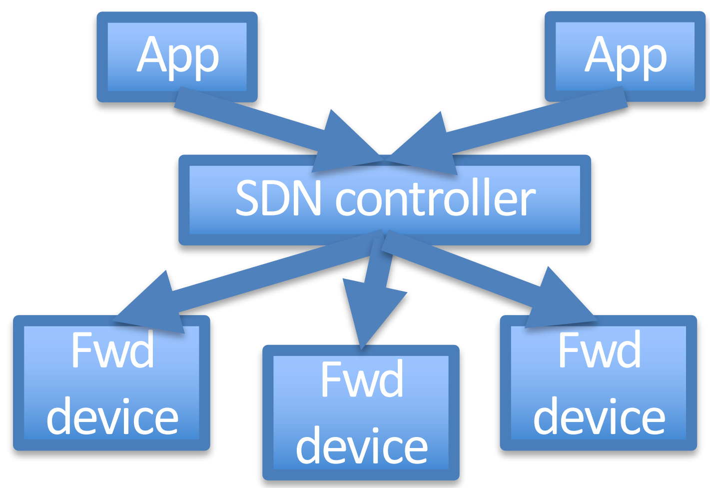
- **Distributed protocols** (e.g., STP, OSPF, BGP, etc.)
- Decision-making resides **on each device**



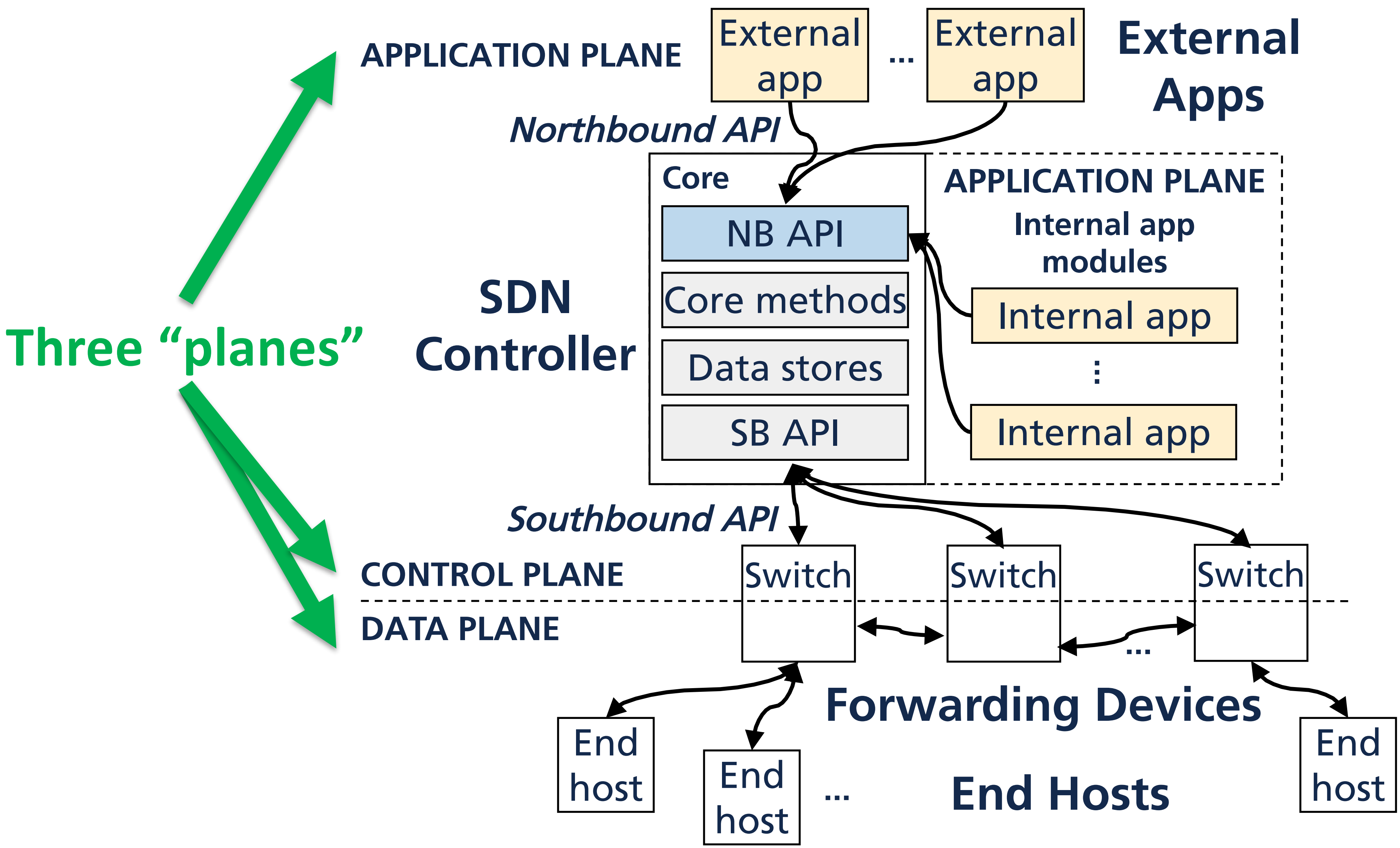
Takeaway: Network forwarding devices are "dumb"

SDN

- **Centralized protocols** implemented in software programs (or apps)
- Decision-making resides **in SDN controller** (OpenFlow as configuration protocol)

