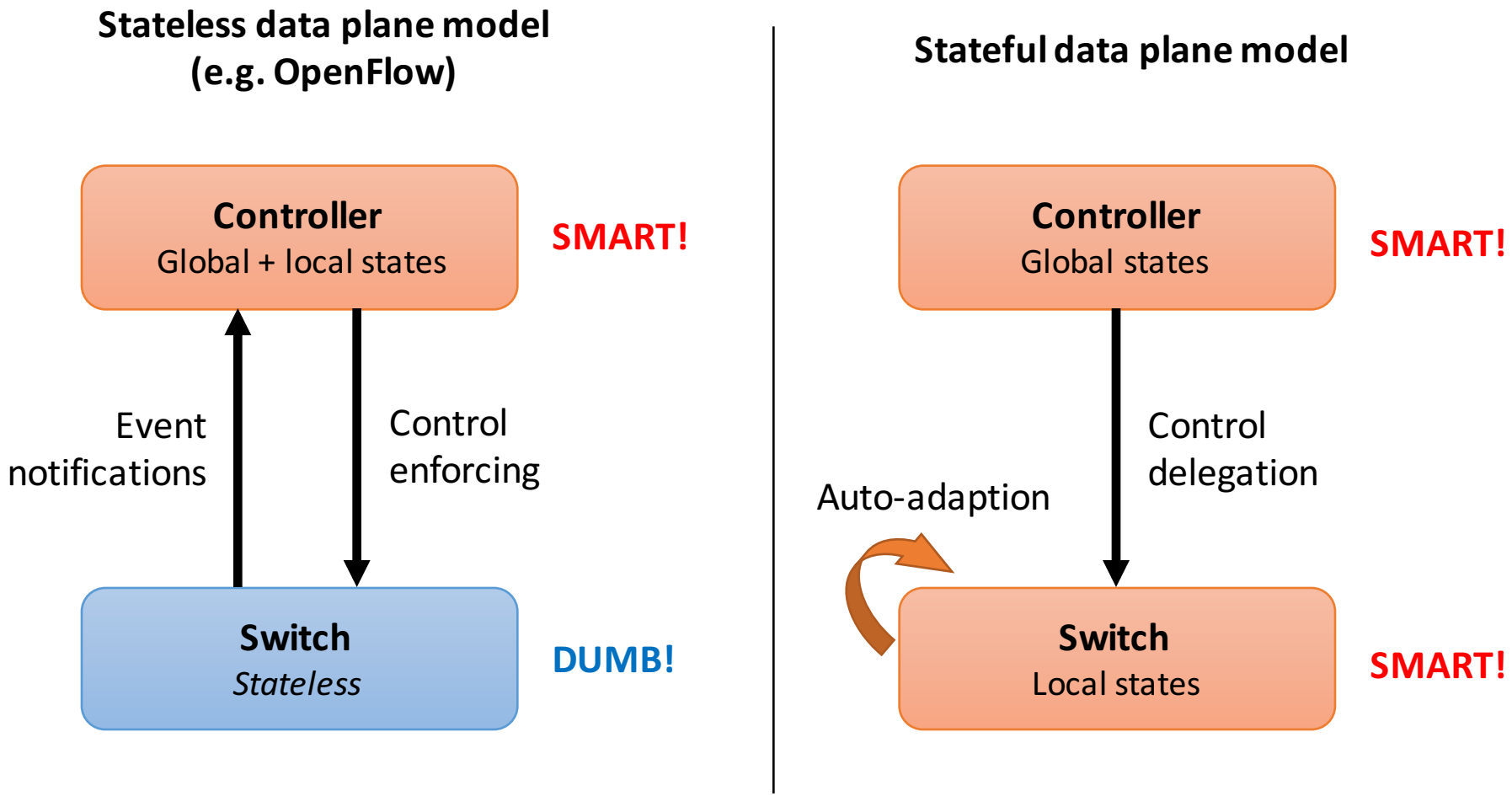
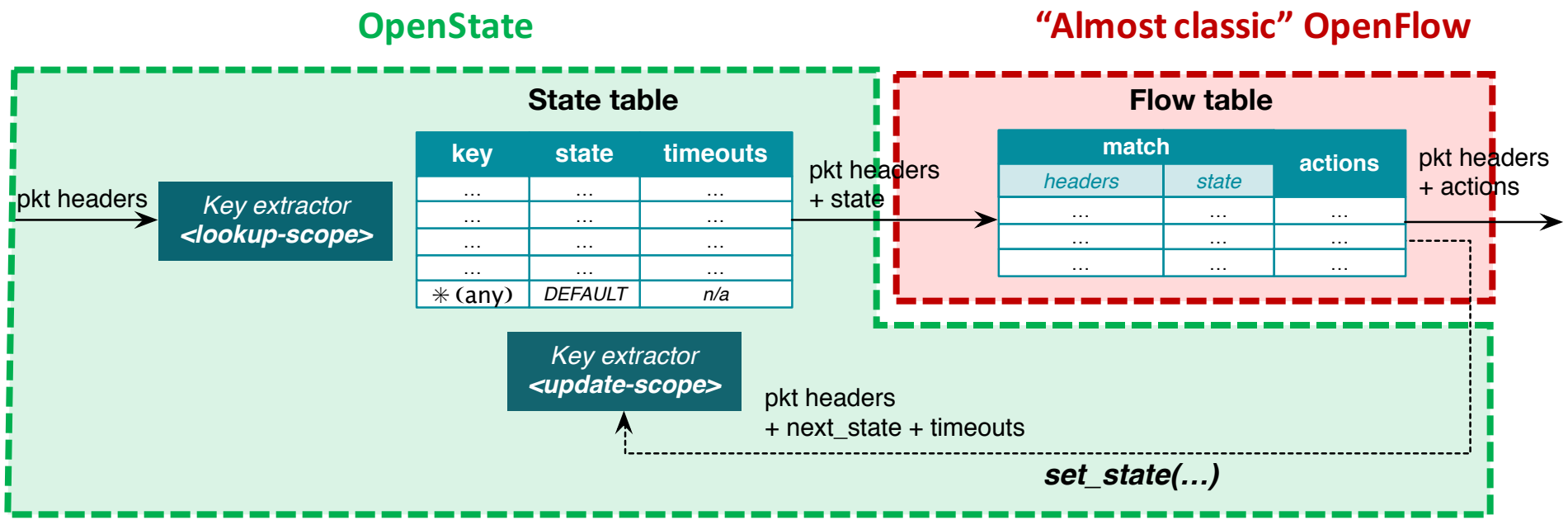
OpenState demo

