

# SDNRacer: Detecting Concurrency Violations in Software-Defined Networks

Jeremie Miserez, **Pavol Bielik**, Ahmed El-Hassany,  
Laurent Vanbever, Martin Vechev

# SDN Concurrency



Internal Host



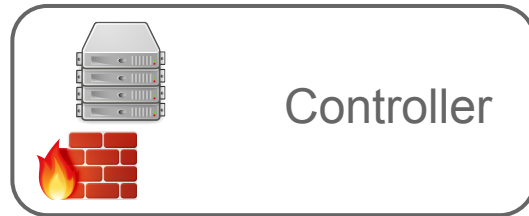
Switch



External Host

# SDN Concurrency

```
if ip_src == H1:  
    flow_mod(10,ip_src,  
            ip_dst,fwd)  
    flow_mod(10,ip_dst,  
            ip_src,fwd)  
    packet_out(pkt,sw)  
else:  
    flow_mod(15,ip_src,  
            ip_dst, drop)
```



Internal Host



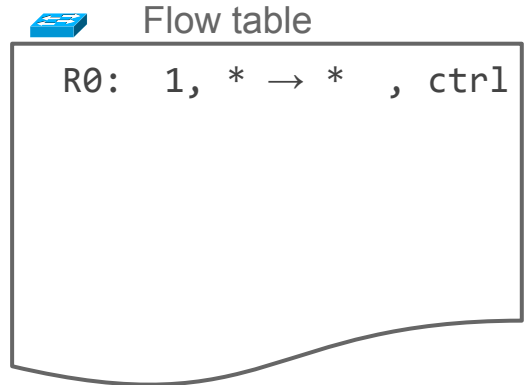
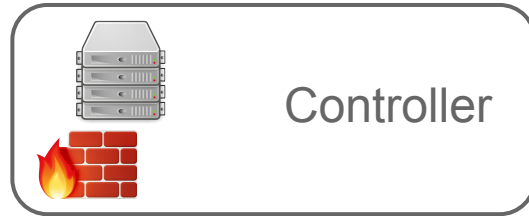
Switch



External Host

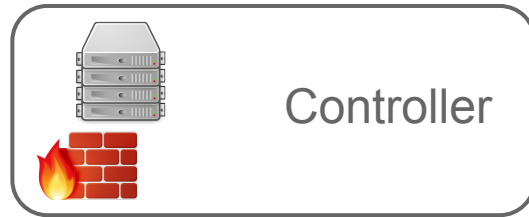
# SDN Concurrency

```
if ip_src == H1:  
    flow_mod(10,ip_src,  
            ip_dst,fwd)  
    flow_mod(10,ip_dst,  
            ip_src,fwd)  
    packet_out(pkt,sw)  
else:  
    flow_mod(15,ip_src,  
            ip_dst, drop)
```

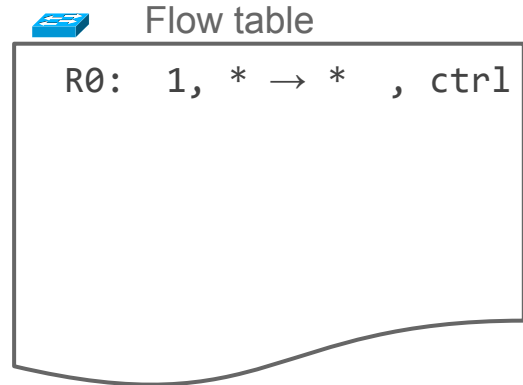


# SDN Concurrency

```
if ip_src == H1:  
    flow_mod(10,ip_src,  
            ip_dst,fwd)  
    flow_mod(10,ip_dst,  
            ip_src,fwd)  
    packet_out(pkt,sw)  
else:  
    flow_mod(15,ip_src,  
            ip_dst, drop)
```



3



Internal Host



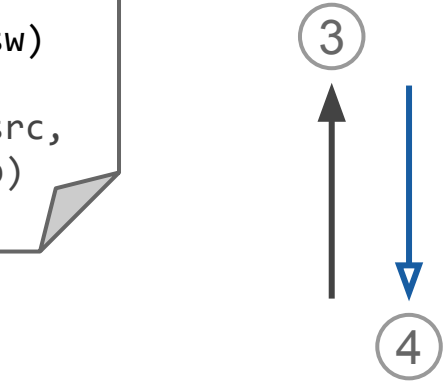
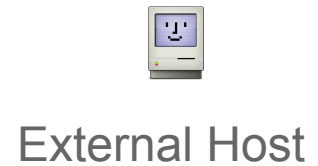
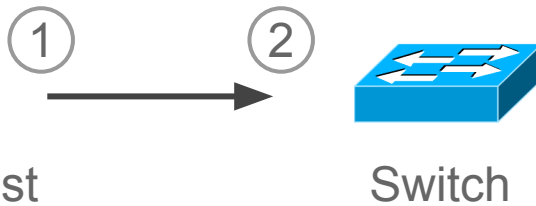
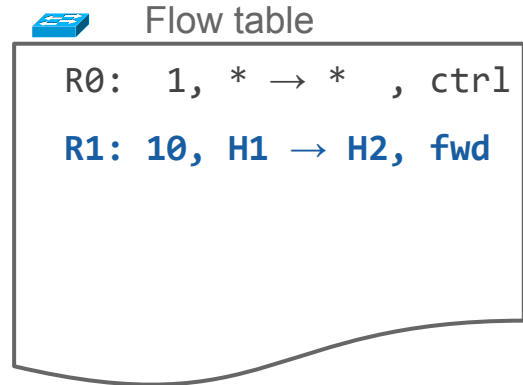
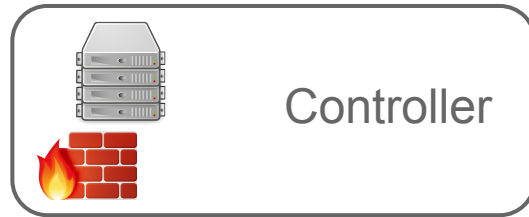
Switch