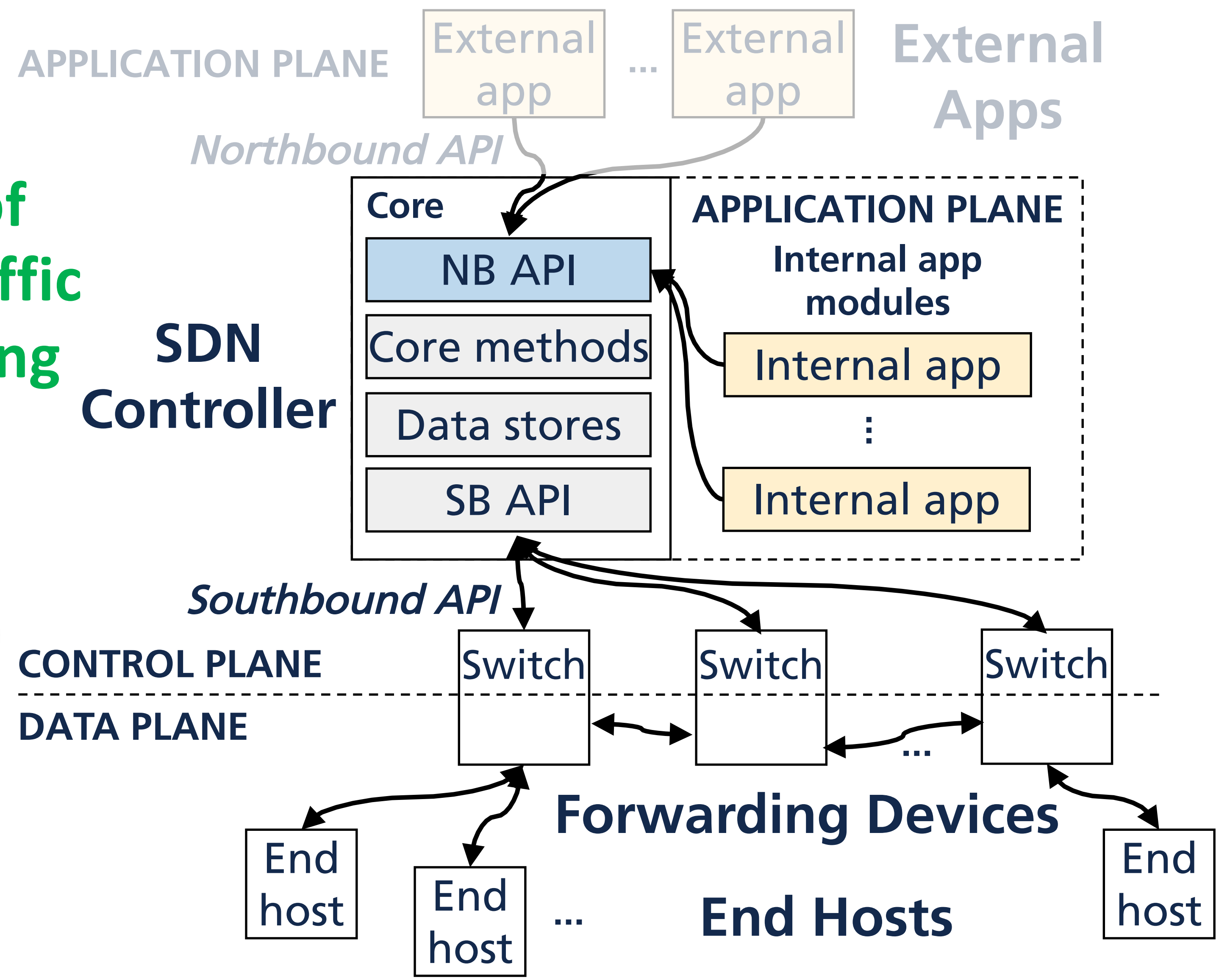


SDN Architecture

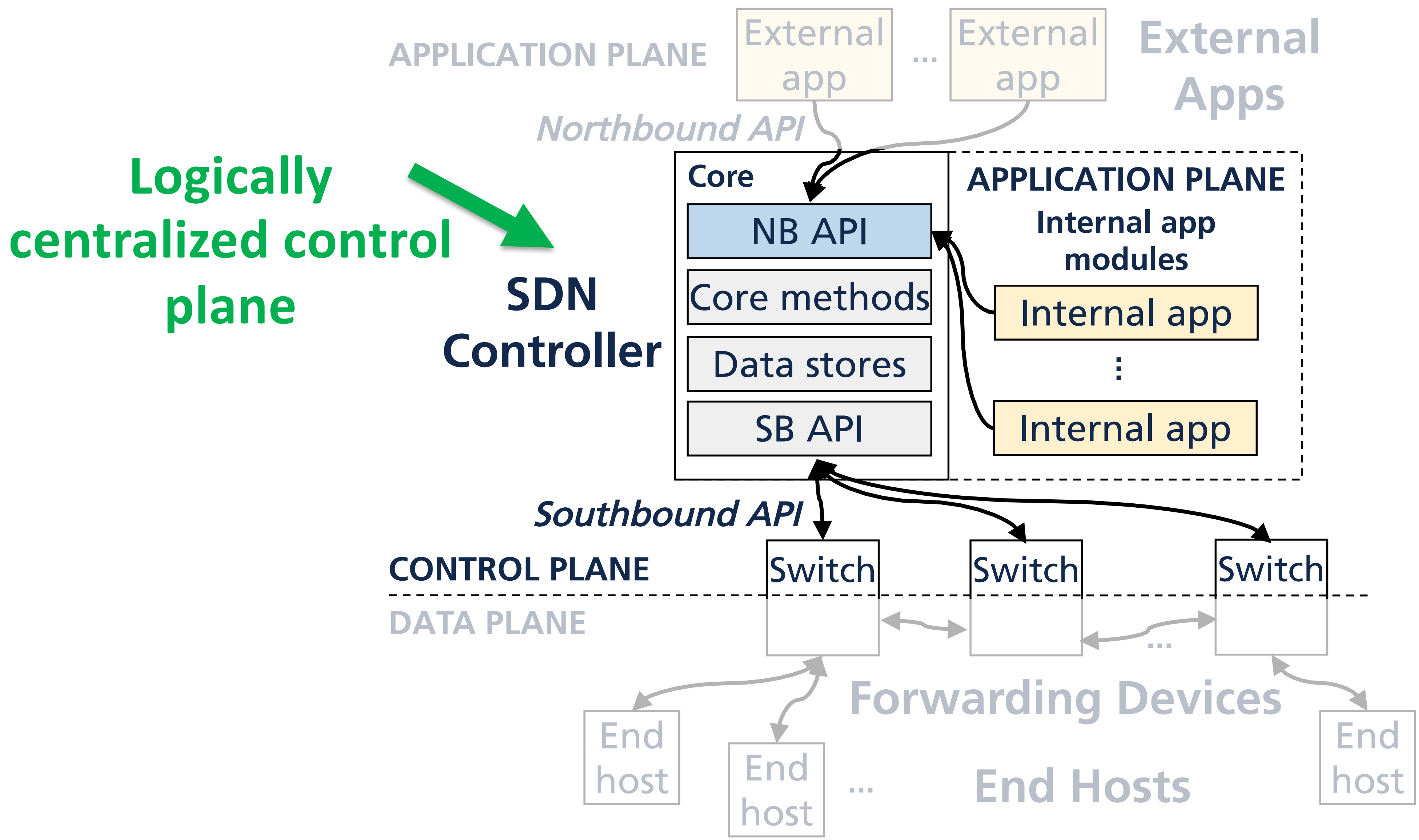


SDN Architecture

Decoupling of traffic from traffic decision-making

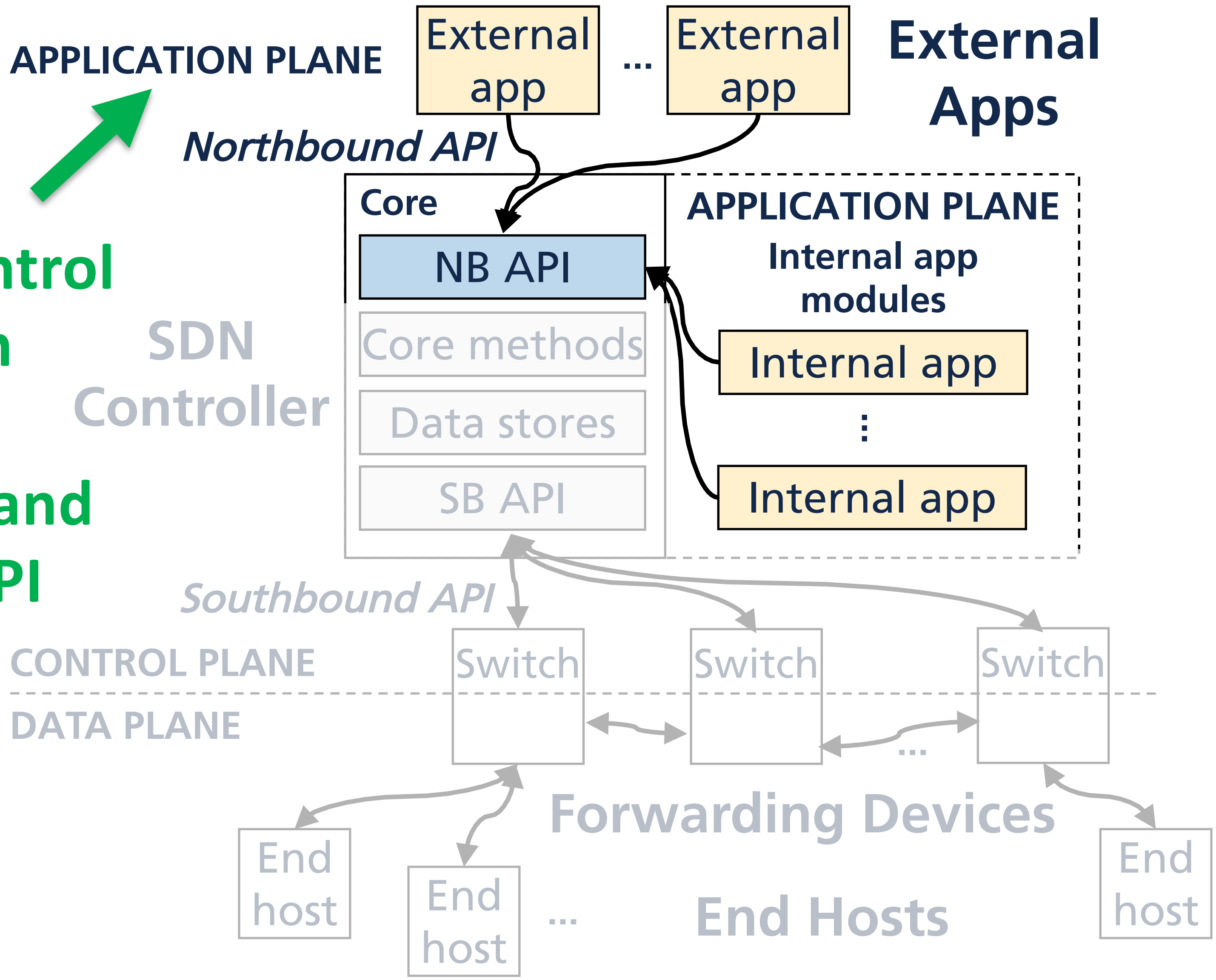


SDN Architecture



SDN Architecture

Extensible control plane with network applications and standard API



SDN Controllers and Frameworks

- Open-source
  - Open Network Operating System (ONOS)
  - OpenDaylight (ODL)
  - Floodlight
- Proprietary
  - HPE Virtual Application Networks (VAN) SDN Controller
  - Many other vendor-specific SDNs (e.g., Google B4)

