Artifact • AEC • Evaluated • PLDI •
Consistent • Complete • Well Documented • Easy to Reuse •

Temporal NetKAT

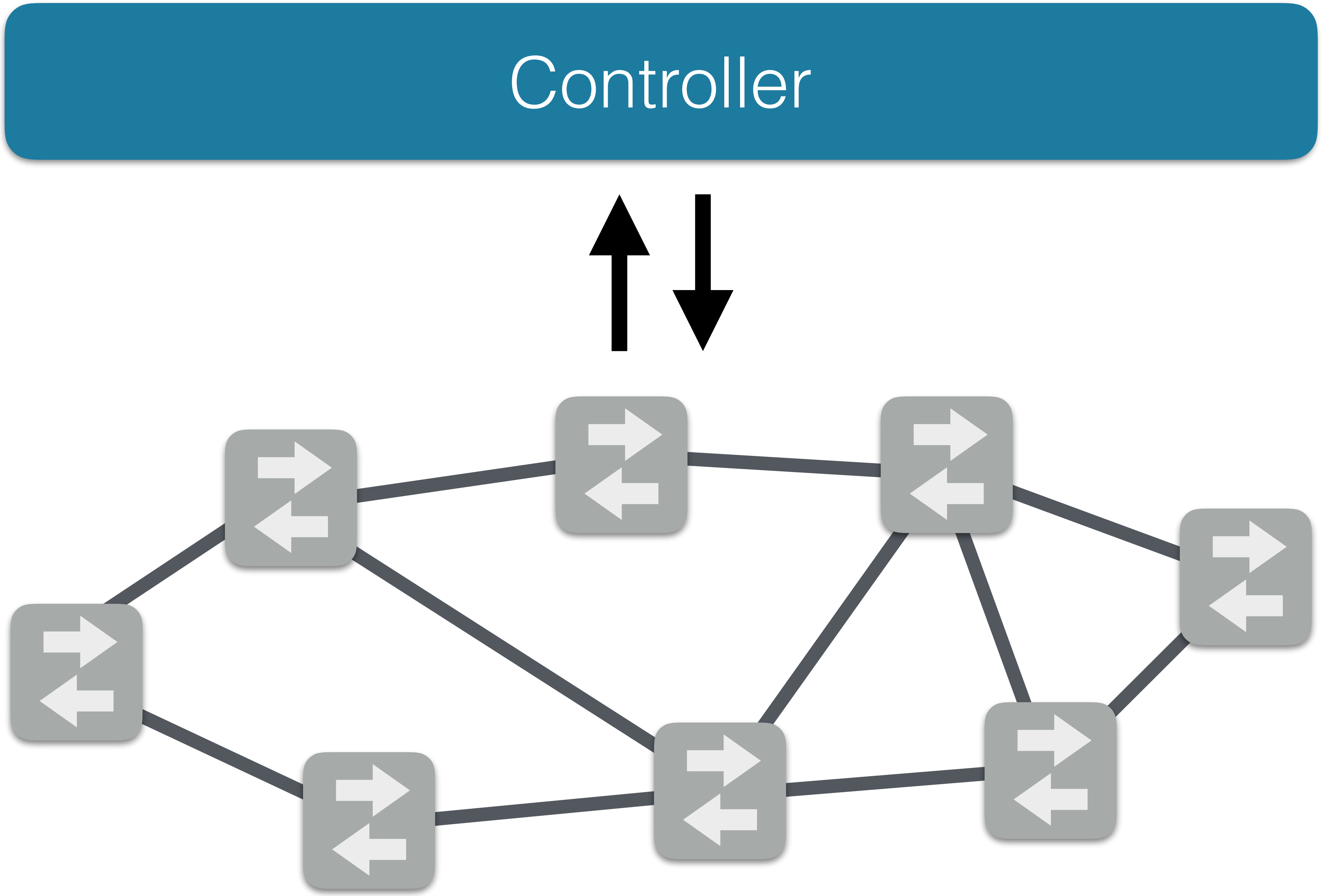
Ryan Beckett
Michael Greenberg*, David Walker