SOSR ‘15, Santa Clara (CA), USA

Stateless vs. Stateful in SDN

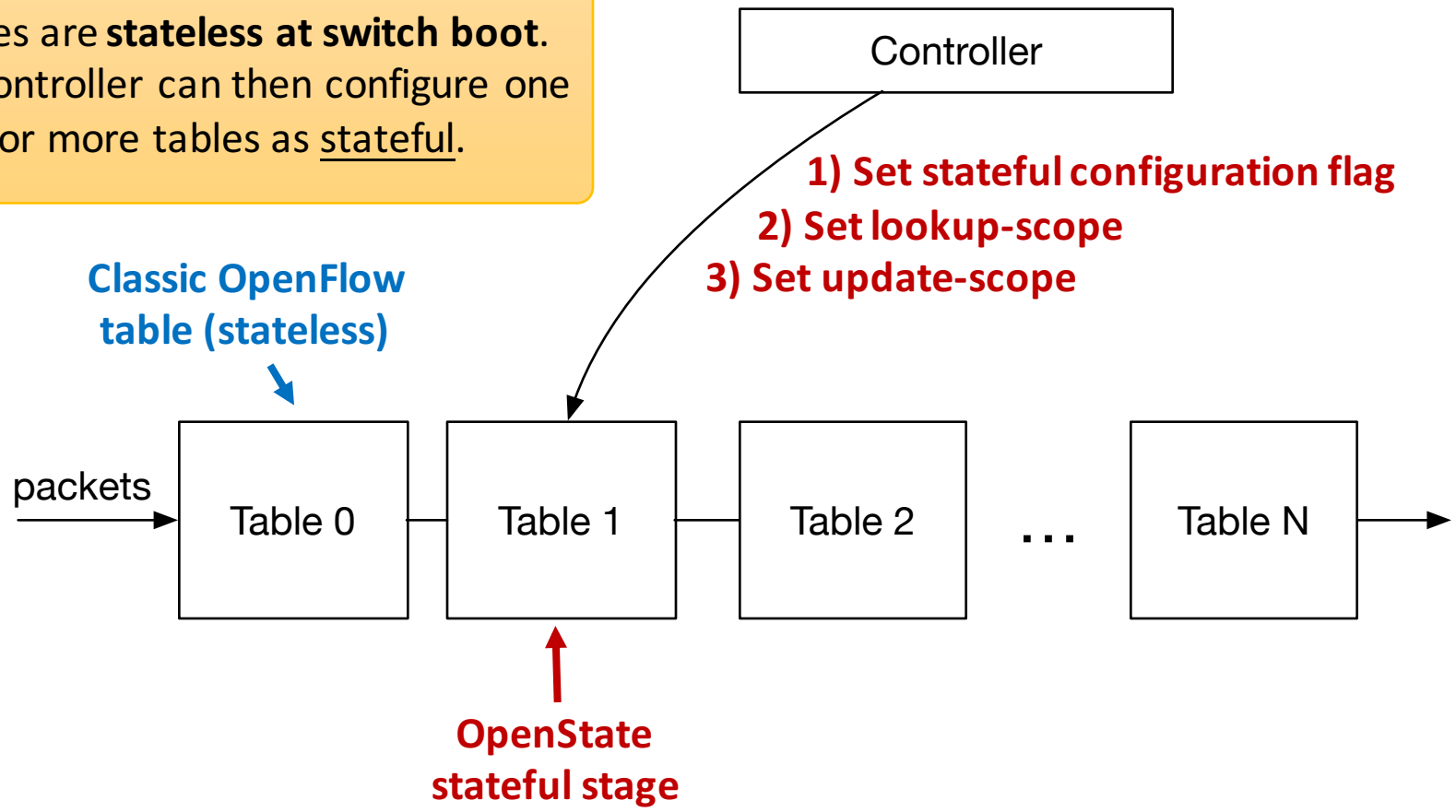


OpenState architecture



OpenState pipeline

Tables are **stateless at switch boot**. The controller can then configure one or more tables as stateful.

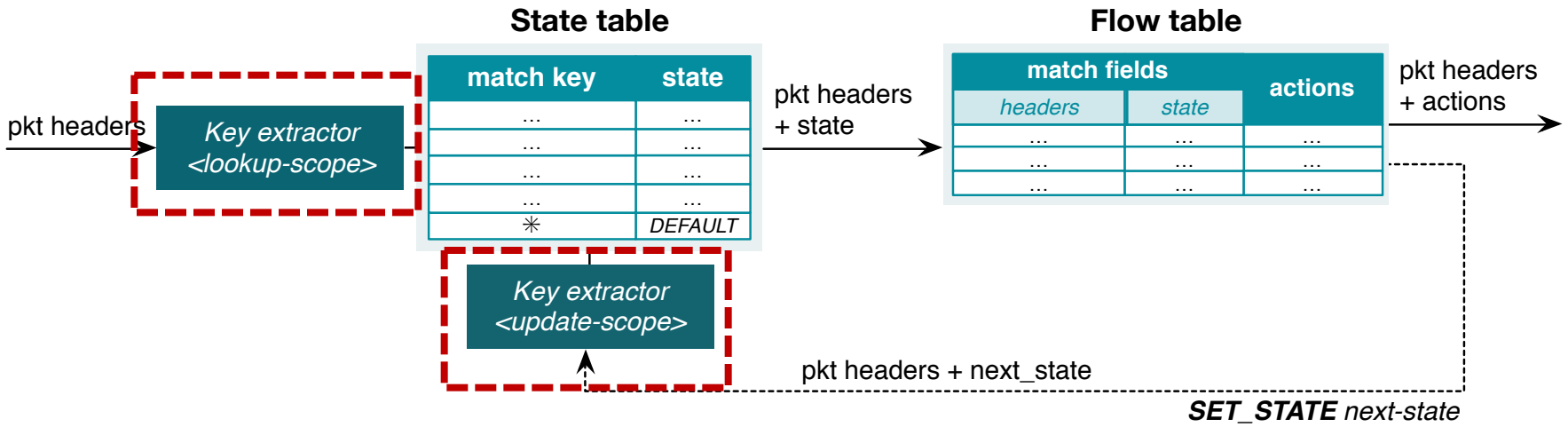


Flow scope / key extractors

- Used to match the state table
  - Lookup and update phase
- Input: packet headers
- Output: variable length bit sequence
  - Concatenated header fields
- Scope = ordered list of header fields
  - E.g. {ip_src} → 32 bit key
  - E.g. {eth_src, eth_dst} → 96 bit key