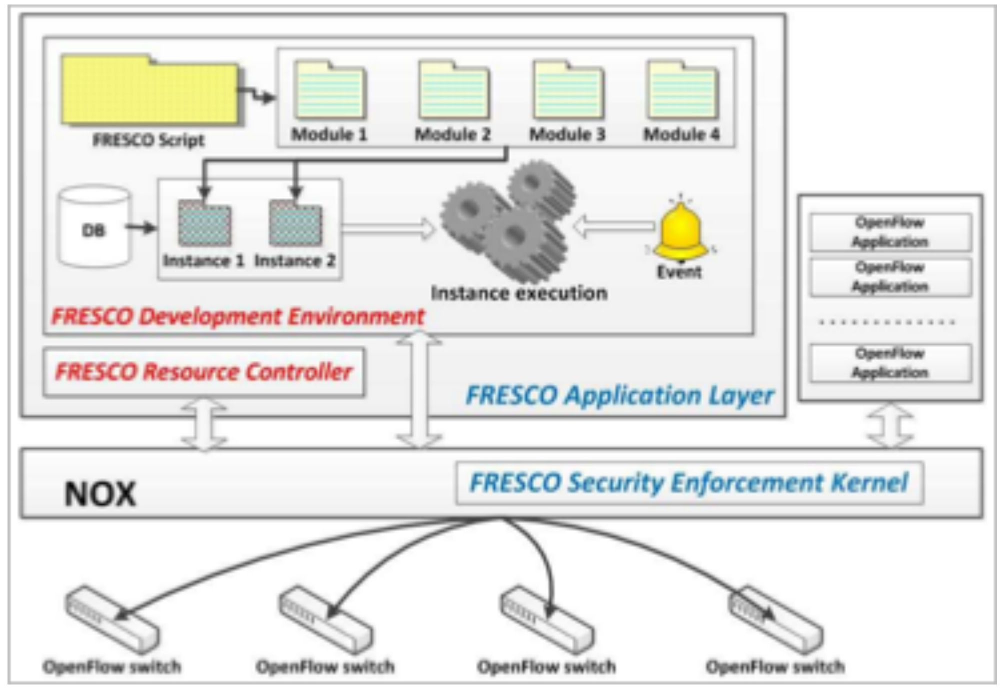
SDN and Security

- SDN as a security service
  - How can SDN help ensure security properties of the hosts in the data plane?
  - Examples: FRESCO, PSI, DFI
- Security of SDN (attacks and defenses)
  - How does the SDN architecture affect the security properties of the overall network?
  - Examples: Cross-app poisoning, ARP spoofing, LLDP spoofing

SDN as a Security Service

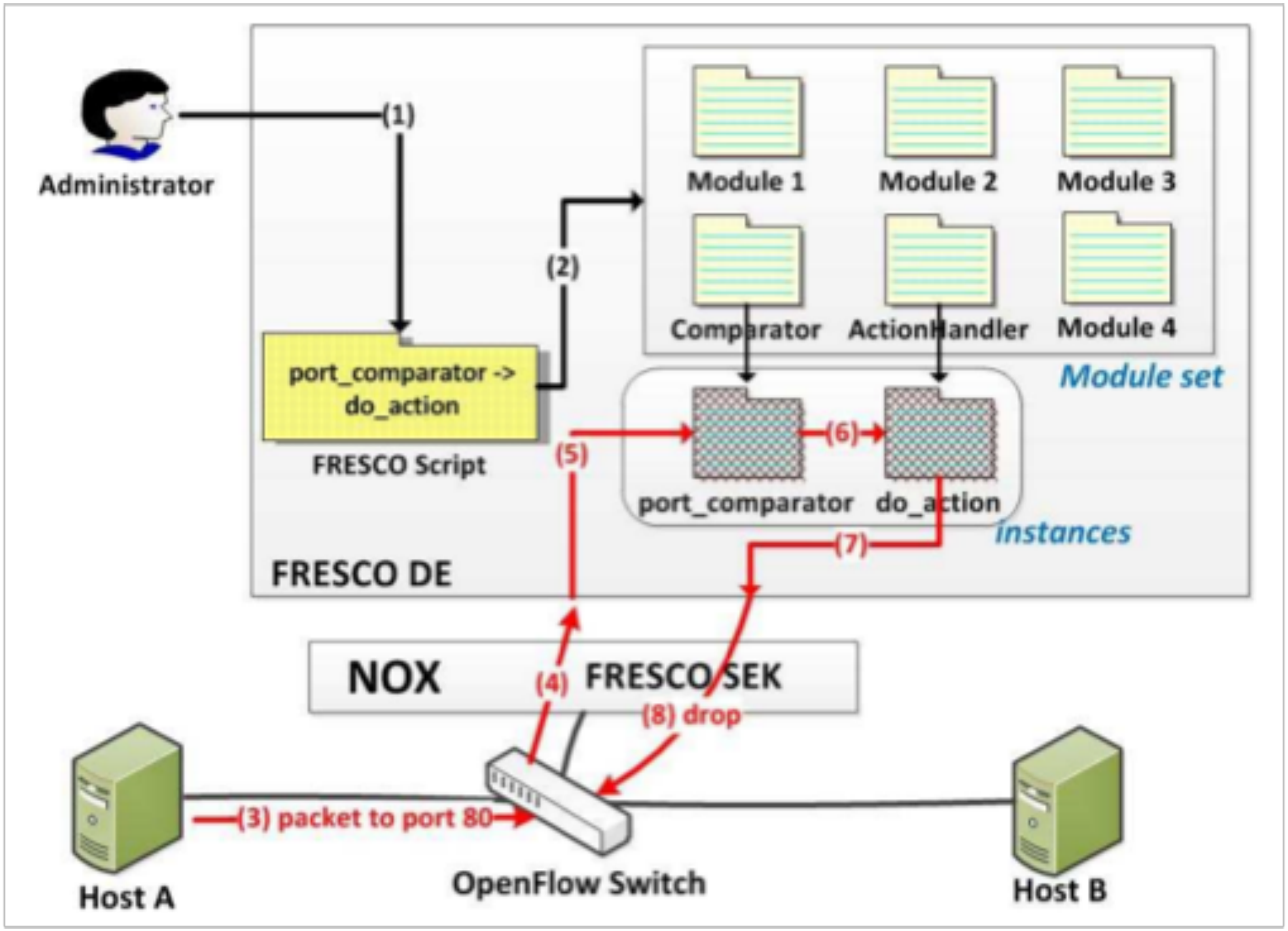
"FRESCO: Modular Composable Security Services for Software-Defined Networks" [NDSS '13]

- Problem: Network security applications are difficult to compose correctly
- Solution: Use SDN to develop coordinated security services
- Scripts of modules, inputs, outputs, actions
- Redirect, mirror, quarantine actions



SDN as a Security Service

"FRESCO: Modular Composable Security Services for Software-Defined Networks" [NDSS '13]



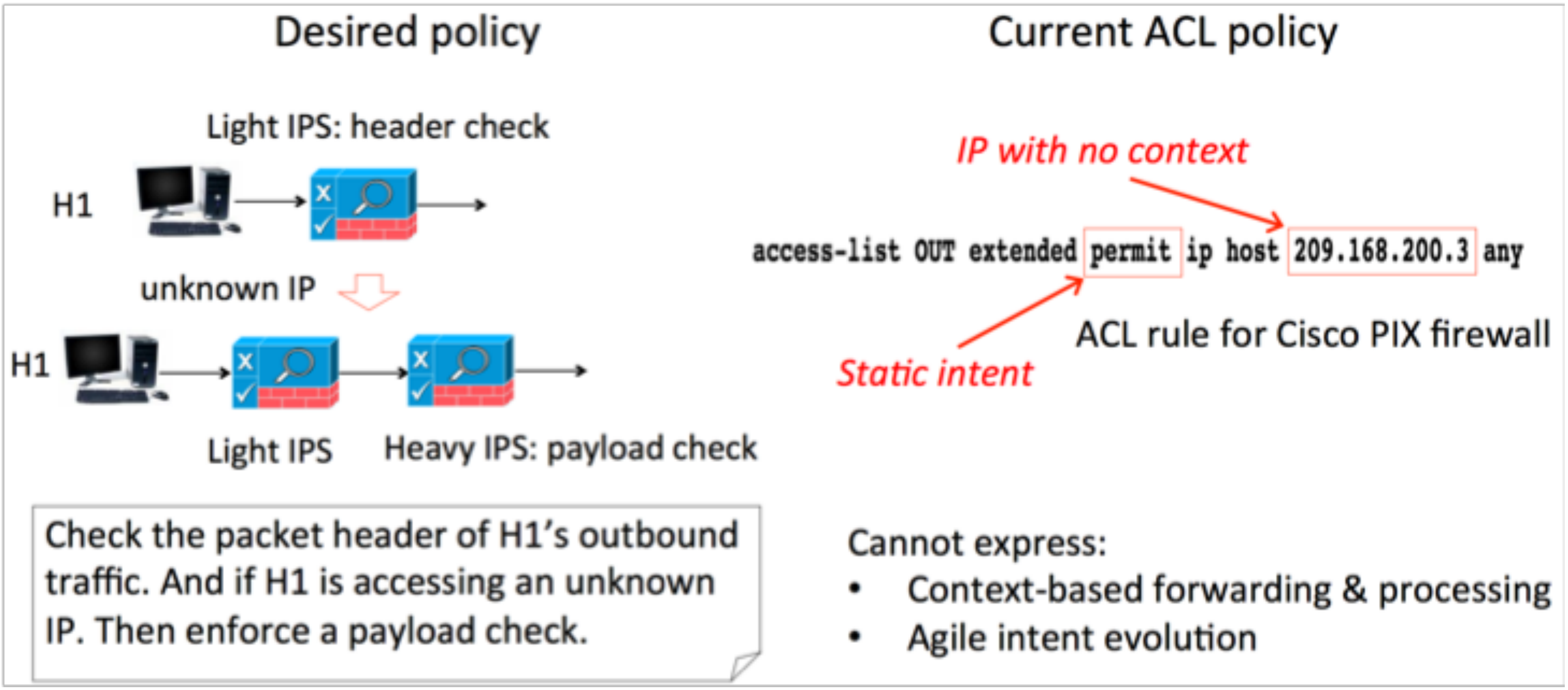
SDN as a Security Service

"PSI: Precise Security Instrumentation for Enterprise Networks" [NDSS '17]

