IEEE INFOCOM

FlowCloak: Defeating Middlebox-Bypass Attacks in Software-Defined Networking

Kai Bu[1], **Yutian Yang[1]**, Zixuan Guo[1], Yuanyuan Yang[2], Xing Li[1], Shigeng Zhang[3]

[1]Zhejiang University [2]Stony Brook University [3]Central South University

Middlebox

Middlebox: Pain Spot in modern networks

- Needs

Varieties of functions: Security & Performance

Widely deployed: A third of network devices

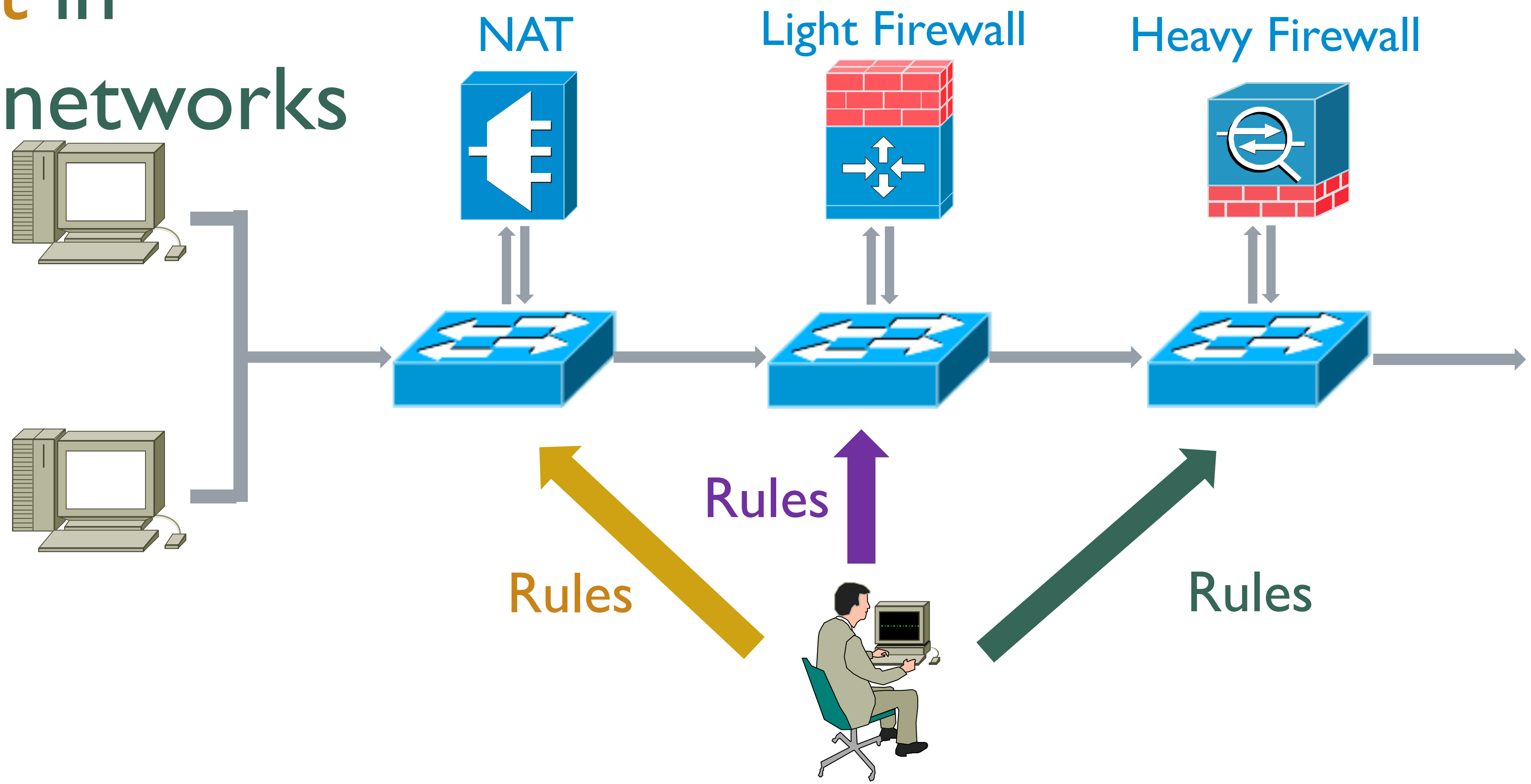
- Troubles

Deployment and configuration:
Complex & Error-prone

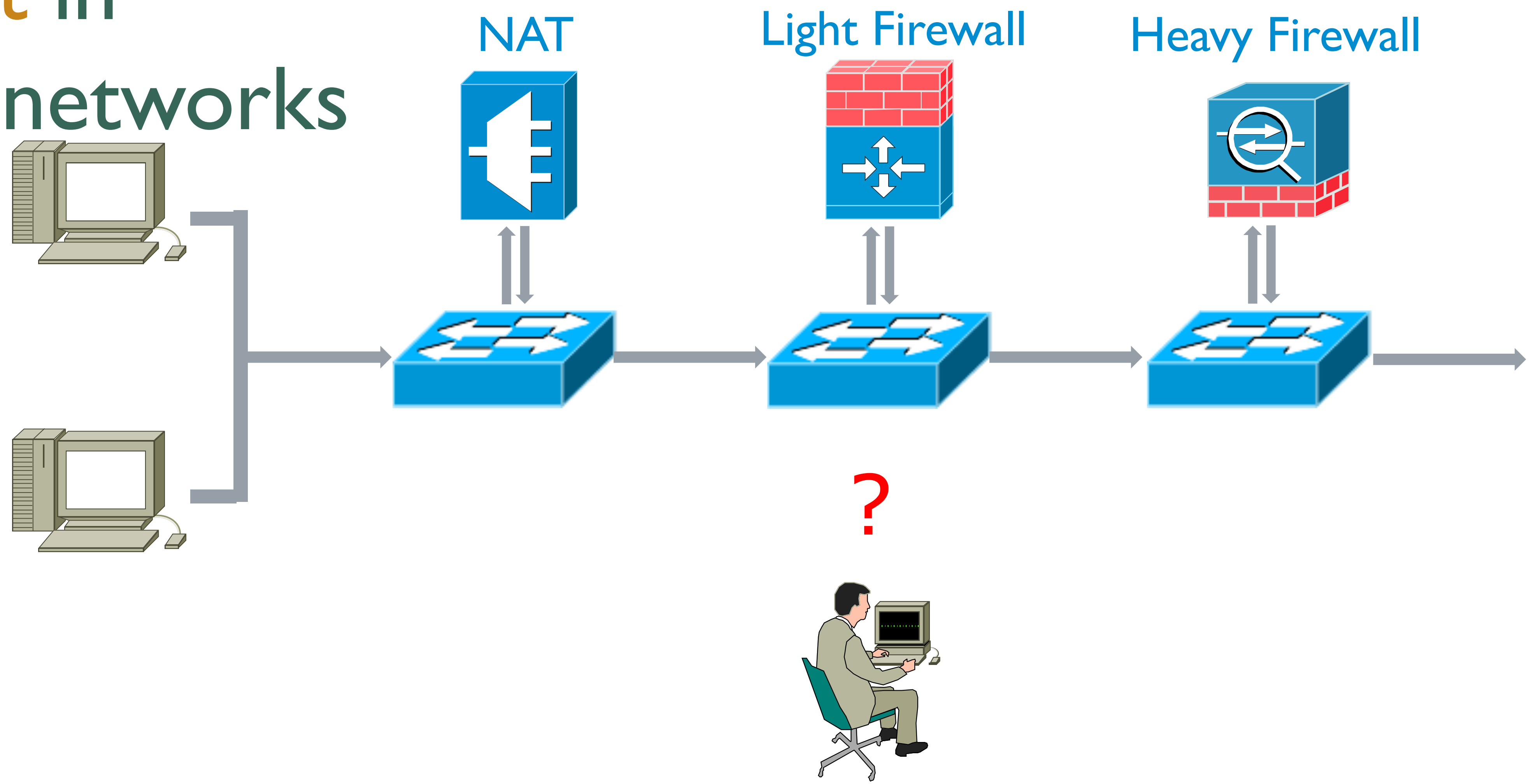
Costs: Personnel, Money, Time

Middlebox: Pain Spot in modern networks

Middlebox: Pain Spot in modern networks



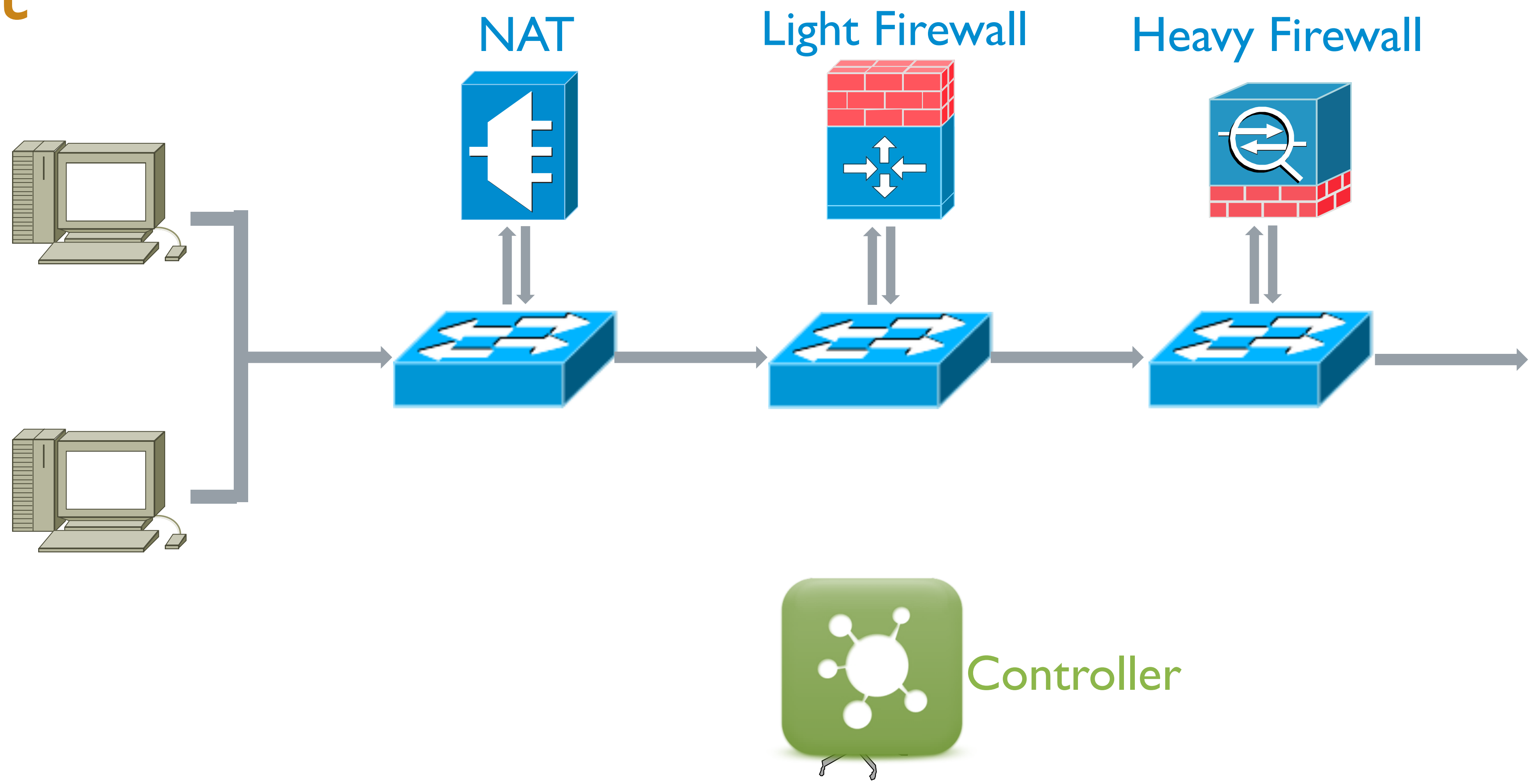
Middlebox: Pain Spot in modern networks



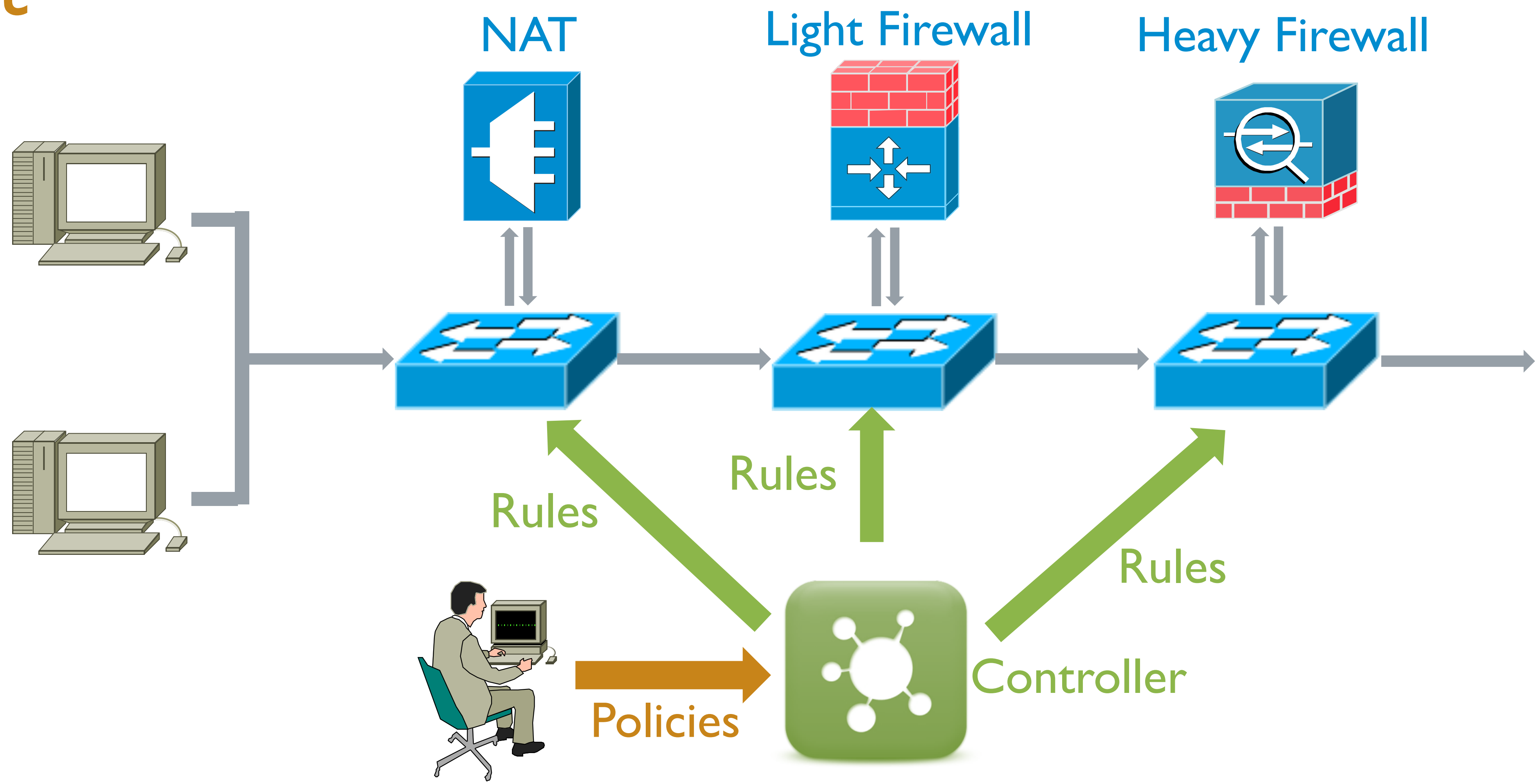
Middlebox:

Pain Spot

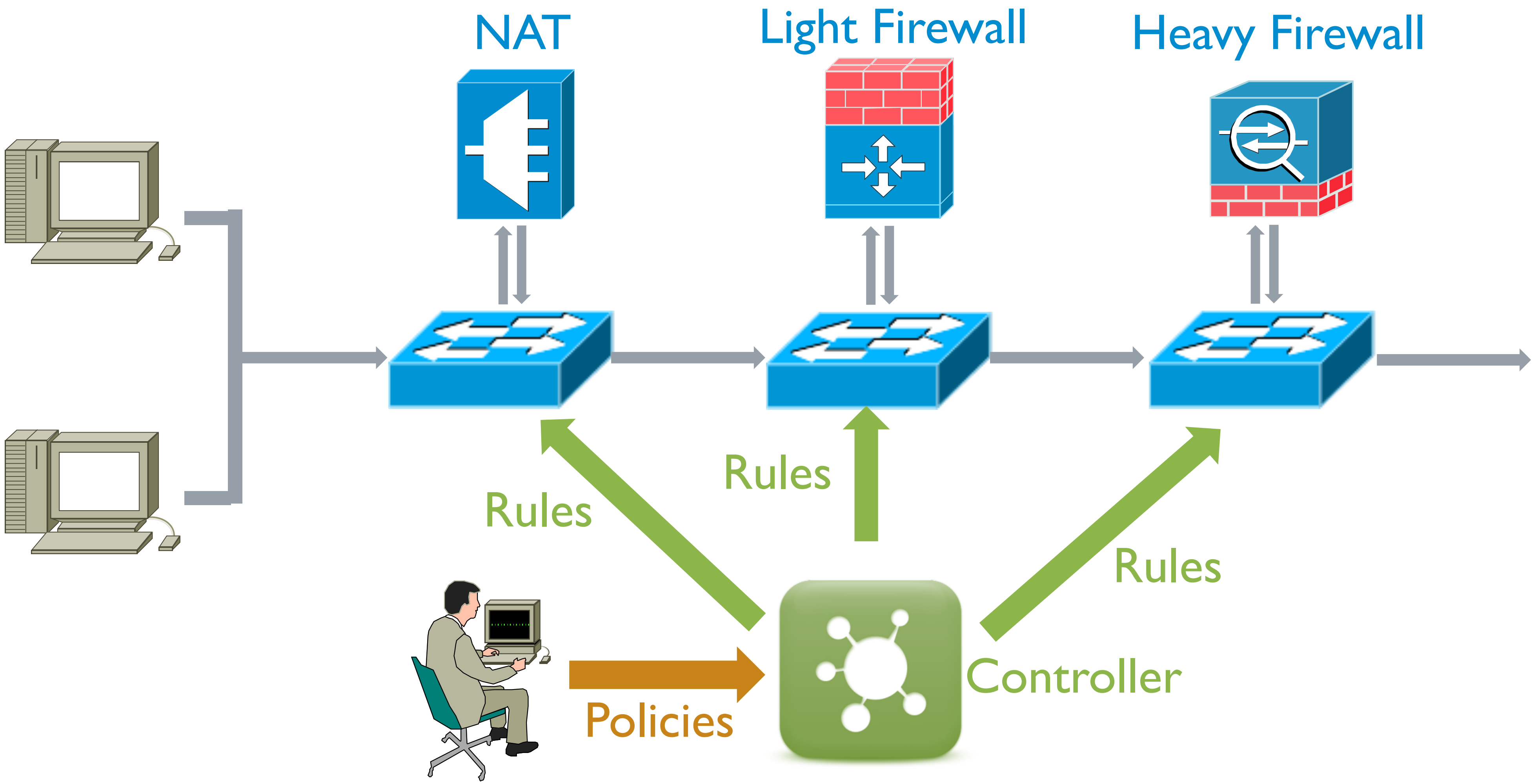
SDN



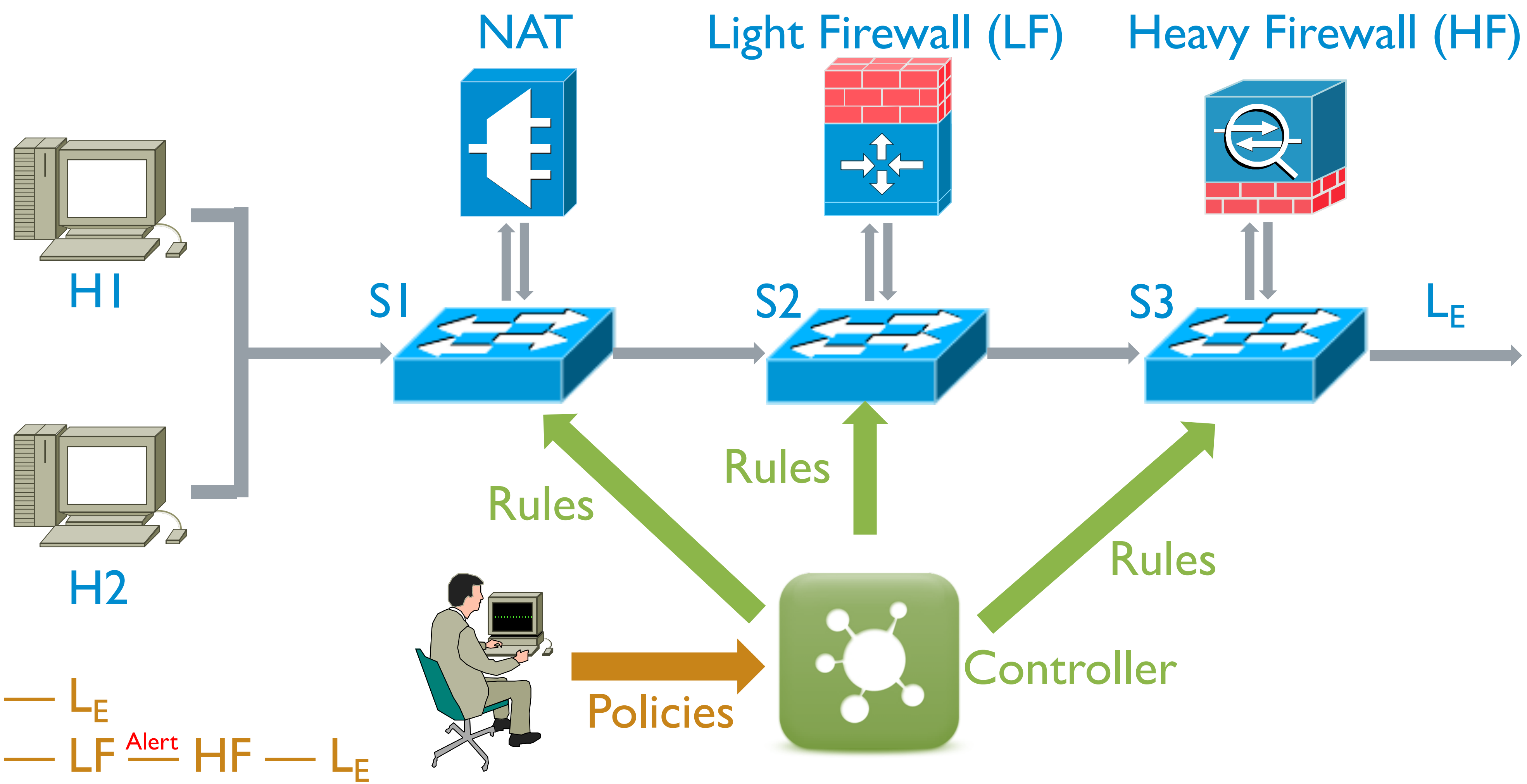
Middlebox: Pain Spot SDN



Middlebox meets SDN



Middlebox meets SDN



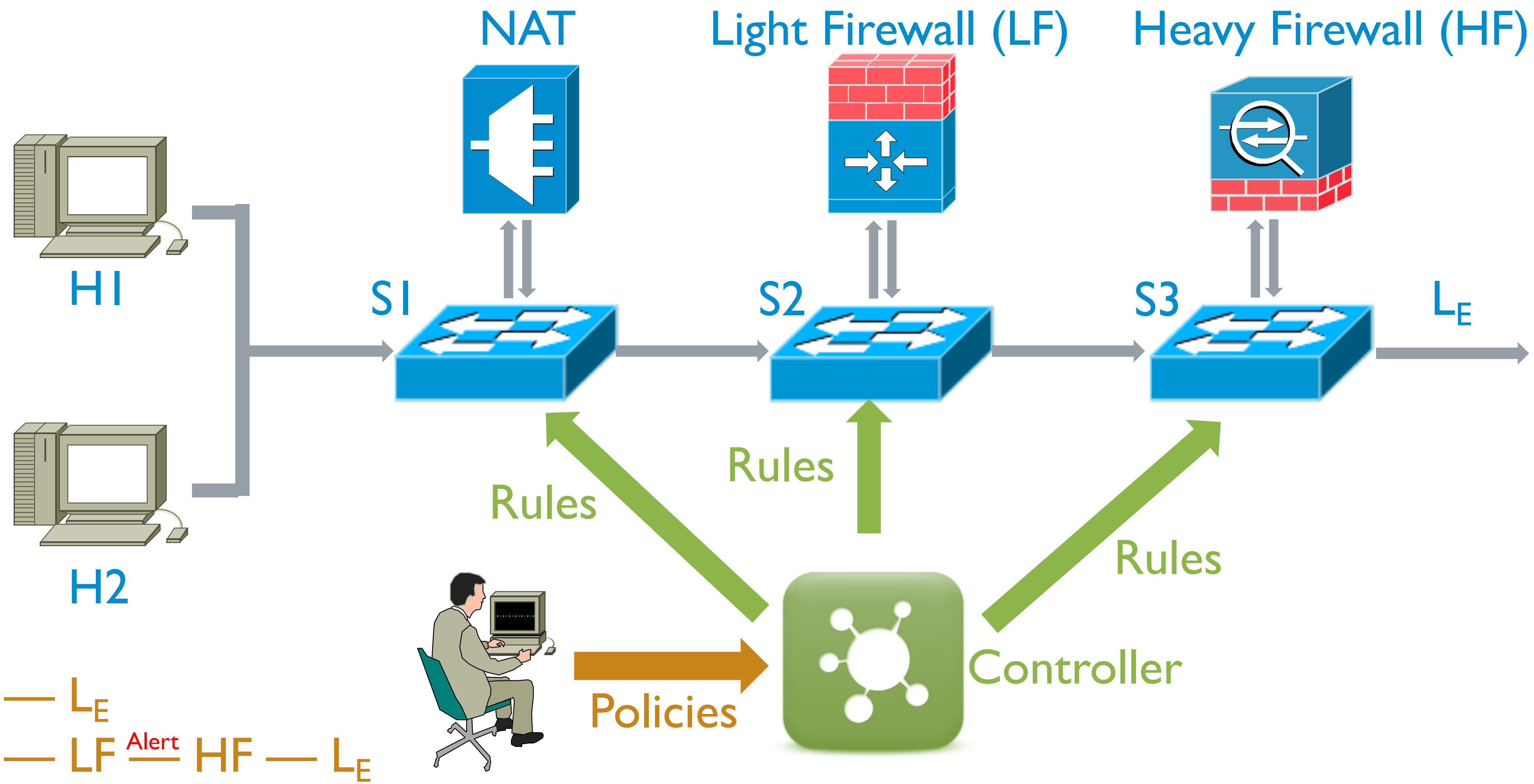
Policies:

(1) H1 — NAT — L_E

(2) H2 — NAT — LF $\overset{\text{Alert}}{—}$ HF — L_E

Forwarding Ambiguity

Middlebox meets SDN



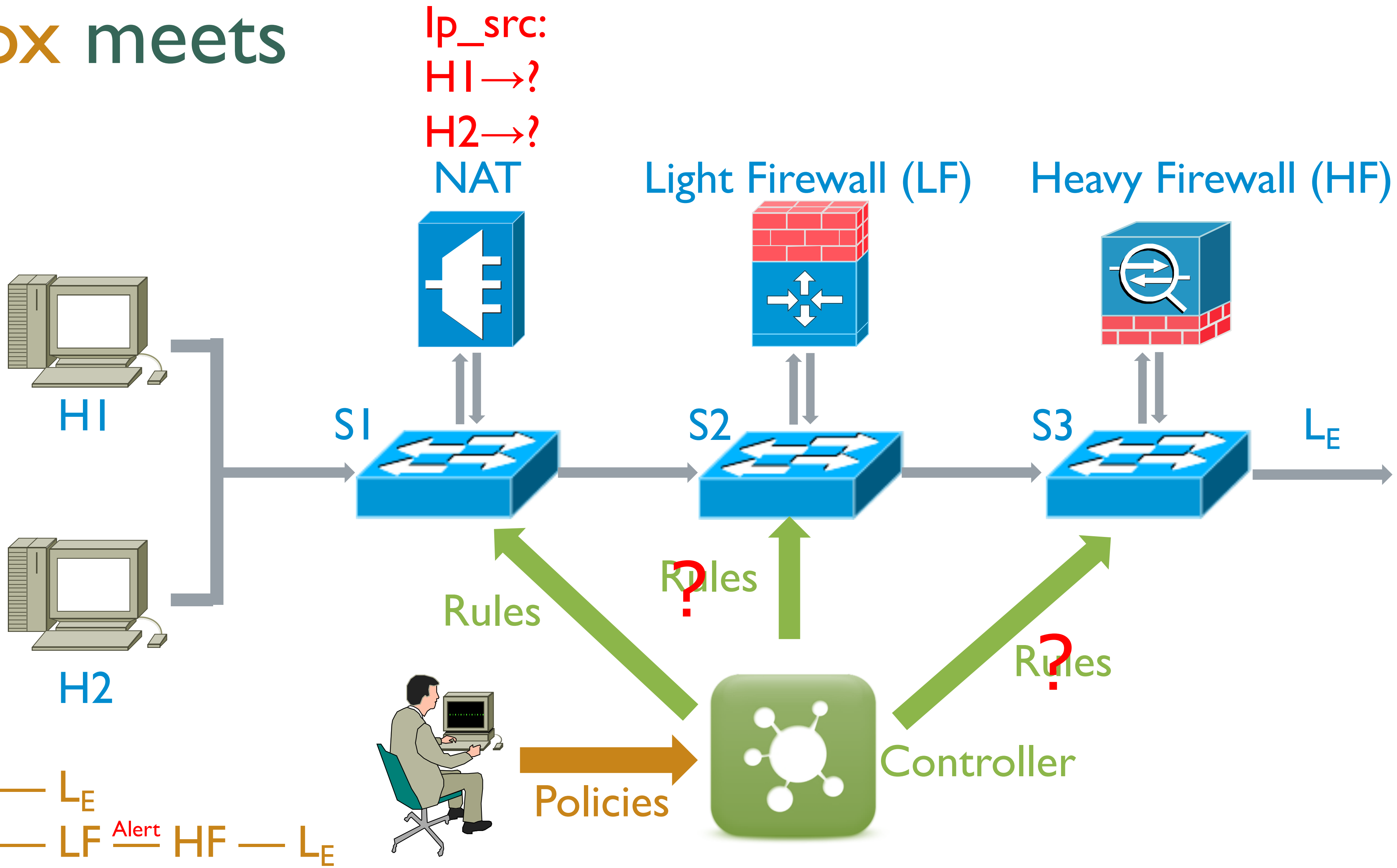
Policies:

(1) H1 — NAT — L_E

(2) H2 — NAT — LF $\overset{\text{Alert}}{—}$ HF — L_E

Forwarding Ambiguity

Middlebox meets SDN



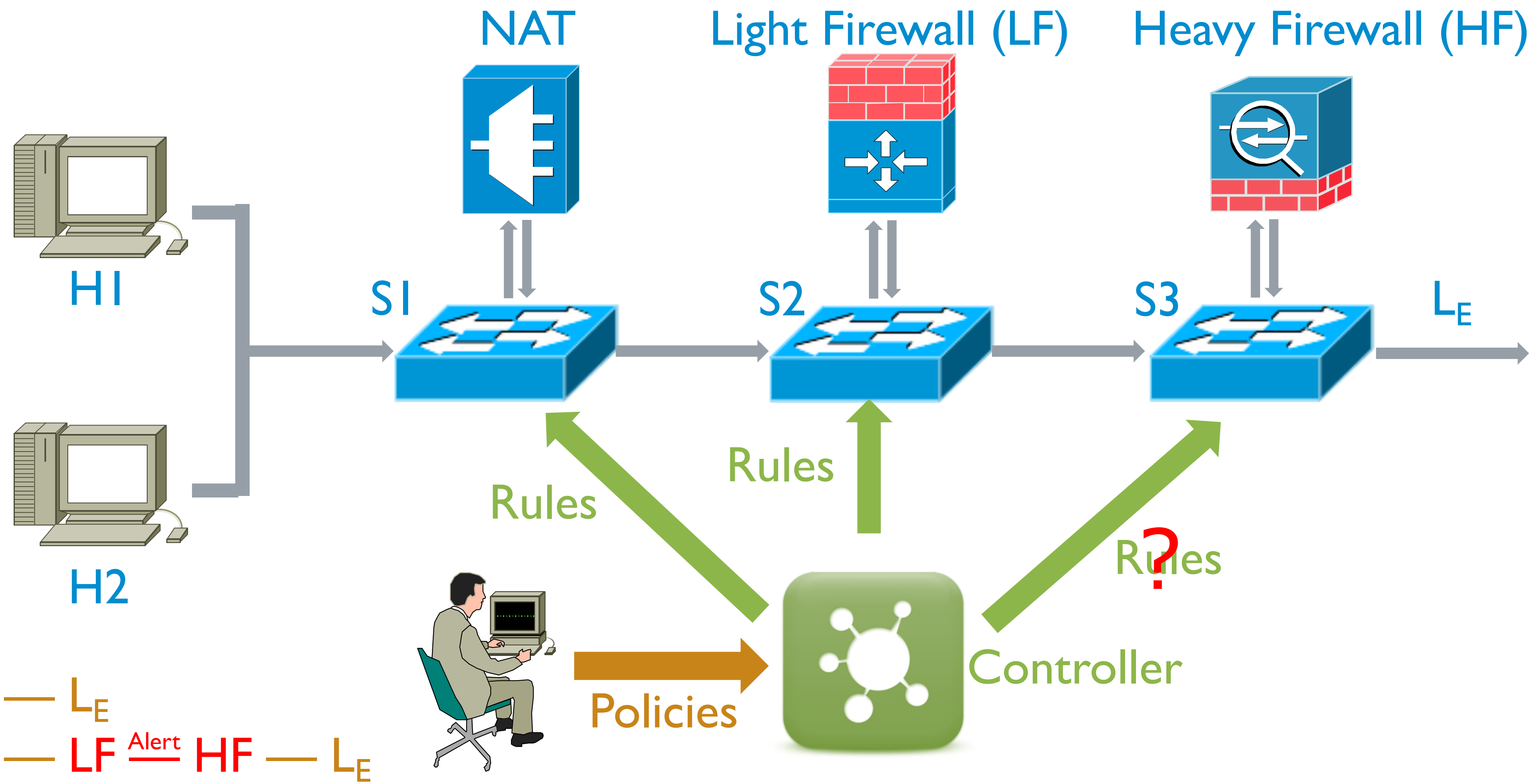
Policies:

(1) H1 — NAT — L_E

(2) H2 — NAT — LF $\overset{\text{Alert}}{—}$ HF — L_E

Forwarding Ambiguity

Middlebox meets SDN



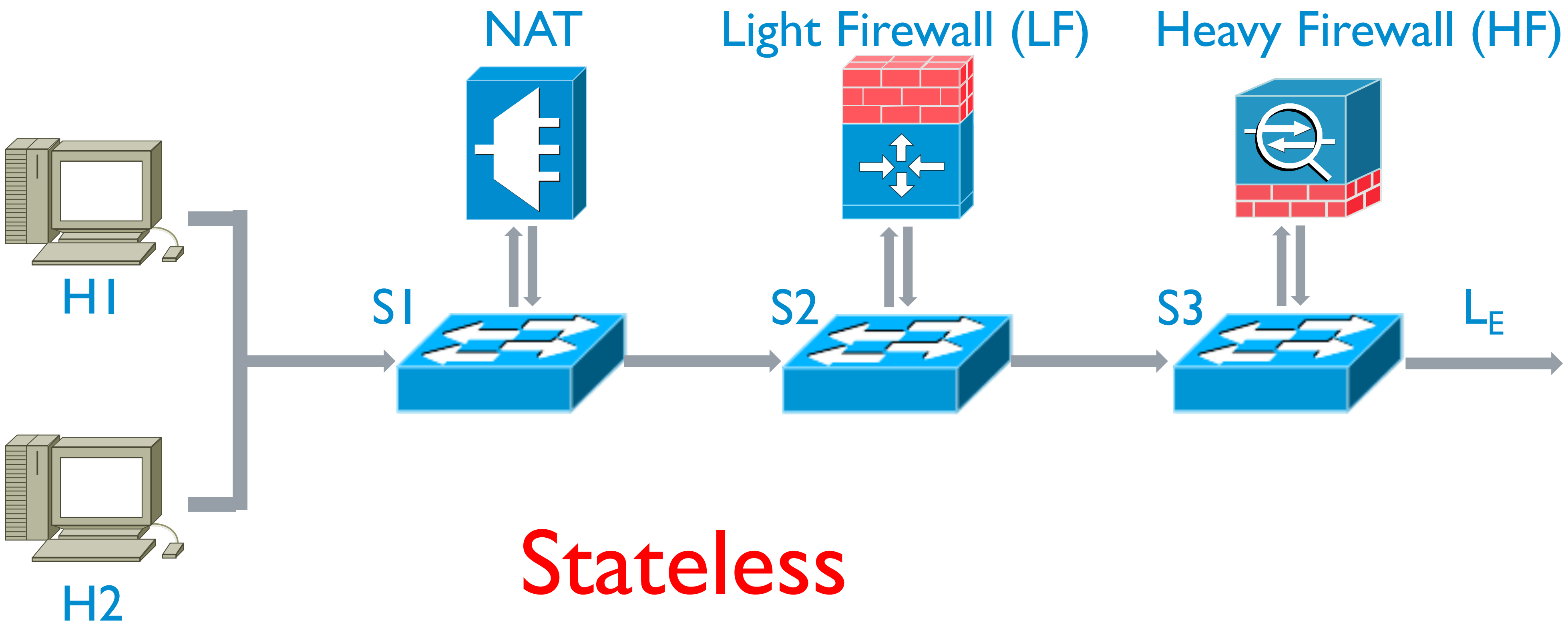
Policies:

(1) H1 — NAT — L_E

(2) H2 — NAT — LF $\xrightarrow{\text{Alert}}$ HF — L_E

Forwarding Ambiguity

Middlebox meets SDN

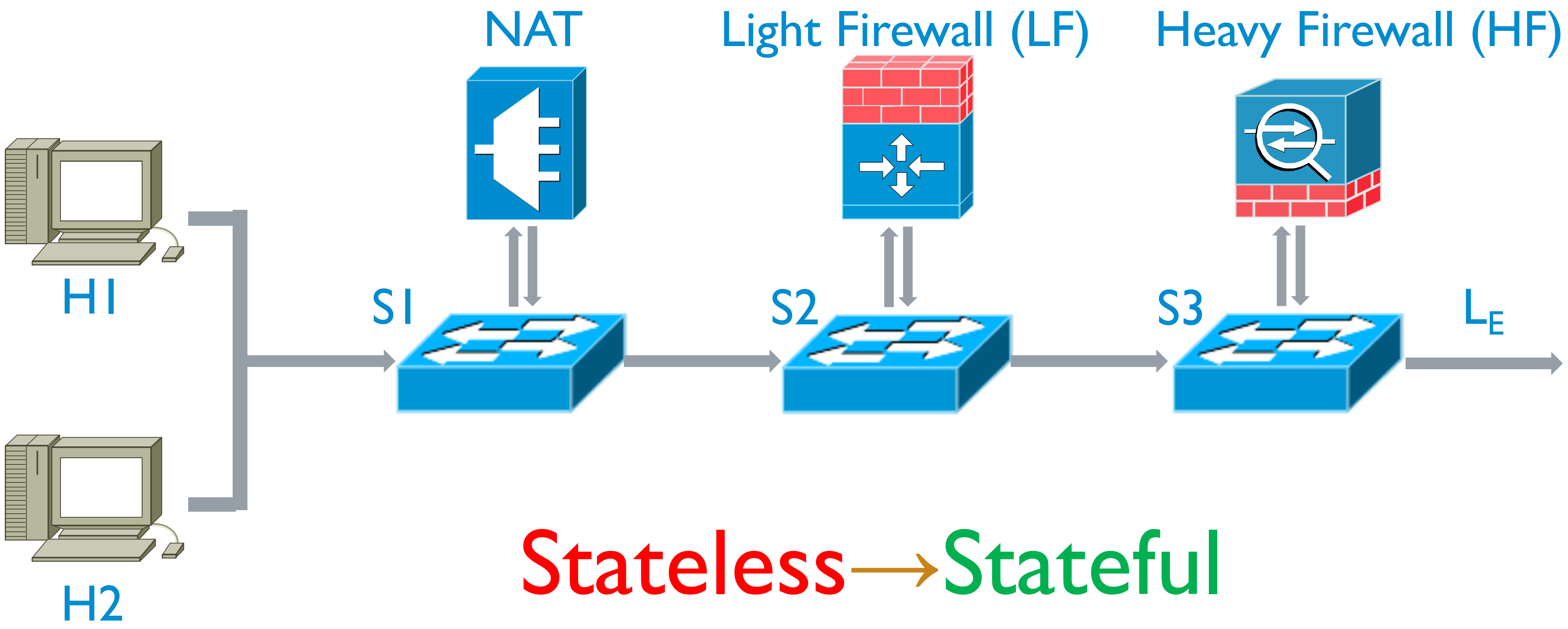


Policies:

(1) H1 — NAT — L_E

(2) H2 — NAT — LF $\overset{\text{Alert}}{—}$ HF — L_E

Middlebox meets SDN



Stateless⟶Stateful

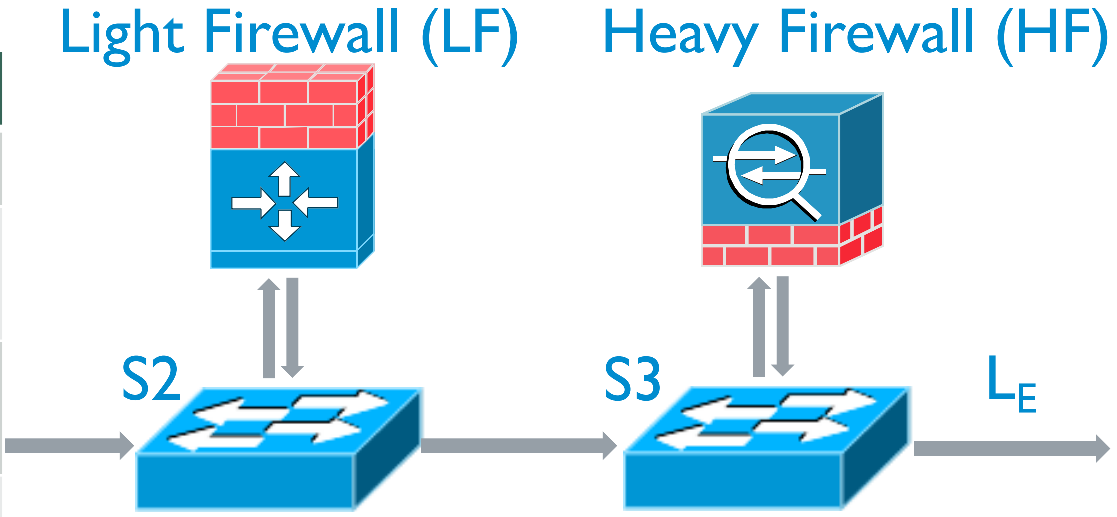
Policies:

(1) H1 — NAT — L_E

(2) H2 — NAT — LF $\overset{\text{Alert}}{—}$ HF — L_E

Middlebox meets SDN

Switch	Some Crucial Rules	
	Matching	Action
S2	tag=<src:H2, NAT>, interface=S2:S1	fwd(LF)
S2	tag=<src:H1,NAT>, interface=S2:S1	fwd(S3)
S3	tag=<src:H2, LF, alert>, interface=S3:S2	fwd(HF)
S3	tag=<src:H2, LF, pass> Interface=S3:S2	fwd(L_E)



Flowtags [NSDI '14]
Stateful Tags on packer header

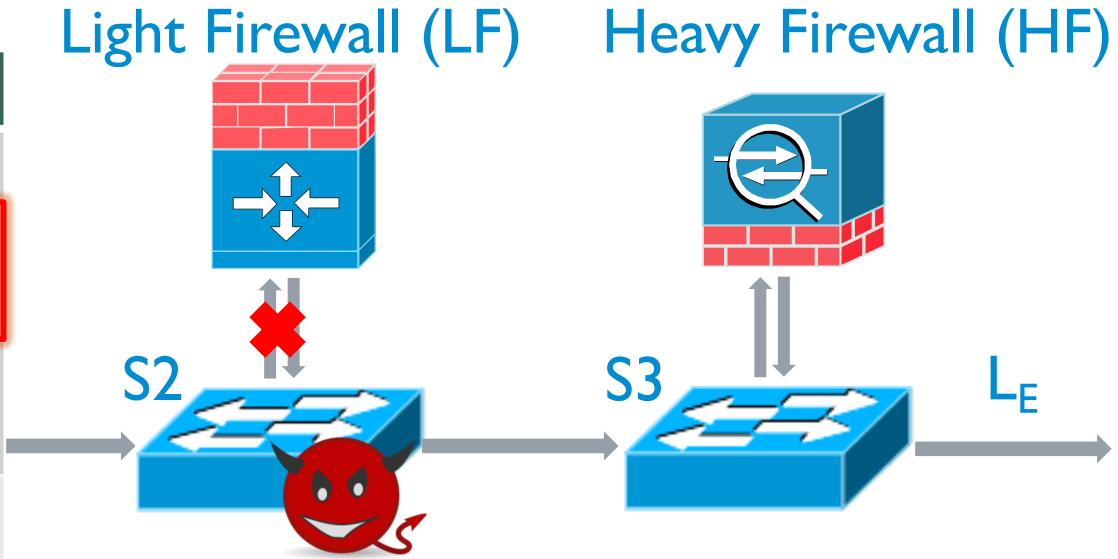
Policies:

(1) H1 — NAT — L_E

(2) H2 — NAT — LF $\overset{\text{Alert}}{—}$ HF — L_E

Middlebox-Bypass Attacks
SDN

Switch	Some Crucial Rules	
	Matching	Action
S2	tag=<src:H2, NAT>, interface=S2:S1	fwd(LF)
S2	tag=<src:H1,NAT>, interface=S2:S1	fwd(S3)
S3	tag=<src:H2, LF, alert>, interface=S3:S2	fwd(HF)
S3	tag=<src:H2, LF, pass> Interface=S3:S2	fwd(L_E)



Policies:

(1) H1 — NAT — L_E

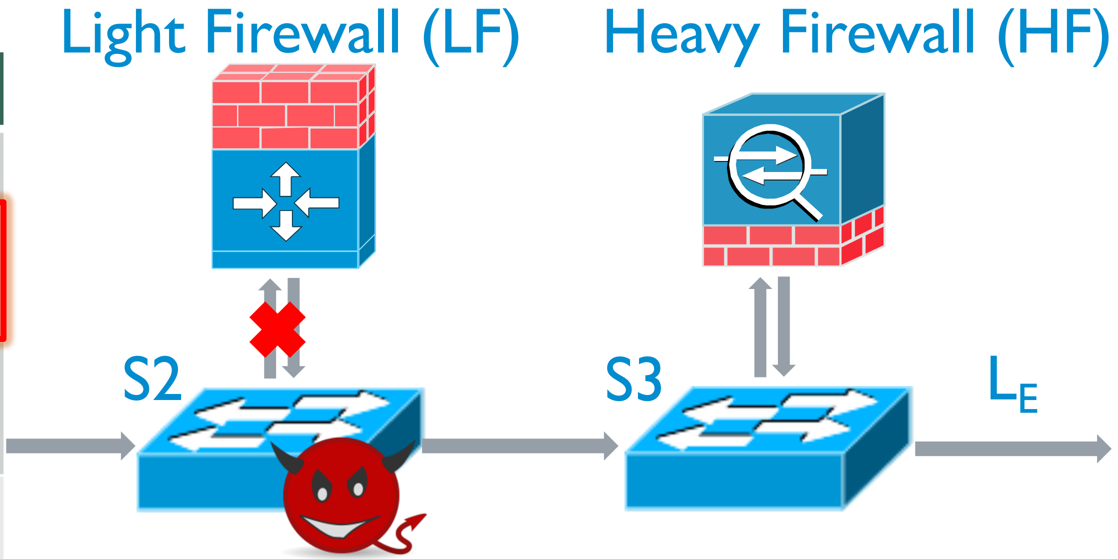
(2) H2 — NAT — LF $\overset{\text{Alert}}{—}$ HF — L_E

Middlebox-Bypass Attacks

NAT

Switch	Some Crucial Rules	
	Matching	Action
S2	tag=<src:H2, NAT>, interface=S2:S1	tag(LF, pass) fwd(HF)
S2	tag=<src:H1,NAT>, interface=S2:S1	fwd(S3)
S3	tag=<src:H2, LF, alert>, interface=S3:S2	fwd(HF)
S3	tag=<src:H2, LF, pass> Interface=S3:S2	fwd(L_E)

H2



Policies:

(1) H1 — NAT — L_E

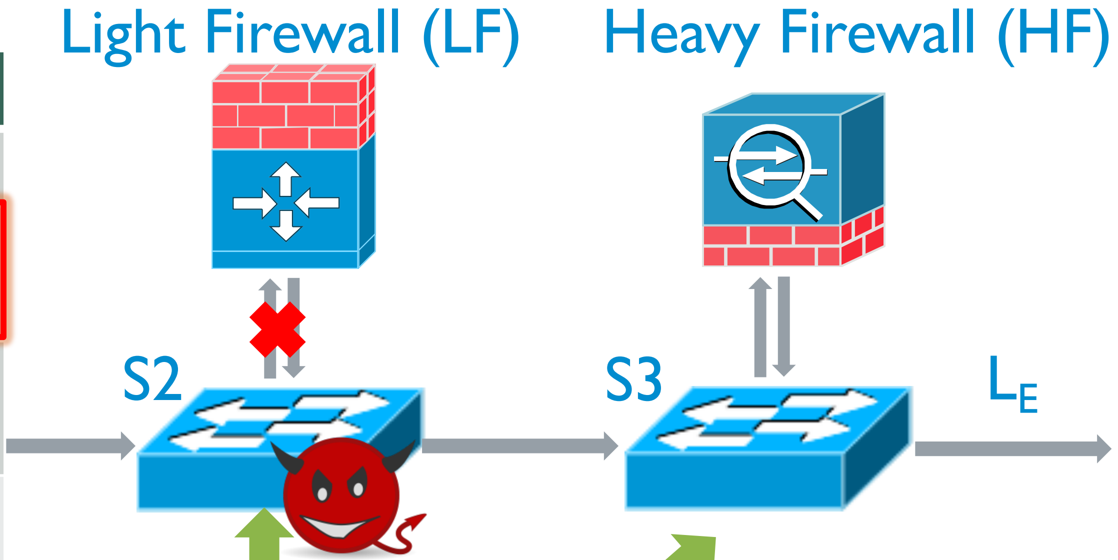
(2) H2 — NAT — LF $\xrightarrow{\text{Alert}}$ HF — L_E

Leads to:

- Severe security breaches
- Performance degradation

Middlebox-Bypass Attacks: More than Hypothesis

Switch	Some Crucial Rules	
	Matching	Action
S2	tag=<src:H2, NAT>, interface=S2:S1	fwd(LF)
S2	tag=<src:H1,NAT>, interface=S2:S1	fwd(S3)
S3	tag=<src:H2, LF, alert>, interface=S3:S2	fwd(HF)
S3	tag=<src:H2, LF, pass> Interface=S3:S2	fwd(L_E)



Policies:

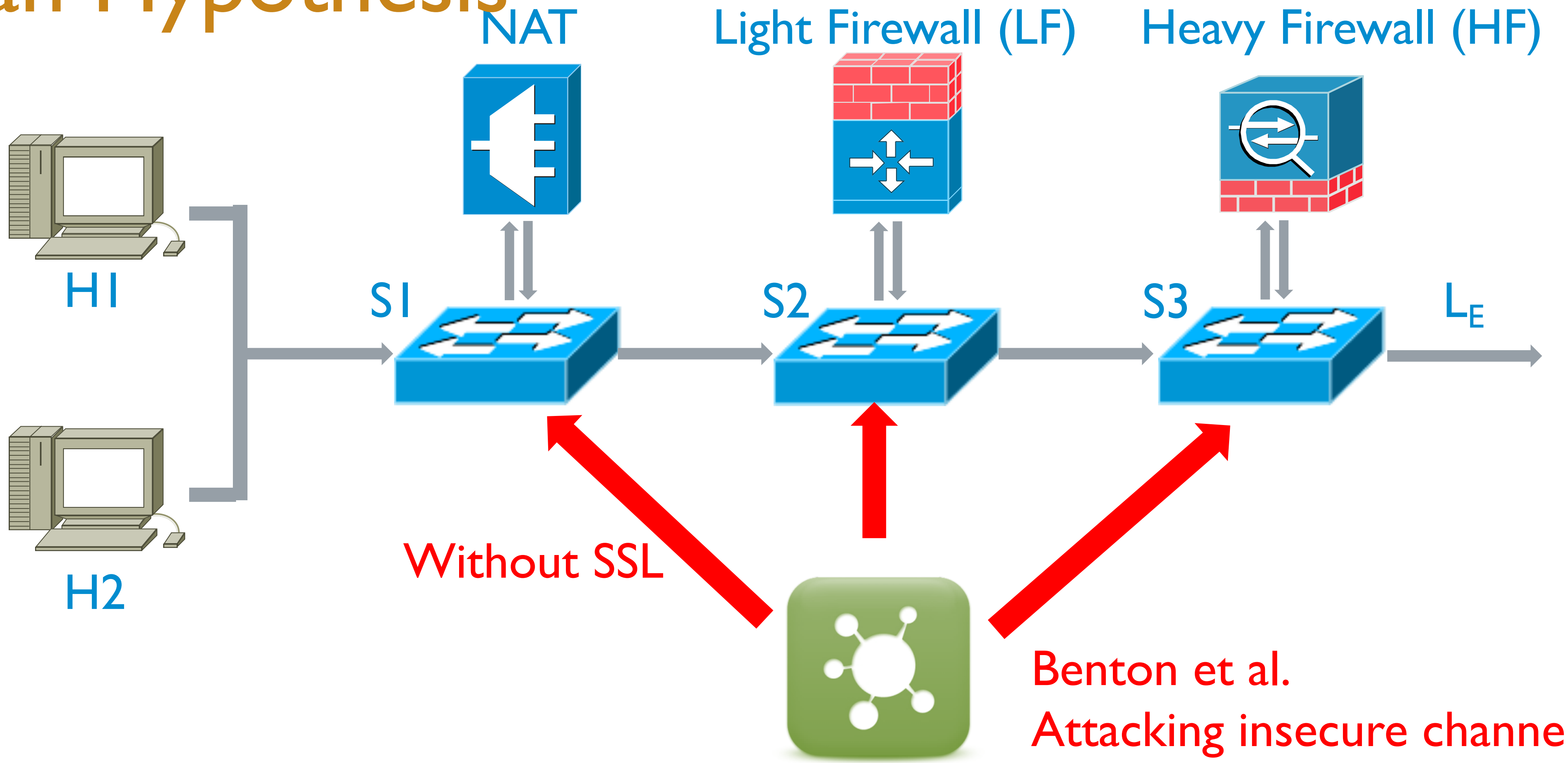
(1) H1 — NAT — L_E

(2) H2 — NAT — LF $\overset{Alert}{—}$ HF — L_E

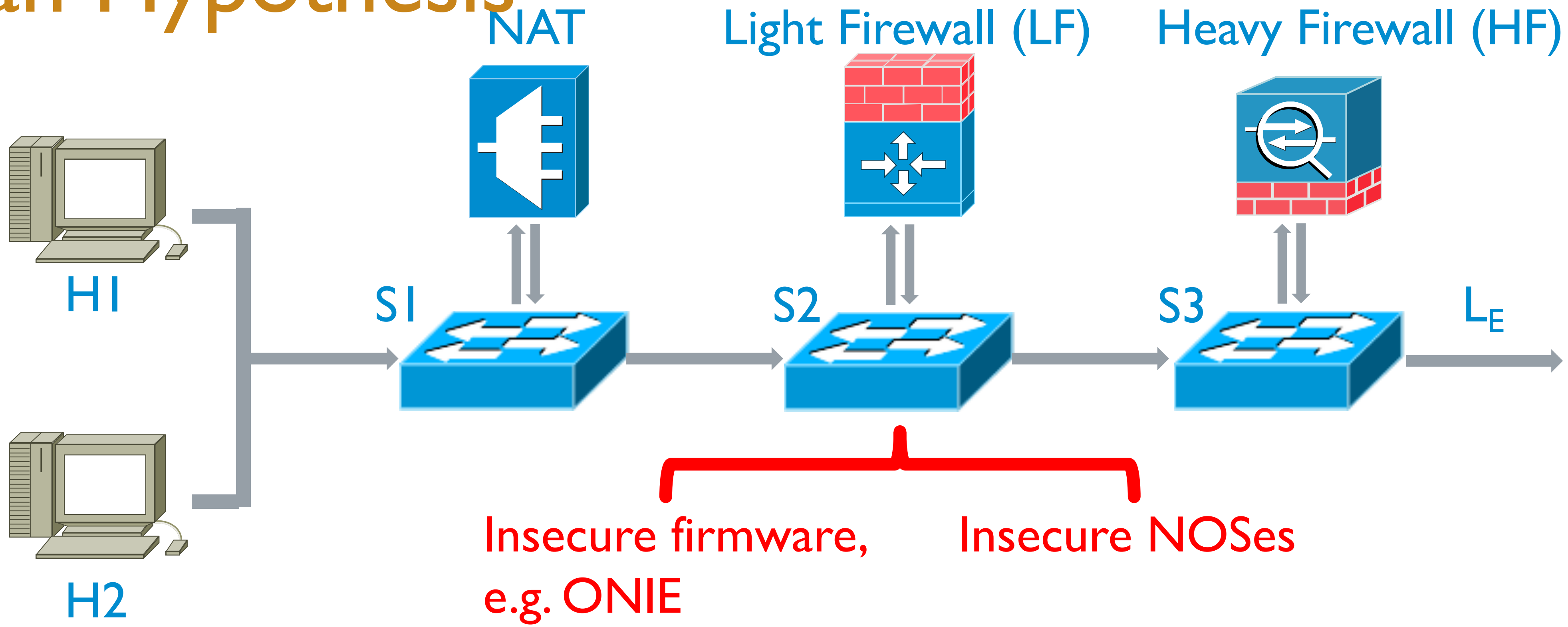
Leads to:

- Security breaches
- Performance degradation

Middlebox-Bypass Attacks: More than Hypothesis



Middlebox-Bypass Attacks: More than Hypothesis



Insecure firmware, e.g. ONIE

Insecure NOSes

Pickett @ DEFCON

Middlebox-Bypass Attacks: Existing malicious switch detection methods

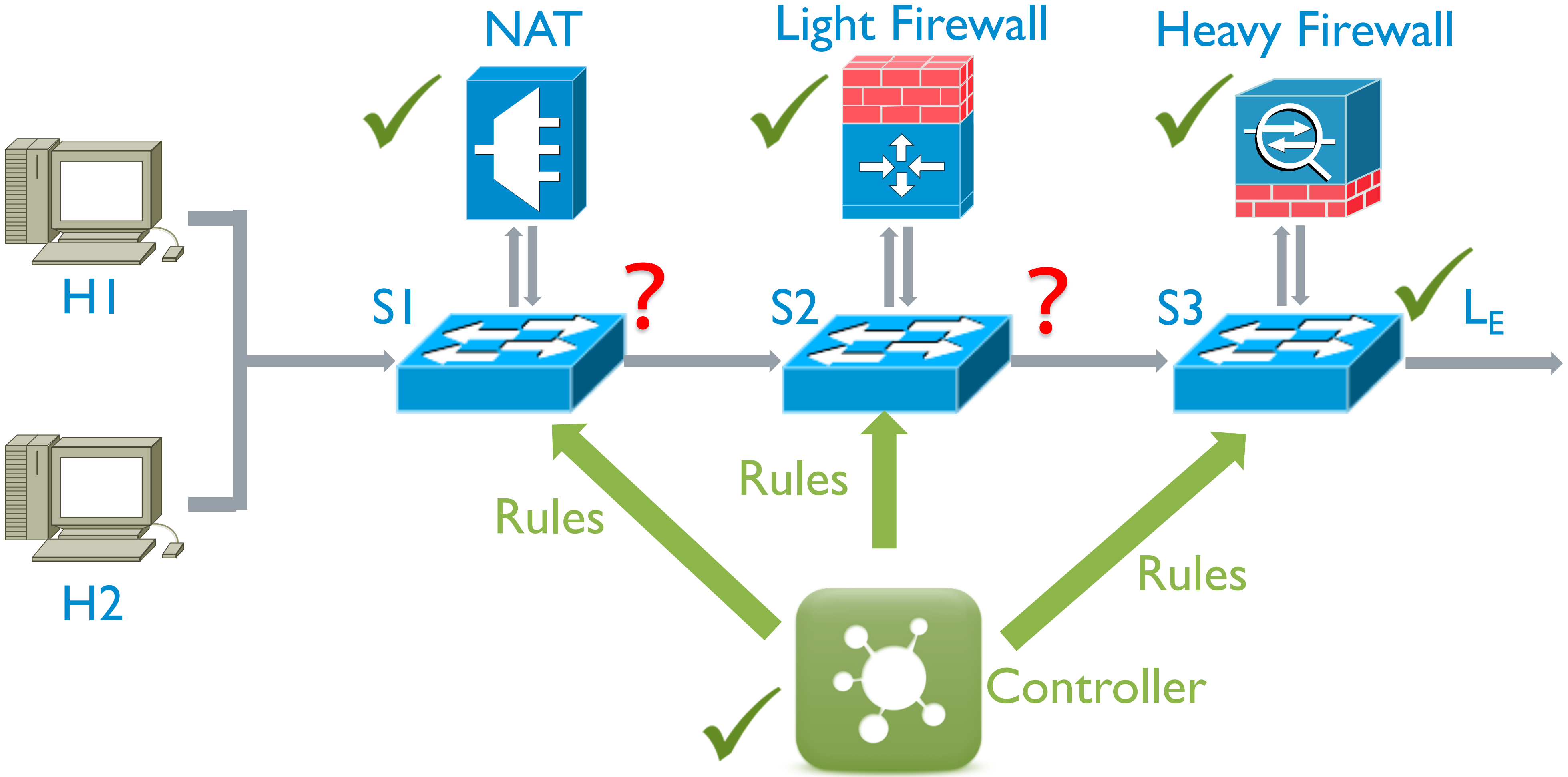
- **Probe-based Methods**
  - Blinded by coward-attack
  - Waste valuable control channel bandwidth
- **Statistics-based Methods**
  - False positive (negative)
  - Waste valuable control channel bandwidth

Middlebox-Bypass Attacks: ~~Existing Secure Methods~~

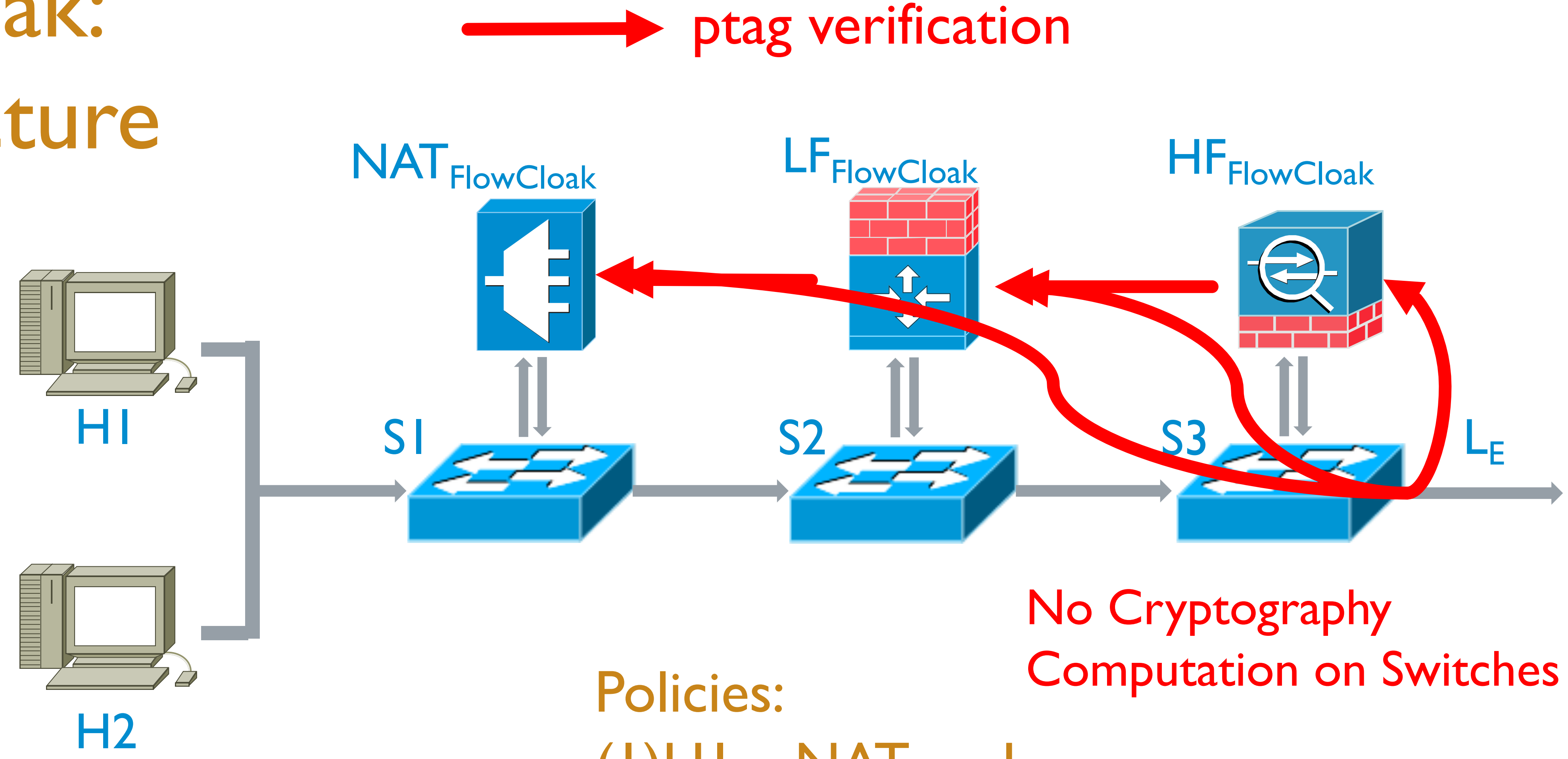
- **Probe-based Methods**
  - Blinded by coward-attack
  - Waste valuable control channel bandwidth
- **Statistics-based Methods**
  - False positive (negative)
  - Waste valuable control channel bandwidth

FlowCloak: Defeating Middlebox-Bypass Attacks in Software-Defined Networking

FlowCloak: Model



FlowCloak: Architecture



No Cryptography Computation on Switches

Policies:

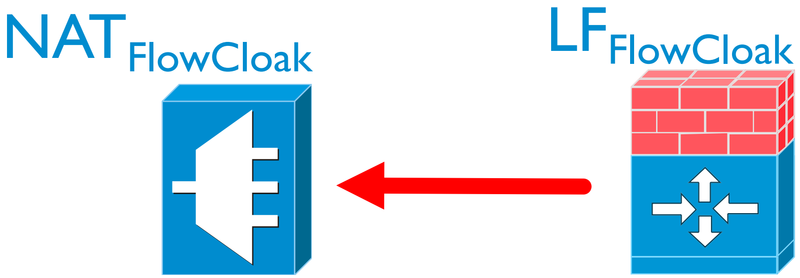
(1)H1—NAT — L_E

(2)H2 — NAT — LF $\xrightarrow{Alert}$ HF —L_E

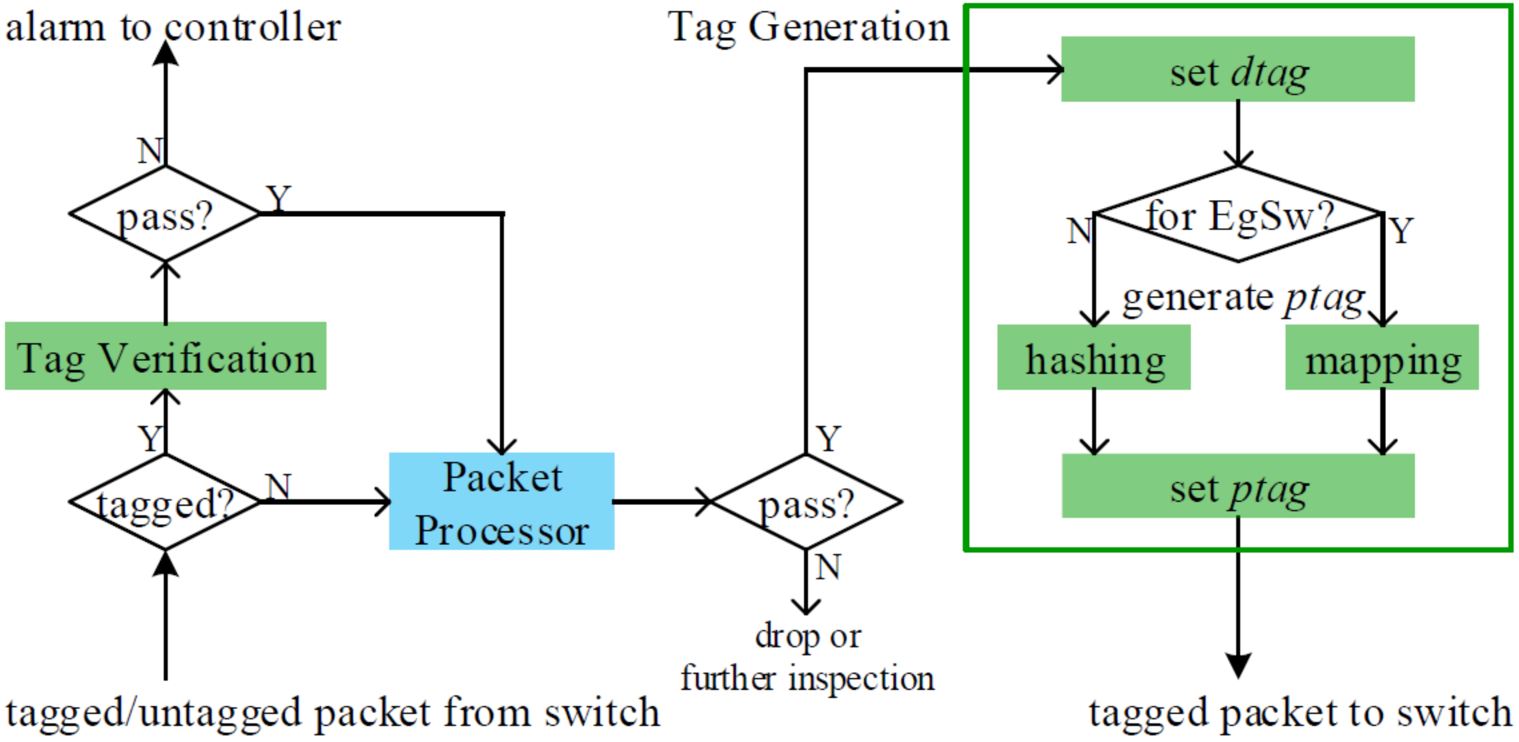
dtag	ptag

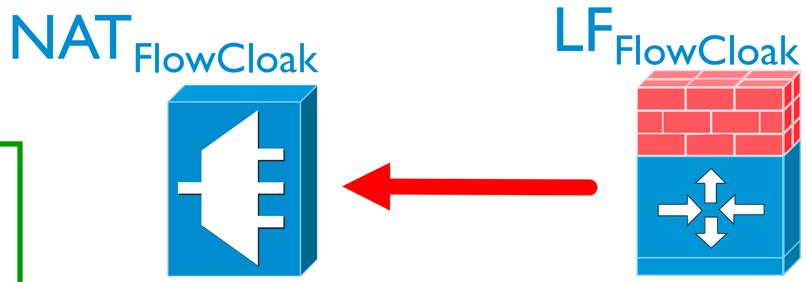
FlowCloak Tag

FlowCloak: Architecture



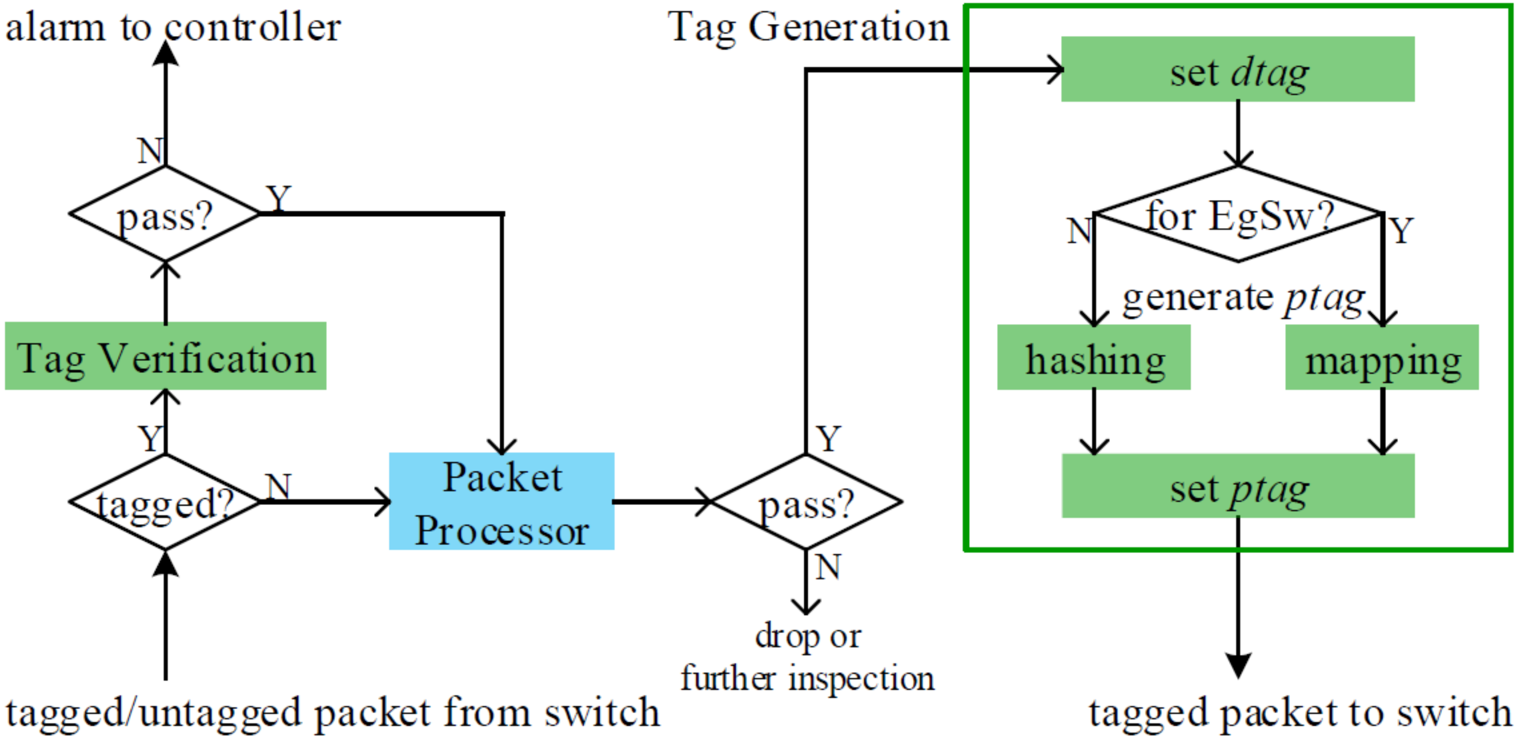
FlowCloak: Middlebox vs. Middlebox



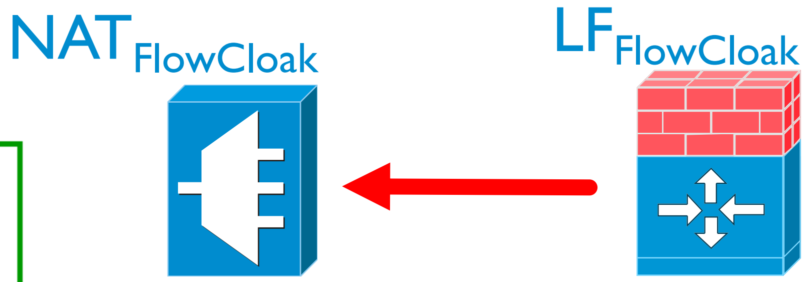


Packet Processing Logic on FC Middleboxes

FlowCloak: Middlebox vs. Middlebox

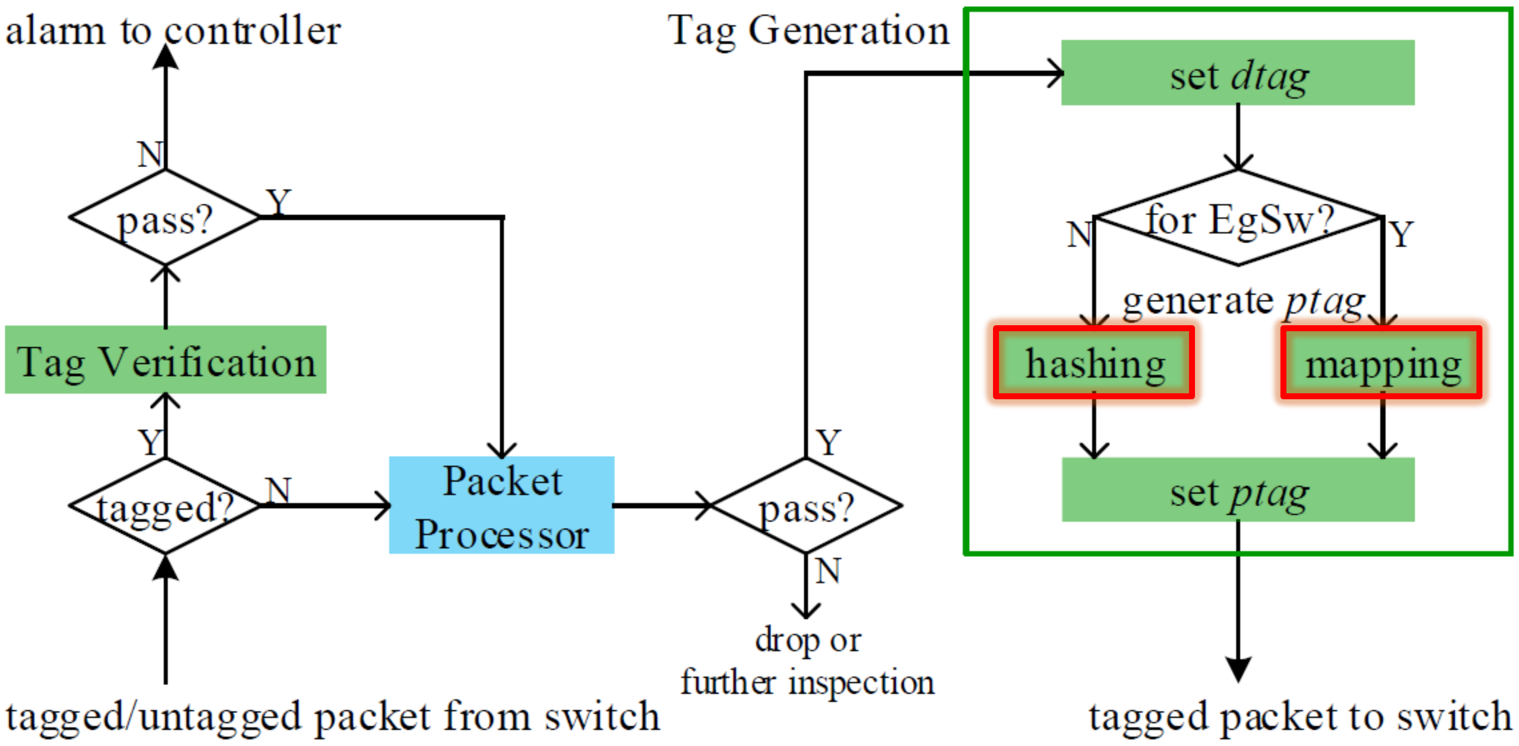


Packet Processing Logic on FC Middleboxes




```
TAGVERIFICATION(P)
  if isexist(P. dtag, dtagmap)  then
     ptag' = Hash(Sample(P. Header))
     if(ptag' == P.Header.ptag)
        return TRUE
     return FALSE
TAGVERIFICATION ends
```

FlowCloak: Middlebox vs. Middlebox



Packet Processing Logic on FC Middleboxes

```
TAGGENERATION(P)
   if next_dev(P) ==
DEV.MIDDLEBOX then
      dtag = flowtags(P, self.ID,
Controller)
      writedtag(P, dtag)
      ptag = Hash(Sample(P. Header))
      writeptag(P, ptag)
   else
      ptag = Map(Sample(P. Header))
TAGGENERATION ends
```

FlowCloak: Middlebox vs. Switch

No cryptography computation: Simulating the hashing function using only match-forward rules

Egress Switch Rules	
Matching	Action
P.SampleDomain=0 && P.Header.ptag=1	forward
P.SampleDomain=1 && P.Header.ptag=0	forward

Hash(b)=~b:
Hash(0)=1
Hash(1)=0

FlowCloak: Middlebox vs. Switch

No cryptography computation: Simulating the hashing function using only match-forward rules

Satisfying Security means Sufficient Rules

Egress Switch Rules	
Matching	Action
P.SampleDomain=0 && P.Header.ptag=1	forward
P.SampleDomain=1 && P.Header.ptag=0	forward

Hash(b)=~b:
Hash(0)=1
Hash(1)=0

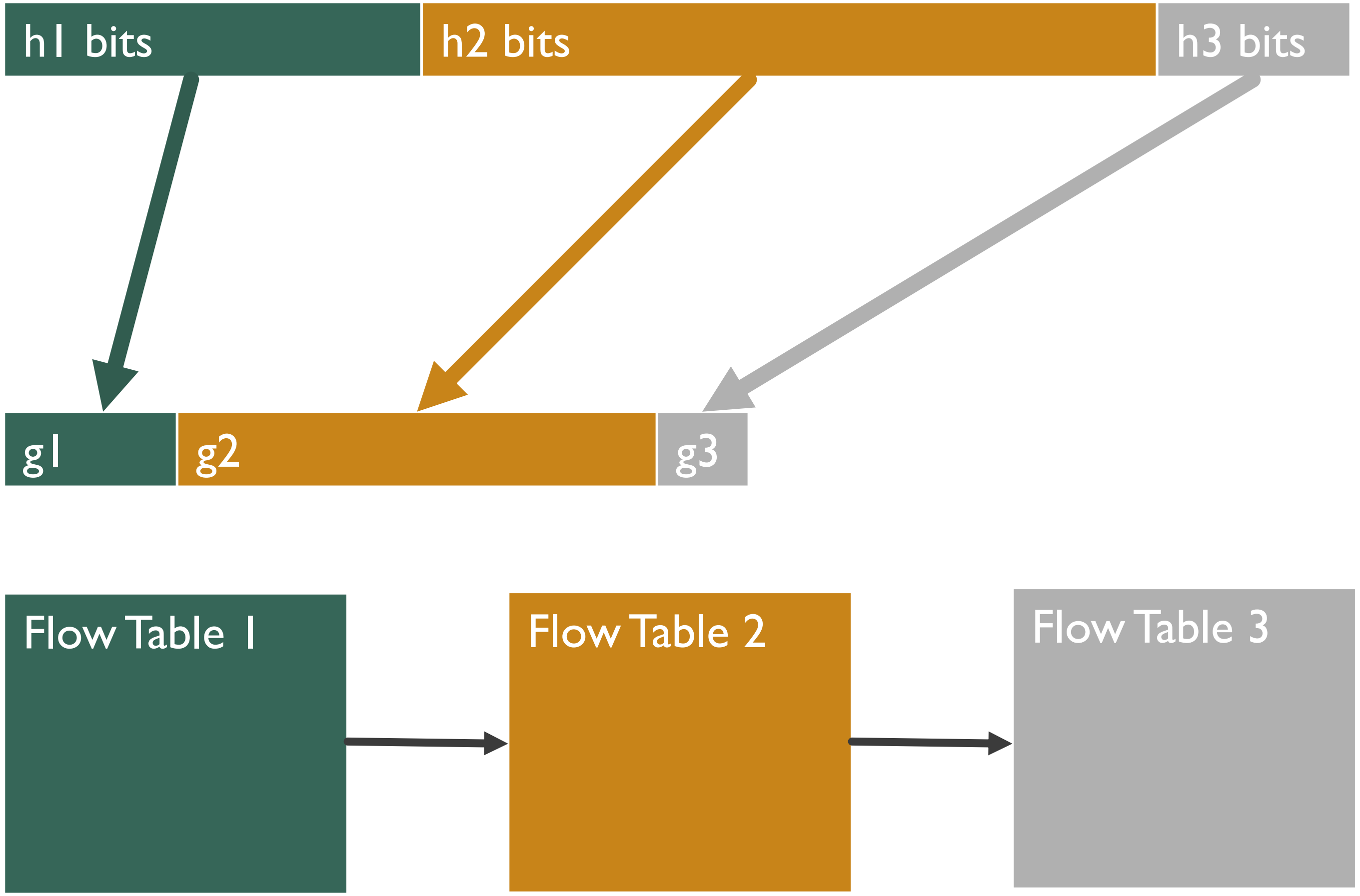
FlowCloak:
Middlebox vs. Switch

Length(P.SampleDomain)=1
2 rules;
…
Length(P.SampleDomain)=n
2^n rules;
Too many rules for limited TCAM capacity