Lookup/update scope

Same packet headers can lookup/update different state entries



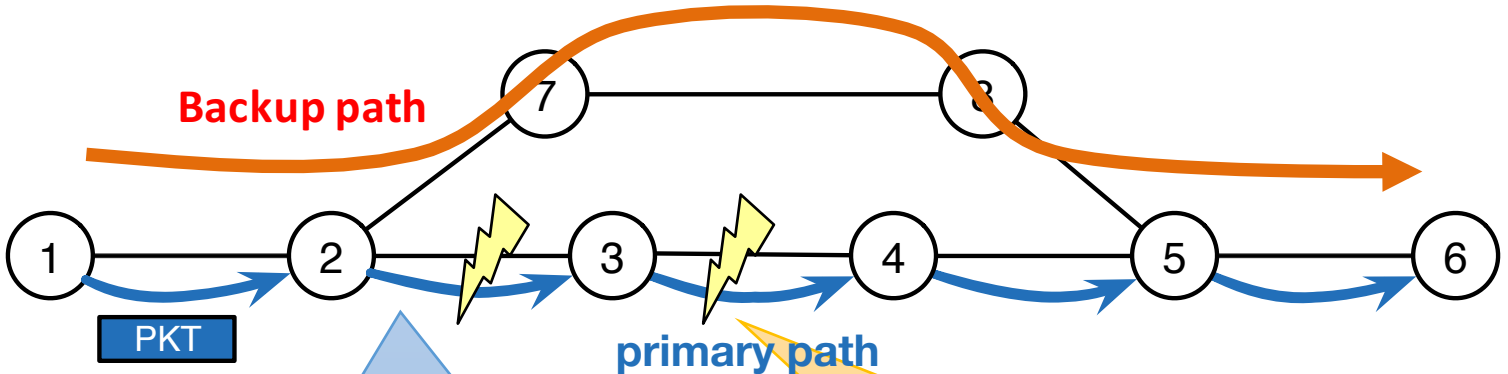
Example applications

Implementation details & demo

Distant failure recovery

…or, how to use tags to perform simple switch-to-switch signaling

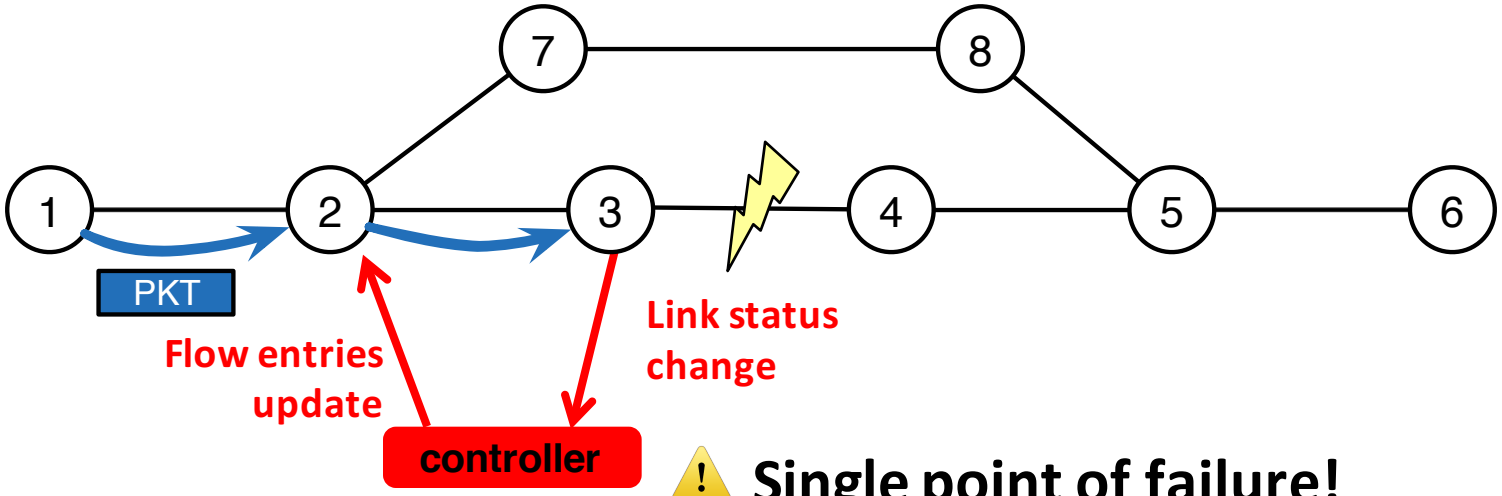
Failure recovery in OpenFlow



“Fast-failover”: Local reroute based on port status (OpenFlow 1.1+)

Weak! What if a local reroute in not available?

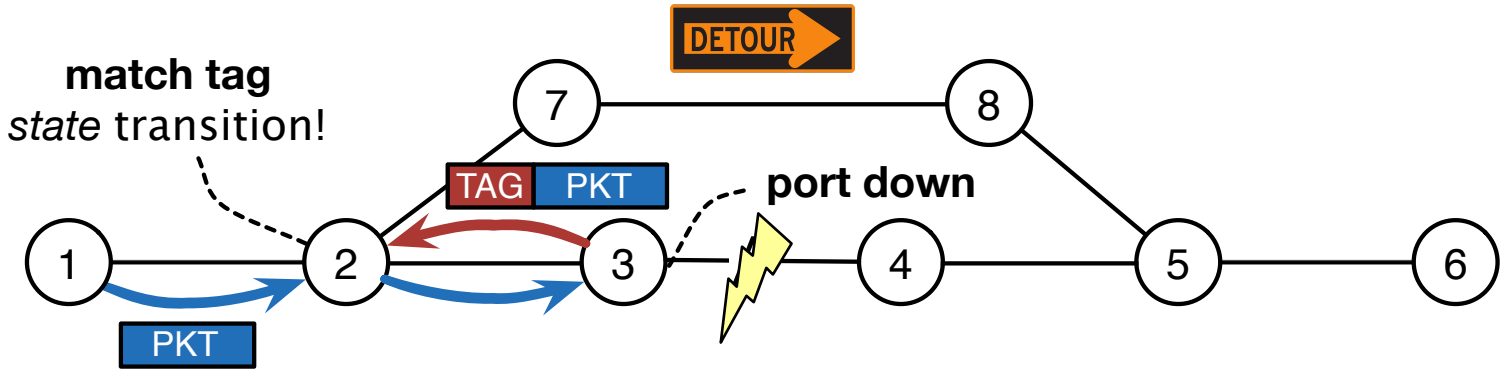
Failure recovery in OpenFlow.. Weak!