- Problem: Network security suffers from lack of isolation, context awareness, and agility to change rapidly
- Solution: Use SDN to implement network traffic inspection policy with agile reconfiguration
- Enforce policies so that traffic must traverse certain devices (e.g., middleboxes)
- Enforcement policies can be dynamic, so reconfigure network forwarding as needed (e.g., suspiciousness of a potentially compromised host)

SDN as a Security Service

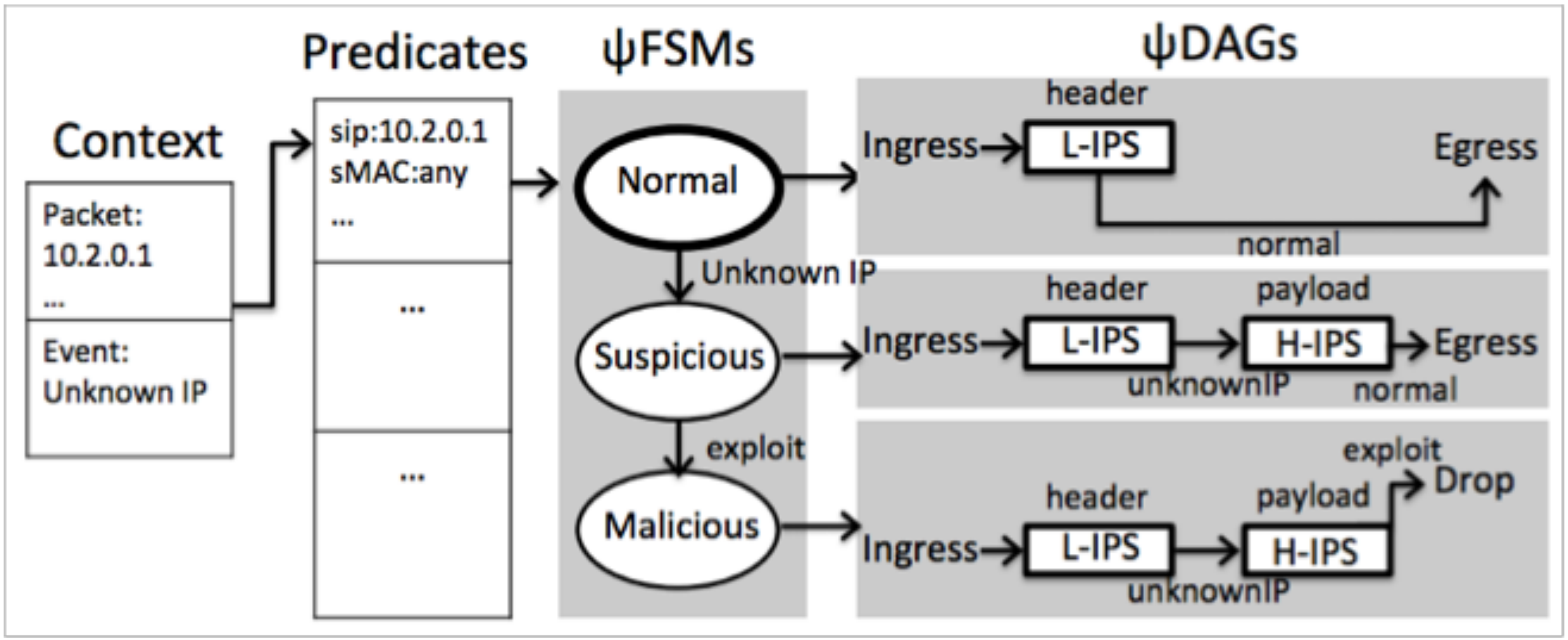
"PSI: Precise Security Instrumentation for Enterprise Networks" [NDSS '17]



Source: Slides by authors from paper presentation

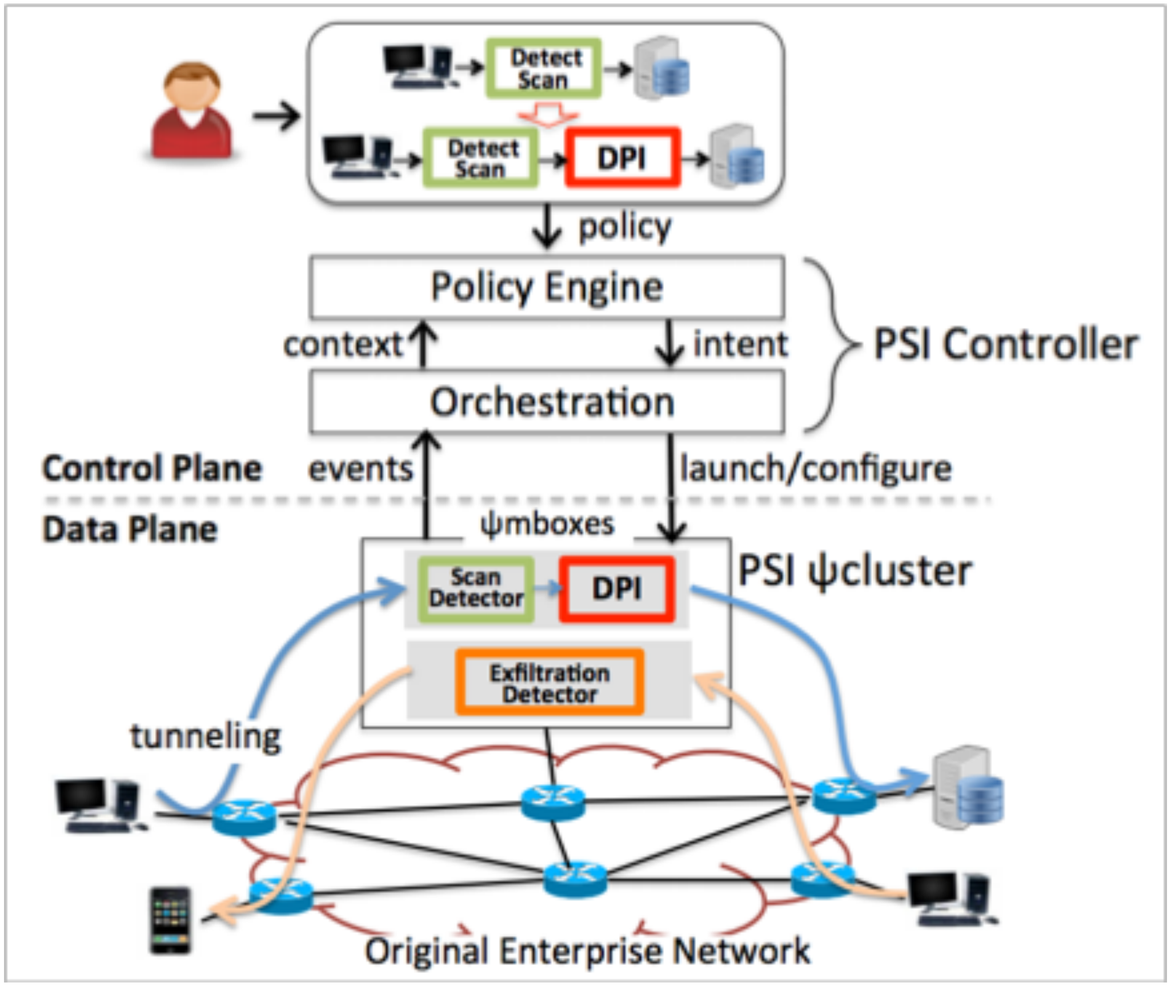
SDN as a Security Service

"PSI: Precise Security Instrumentation for Enterprise Networks" [NDSS '17]