Princeton University

Pomona College*

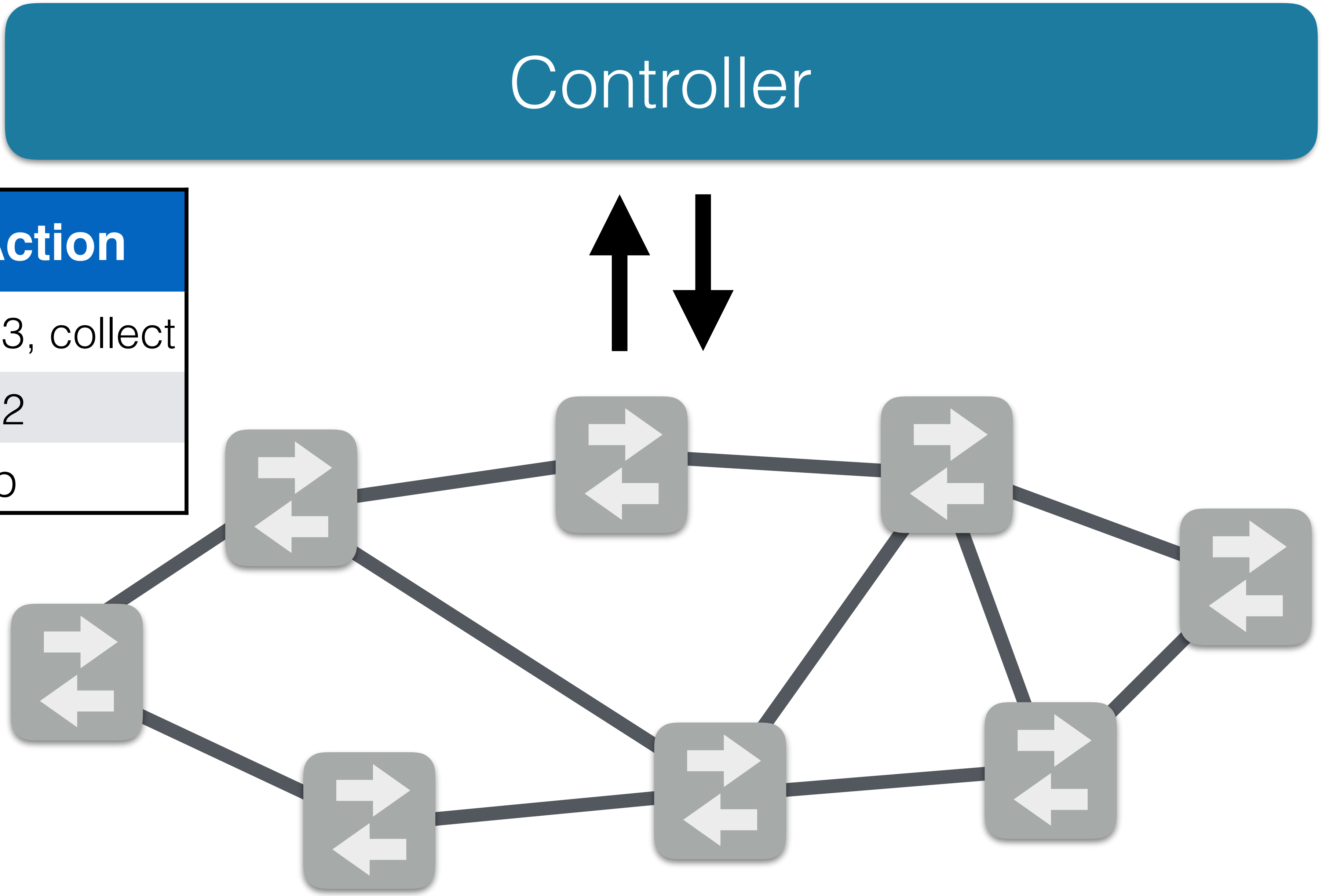
Software-Defined Networking

Google Microsoft AT&T

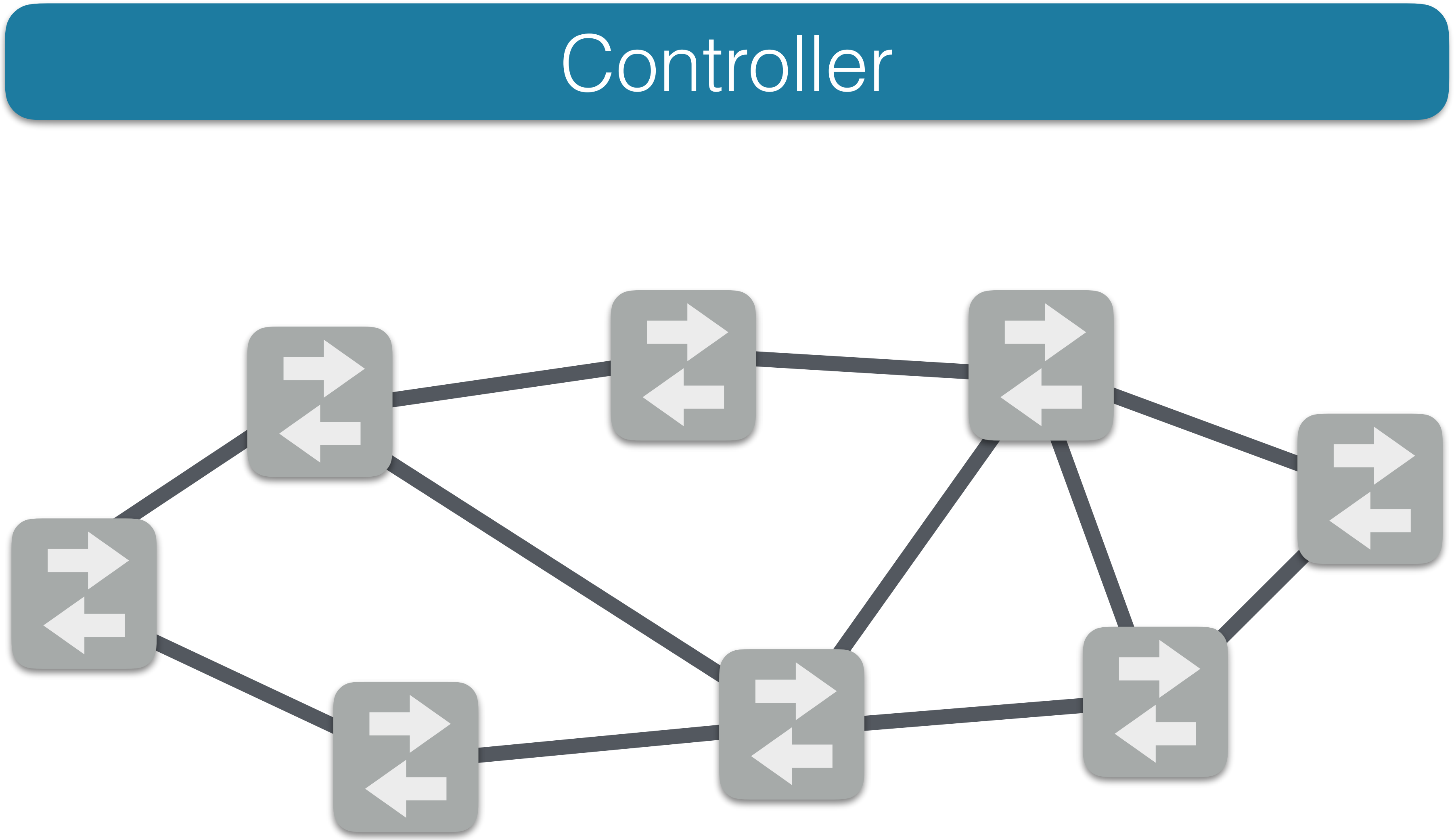


Software-Defined Networking

Google Microsoft AT&T

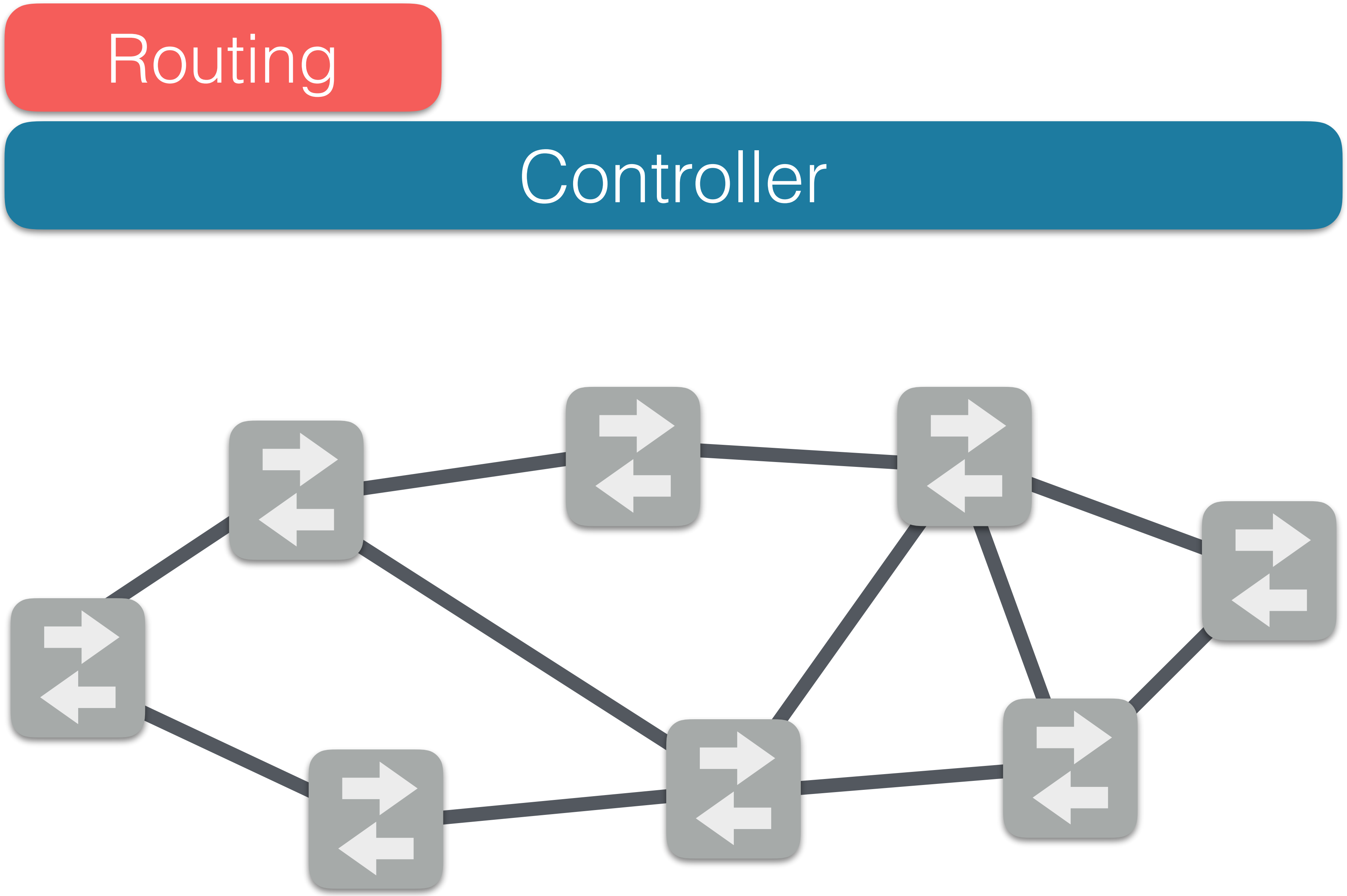


Match	Action
dst=1.2.3.4	pt←3, collect
src=5.6.7.8	pt←2
*	drop



Controller

L
Maple
FlowLog
Frenetic
NetKAT



Routing

Controller

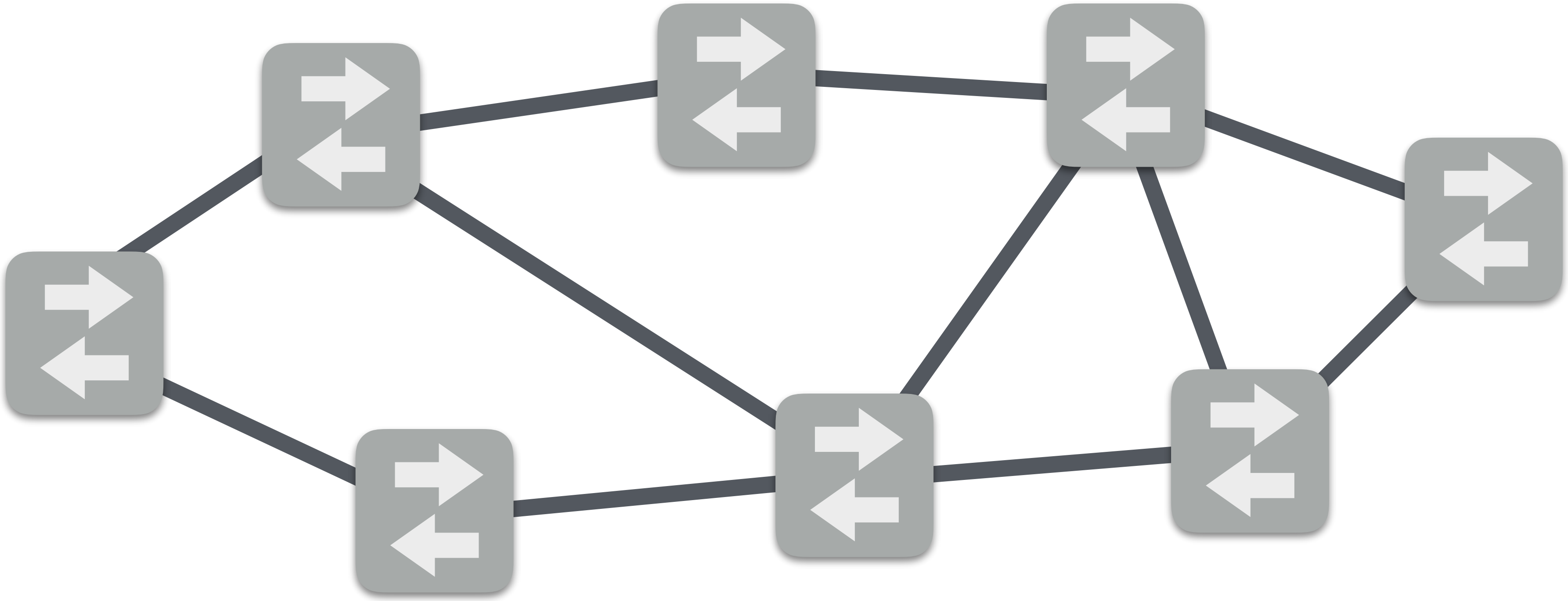
L
Maple
FlowLog
Frenetic
NetKAT

ndb
Path Queries

Routing

Debugging

Controller



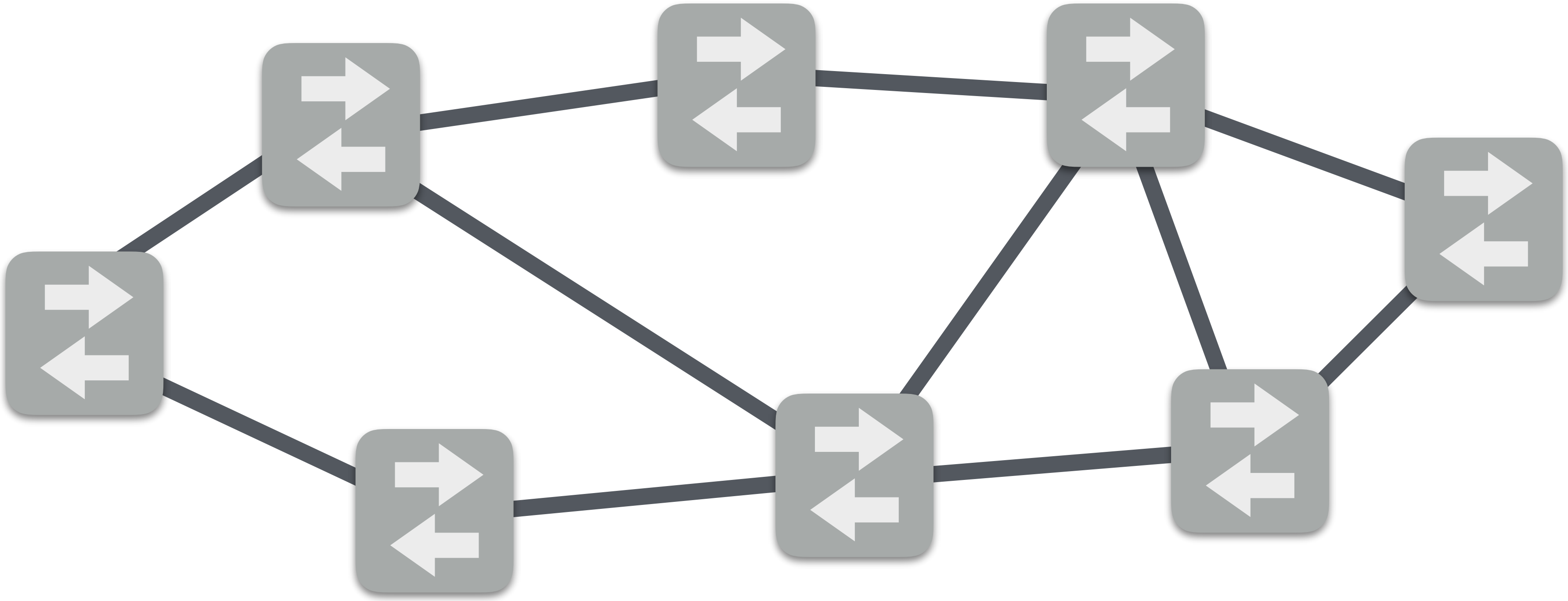
L
Maple
FlowLog
Frenetic
NetKAT

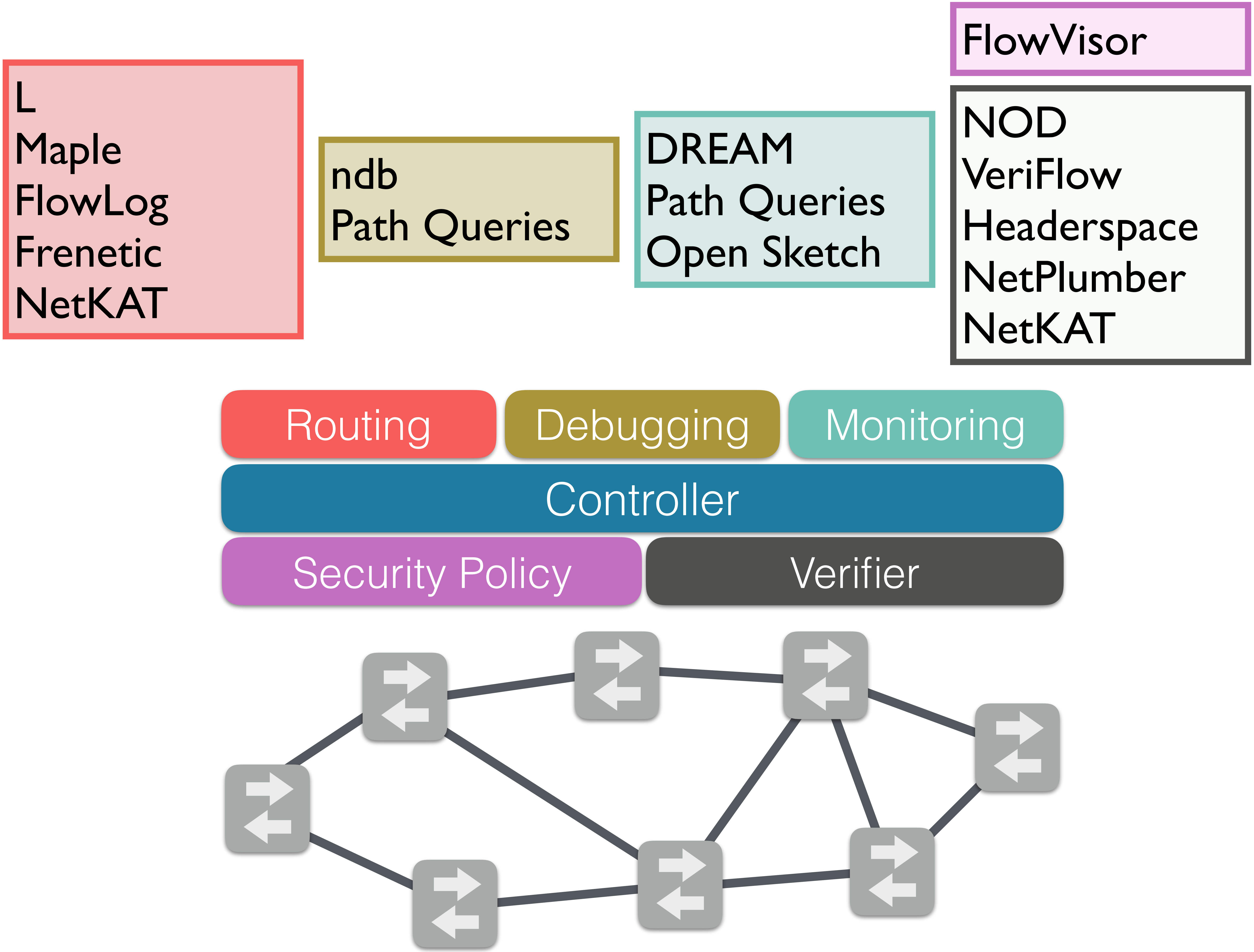
ndb
Path Queries

DREAM
Path Queries
Open Sketch

Routing | Debugging | Monitoring

Controller





L
Maple
FlowLog
Frenetic
NetKAT

ndb
Path Queries

DREAM
Path Queries
Open Sketch

FlowVisor

NOD
VeriFlow
Headerspace
NetPlumber
NetKAT

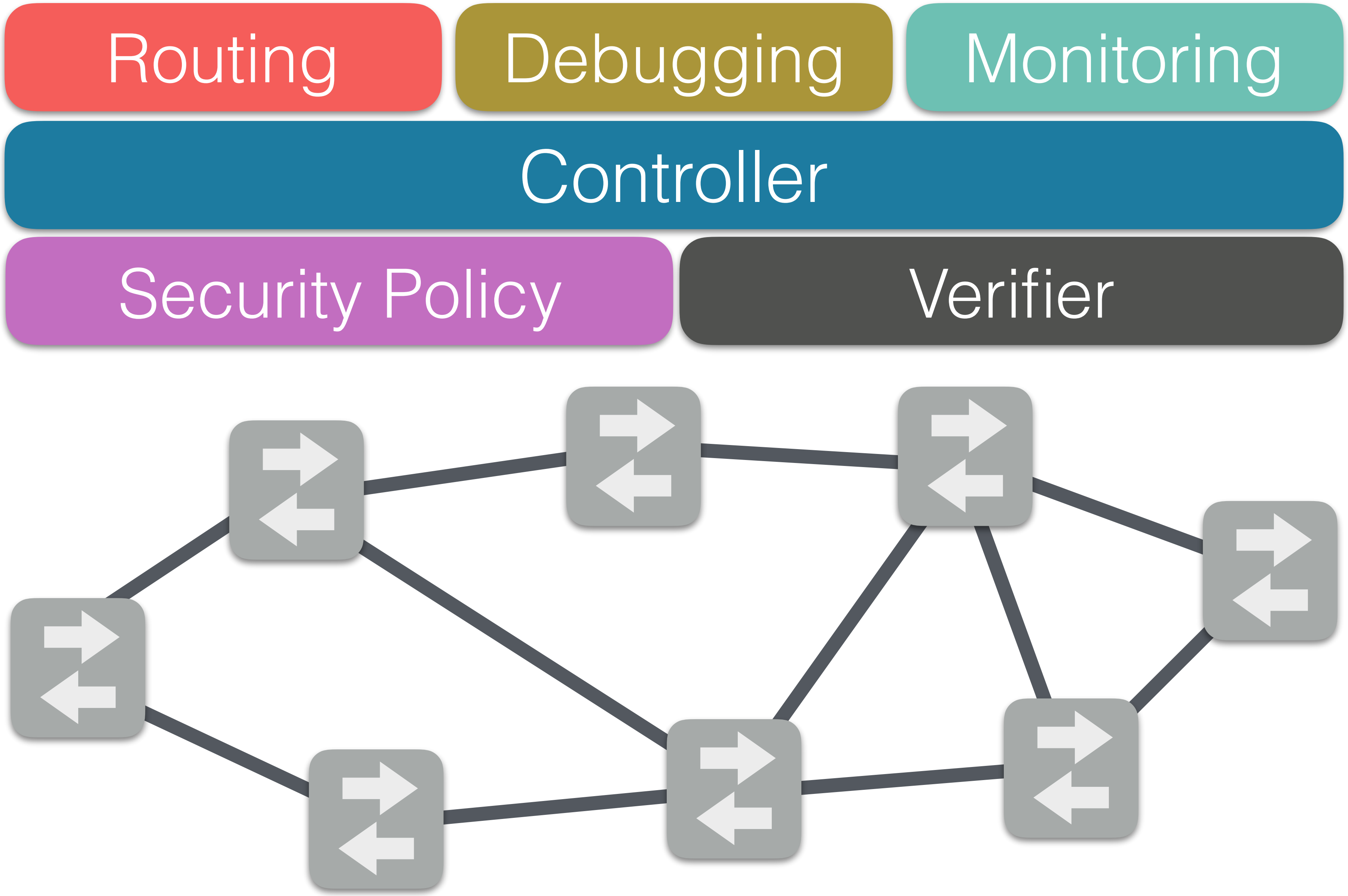
Routing
Debugging
Monitoring

Controller

Security Policy
Verifier

Shared Abstraction:

Dynamic forwarding based on a packet's history!