Egress Switch Rules	
Matching	Action
P.SampleDomain=0 && P.Header.ptag=1	forward
P.SampleDomain=1 && P.Header.ptag=0	forward

Hash(b)=~b:
Hash(0)=1
Hash(1)=0

FlowCloak: Middlebox vs. Switch

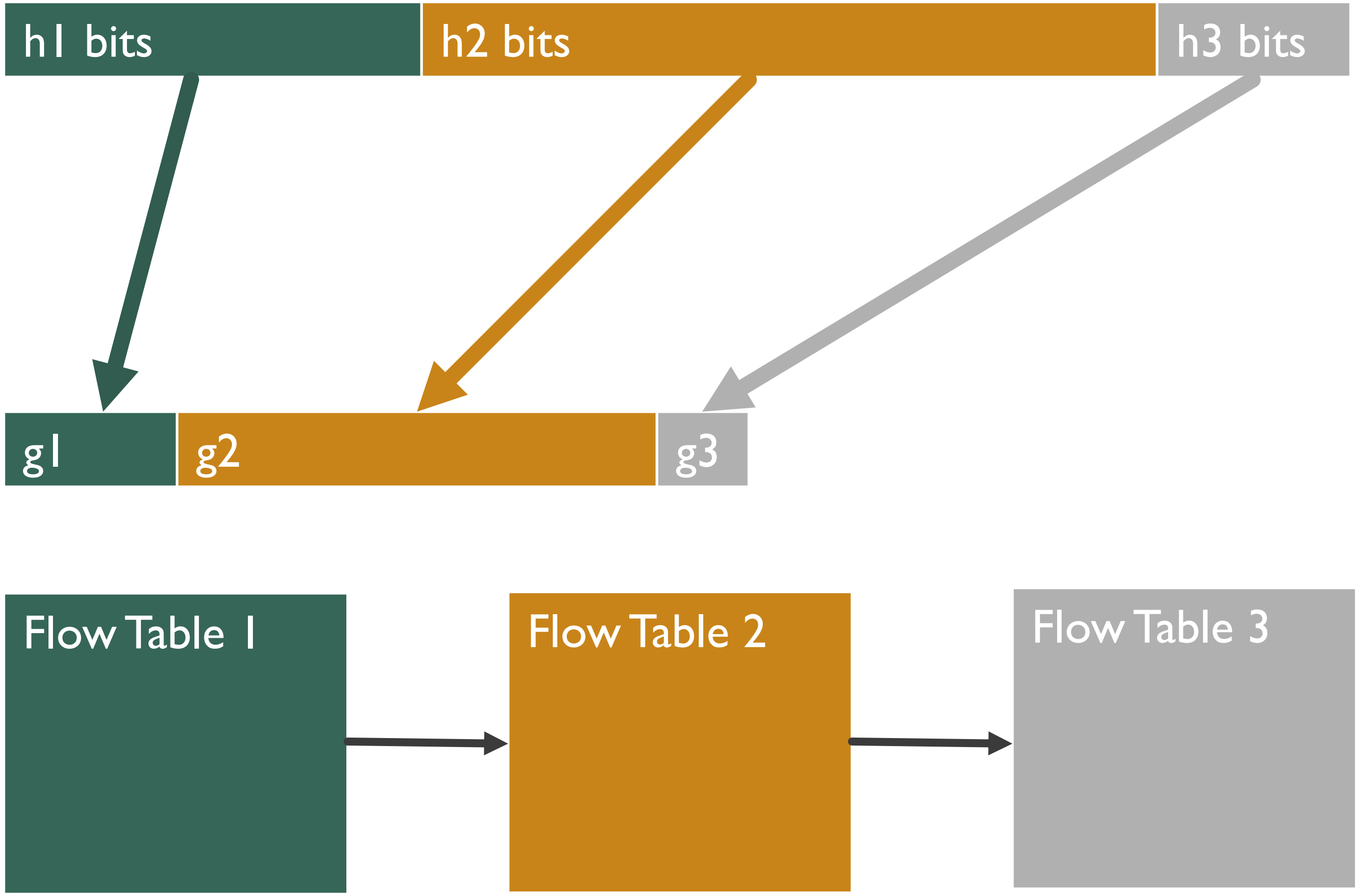


Multi-tag technology

Middlebox Side:
Multi-tag generation based on parallel generation and hashing table.

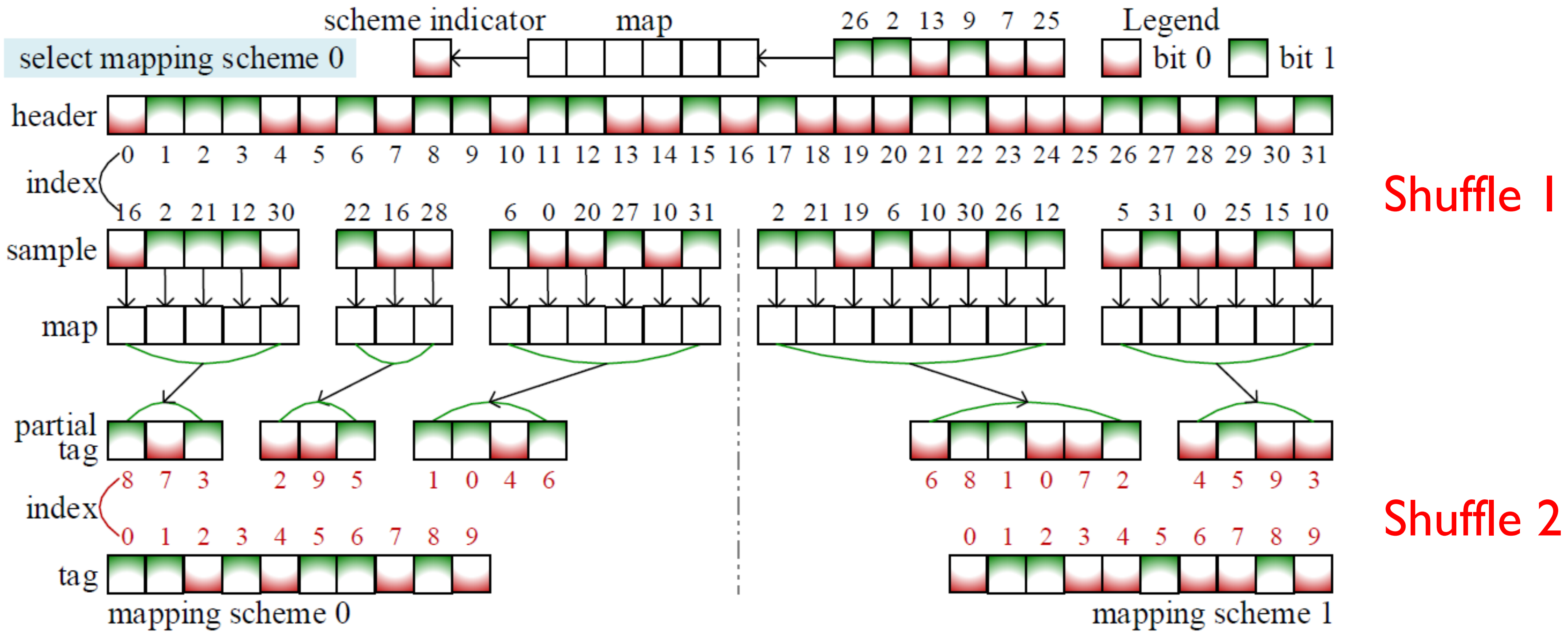
Switch Side:
Multi-tag verification using only $\sum_{i=1}^{n} 2^\wedge hi$ rules rather than $\prod_{i=1}^{n} 2^\wedge hi$ rules

FlowCloak:
Middlebox vs. Switch



Caveat:
Each tag becomes shorter
→Attacking each part
becomes easier?

FlowCloak: Middlebox vs. Switch

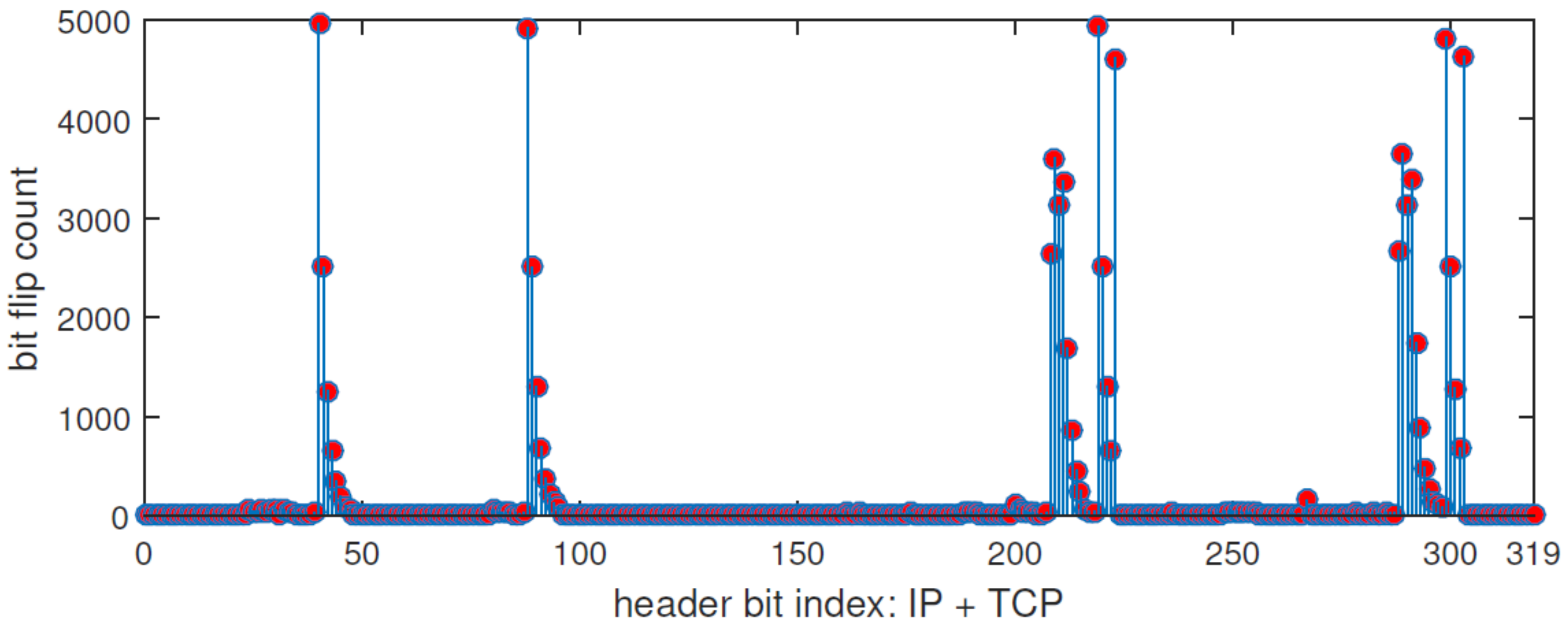


More sophisticated mapping:
multiple mapping schemes + nonconsecutive sample bits + double shuffle

FlowCloak: Evaluation -- Environment

- Middlebox: Snort, 387/29300 lines of C code modified
- SDN: OpenDayLight Carbon as Controller
  OVS v2.5.3 as switches
  Mininet for network simulation
- Hardware: Each Snort instance is assigned with
  8GB memory and 2 2.3GHz(E5-2670 v3) CPUs

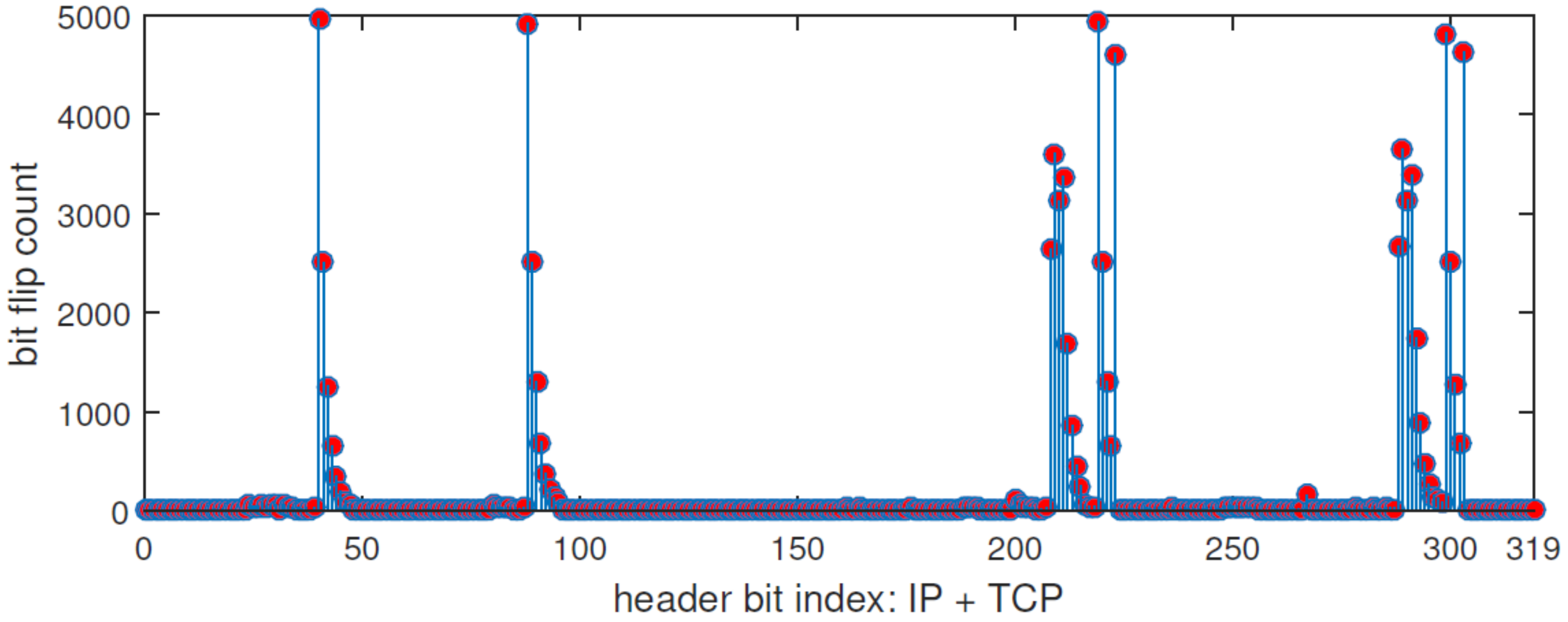
FlowCloak: Evaluation -- Feasibility



packet header dynamics in 5000 sniffed co-flow packets

Is there sufficient diversity in packet headers?