External Host

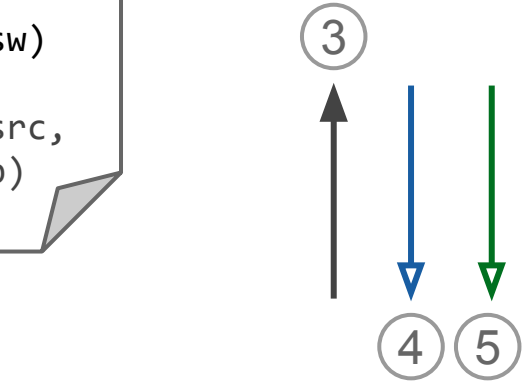
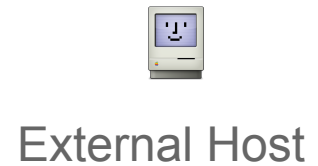
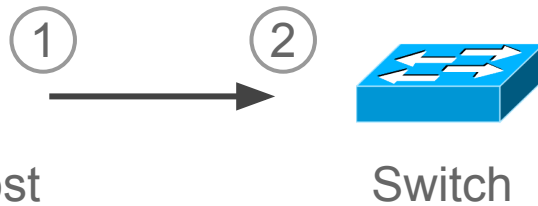
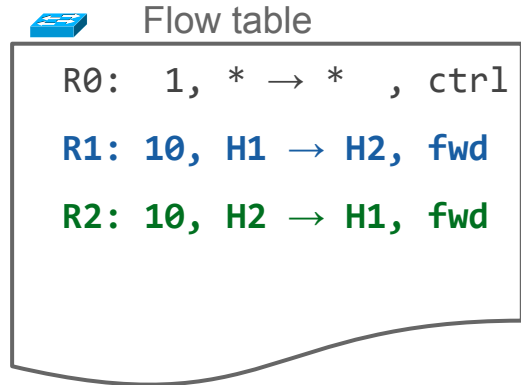
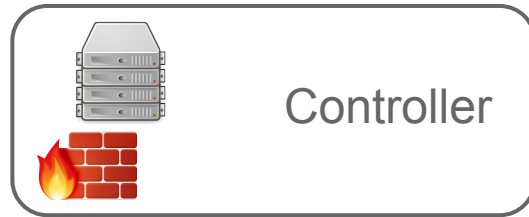
# SDN Concurrency

```
if ip_src == H1:  
    flow_mod(10, ip_src,  
            ip_dst, fwd)  
    flow_mod(10, ip_dst,  
            ip_src, fwd)  
    packet_out(pkt, sw)  
else:  
    flow_mod(15, ip_src,  
            ip_dst, drop)
```



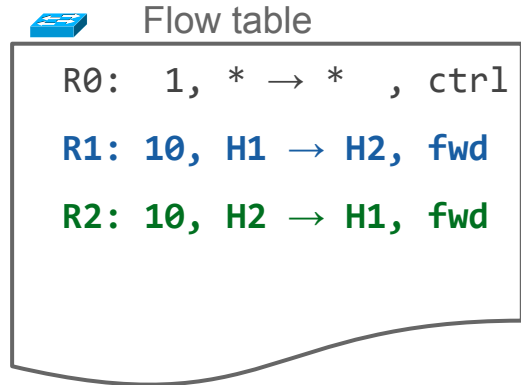
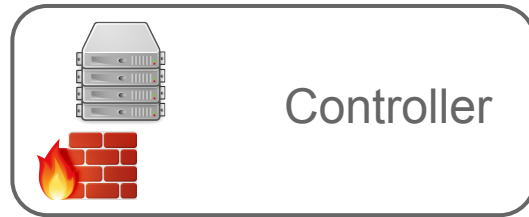
# SDN Concurrency

```
if ip_src == H1:  
    flow_mod(10, ip_src,  
            ip_dst, fwd)  
    flow_mod(10, ip_dst,  
            ip_src, fwd)  
    packet_out(pkt, sw)  
else:  
    flow_mod(15, ip_src,  
            ip_dst, drop)
```



# SDN Concurrency

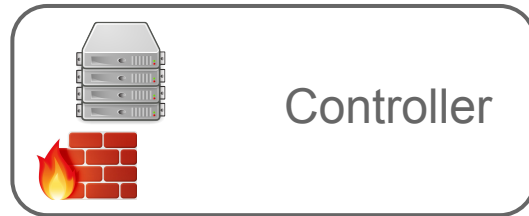
```
if ip_src == H1:  
    flow_mod(10, ip_src,  
            ip_dst, fwd)  
    flow_mod(10, ip_dst,  
            ip_src, fwd)  
    packet_out(pkt, sw)  
else:  
    flow_mod(15, ip_src,  
            ip_dst, drop)
```





# SDN Concurrency

```
if ip_src == H1:  
    flow_mod(10, ip_src,  
            ip_dst, fwd)  
    flow_mod(10, ip_dst,  
            ip_src, fwd)  
    packet_out(pkt, sw)  
else:  
    flow_mod(15, ip_src,  
            ip_dst, drop)
```



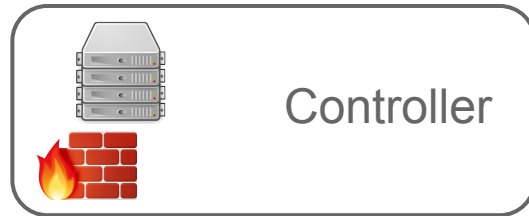
Flow table


R0:	1, *	→ *	, ctrl
R1:	10, H1	→ H2,	fwd
R2:	10, H2	→ H1,	fwd



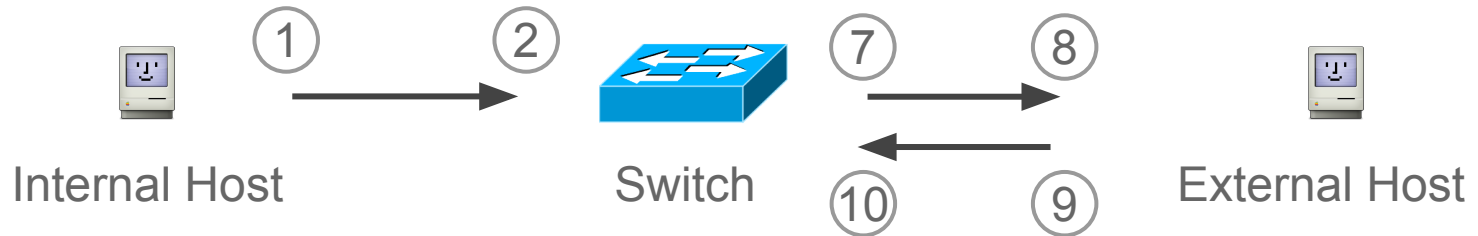
# SDN Concurrency

```
if ip_src == H1:  
    flow_mod(10, ip_src,  
            ip_dst, fwd)  
    flow_mod(10, ip_dst,  
            ip_src, fwd)  
    packet_out(pkt, sw)  
else:  
    flow_mod(15, ip_src,  
            ip_dst, drop)
```