Overview

Temporal NetKAT

- Extend NetKAT with **queries over a packet's history**
- Study the new paradigm on several **applications**
- Define a **semantics** and equational theory for the language
- Prove **soundness** and network-wide **completeness**
- Describe and implement a **compilation strategy**
- **Evaluate** the compiler performance on several networks

Temporal NetKAT

NetKAT — Overview

Predicates

```
a,b ::= f = n     test
      | 1         true
      | 0         false
      | a + b     or
      | a · b     and
      | ¬a        negation
```

Policies

```
p,q ::= a         predicate
      | f ← v     assignment
      | p + q     union
      | p · q     sequence
      | p*        iteration
```

Based on KAT

[Kozen & Smith ’96]

Extended to networks

[Anderson et al ’14]

NetKAT — Overview

Predicates

```
 a,b ::= f = n     test
       | 1         true
       | 0         false
       | a + b     or
       | a · b     and
       | ¬a        negation
```

Boolean Algebra

Policies

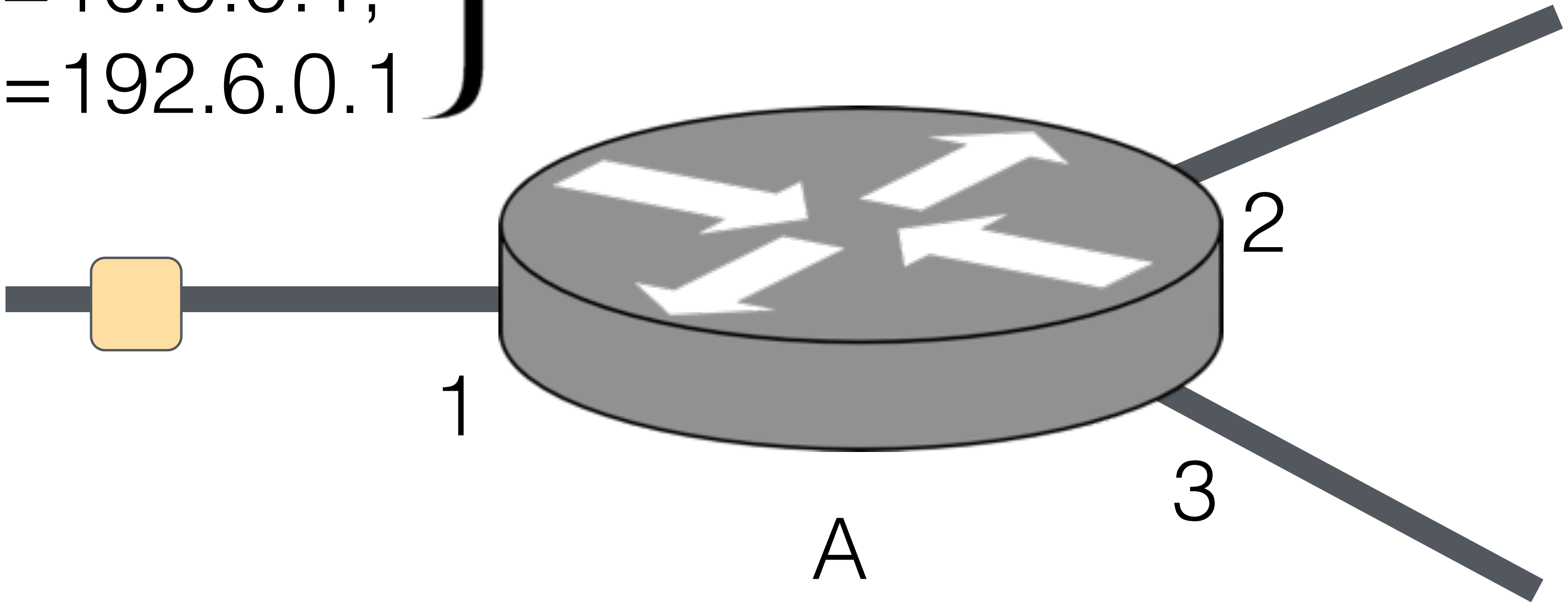
```
 p,q ::= a         predicate
       | f ← v     assignment
       | p + q     union
       | p · q     sequence
       | p*        iteration
```

Kleene Algebra

NetKAT — Overview

Packet: A record of fields and values

$$\left\{\begin{array}{l} \text{sw=A,} \\ \text{pt=1,} \\ \text{src=10.0.0.1,} \\ \text{dst=192.6.0.1} \end{array}\right\}$$

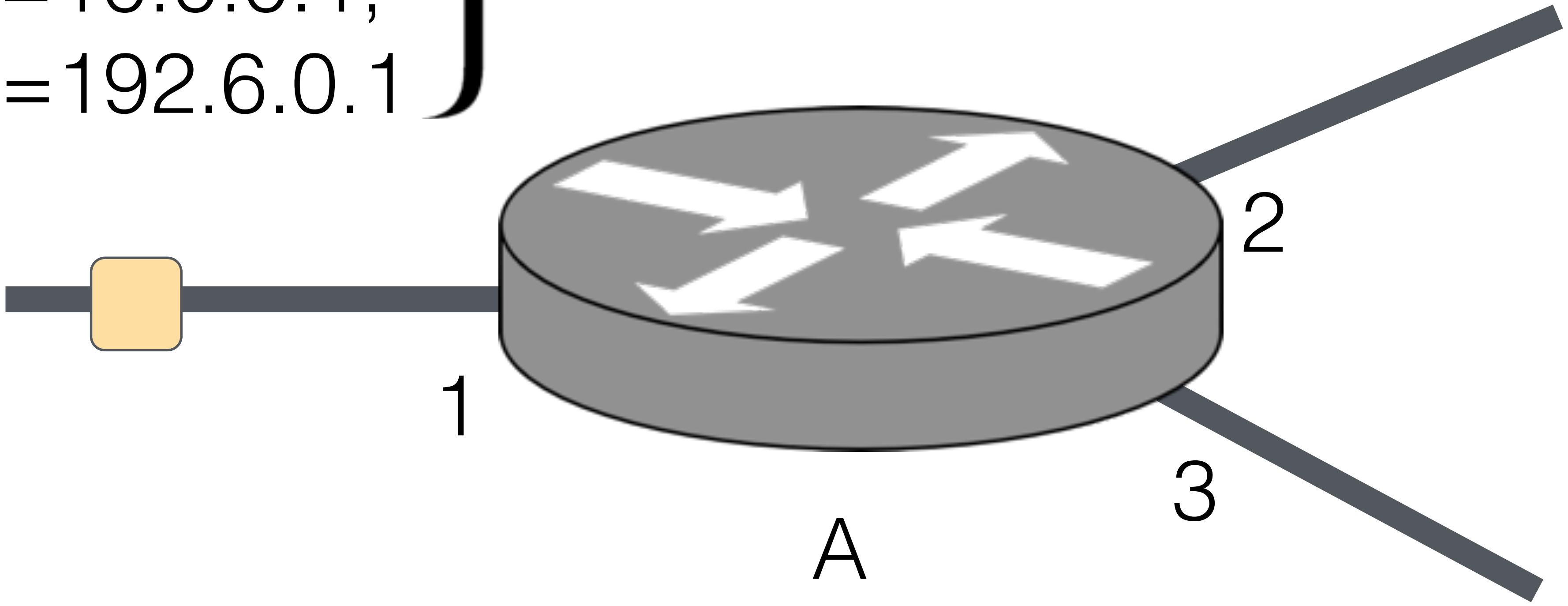


NetKAT — Overview

Language Features:

- Match packets
- Modify packets

$$\left\{ \begin{array}{l} \text{sw=A,} \\ \text{pt=1,} \\ \text{src=10.0.0.1,} \\ \text{dst=192.6.0.1} \end{array} \right\}$$



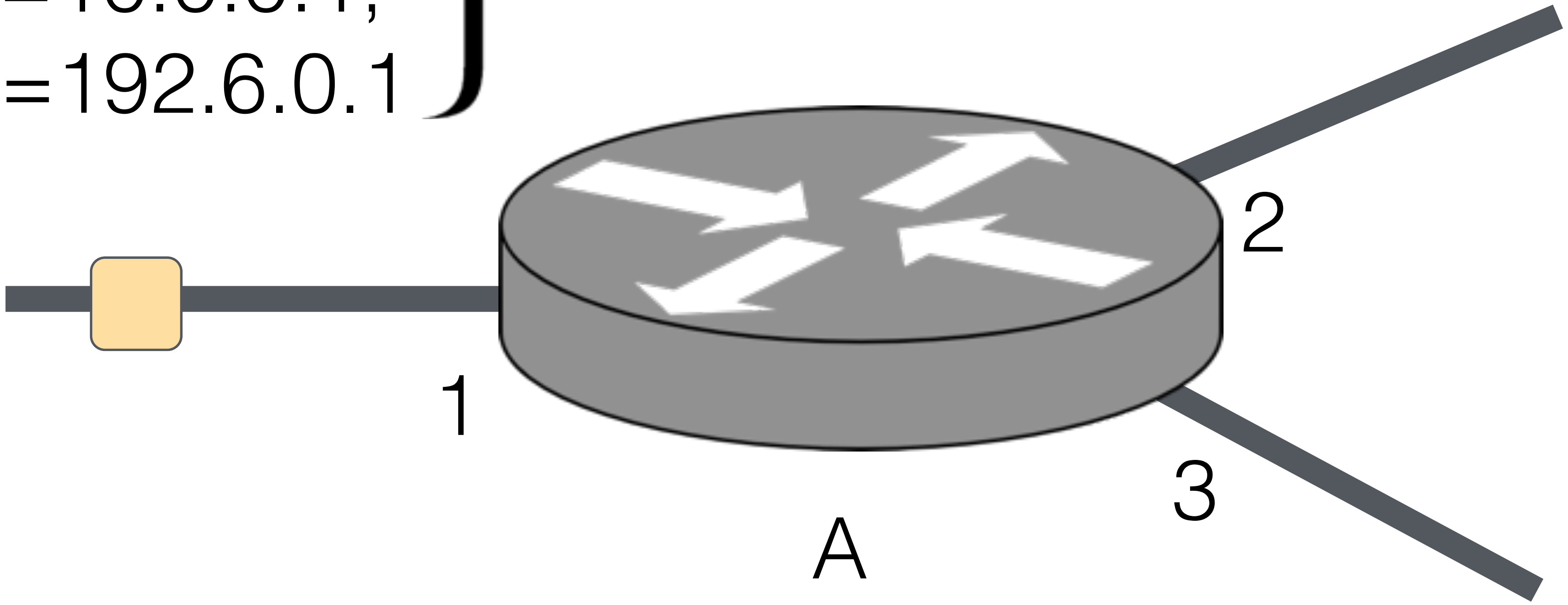
NetKAT — Overview

Language Features:

- **Match packets**
- Modify packets

{
sw=A,
pt=1,
src=10.0.0.1,
dst=192.6.0.1
}

¬src=10.0.0.1 + sw=A



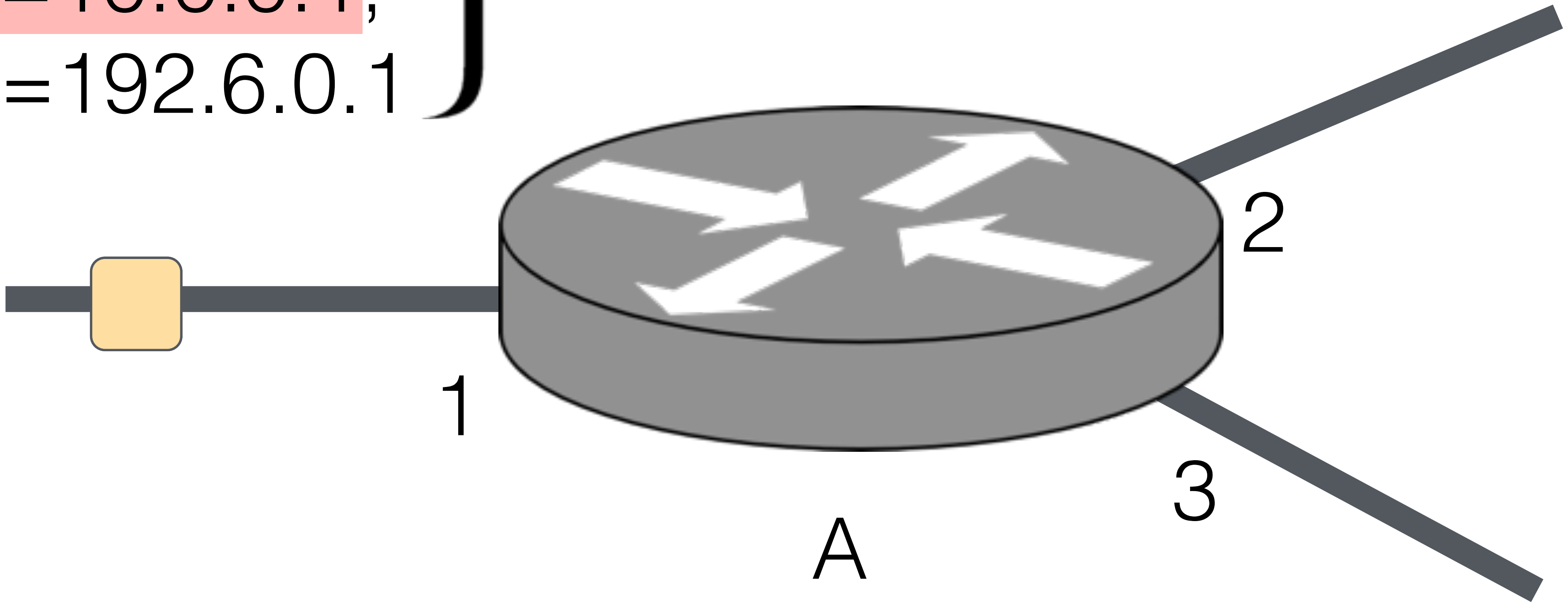
NetKAT — Overview

Language Features:

- Match packets
- **Modify packets**

$$\left\{\begin{array}{l}\text{sw=A,}\\ \text{pt=1,}\\ \text{src=10.0.0.1,}\\ \text{dst=192.6.0.1}\end{array}\right\}$$

src←12.12.0.1 · pt←2



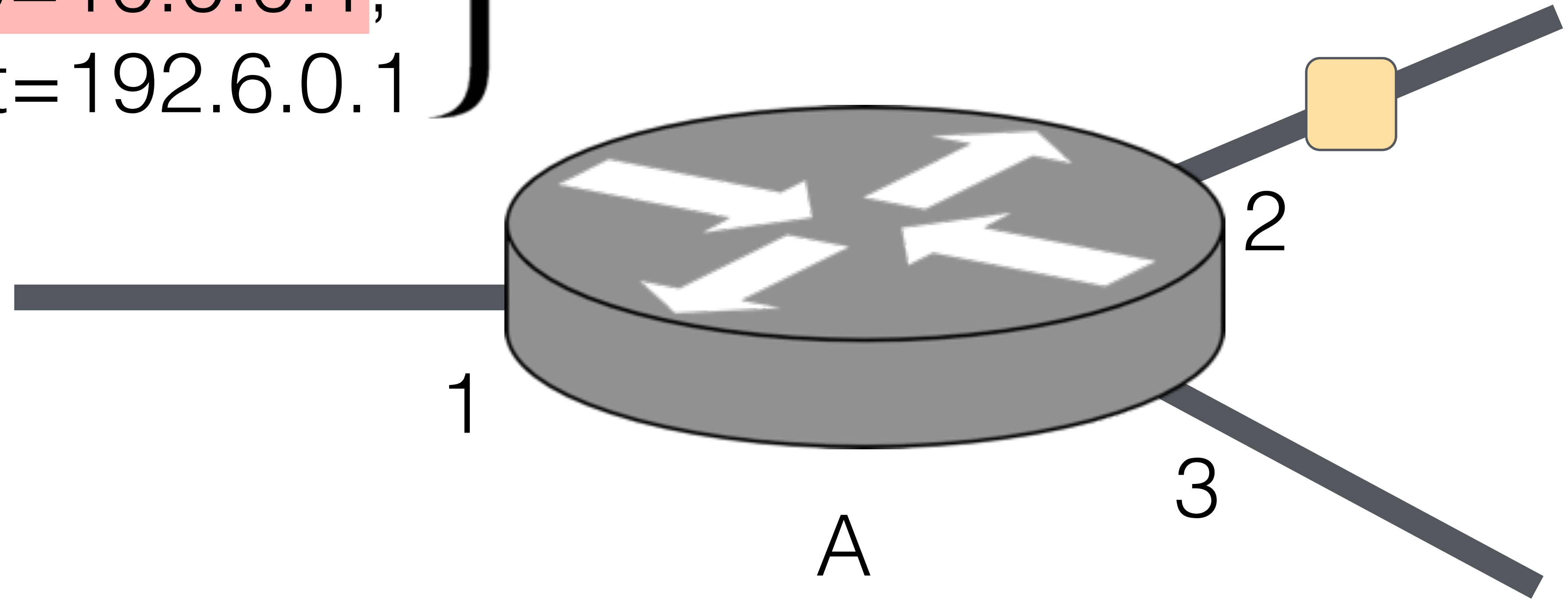
NetKAT — Overview

Language Features:

- Match packets
- **Modify packets**

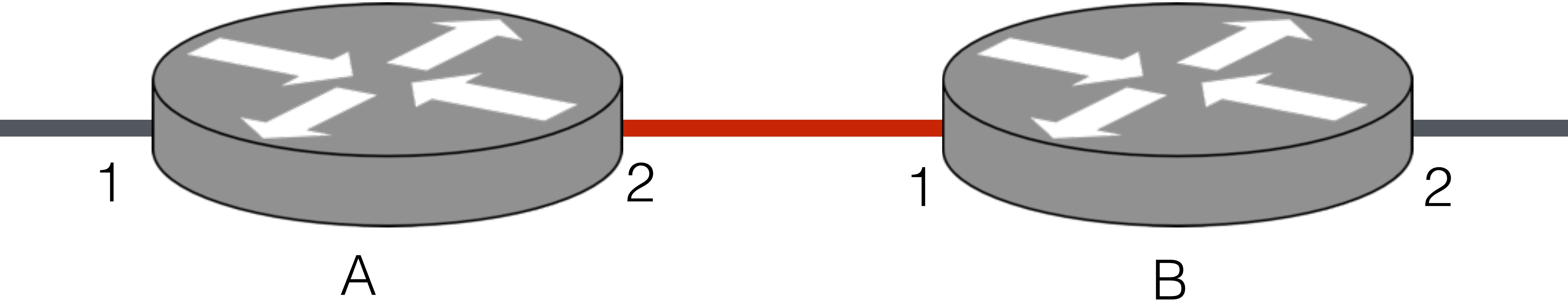
$$\left\{\begin{array}{l} sw=A, \\ pt=1, \\ src=10.0.0.1, \\ dst=192.6.0.1 \end{array}\right\}$$