SDN as a Security Service

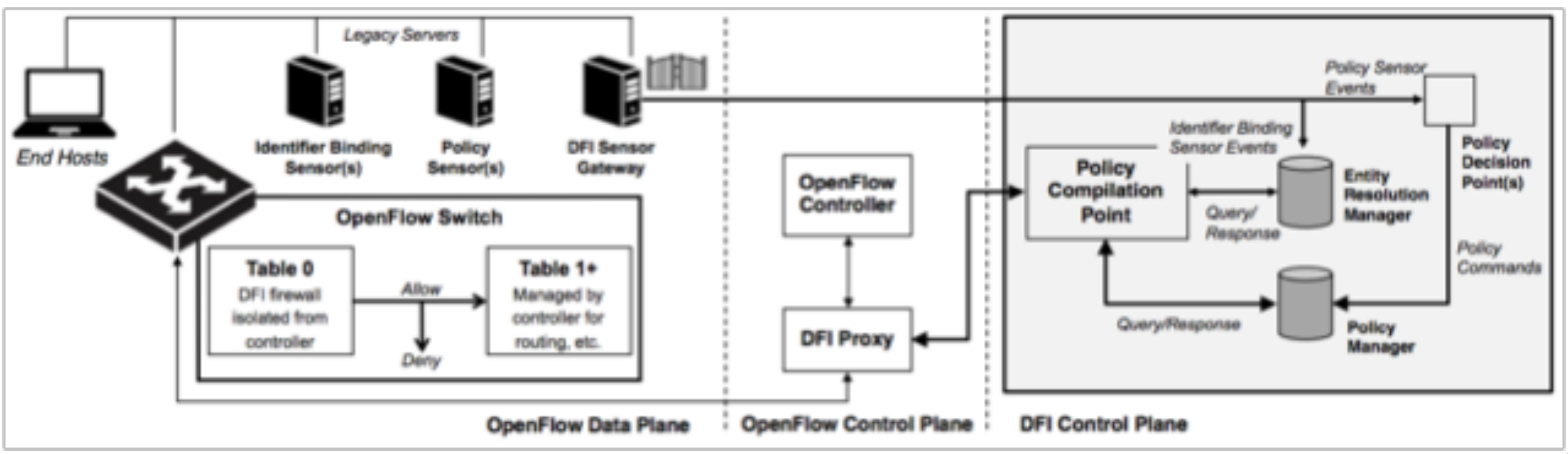
"PSI: Precise Security Instrumentation for Enterprise Networks" [NDSS '17]



SDN as a Security Service

"Controller-Oblivious Dynamic Access Control in Software-Defined Networks" [DSN '19]

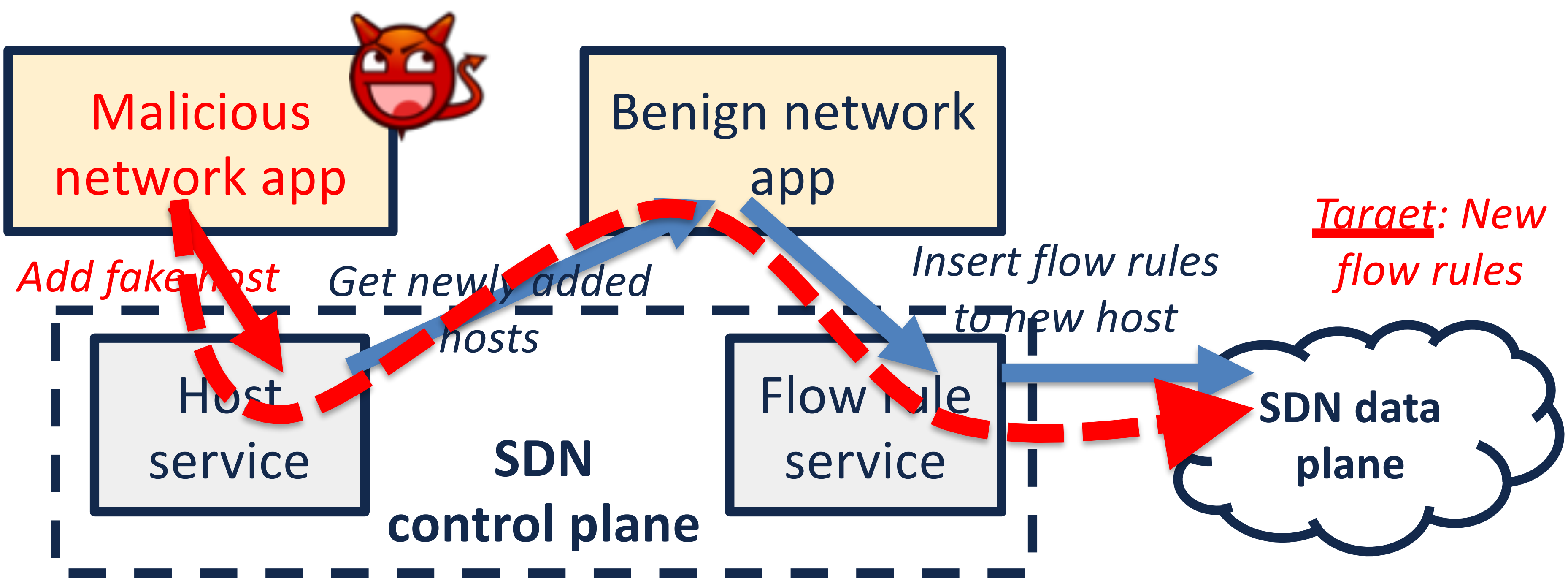
- Problem: Fine-grained, least-privileges access control is necessary to prevent lateral movement, but networks don't respond to non-traditional events (e.g., user logging off)
- Solution: Use SDN and sensors around network to implement access control policies (dynamic flows)



SDN Attacks: Cross-App Poisoning

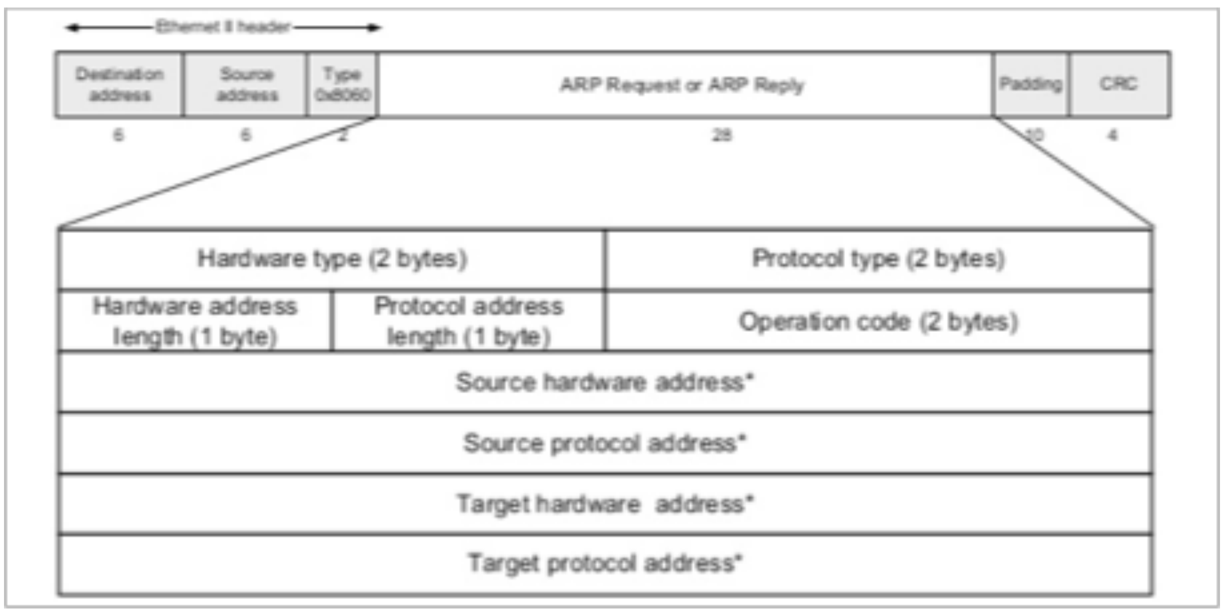
"Cross-App Poisoning in Software-Defined Networking" [CCS '18]

- Problem: Malicious network applications can influence other applications through API calls
- Solution: Cross-app poisoning "gadgets" (compare with ROP gadgets); prevent bad information flow



Recall: Address Resolution Protocol

- Address Resolution Protocol (ARP) lets hosts map IP addresses to MAC addresses
- Host who needs MAC address *M* corresponding to IP address *N* broadcasts an ARP packet to LAN asking, "who has IP address *N*?"