Can rely on controller intervention, but:

- Long recovery latency
  - detection + signaling + flow update
- Failure of control channel (controller unreachable)
- Signaling congestion (controller unresponsive)

Failure recovery with OpenState



- Signaling using same data packets
  - Tag = unreachable node
  - Packets “bounced back” until a convenient redirect point
- Flow-states used to update the routing

➔ **No extra signaling**

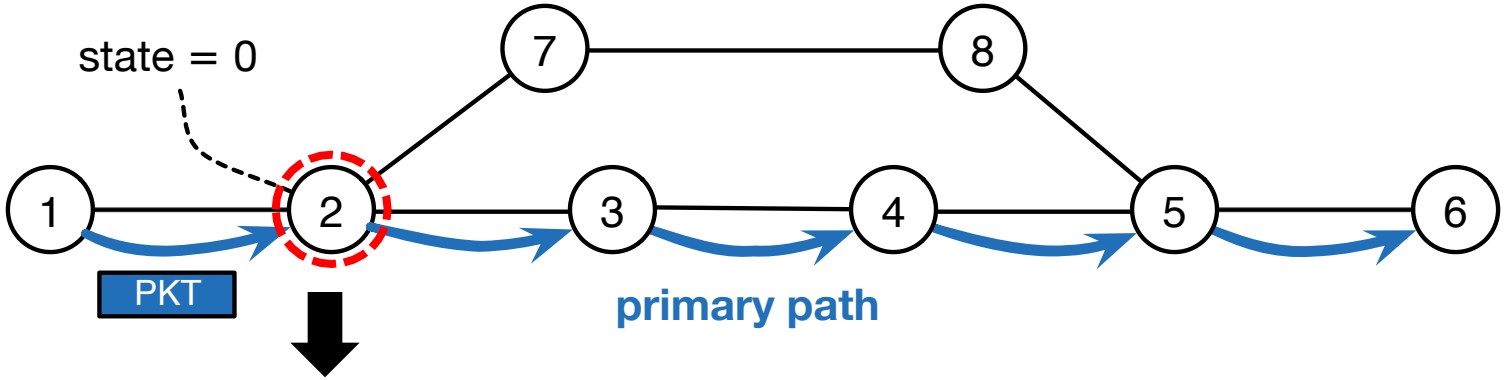
➔ **No packet loss after failure detection**

➔ **Controller not involved**

Failure recovery

Example

Normal conditions (no failures)



lookup-scope=[eth_src, eth_dst]
update-scope=[eth_src, eth_dst]

L2 flows

State table

Key	State
…	…
…	…
* (any)	0

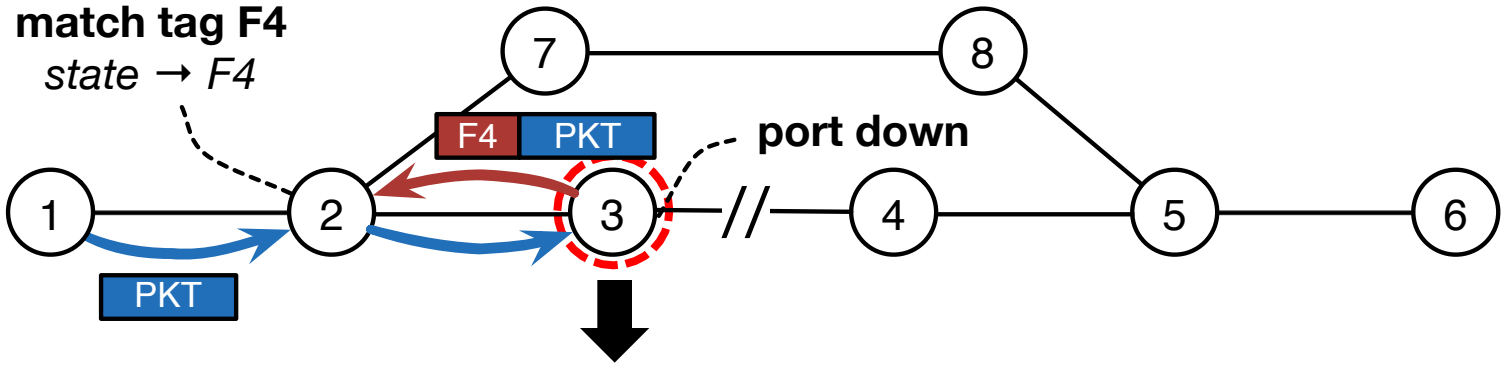
Flow table

Match	Instructions
src=1, dst=6, state=0	fwd(3)
…	…
…	…

Failure recovery

Example

Packets "bounced back" in case of failure



Match	Instructions
src=1, dst=6	group(A)
…	…
…	…

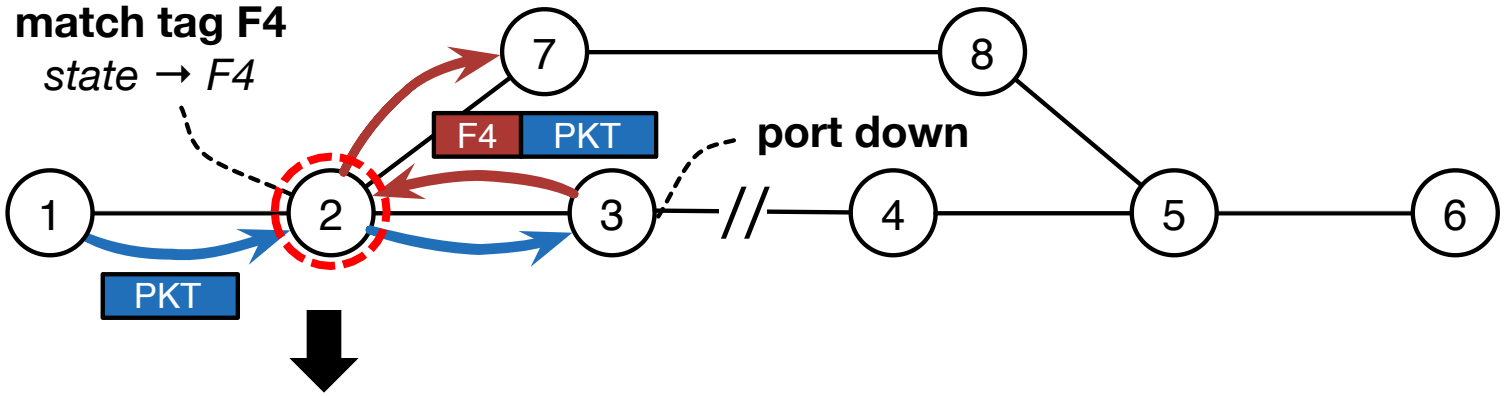
Group table

ID	Type	Action buckets
A	FAST-FAILOVER	<output(2)>, <push_tag(F4), output(1)>,
…	…	…