$src \leftarrow 12.12.0.1 \cdot pt \leftarrow 2$



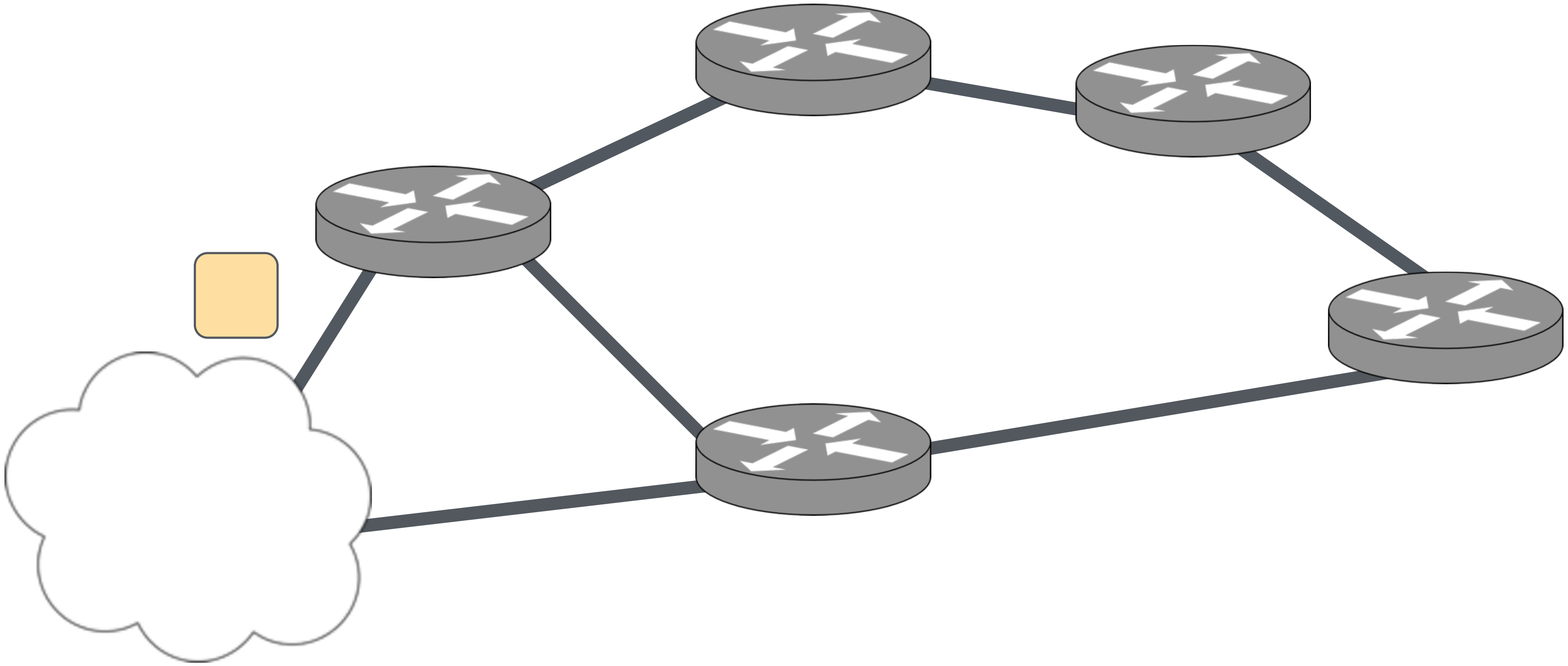
NetKAT — Topology

$(sw{=}A \cdot pt{=}2) \cdot sw{\leftarrow}B \cdot pt{\leftarrow}1$



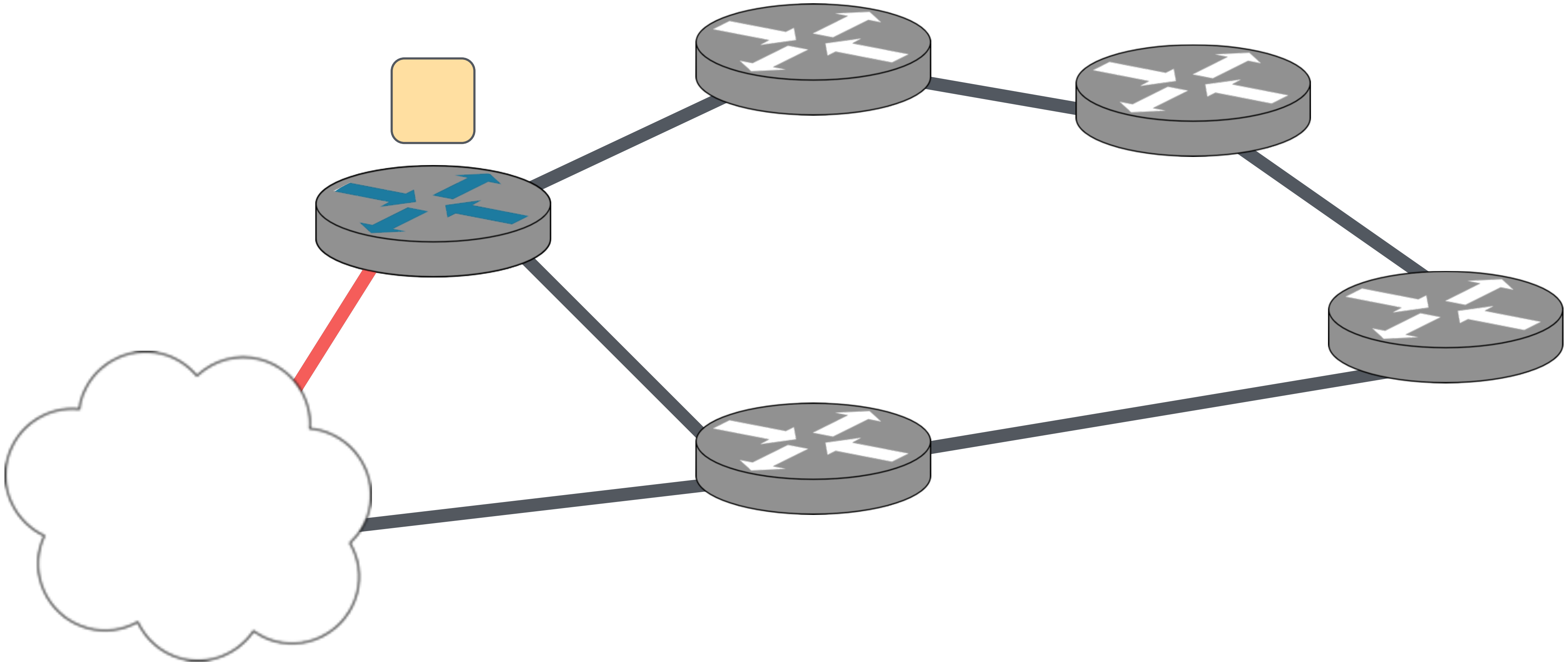
NetKAT — Network

Kleene Star: (topology · switch)*



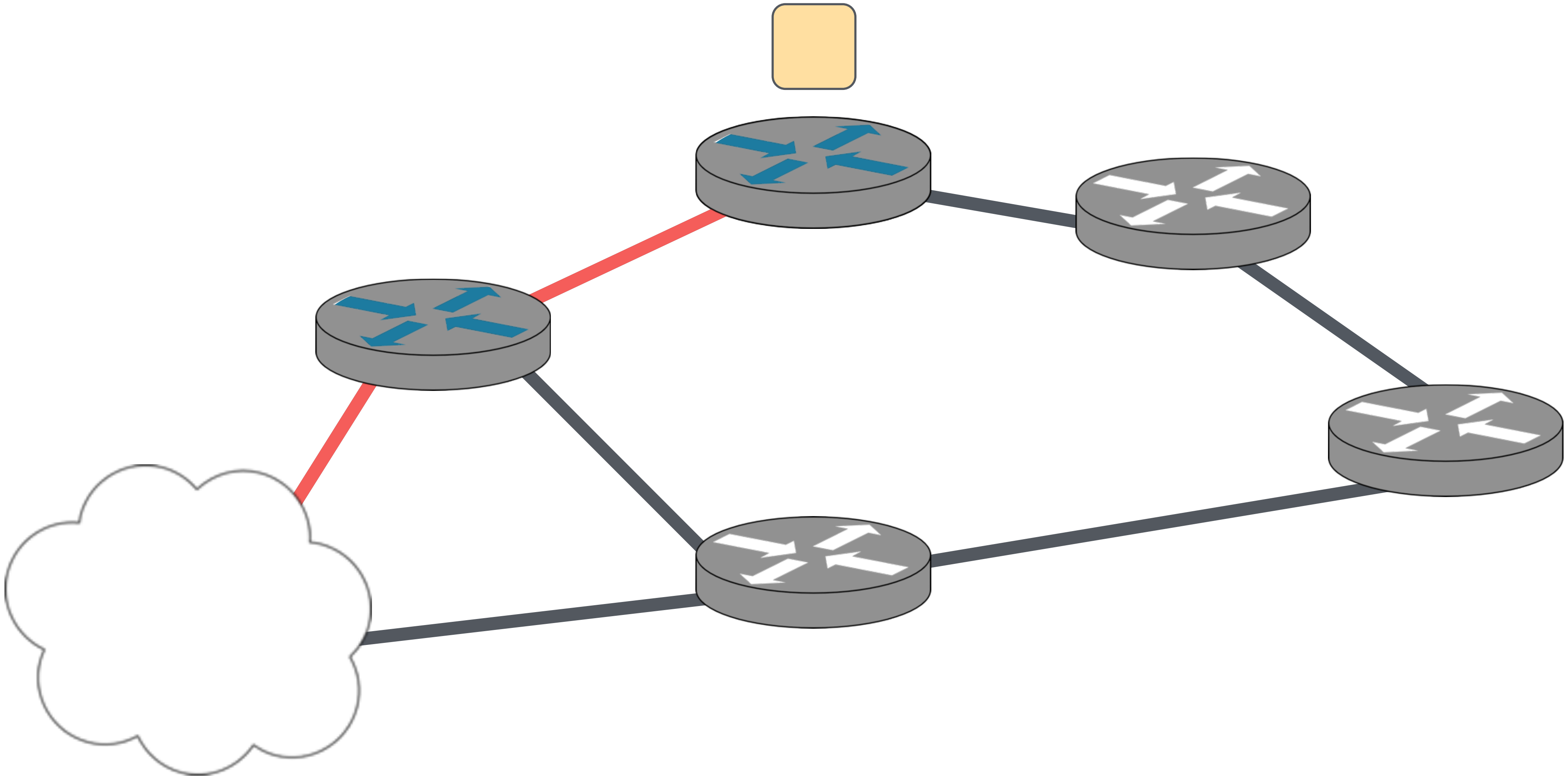
NetKAT — Network

Kleene Star: (topology · switch)*



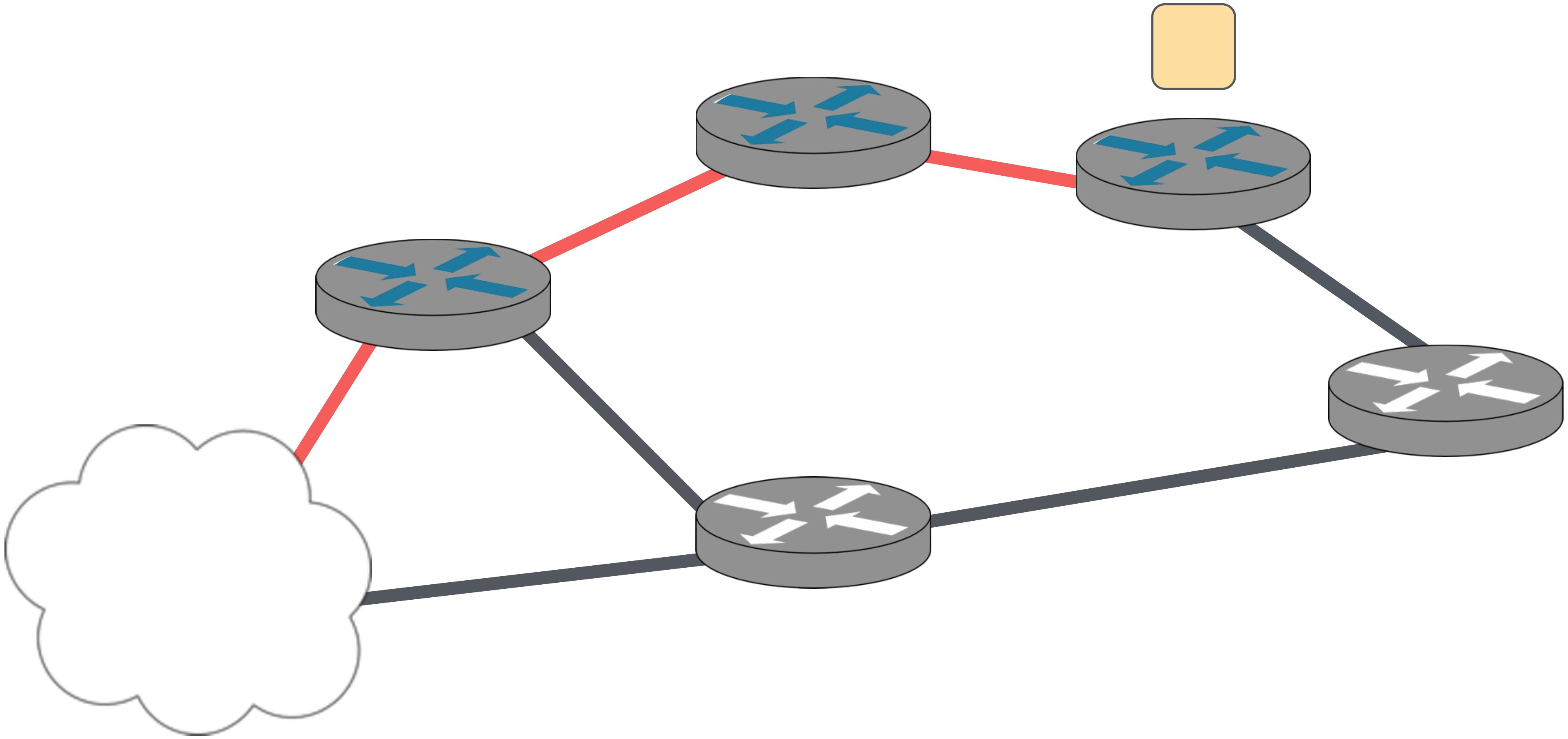
NetKAT — Network

Kleene Star: (topology · switch)*



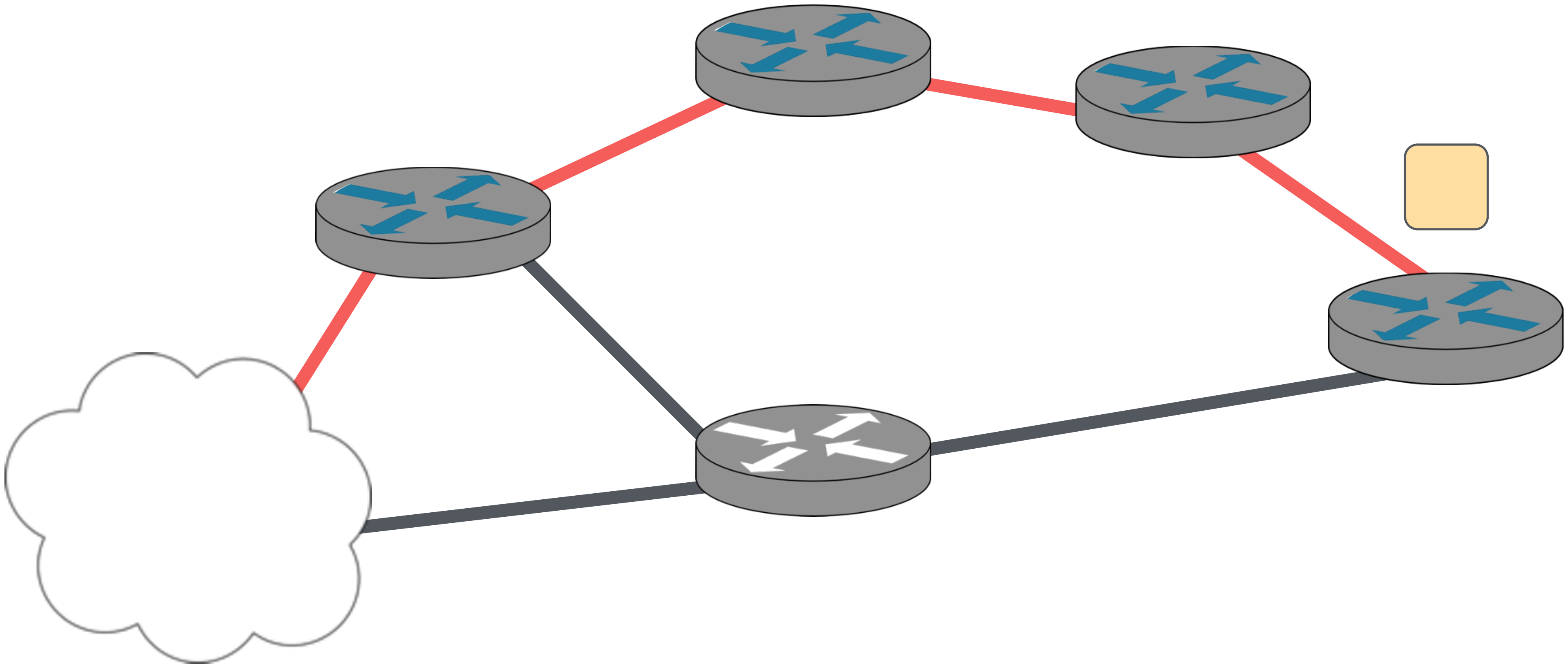
NetKAT — Network

Kleene Star: (topology · switch)*



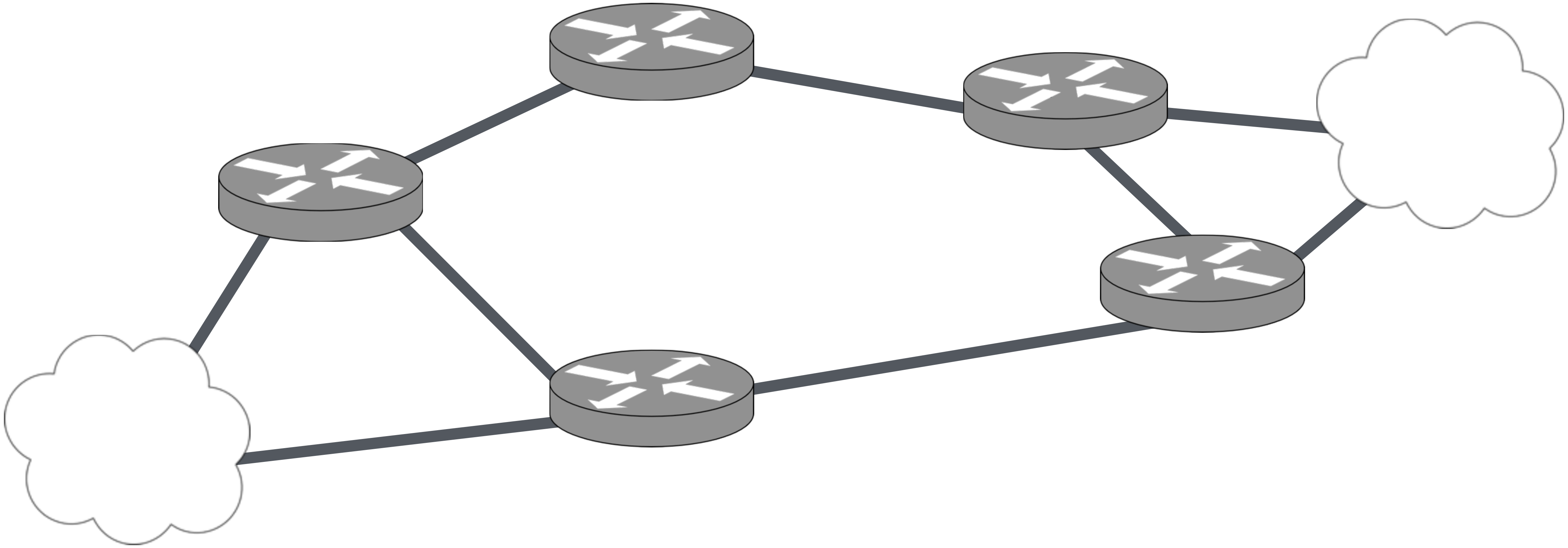
NetKAT — Network

Kleene Star: (topology · switch)*



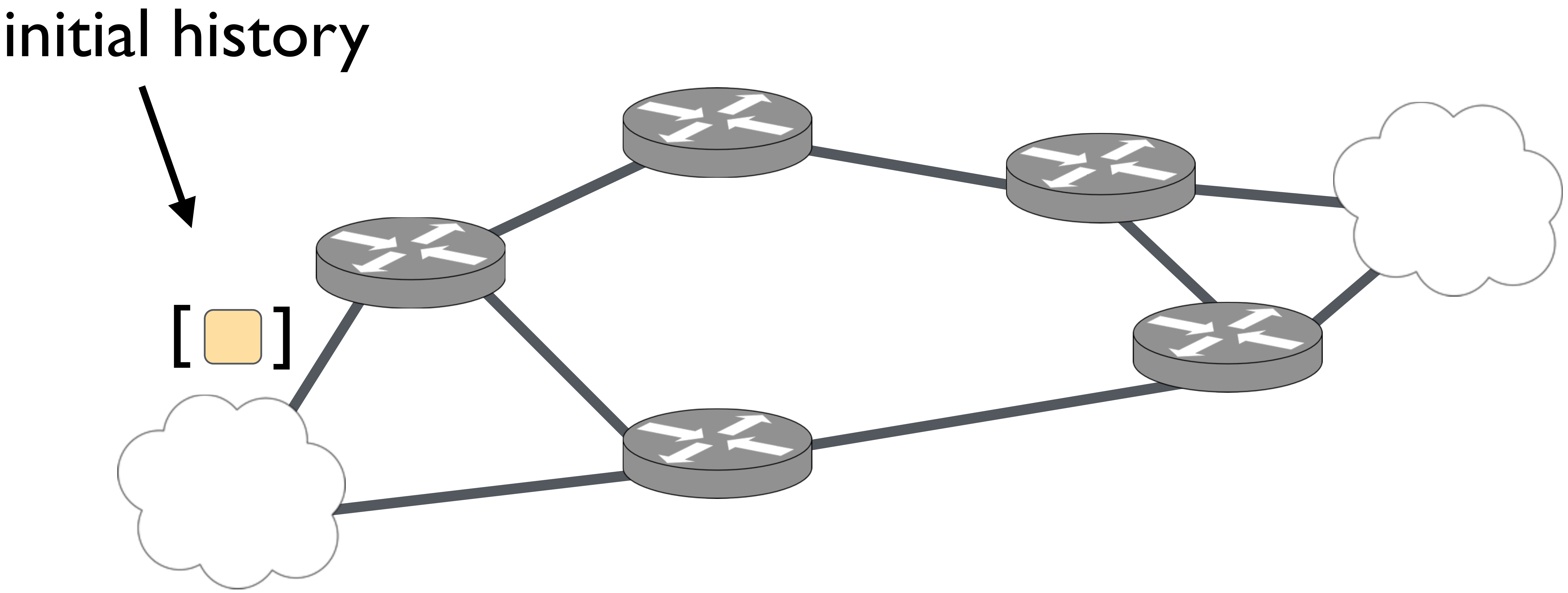
NetKAT: Packet History

A policy takes a packet history to a set of histories



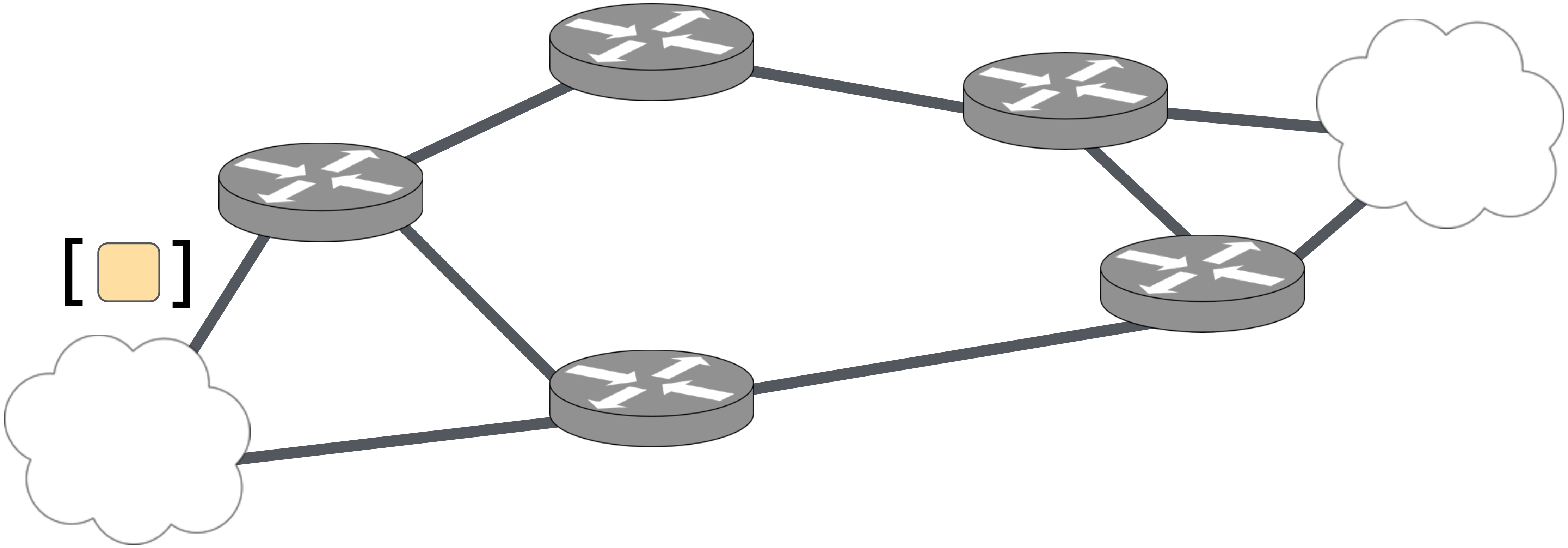
NetKAT: Packet History

A policy takes a packet history to a set of histories