Recall: ARP Security

- Any host on the LAN can send ARP requests and replies: *any host can claim to be another host on the local network!*
  - This is called *ARP spoofing*
- This allows any host X to force IP traffic between any two other hosts A and B to flow through X *(MitM!)*
  - Claim $N_{\mathbf{A}}$ is at attacker's MAC address $M_{\mathbf{X}}$
  - Claim $N_{\mathbf{B}}$ is at attacker's MAC address $M_{\mathbf{X}}$
  - Re-send traffic addressed to $N_{\mathbf{A}}$ to $M_{\mathbf{A}}$, and vice versa

SDN Attacks: ARP Spoofing

"Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures" [NDSS '15]

- In SDN: the host location hijacking attack
- Why this attack works:
  - SDN controllers maintain a database of host objects, derived from data plane packet information
  - As consequence of flattening network stack, controllers use proxy ARP (i.e., single ARP table) → *attacker and victim can be in separate broadcast domains of network*
  - Any data plane host can masquerade as another data plane host
  - No authentication of ARP packets

SDN Defenses: ARP Spoofing

"Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures" [NDSS '15]

- Static ARP
  - Manually configure ARP table based on known MAC addresses (but hard to change)
- TopoGuard [NDSS '15]
  - Uses update semantics (e.g., has host moved?) to mitigate (but not prevent) ARP spoofing
- SecureBinder [USENIX Security '17]
  - Use IEEE 802.1x as a root of trust

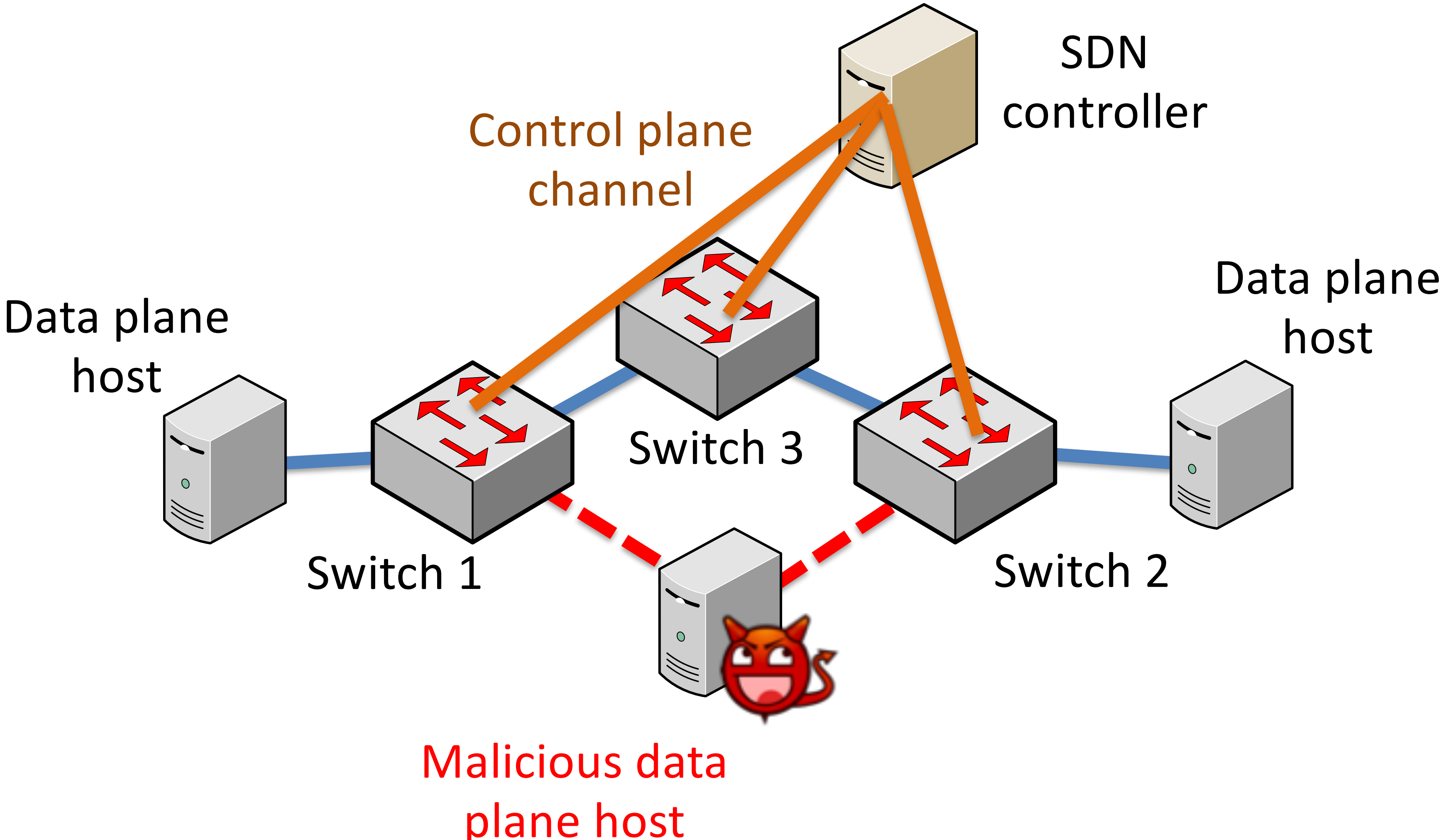
SDN Attacks: LLDP Spoofing

"Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures" [NDSS '15]

- Link-layer discovery protocol (LLDP) used to learn links (and thus topology) via adjacent switches
- Must use data plane
- Attacker goals:
  1. Create a MitM attack to route traffic through an attacker-controlled host
  2. Create a "black hole" attack to force data plane traffic to be dropped through non-existent links

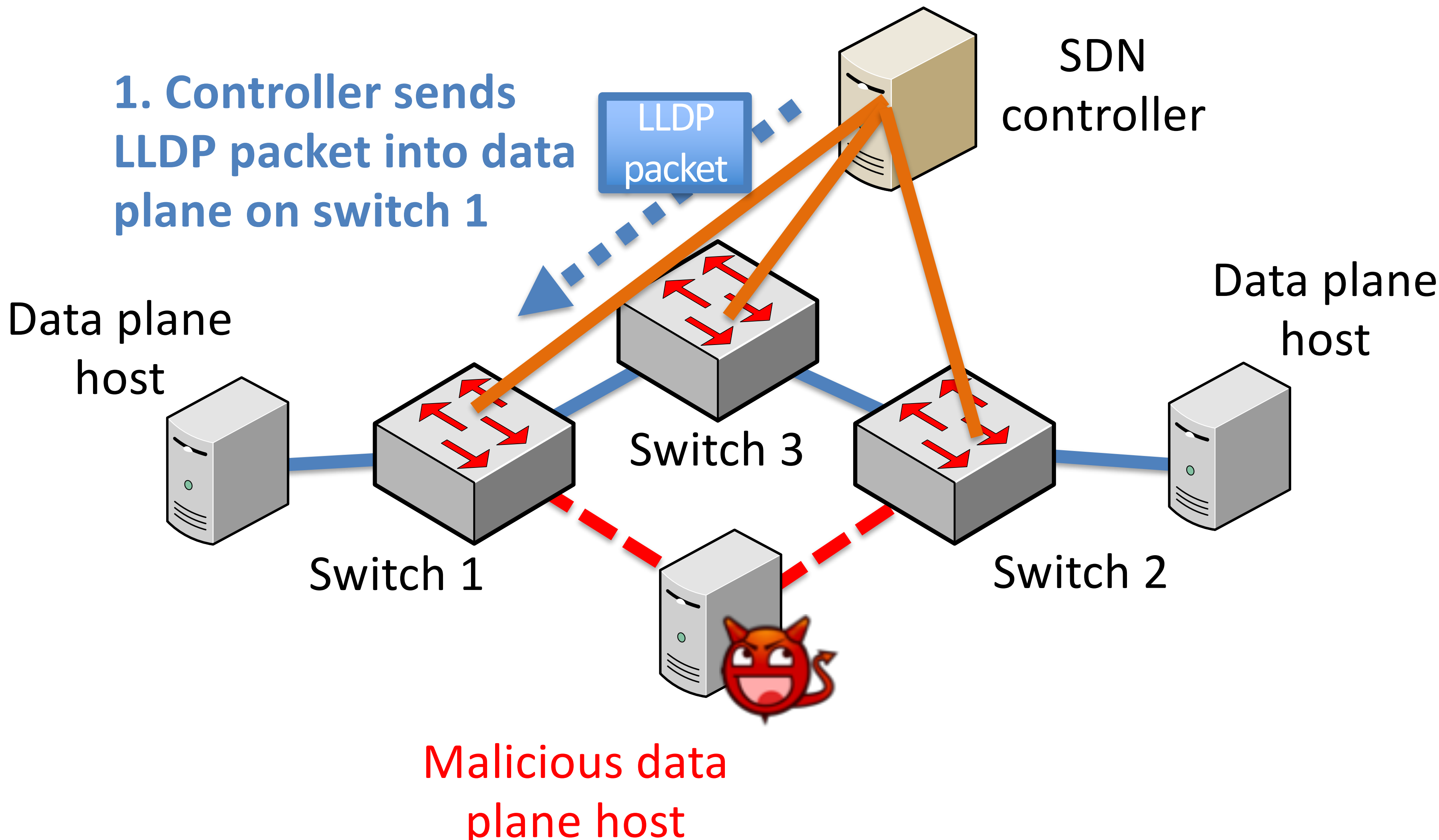
SDN Attacks: LLDP Spoofing

"Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures" [NDSS '15]