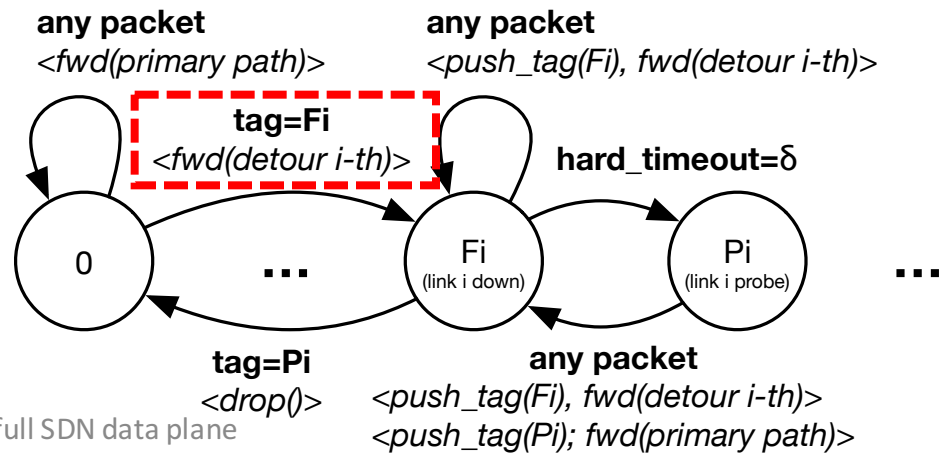
Failure recovery

Example

State transition at a predetermined reroute node

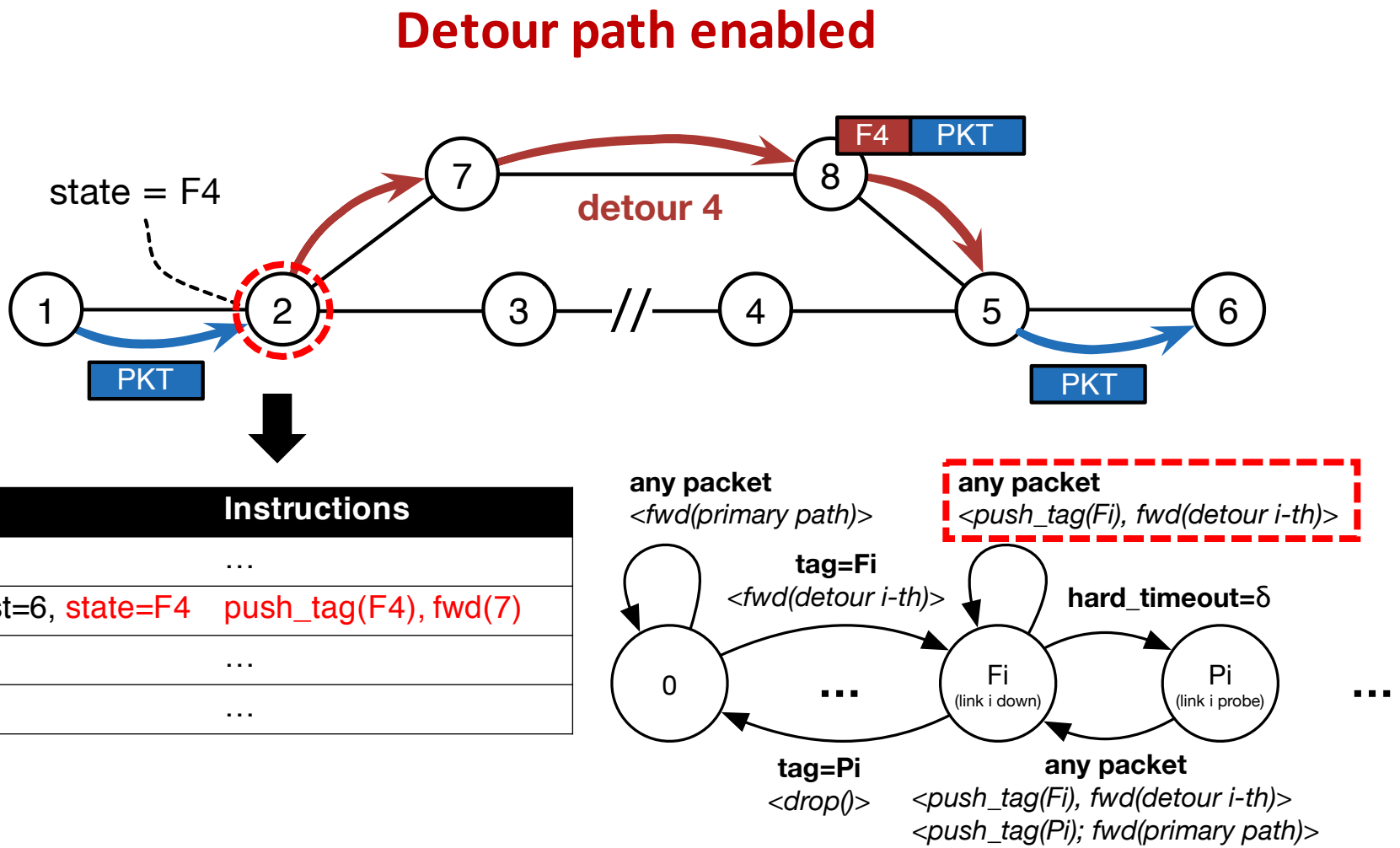


Match	Instructions
…	…
…	…
tag=F4	set_state(F4, hard_to=10s, hard_rollback=P4) fwd(7)
…	…