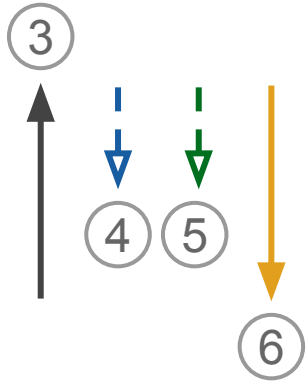
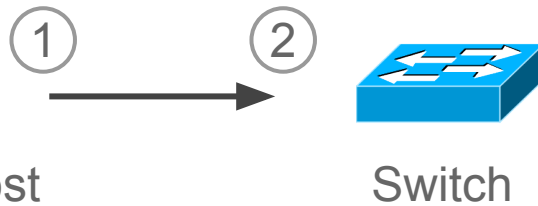
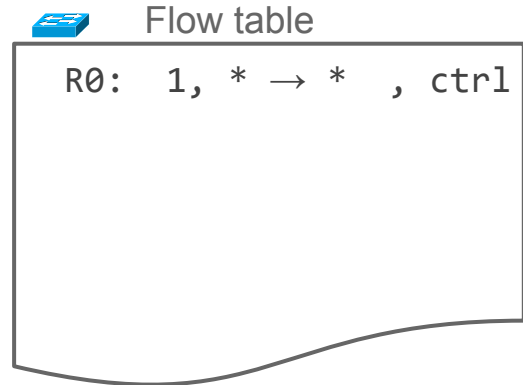
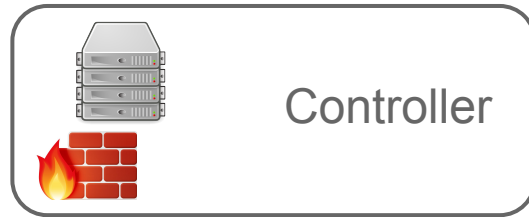
 Flow table

R0:	1, *	→ *	, ctrl
R1:	10, H1	→ H2,	fwd
R2:	10, H2	→ H1,	fwd



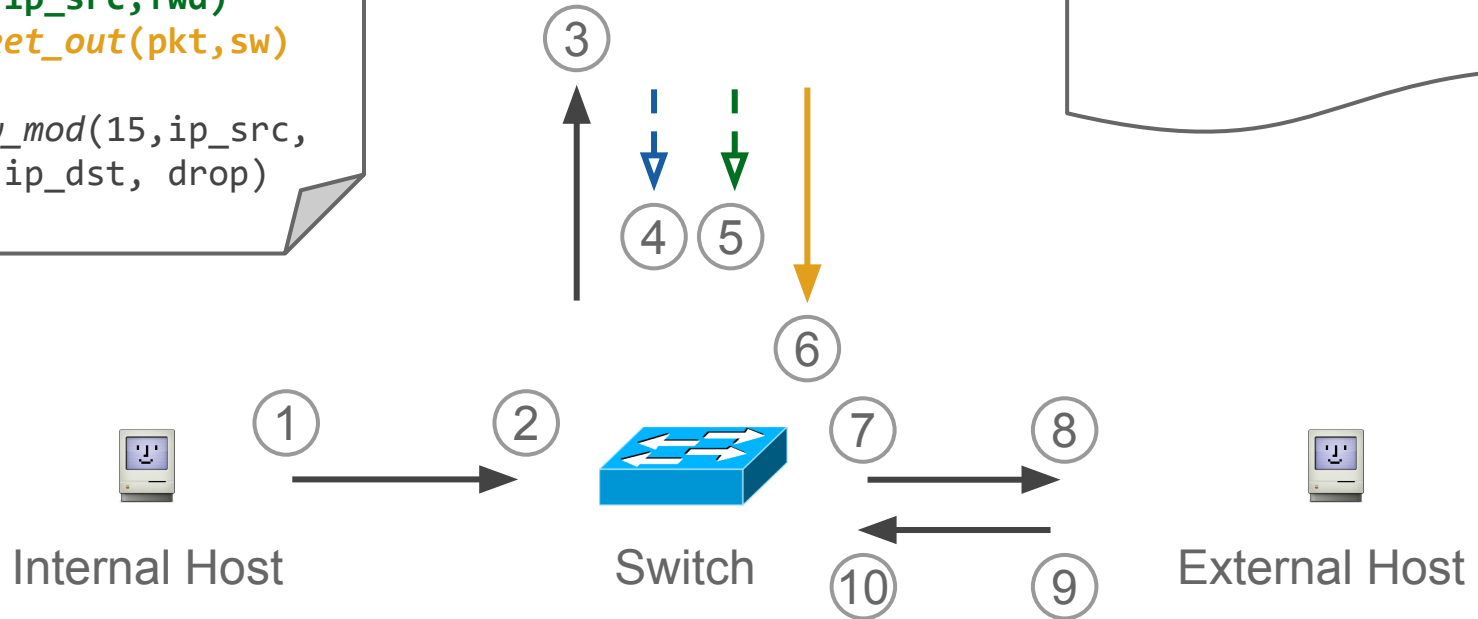
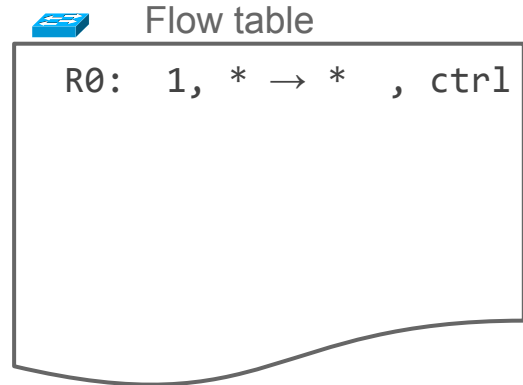
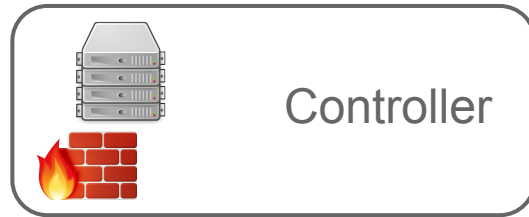
# SDN Concurrency

```
if ip_src == H1:  
    flow_mod(10, ip_src,  
            ip_dst, fwd)  
    flow_mod(10, ip_dst,  
            ip_src, fwd)  
    packet_out(pkt, sw)  
else:  
    flow_mod(15, ip_src,  
            ip_dst, drop)
```



