Flow-level State Transition as a New Switch Primitive for SDN

(HotSDN'14)

Masoud Moshref, Apoorv Bhargava,

Adhip Gupta, Minlan Yu, Ramesh Govindan

USC University of Southern California

Motivation

Current practice

- **Proactive needs a priori knowledge**
- **Reactive has high delay**

Opportunity: Local state is enough for many policies (stateful firewall, FTP monitoring, large source IP detection)
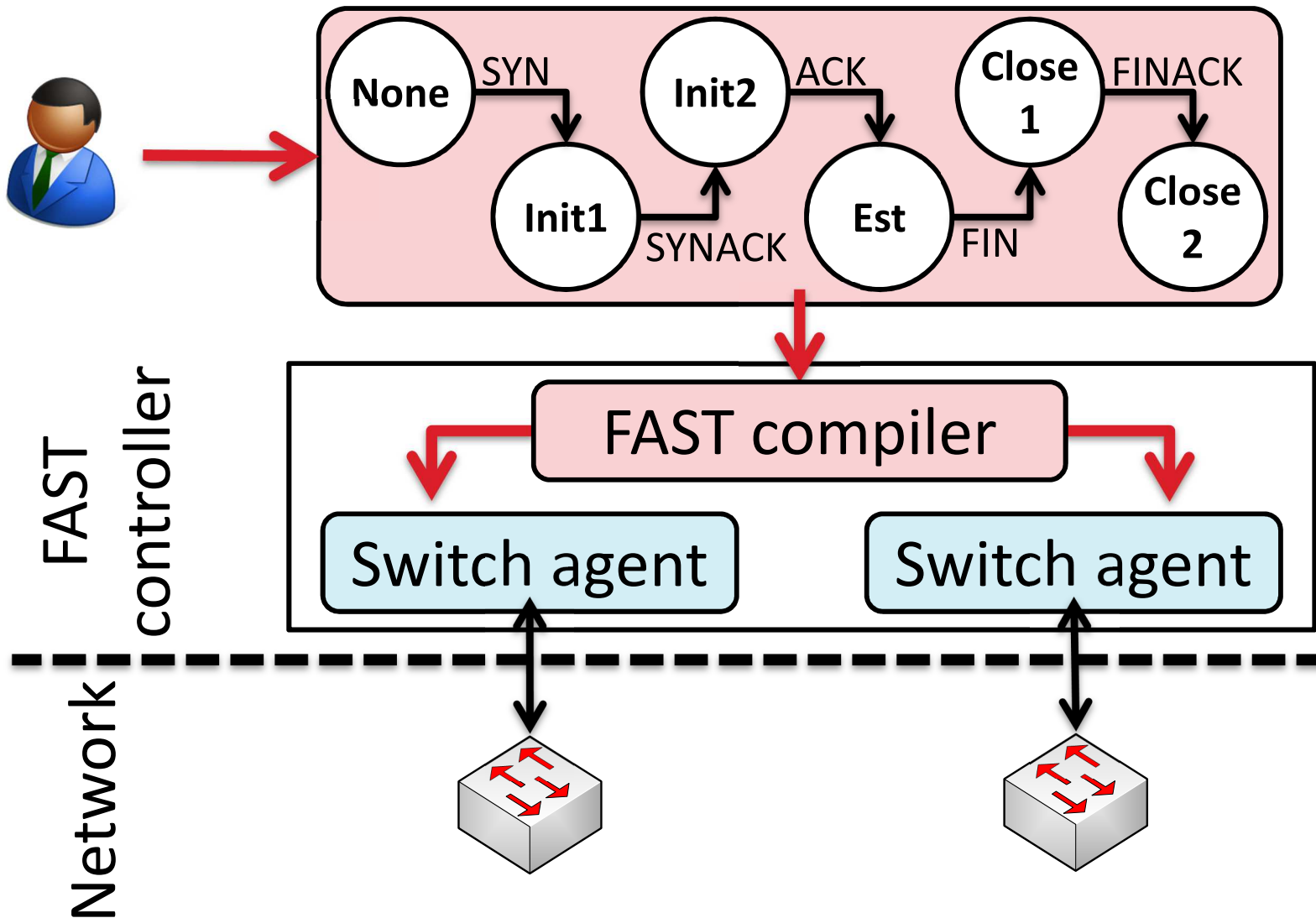
Key idea: State machine is a general but efficient abstraction to allow dynamic actions at switches