Failure recovery

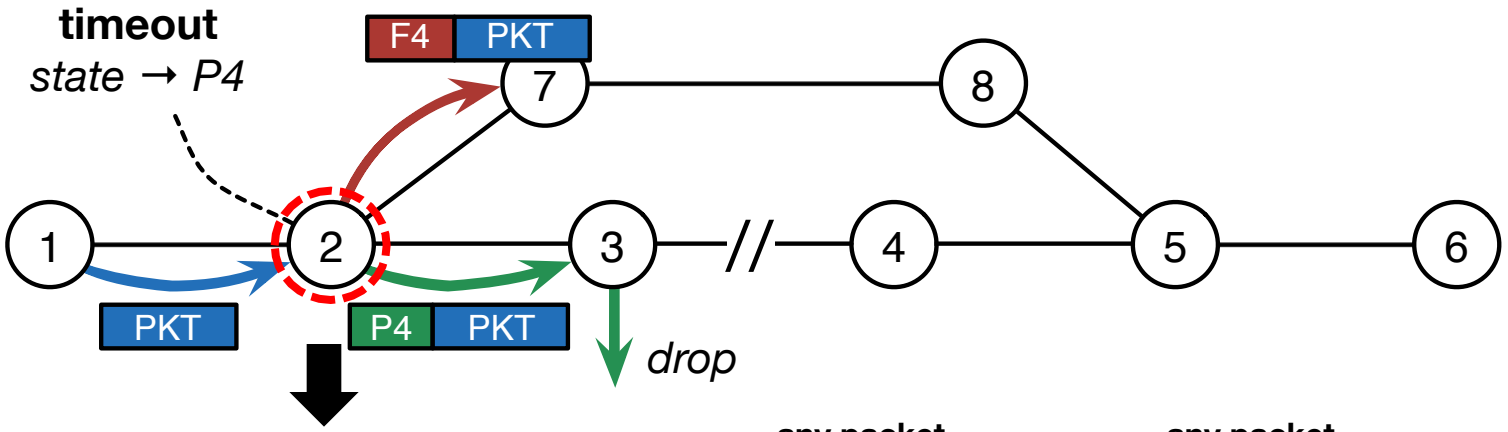
Example



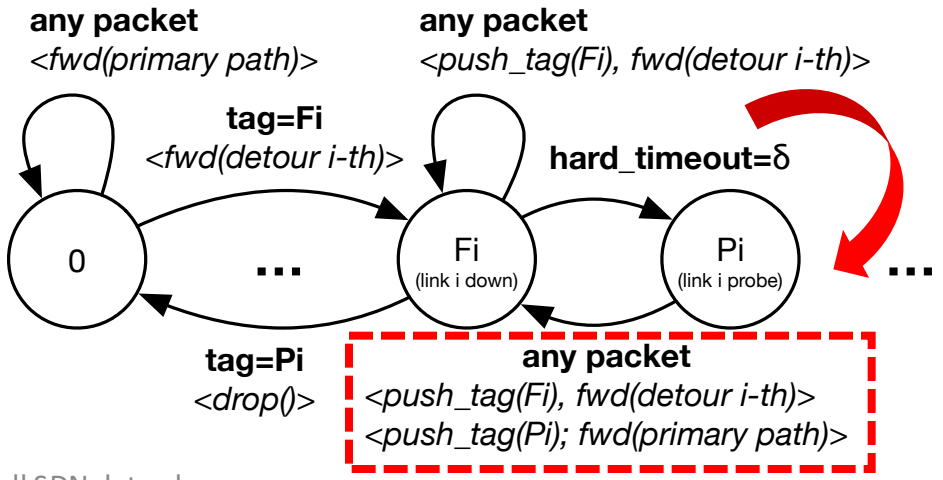
Failure recovery

Example

State hard timeout to generate probe packets



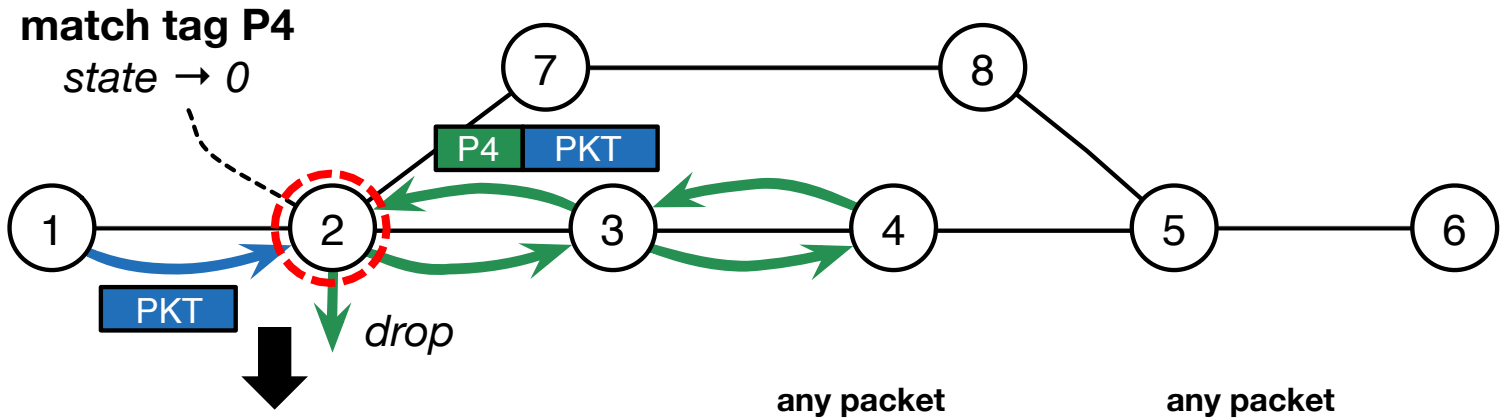
Match	
…	…
…	…
…	…
src=1, dst=6, state=P4	set_state(F4, hard_to=10s, hard_rollback=P4), <push_tag(F4), fwd(7)> <push_tag(P4), fwd(3)>