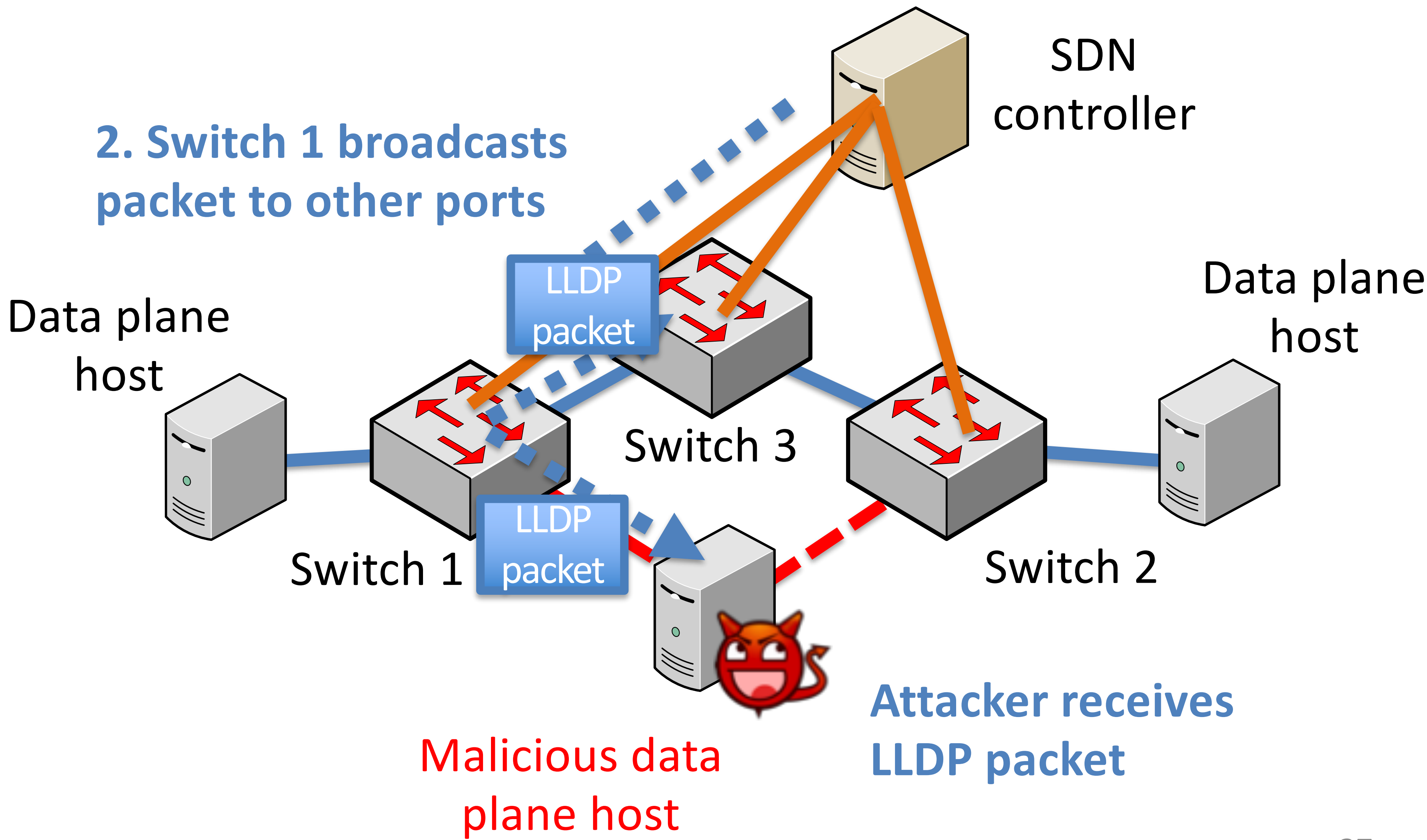
SDN Attacks: LLDP Spoofing

"Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures" [NDSS '15]



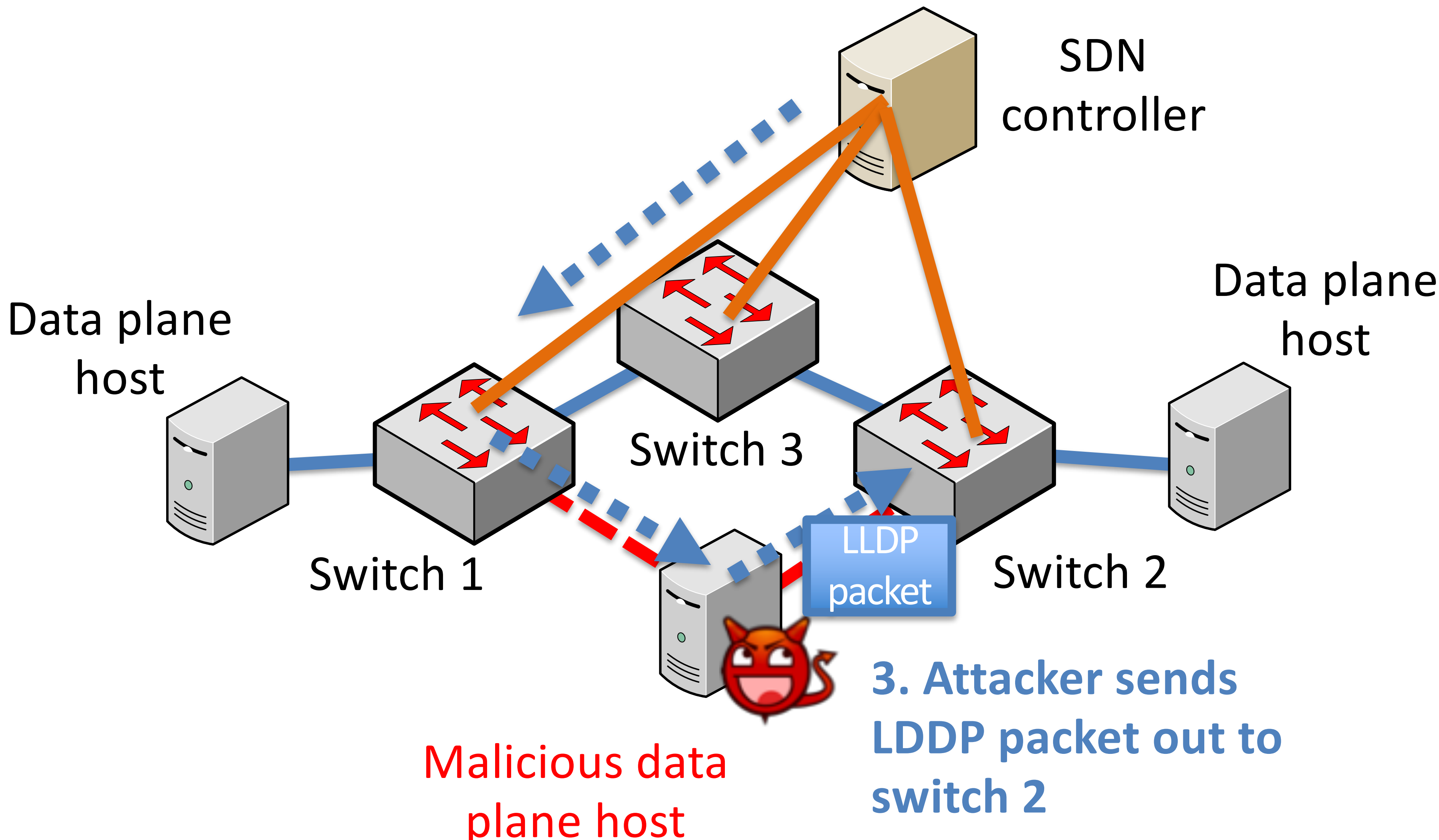
SDN Attacks: LLDP Spoofing

"Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures" [NDSS '15]