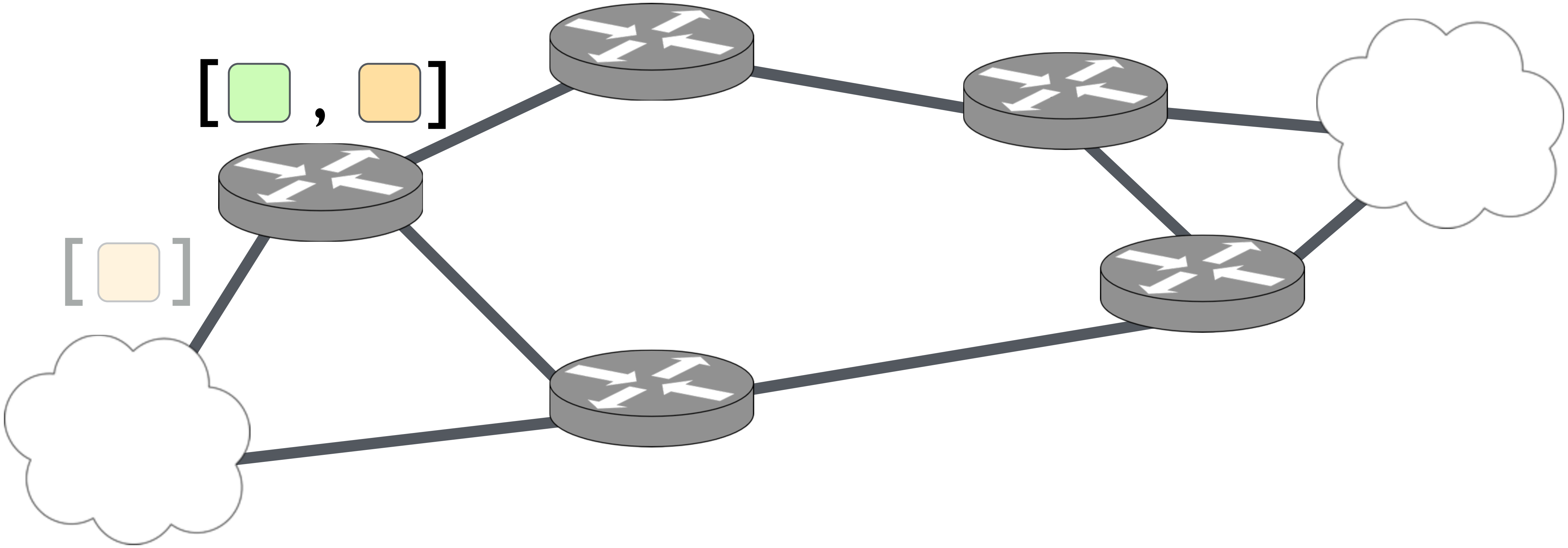
NetKAT: Packet History

A policy takes a packet history to a set of histories



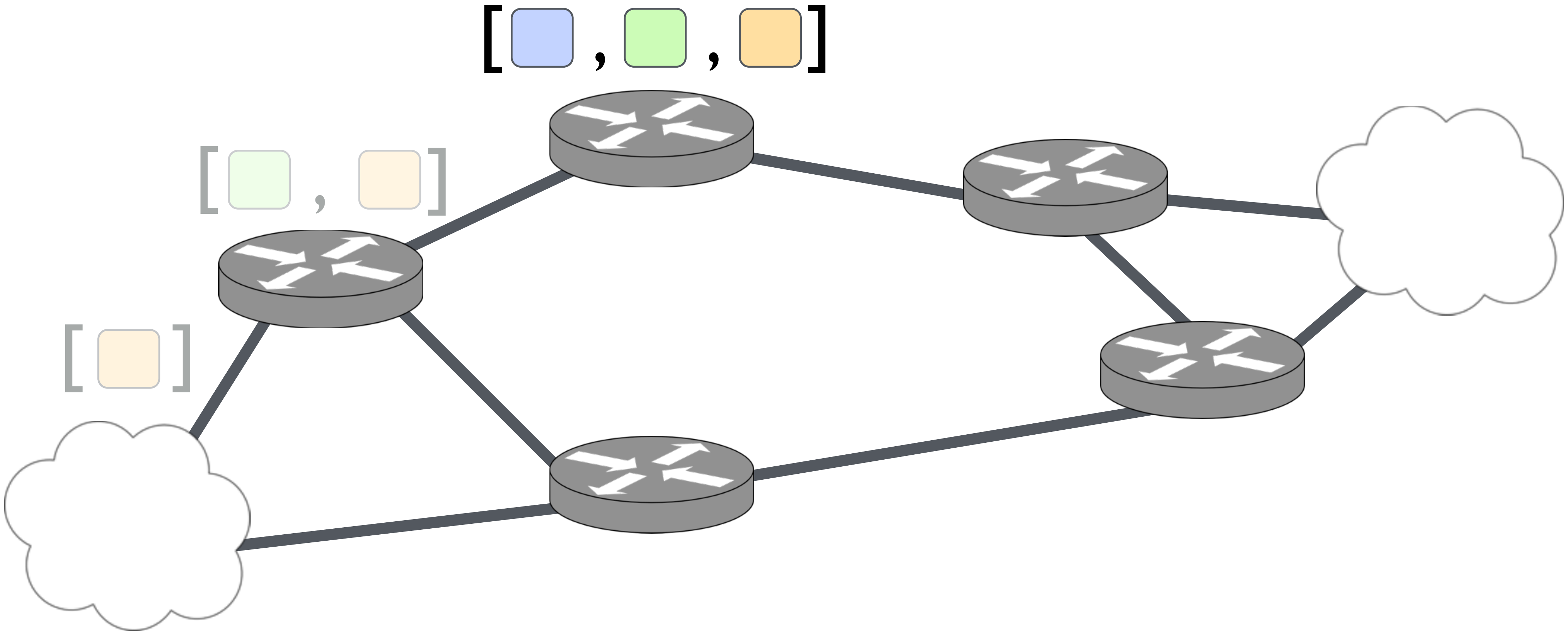
NetKAT: Packet History

A policy takes a packet history to a set of histories



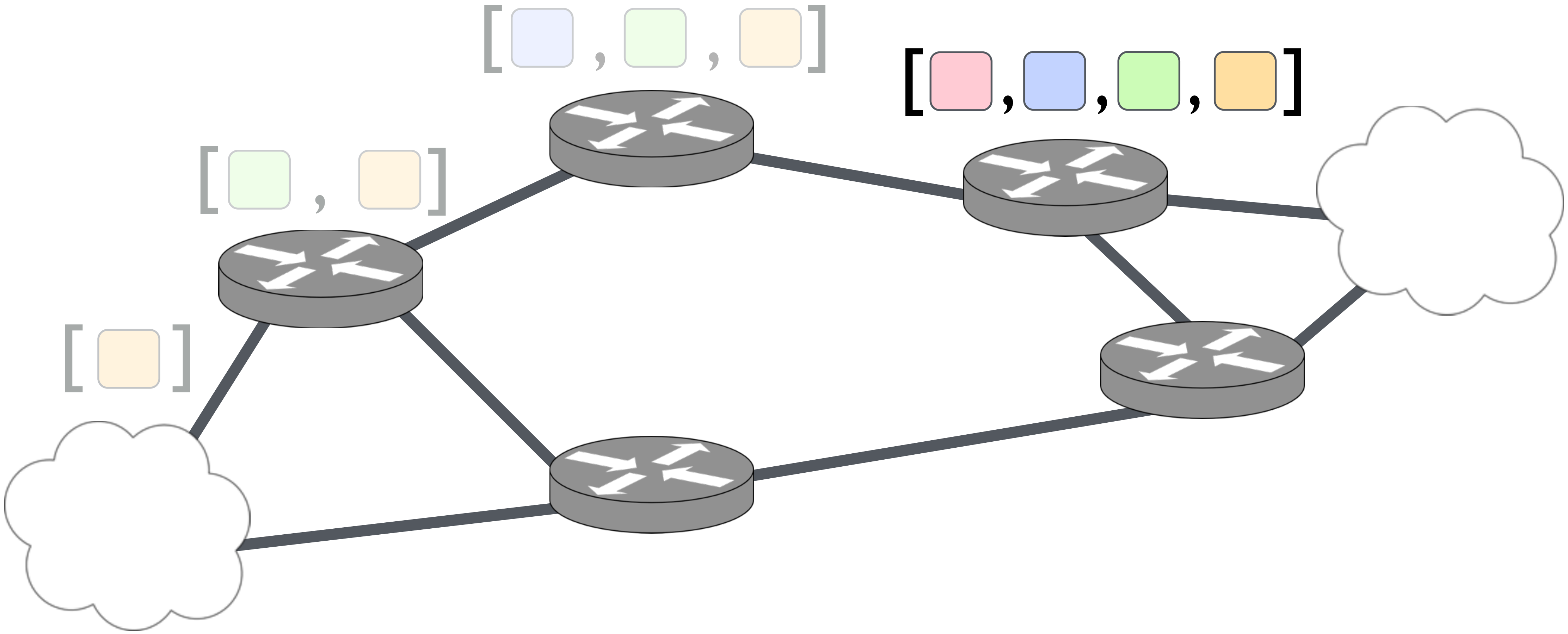
NetKAT: Packet History

A policy takes a packet history to a set of histories

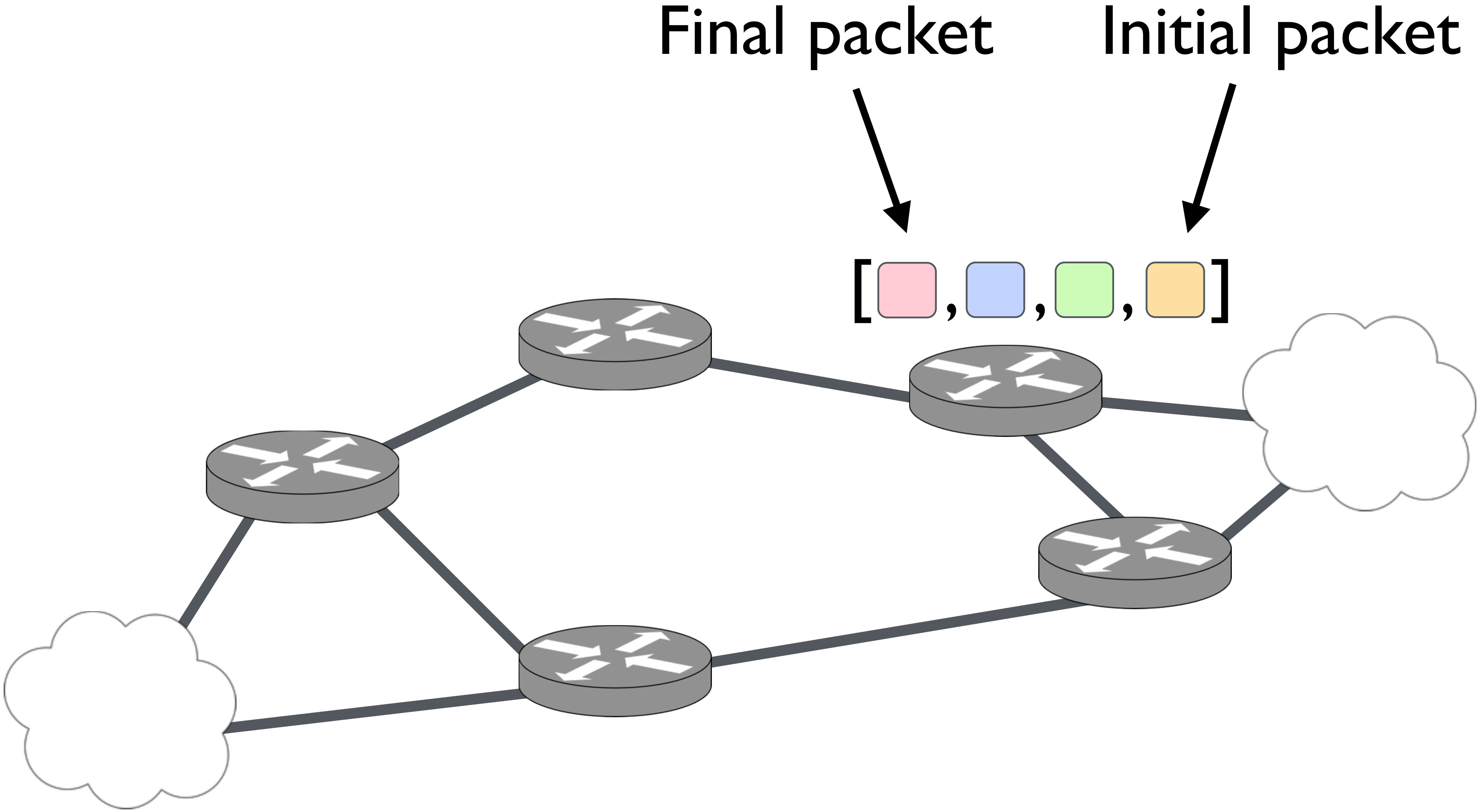


NetKAT: Packet History

A policy takes a packet history to a set of histories



NetKAT: Packet History

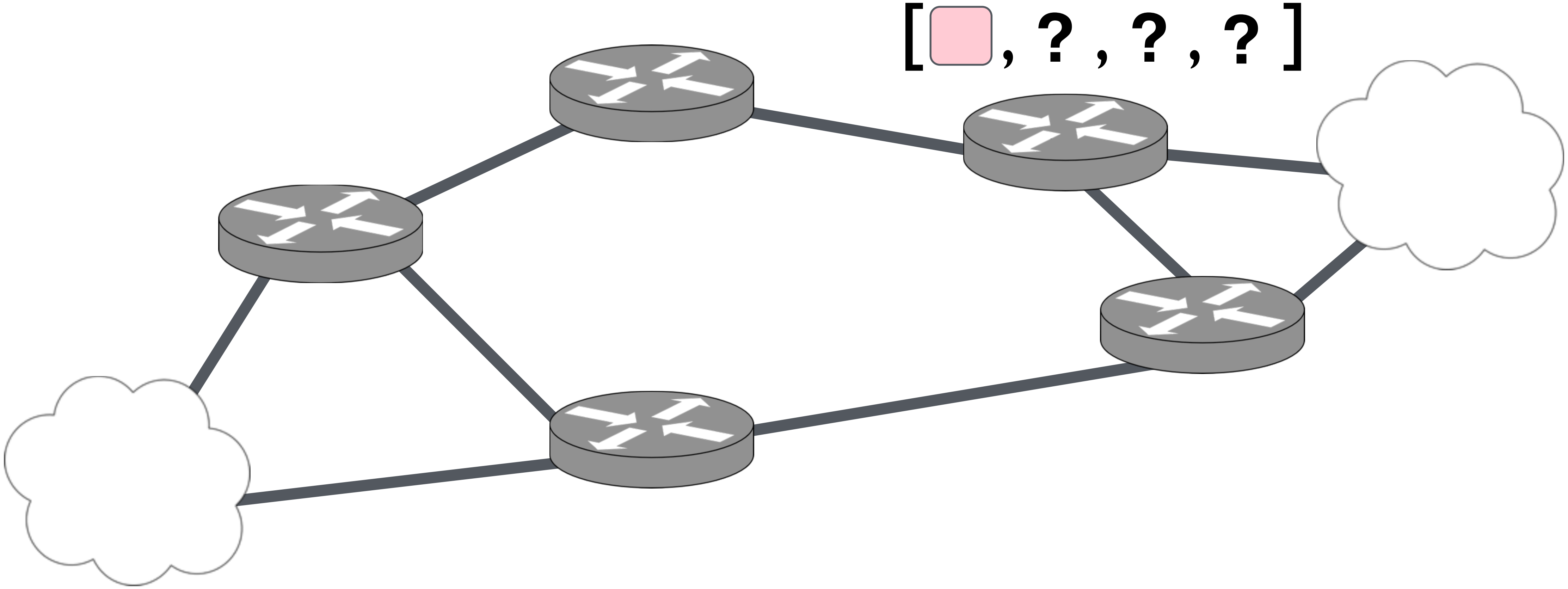


Final packet

Initial packet

NetKAT: Packet History

In practice, packets do not carry their history



Temporal NetKAT

Predicates

```
 a,b ::= f = n      test
       | 1          identity
       | 0          drop
       | a + b      or
       | a · b      and
       | ¬a         negation
       | ○a         last
       | (a S b)    since
```