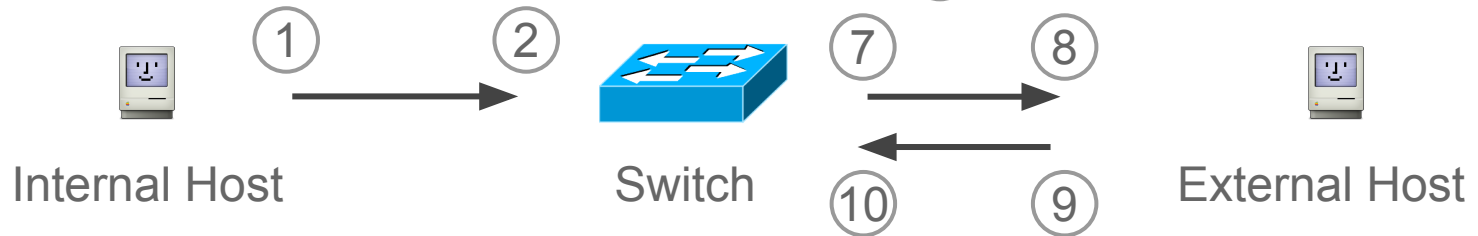
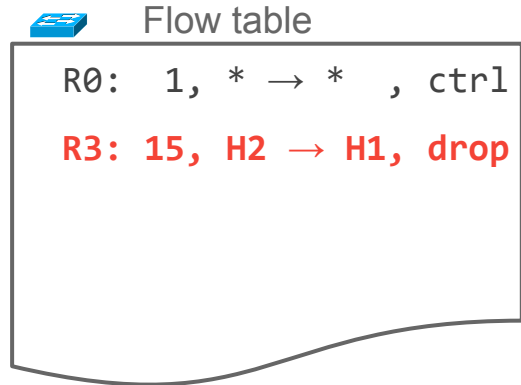
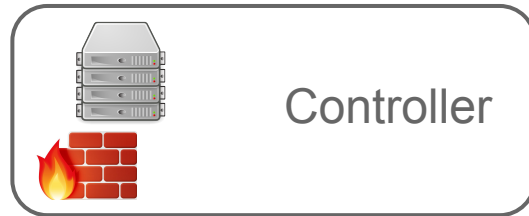
# SDN Concurrency

```
if ip_src == H1:  
    flow_mod(10, ip_src,  
            ip_dst, fwd)  
    flow_mod(10, ip_dst,  
            ip_src, fwd)  
    packet_out(pkt, sw)  
else:  
    flow_mod(15, ip_src,  
            ip_dst, drop)
```



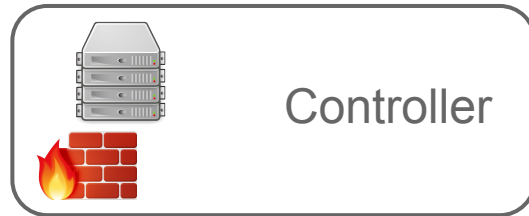
# SDN Concurrency

```
if ip_src == H1:  
    flow_mod(10, ip_src,  
            ip_dst, fwd)  
    flow_mod(10, ip_dst,  
            ip_src, fwd)  
    packet_out(pkt, sw)  
else:  
    flow_mod(15, ip_src,  
            ip_dst, drop)
```



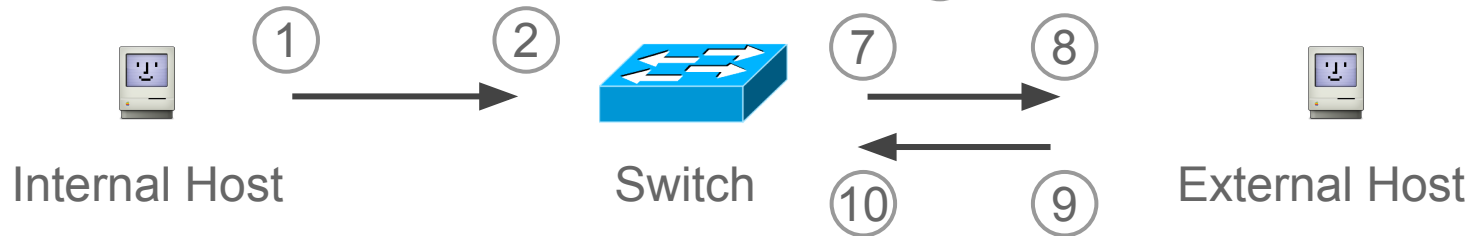
# SDN Concurrency

```
if ip_src == H1:  
    flow_mod(10, ip_src,  
            ip_dst, fwd)  
    flow_mod(10, ip_dst,  
            ip_src, fwd)  
    packet_out(pkt, sw)  
else:  
    flow_mod(15, ip_src,  
            ip_dst, drop)
```