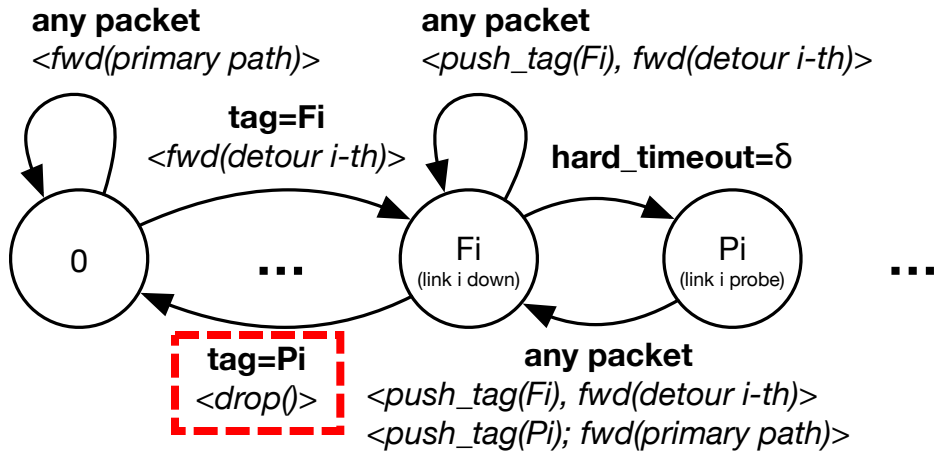
Failure recovery

Example

Primary path re-established



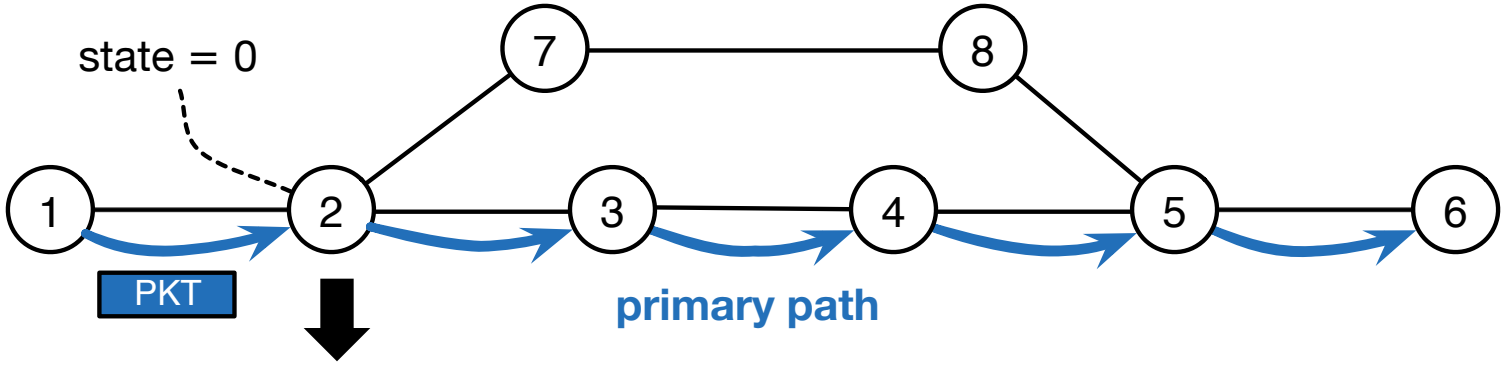
Match	
…	…
…	…
…	…
…	…
tag=P4	set_state(0), drop()