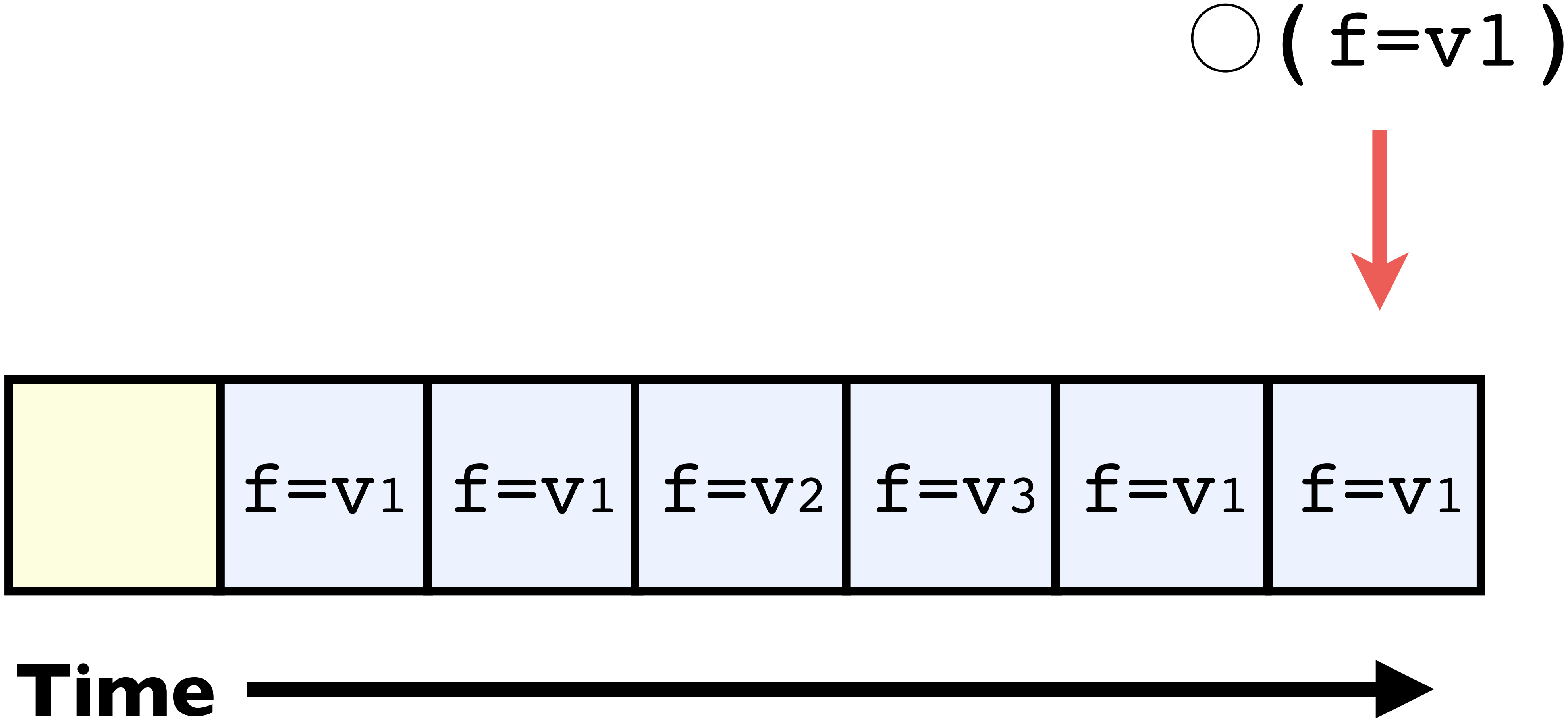
LTL_f

Policies

```
 p,q ::= a          predicate
       | f ← v      assignment
       | p + q      union
       | p · q      sequence
       | p*         iteration
```

Kleene Algebra

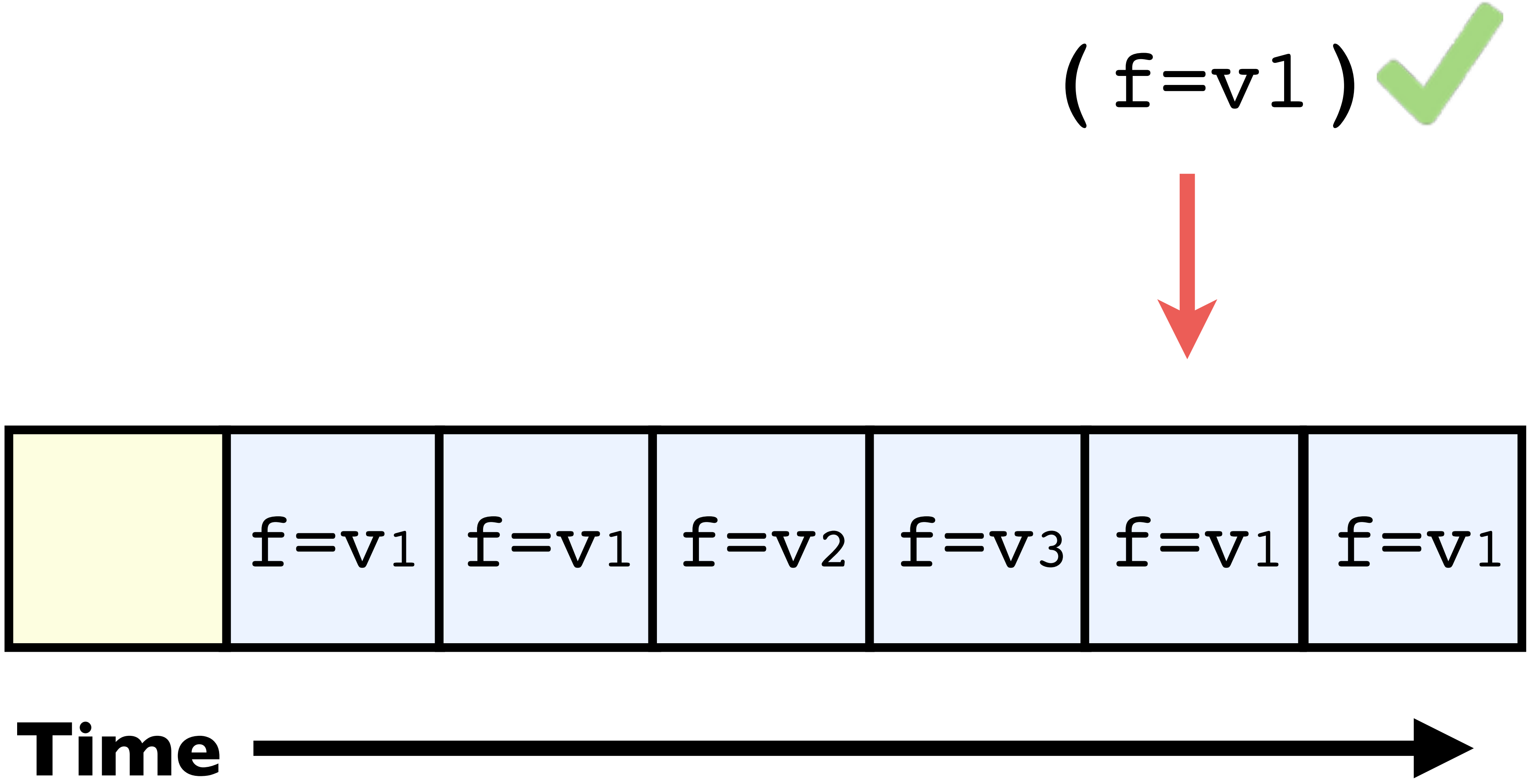
Temporal NetKAT



○ (f=v1)

	f=v1	f=v1	f=v2	f=v3	f=v1	f=v1

Temporal NetKAT



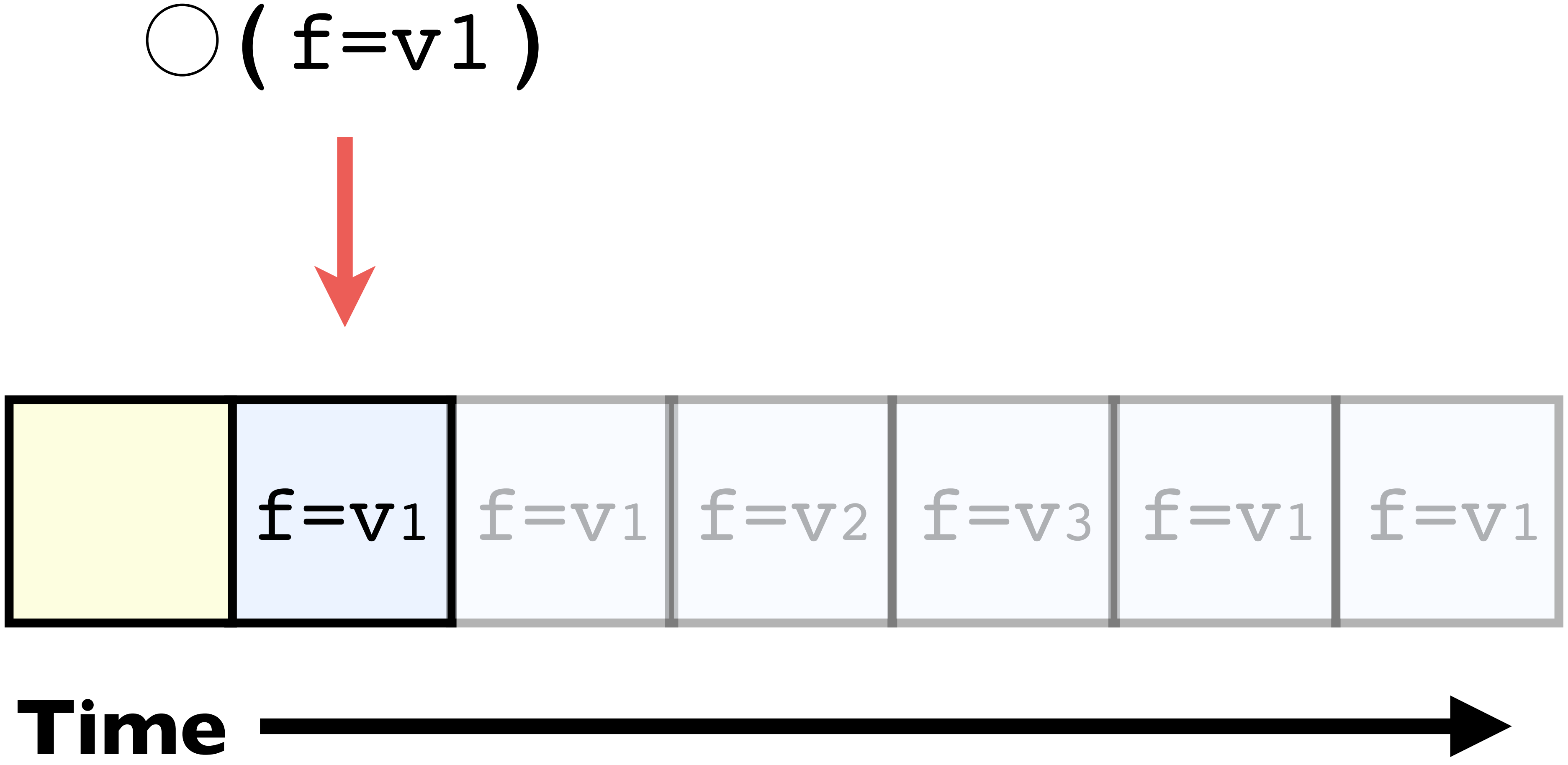
(f=v1) ✓

	$f=v_1$	$f=v_1$	$f=v_2$	$f=v_3$	$f=v_1$	$f=v_1$

Time

Temporal NetKAT

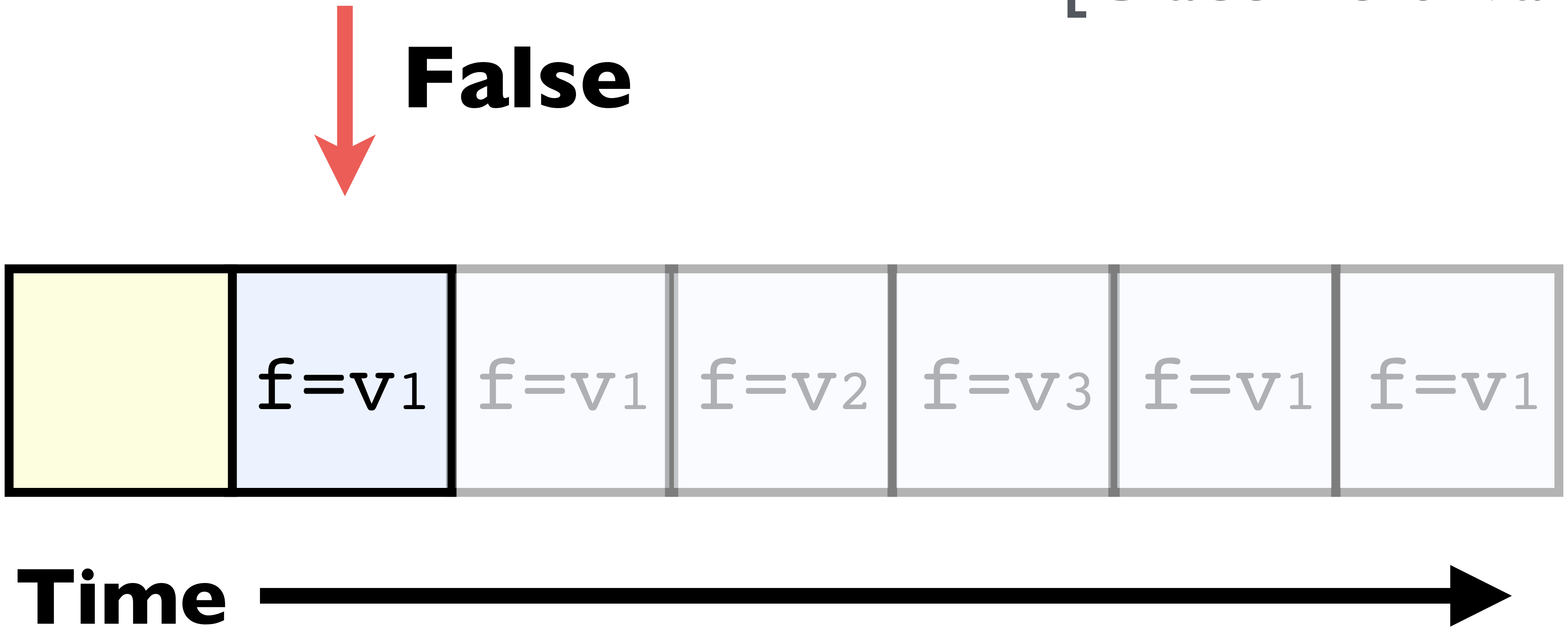
What to do when there is no history?



Temporal NetKAT

What to do when there is no history?