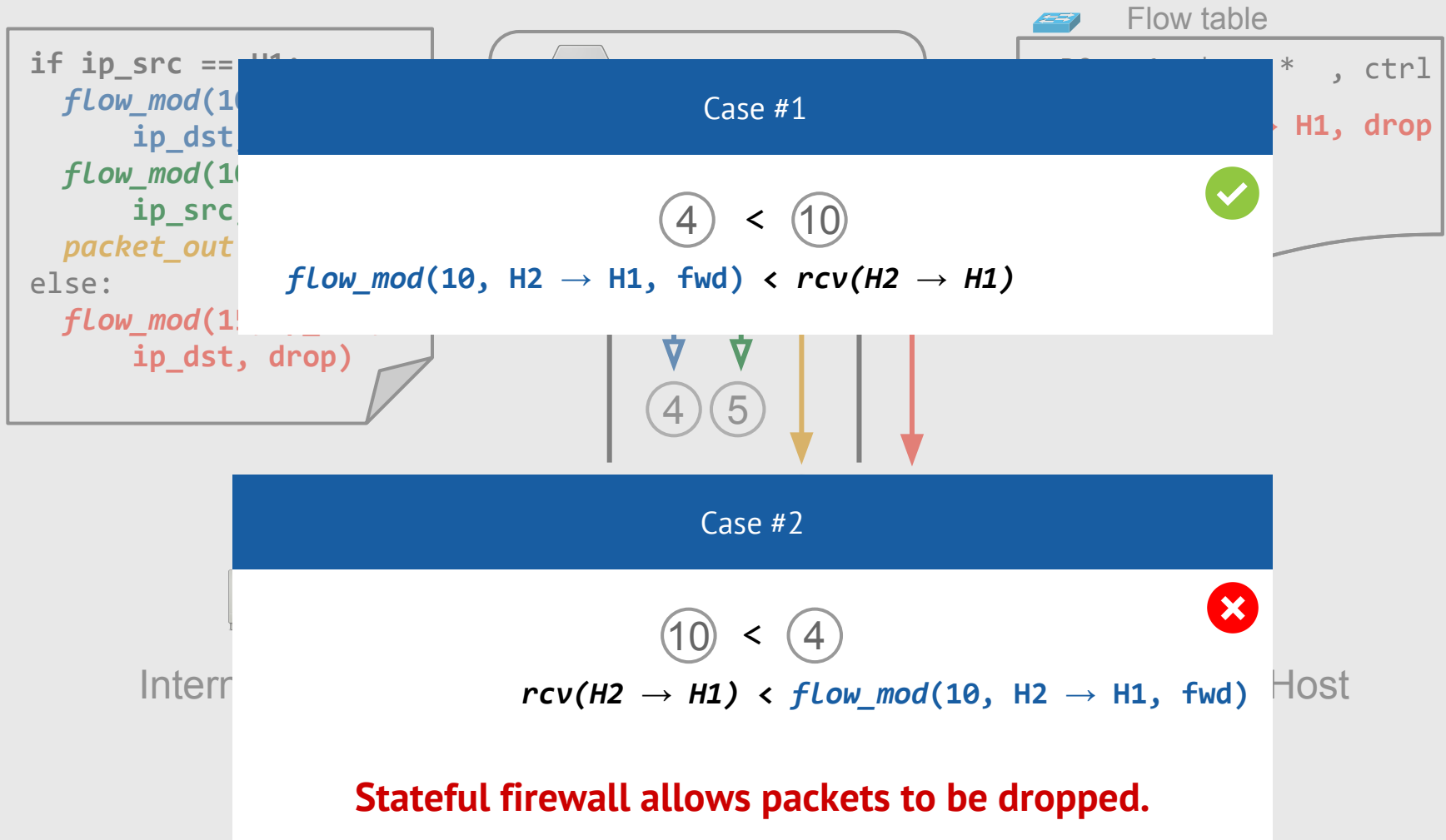


Flow table

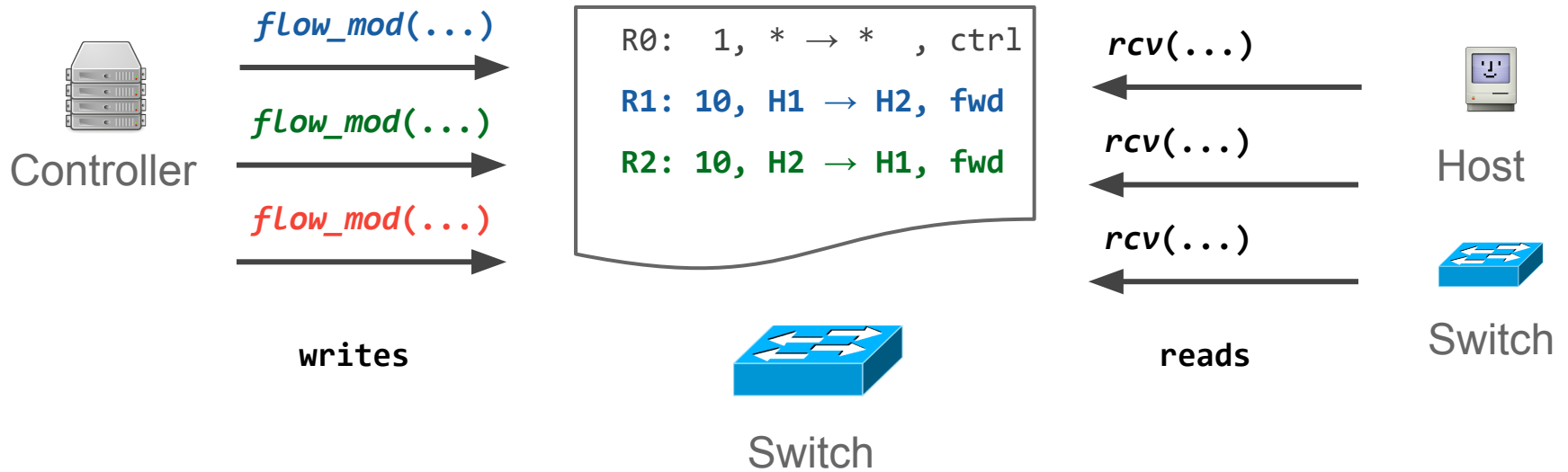
R0:	1,	*	→	*	,	ctrl
R3:	15,	H2	→	H1,	drop	
R1:	10,	H1	→	H2,	fwd	
R2:	10,	H2	→	H1,	fwd	



# Root Cause

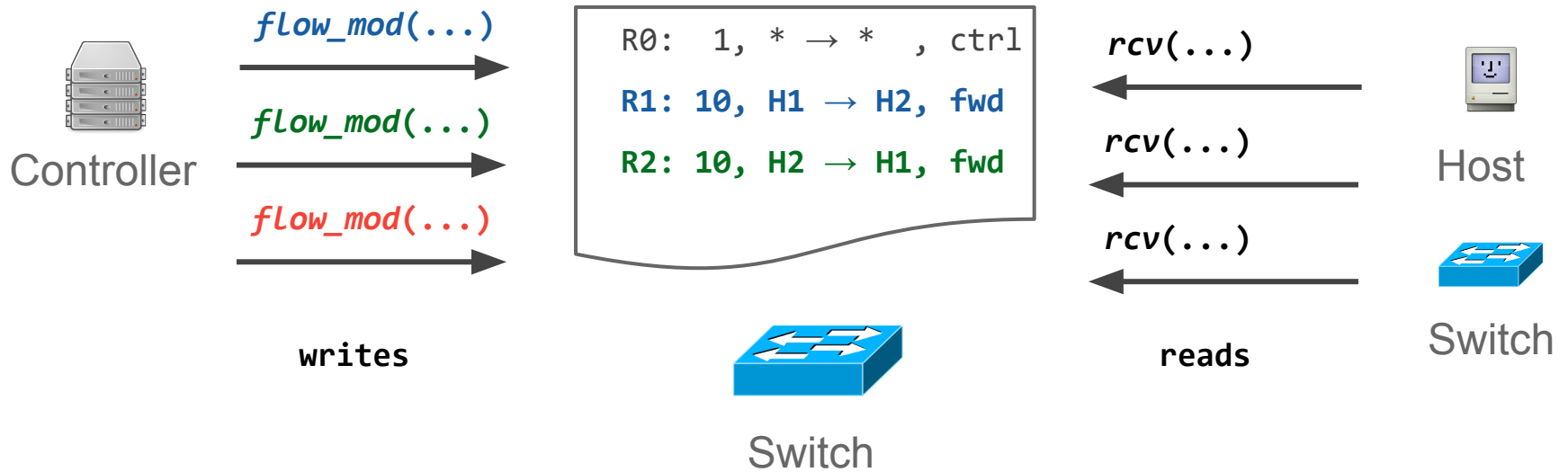


# Detecting SDN Asynchrony





# Detecting SDN Asynchrony



Detect Data Races on Switch Flow Table

Data Race: two unordered events accessing the same flow table, where at least one of the events is a write.