Failure recovery

Example

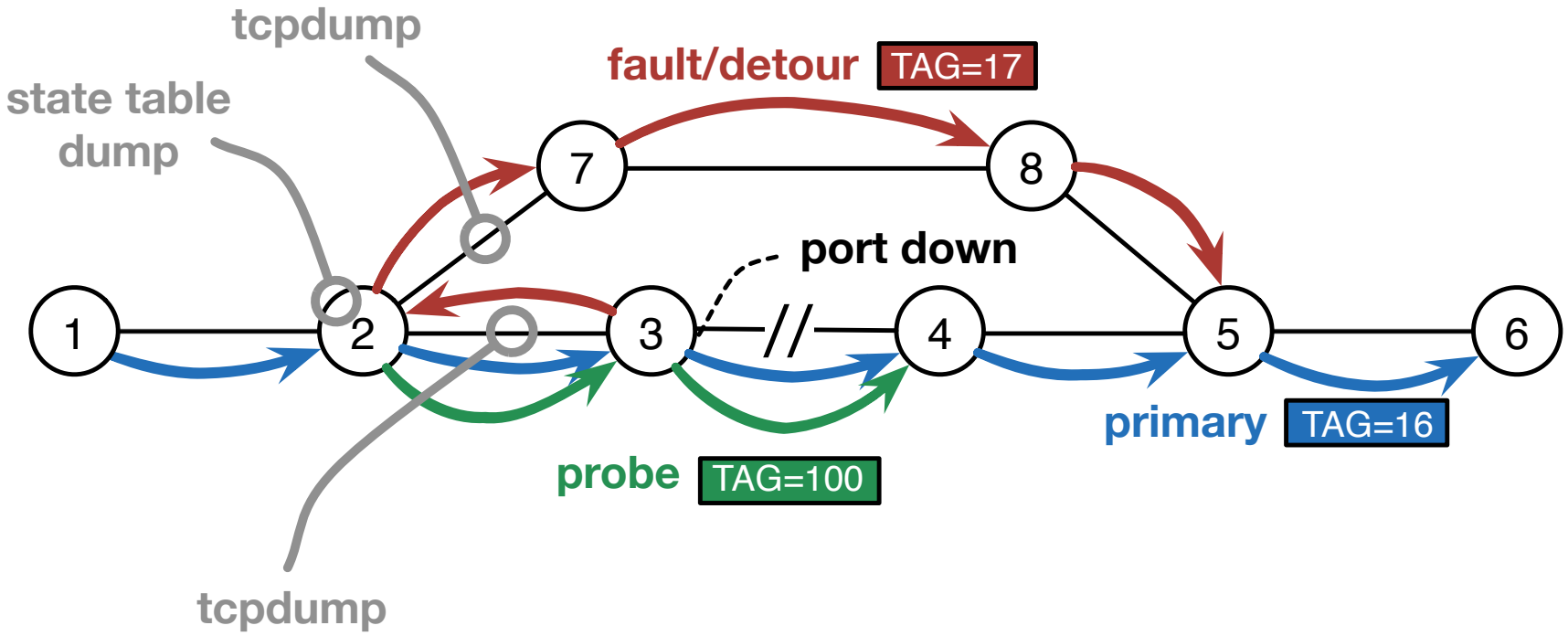
Failure solved



Match	Instructions
src=1, dst=6, state=0	fwd(3)
…	…
…	…

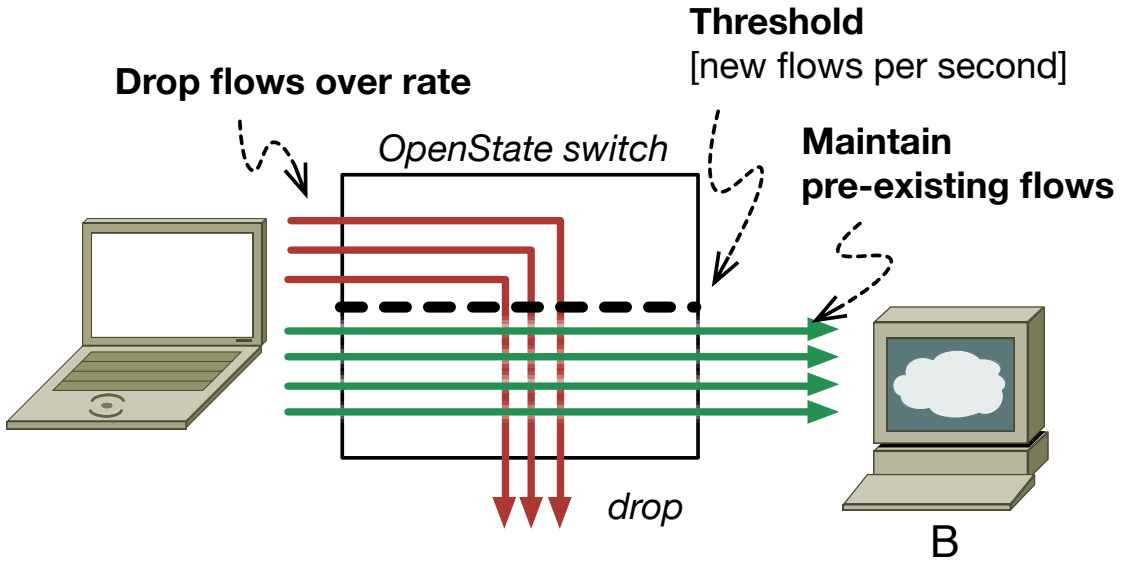
Failure recovery

Demo setup



DDoS mitigation building blocks

- **GOAL: measure the rate of new flows toward a given target**
  1. Block new connections initiated after a given threshold is reached
  2. Keep forwarding of all previous connections
- From this simple mechanism we can create **a more complex DDoS detection and mitigation scenario**



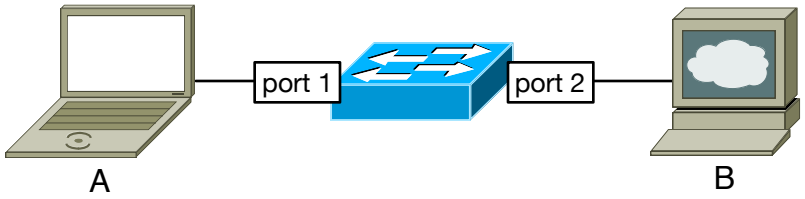
DDoS building blocks

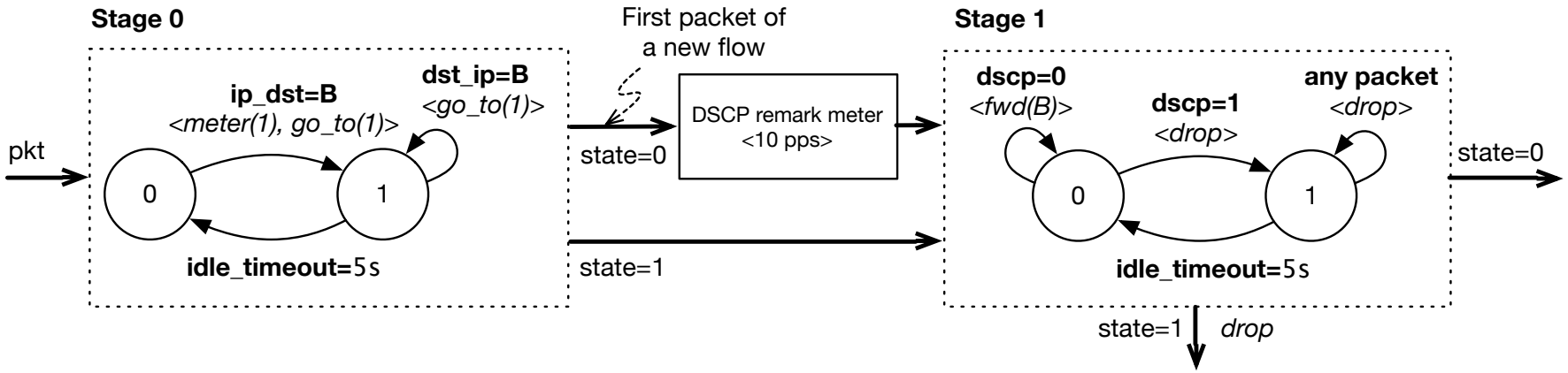
2 stateful stages

1. **Measurement stage**
   A. All “first packets” of any TCP flows are given as input to a DSCP meter
   B. The flow state is changed from 0 (i.e. new flow) to 1
   C. If the meter exceeds a threshold the packet is marked
2. **Forwarding stage**
   A. The first packet of a new flow set the verdict for that flow
   B. If the packet is not marked the flow status is set to 0 (i.e. a flow generated in a normal state)
   C. If the packet is marked the flow status is set to 1 (i.e. A flow generated after the threshold is reached)
   D. All packets whose flow is in state 1 are DROPED, FORWARDED otherwise

DDoS

Behavioral model





DDoS

Table configuration

Stage 1: Measurement

Key extractors:

Lookup-scope = {ip_src, ip_dst, tcp_src, tcp_dst}

Update-scope = {ip_src, ip_dst, tcp_src, tcp_dst}

Flow identification:
L3-L4 4-tuple

Flow table (table_id = 0)

	Match	Instructions/Actions
First packet of a TCP flow towards B	ip_dst=B, state=0	set_state(1, idle_to=5s); meter(1); goto(1)
Subsequent packets towards B	ip_dst=B, state=1	go_to(1)
Packet towards A	ip_dst=A	output(1)

DDoS

Table configuration

Stage 2: Forwarding

Key extractors:

Lookup-scope = {ip_src, ip_dst, tcp_src, tcp_dst}

Update-scope = {ip_src, ip_dst, tcp_src, tcp_dst}

Flow table (table_id = 1)

	Match	Instructions/Actions
First packet of a TCP flow when destination is already "under attack"	dscp=1	set_state(1, idle_to=5s); drop()
Subsequent packets of a TCP flow when destination is already "under attack"	dscp=0, state=1	drop()
Packets of a flow generated "before the attack"	dscp=0, state=0	output(2)