- Controller proactively programs state transitions and actions at switches
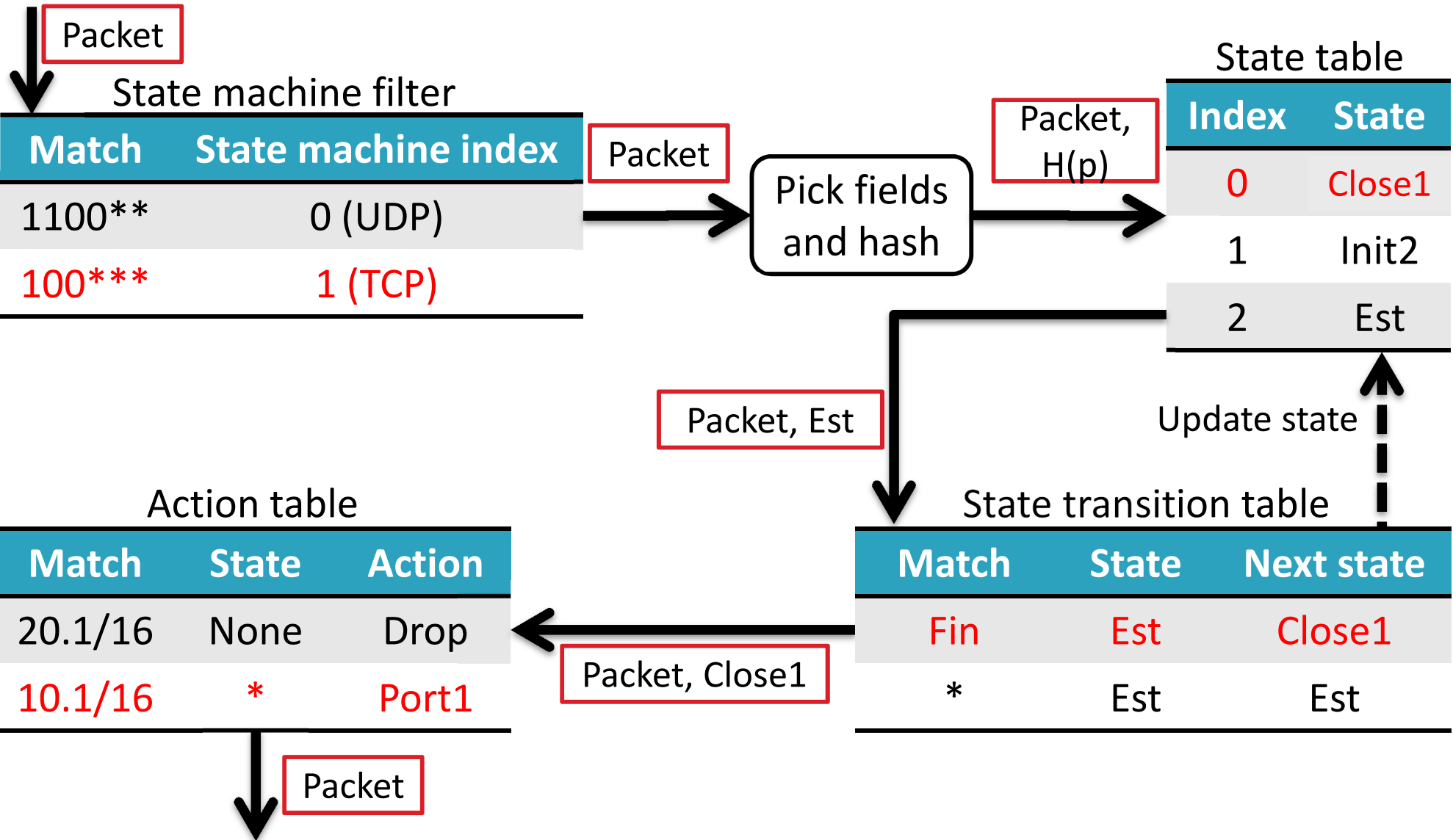- Switches run state machines and actions of a state

Examples:

- **Stateful firewall:** TCP state machine with actions that drop uninitiated flows
- **FTP Monitoring:** Track the states of control channel & allow data channel traffic
- **Large source IP detection:** Keep a counter per IP and compare it against a threshold

Controller translates state machines to switch API

FAST data plane is implementable in hardware switch components

Packet

State machine filter

Match	State machine index
1100**	0 (UDP)
100***	1 (TCP)

Packet

Pick fields and hash

Packet, H(p)

State table

Index	State
0	Close1
1	Init2
2	Est

Update state

Packet, Est

State transition table

Match	State	Next state
Fin	Est	Close1
*	Est	Est

Packet, Close1

Action table

Match	State	Action
20.1/16	None	Drop
10.1/16	*	Port1

Packet

Delay of going through all TCP states for FAST is small
1 packet, 1 flow : FAST: 28x faster (3ms)
> 64 concurrent flows: 6ms

FAST state lookup has small overhead:
Iperf throughput (Gbps): <5% overhead