# SDNRacer: Detecting Concurrency Violations

SDN Network

Capturing  
Asynchrony

Conflict Detection

Commutativity

## How to capture asynchrony in SDN?

Define high level events:

- Flow table write:
  - FLOW\_MOD (ADD)
  - FLOW\_MOD (DELETE)
  - FLOW\_MOD (MODIFY)
- Flow table read:
  - Lookup rule for packet
- Send packet/message
- Receive packet/message
- Switch → Controller:
  - PACKET\_IN
  - FLOW\_REMOVED
  - PORT\_STATUS
- Controller → Switch:
  - FLOW\_MOD
  - PORT\_MOD
  - PACKET\_OUT
  - BARRIER\_REQUEST

# SDNRacer: Detecting Concurrency Violations

SDN Network

Capturing  
Asynchrony

Conflict Detection

Commutativity

## How to capture asynchrony in SDN?

Define happens-before relation between the events.



# SDNRacer: Detecting Concurrency Violations

SDN Network

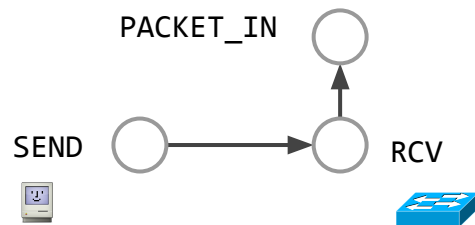
Capturing  
Asynchrony

Conflict Detection

Commutativity

## How to capture asynchrony in SDN?

Define happens-before relation between the events.



# SDNRacer: Detecting Concurrency Violations

SDN Network

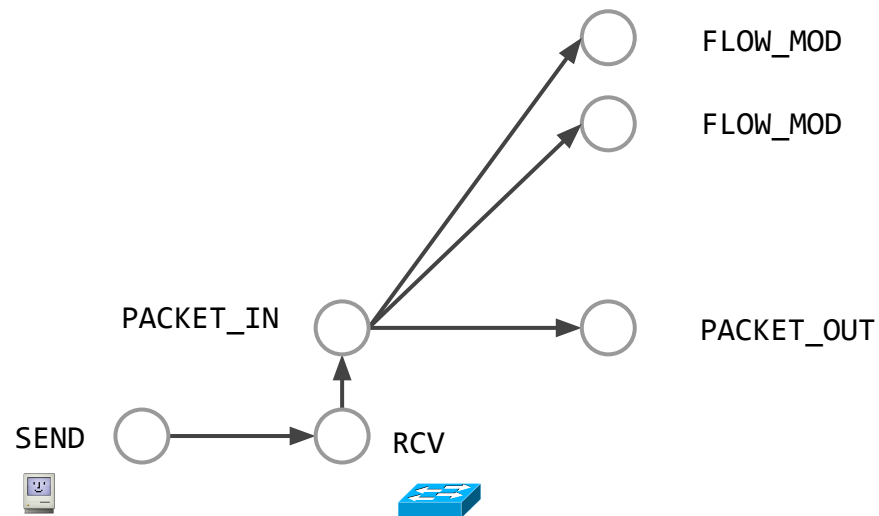
Capturing  
Asynchrony

Conflict Detection

Commutativity

## How to capture asynchrony in SDN?

Define happens-before relation between the events.



# SDNRacer: Detecting Concurrency Violations

SDN Network

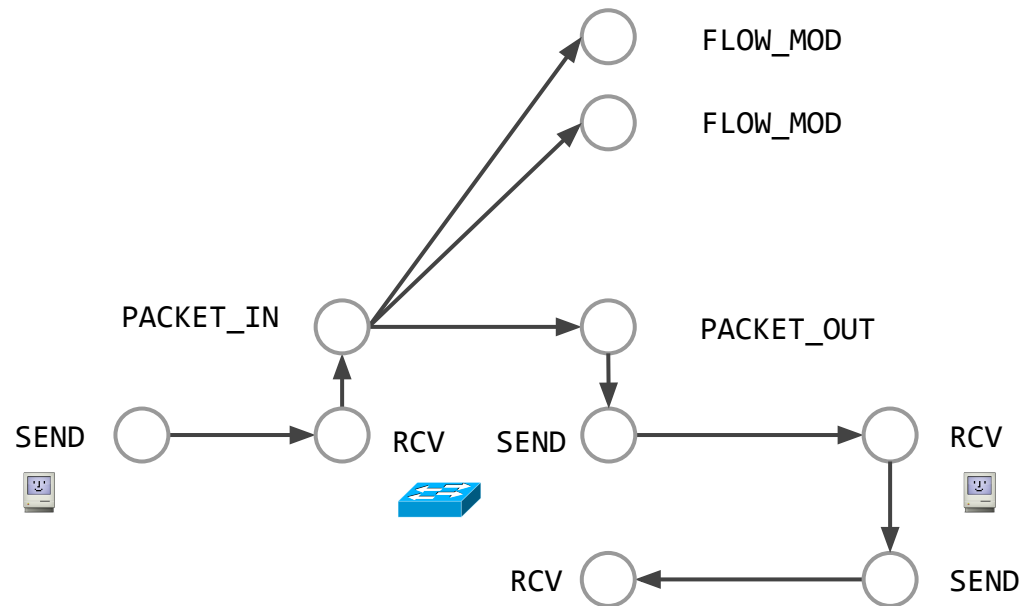
Capturing  
Asynchrony

Conflict Detection

Commutativity

## How to capture asynchrony in SDN?

Define happens-before relation between the events.