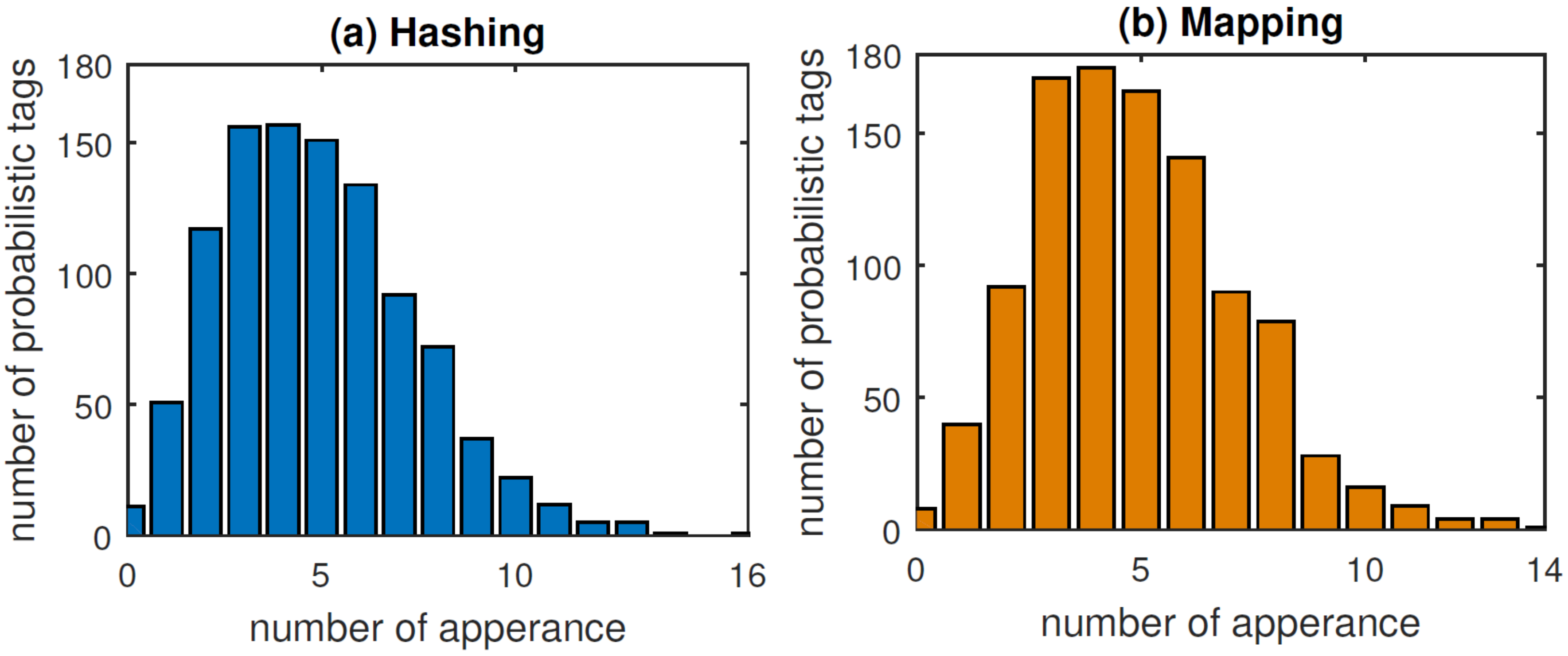
FlowCloak: Evaluation -- Feasibility



packet header dynamics in 5000 sniffed co-flow packets

Fortunately, we get enough dynamics in packet headers.

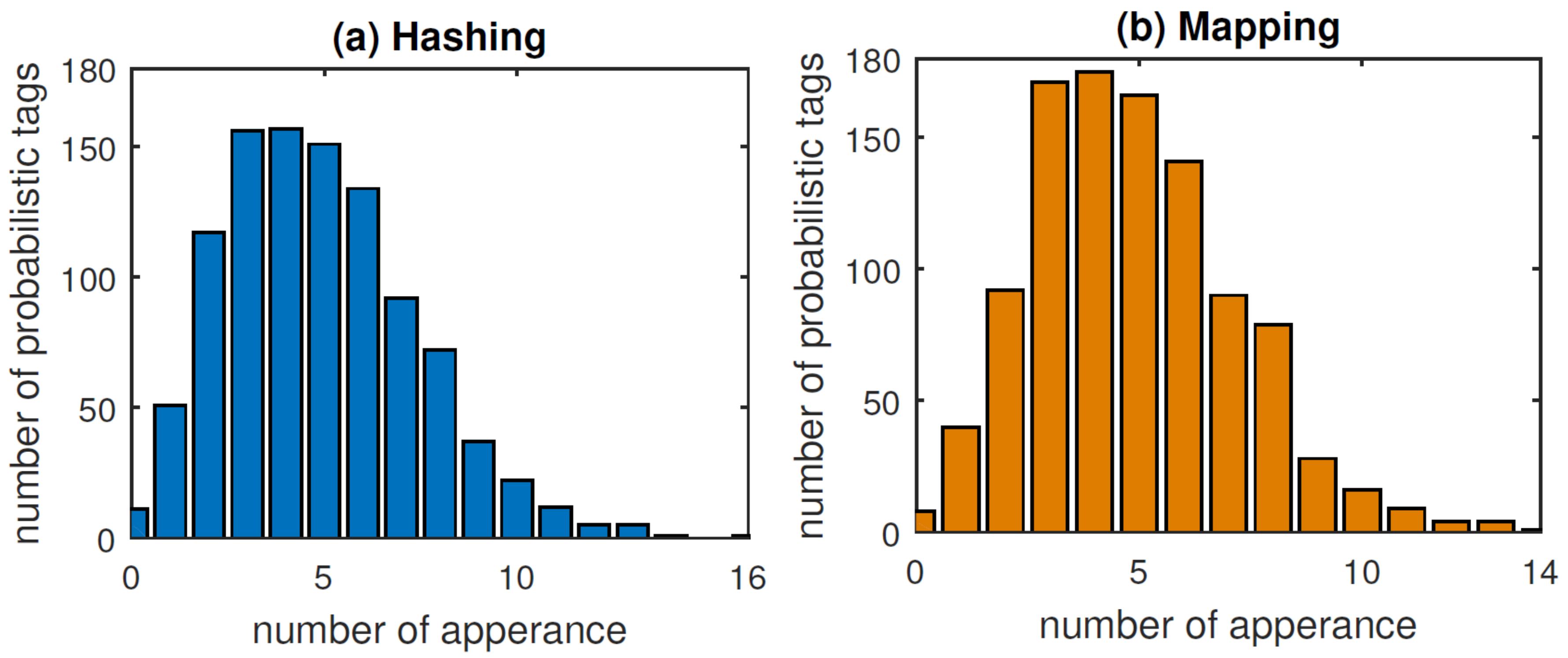
FlowCloak: Evaluation -- Robustness



ptag distribution under hashing and mapping

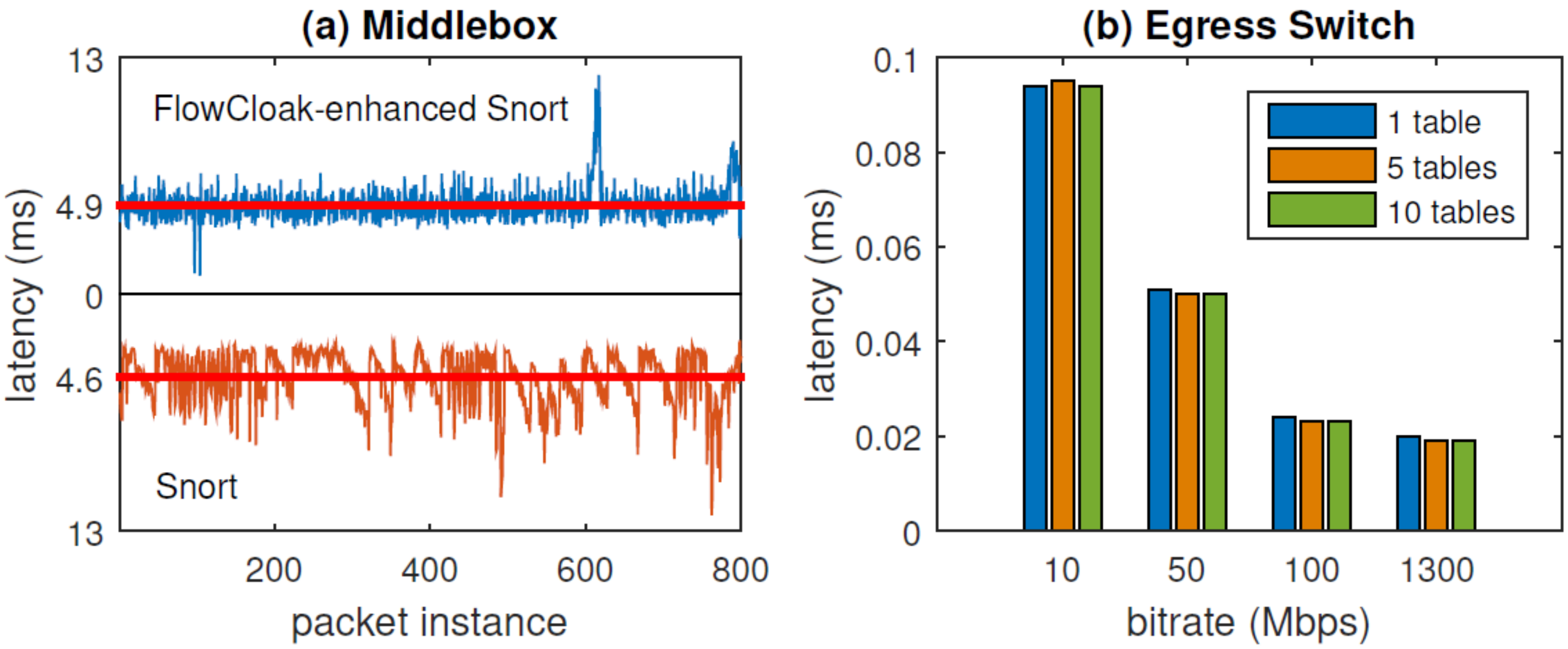
Can attackers find any pattern in ptag?

FlowCloak: Evaluation -- Robustness



ptag distribution under hashing and mapping
Both approximate binomial distribution

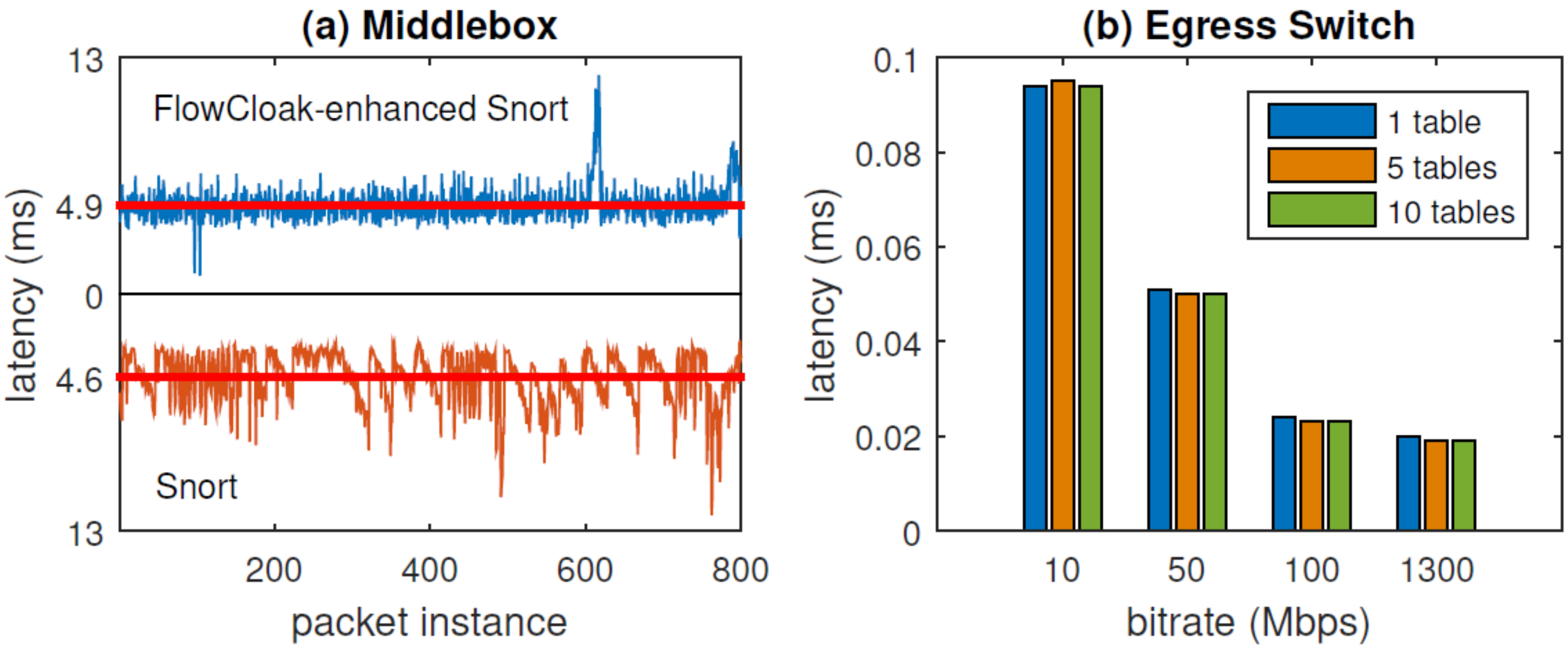
FlowCloak: Evaluation -- Efficiency



overhead of FlowCloak

Is performance degradation acceptable?

FlowCloak: Evaluation -- Efficiency



overhead of FlowCloak

Latency induced by FlowCloak on Middlebox: 0.3 ms

Latency induced by multiple flow tables: no obvious delay

FlowCloak:

FlowCloak: Defeating Middlebox-Bypass Attacks in Software-Defined Networking

FlowCloak: Defeating Middlebox-Bypass Attacks in Software-Defined Networking

Middlebox meets SDN

↓

Middlebox-bypass attacks

↓

FlowCloak & Multi-tag technology

↓

Efficient, Accurate & Robust

?

Thank You

ytyang@zju.edu.cn