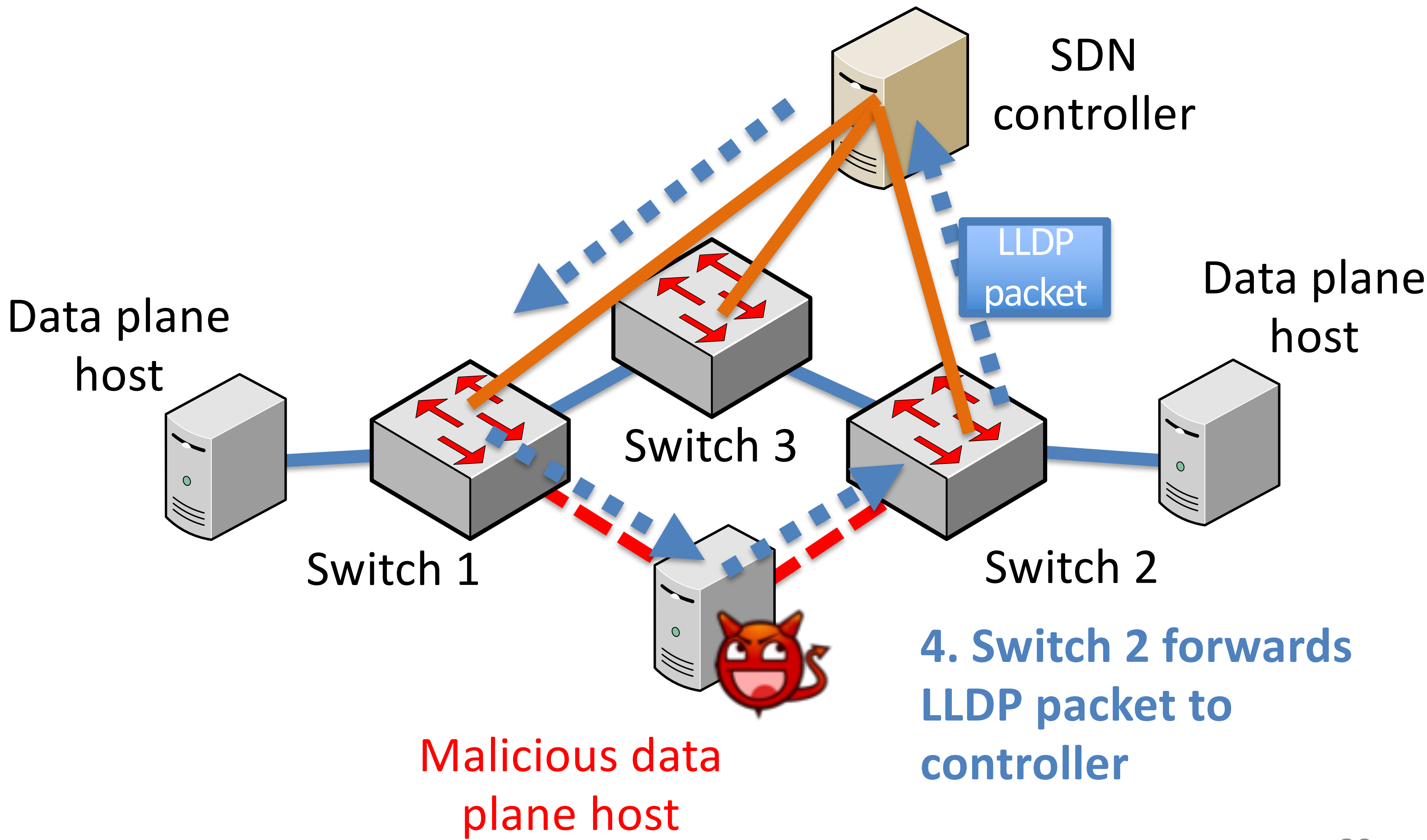
SDN Attacks: LLDP Spoofing

"Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures" [NDSS '15]



SDN Attacks: LLDP Spoofing

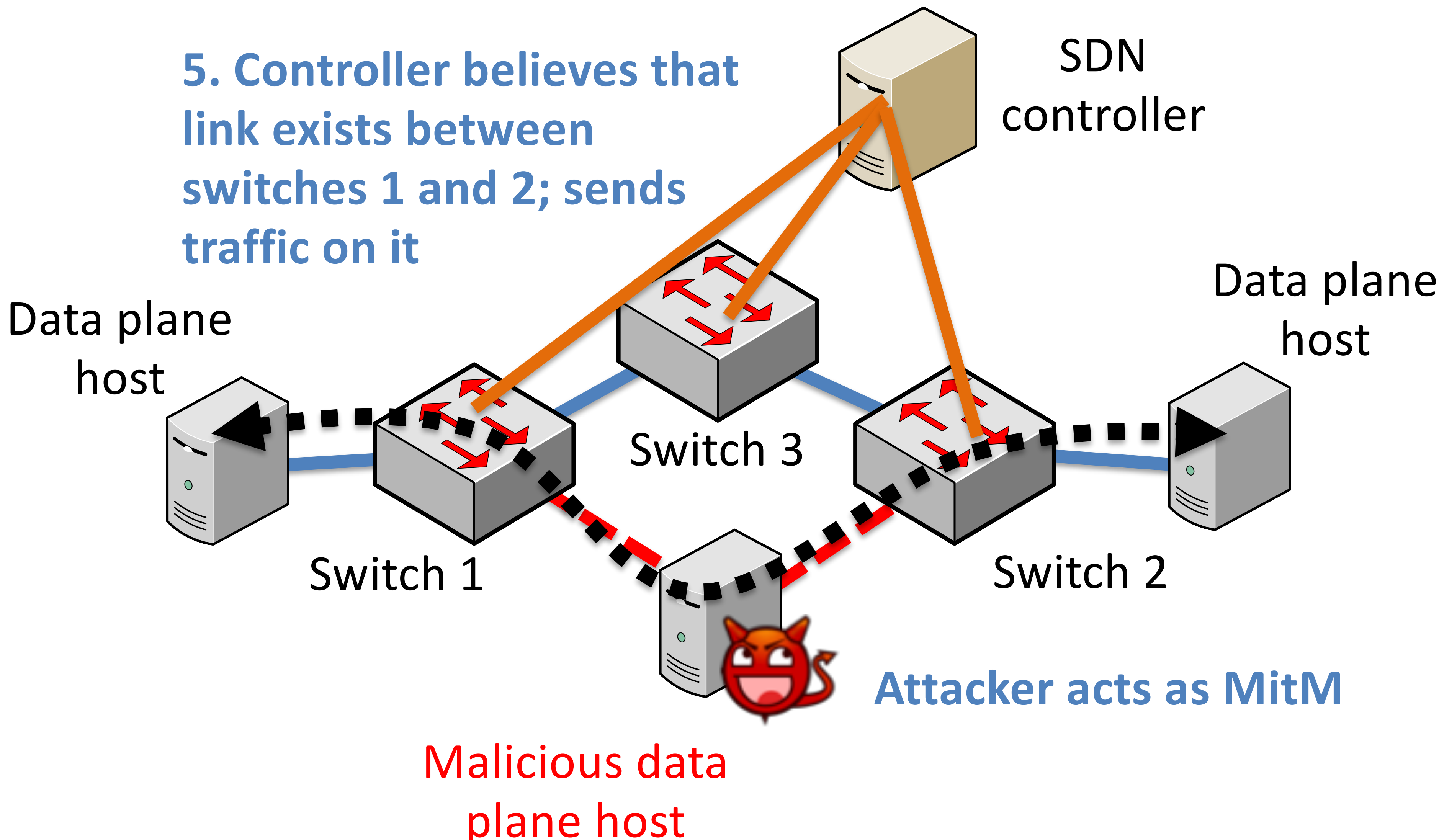
"Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures" [NDSS '15]



4. Switch 2 forwards LLDP packet to controller

SDN Attacks: LLDP Spoofing

"Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures" [NDSS '15]



SDN Attacks: LLDP Spoofing

"Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures" [NDSS '15]

- In SDN: the link fabrication attack
- Why this attack works:
  - SDN controllers maintain a topology graph (links, switches, and hosts) of the current network "state" so network applications can query it
  - LLDP packets must be sent through the data plane
  - Integrity of LLDP packets are (generally) not checked
    - Spoof, replay, or tunnel LLDP packets

SDN Defenses: LLDP Spoofing

"Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures" [NDSS '15]

- Spoofing: Signed LLDP messages
  - Uses Hash-based Message Authentication Code (HMAC) signed by the SDN controller
  - Provides integrity and authenticity guarantees that the LLDP packet came from the SDN controller
- Tunneling: Verify if port is HOST or SWITCH
  - Host and switch ports are mutually exclusive
  - Hosts will generate their own traffic (e.g., ARP, DNS), so these are not SWITCH ports

SDN Security Research

- Active area of research!
- SDN as network operating system
  - Complexity of SDN framework creates new security challenges
  - What security concerns from OS design can we incorporate into SDN controller design?
- SDN application trust
  - SDN apps now have "app stores"
  - What assumptions should we make about trustworthiness of SDN apps?

The End

- Questions?
- Announcements:
  - Networking Project, Checkpoint 1 due 6pm
  - Adam returning on Friday (lecture on TLS)