Instrumentation to track packet and message identities

# SDNRacer: Detecting Concurrency Violations

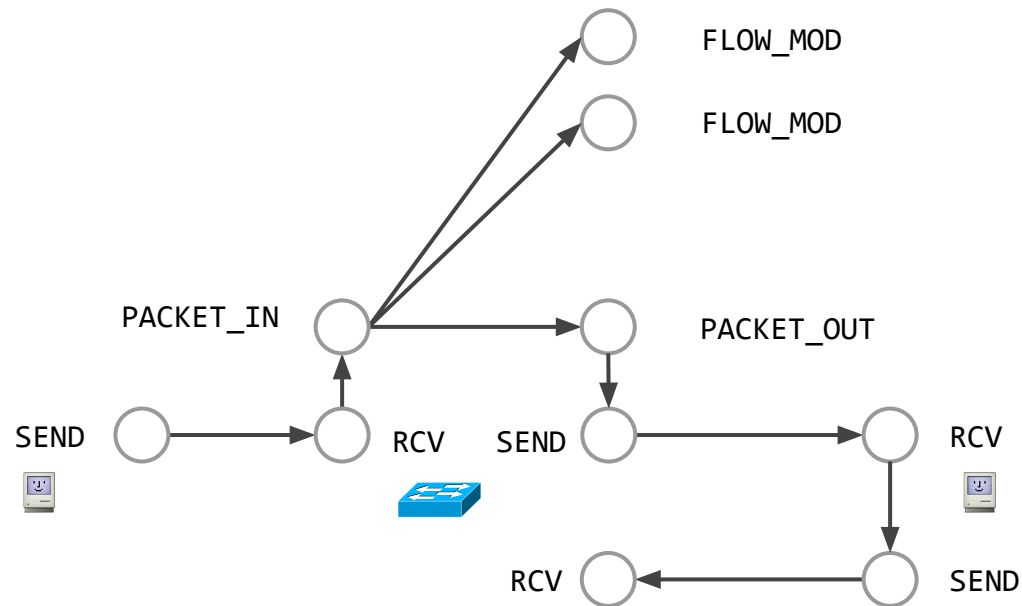
SDN Network

Capturing  
Asynchrony

Conflict Detection

Commutativity

Which high-level events are conflicting?



# SDNRacer: Detecting Concurrency Violations

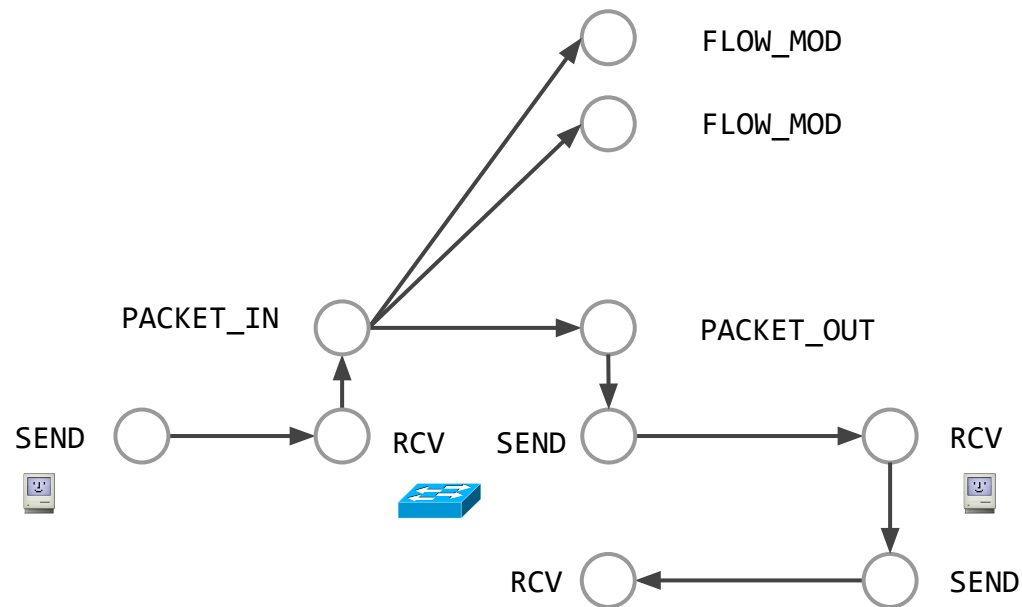
SDN Network

Capturing  
Asynchrony

Conflict Detection

Commutativity

Which high-level events are conflicting?



Phrased as a graph connectivity query



# SDNRacer: Detecting Concurrency Violations

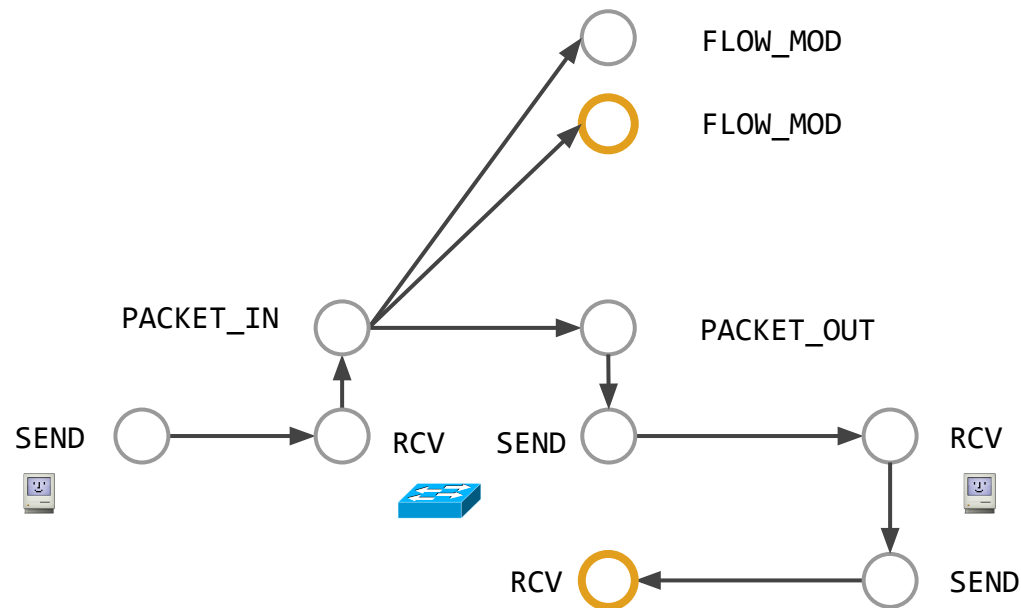
SDN Network

Capturing  
Asynchrony

Conflict Detection

Commutativity

Which high-level events are conflicting?



Phrased as a graph connectivity query

# SDNRacer: Detecting Concurrency Violations

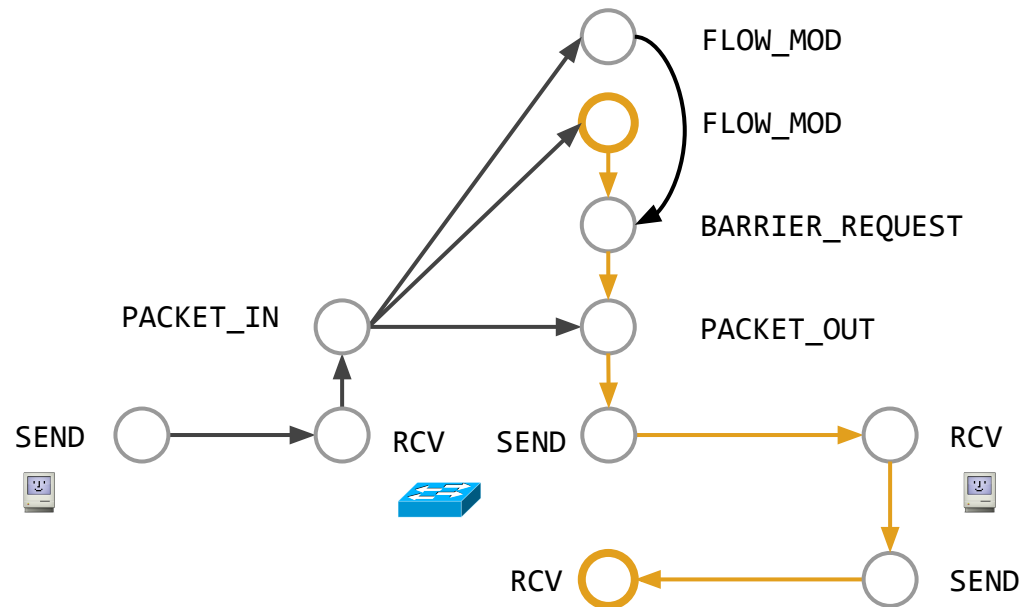
SDN Network

Capturing  
Asynchrony

Conflict Detection

Commutativity

Which high-level events are conflicting?



Phrased as a graph connectivity query

# Evaluation

POX l2\_multi controller with random network traffic

# Events	# Races
500	564
1000	2822
4000	4026

# SDNRacer: Detecting Concurrency Violations

SDN Network

Capturing  
Asynchrony

Conflict Detection

Commutativity

Which high-level events commute?

$\varphi$  FLOW\_MOD(10, H1 → H2, fwd)  
FLOW\_MOD(10, H2 → H1, fwd)



Flow table

```
R0: 1, * → * , ctrl  
R1: 10, H1 → H2, fwd  
R2: 10, H2 → H1, fwd
```



# SDNRacer: Detecting Concurrency Violations

SDN Network

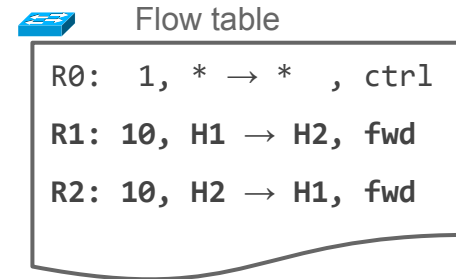
Capturing  
Asynchrony

Conflict Detection

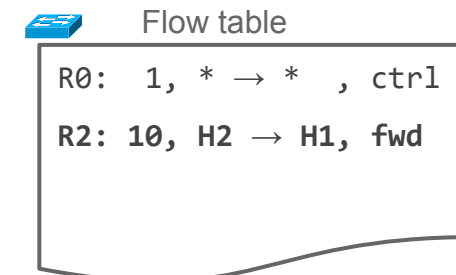
Commutativity

Which high-level events commute?

$\varphi$  FLOW\_MOD(10, H1 → H2, fwd)  
FLOW\_MOD(10, H2 → H1, fwd)



$\varphi$  RCV(H2 → H1)  
FLOW\_MOD(10, H2 → H1, fwd)



# Evaluation

POX l2\_multi controller with random network traffic

# Events	# Races	# Commutative Races	# Analysis Time
500	564	321 (57%)	0.7s
1000	2822	1900 (67%)	3.5s
4000	4026	2702 (67%)	16.1s

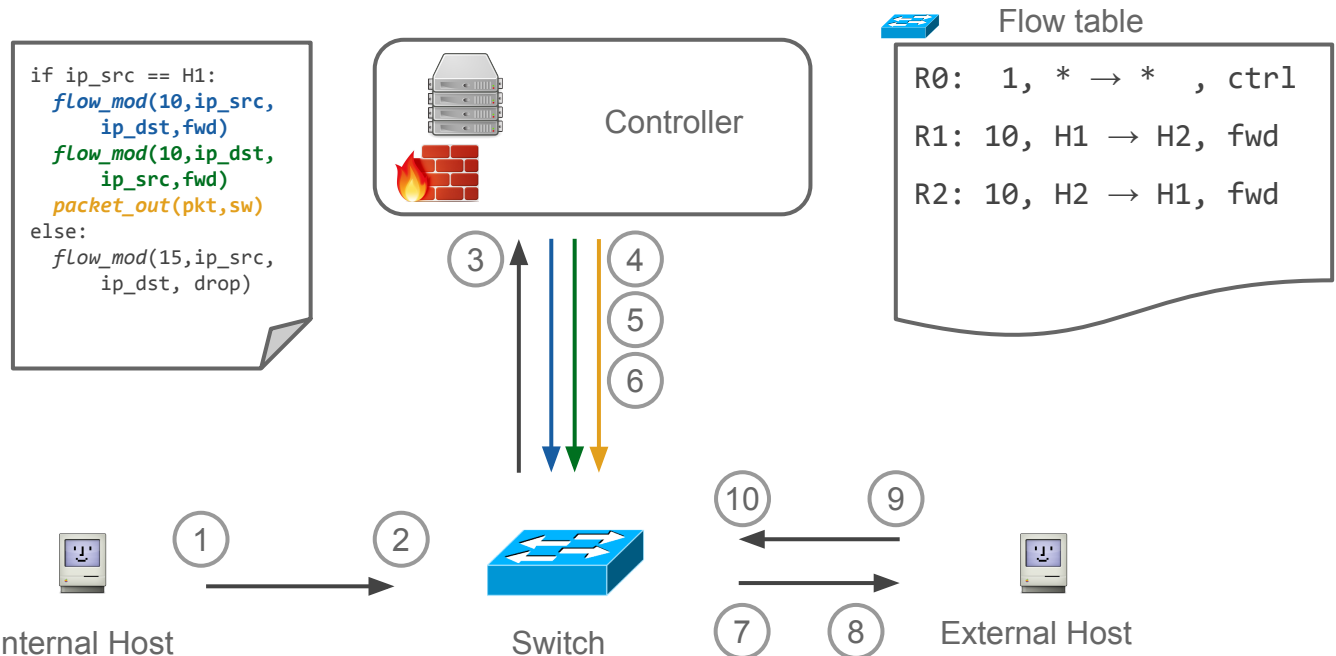
# SDNRacer: Detecting Concurrency Violations

SDN Network

Capturing  
Asynchrony

Conflict Detection

Commutativity



Concurrency violations:

FLOW\_MOD(10, H2 → H1, fwd) **4** ⚡ **10** RCV(H2 → H1)