$\bigcirc$(f=v1)



False

	f=v1	f=v1	f=v2	f=v3	f=v1	f=v1

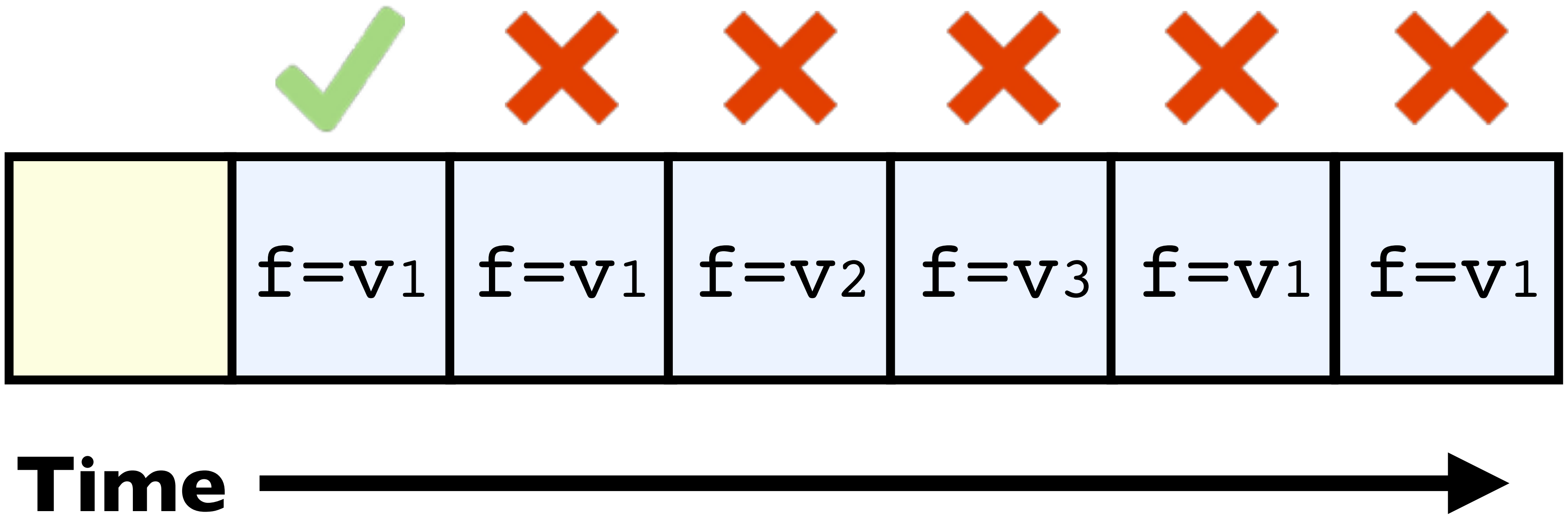
Time

Finite trace semantics

LTL_f [Giacomo & Vardi ‘13]

Temporal NetKAT

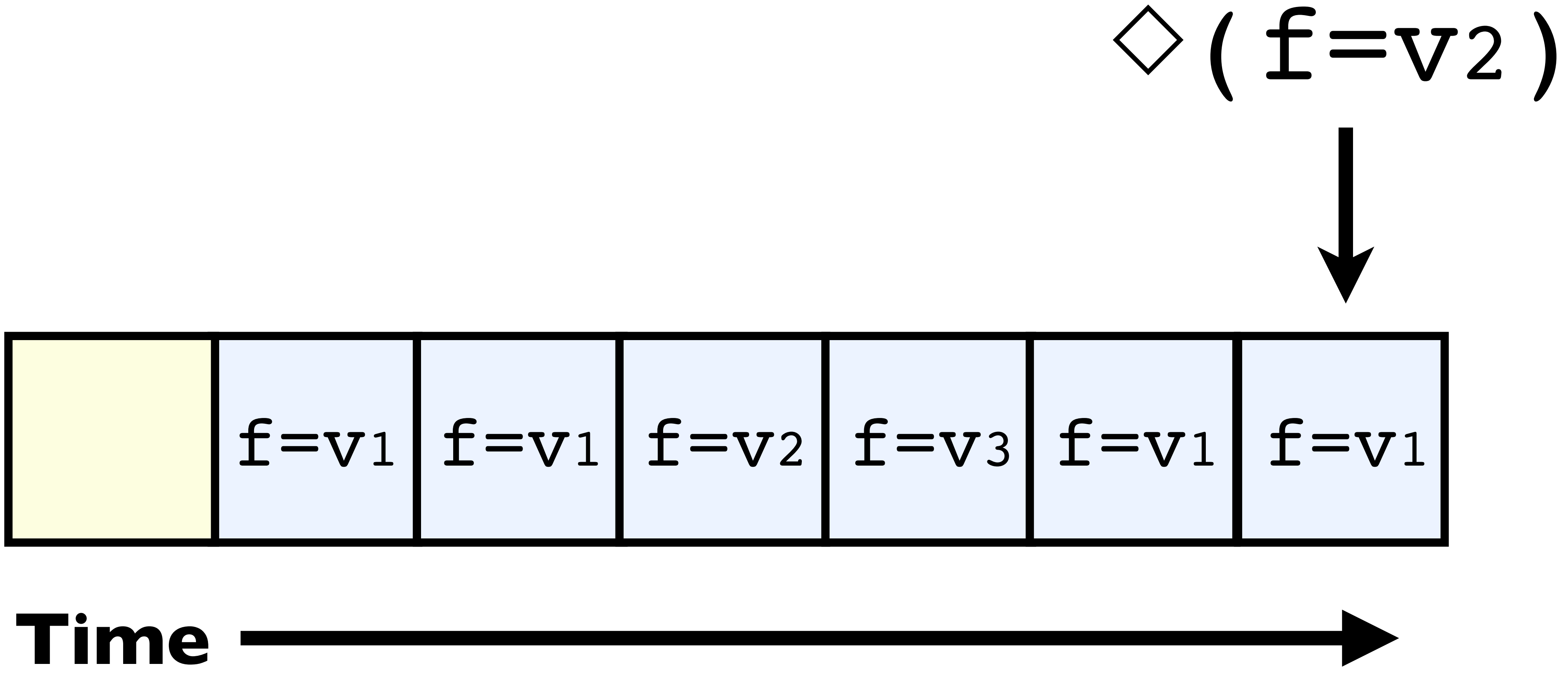
$$\texttt{start} = \neg\bigcirc 1$$



	✓	✗	✗	✗	✗	✗
	$f=v_1$	$f=v_1$	$f=v_2$	$f=v_3$	$f=v_1$	$f=v_1$

Temporal NetKAT

Ever $\Diamond \mathtt{a} = (\mathtt{1}\ \mathcal{S}\ \mathtt{a})$



$\Diamond(\mathtt{f=v_2})$

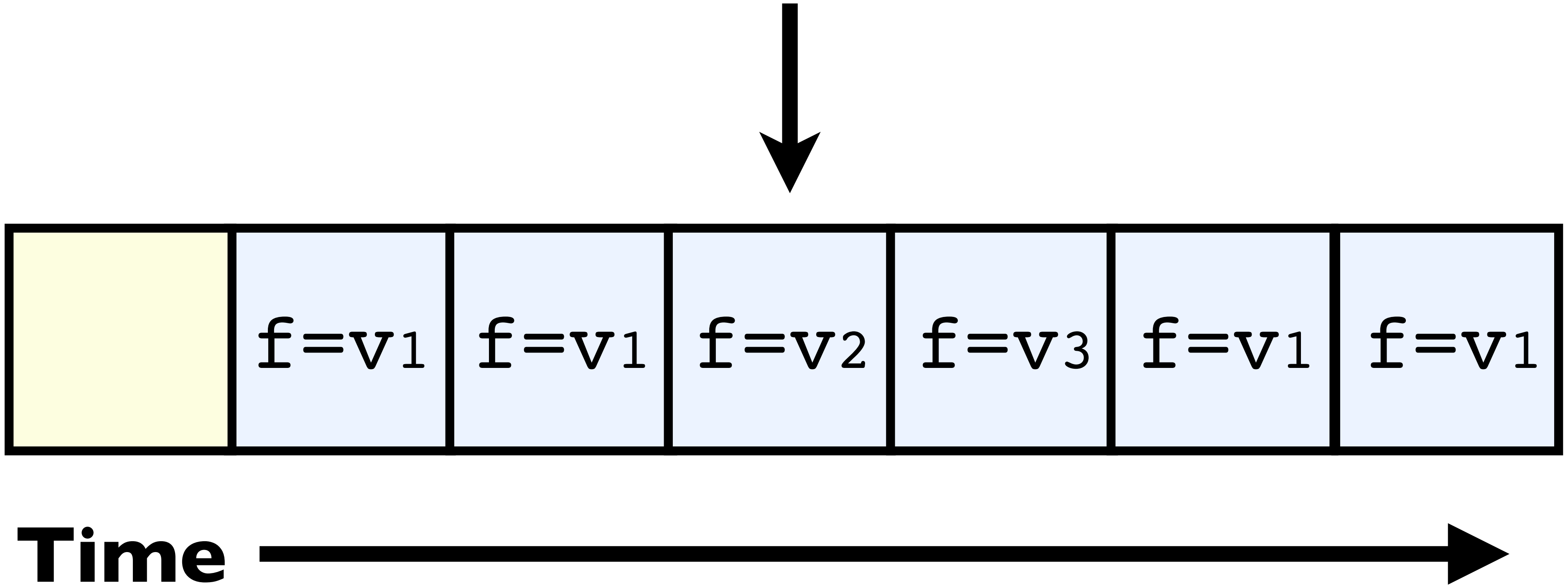
	$\mathtt{f=v_1}$	$\mathtt{f=v_1}$	$\mathtt{f=v_2}$	$\mathtt{f=v_3}$	$\mathtt{f=v_1}$	$\mathtt{f=v_1}$

Time

Temporal NetKAT

Ever $\Diamond \texttt{a} = (\texttt{1}\ \mathit{S}\ \texttt{a})$

$\Diamond(\texttt{f=v}_2)$ ✓

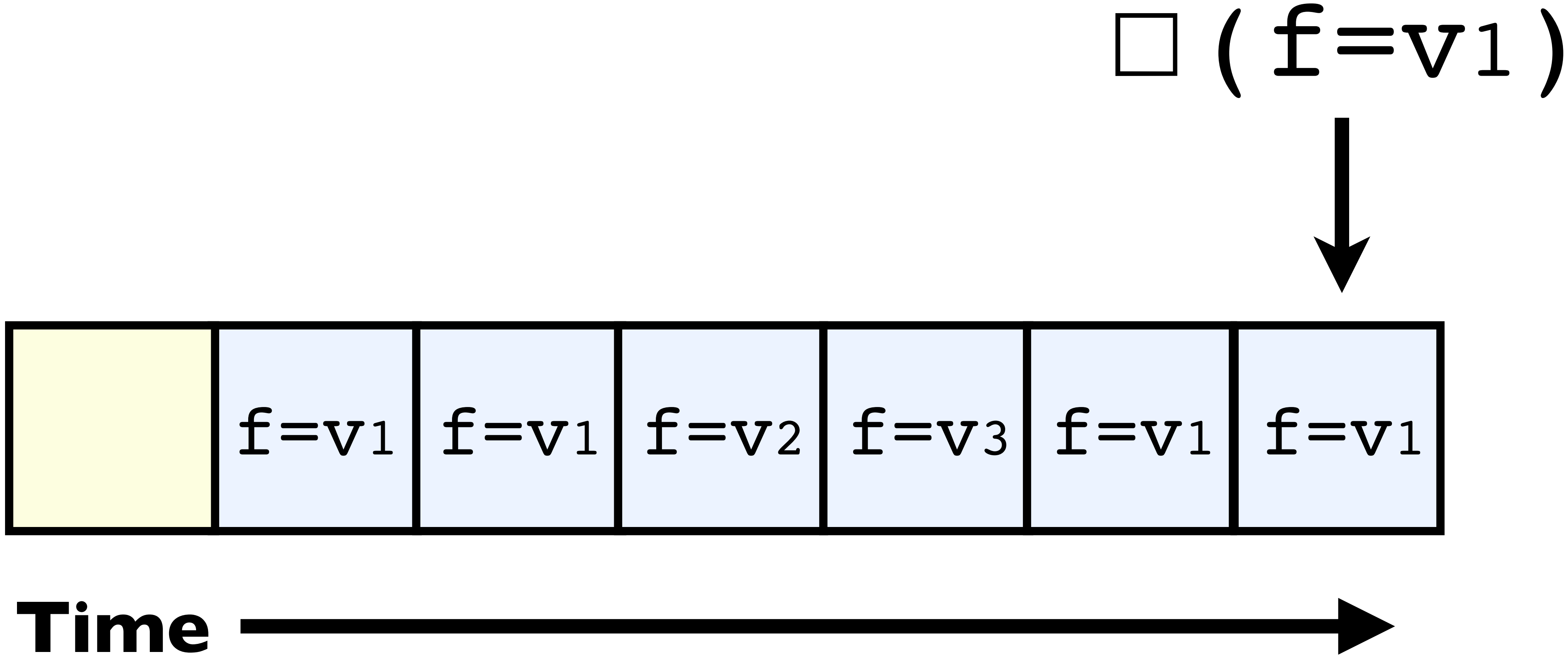


	$\texttt{f=v}_1$	$\texttt{f=v}_1$	$\texttt{f=v}_2$	$\texttt{f=v}_3$	$\texttt{f=v}_1$	$\texttt{f=v}_1$

Time

Temporal NetKAT

Always $\Box a = \neg\Diamond\neg a$



$\Box(f=v_1)$

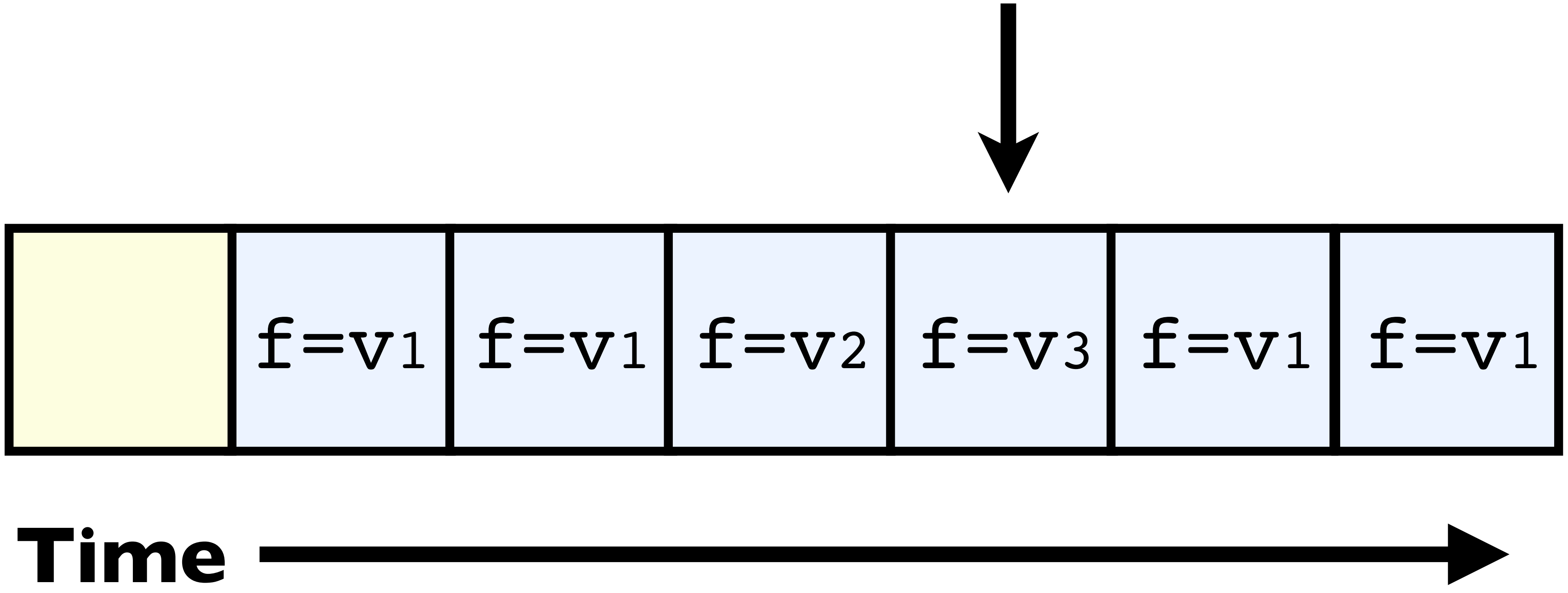
	$f=v_1$	$f=v_1$	$f=v_2$	$f=v_3$	$f=v_1$	$f=v_1$

Time

Temporal NetKAT

Always $\square a \;=\; \neg\Diamond\neg a$

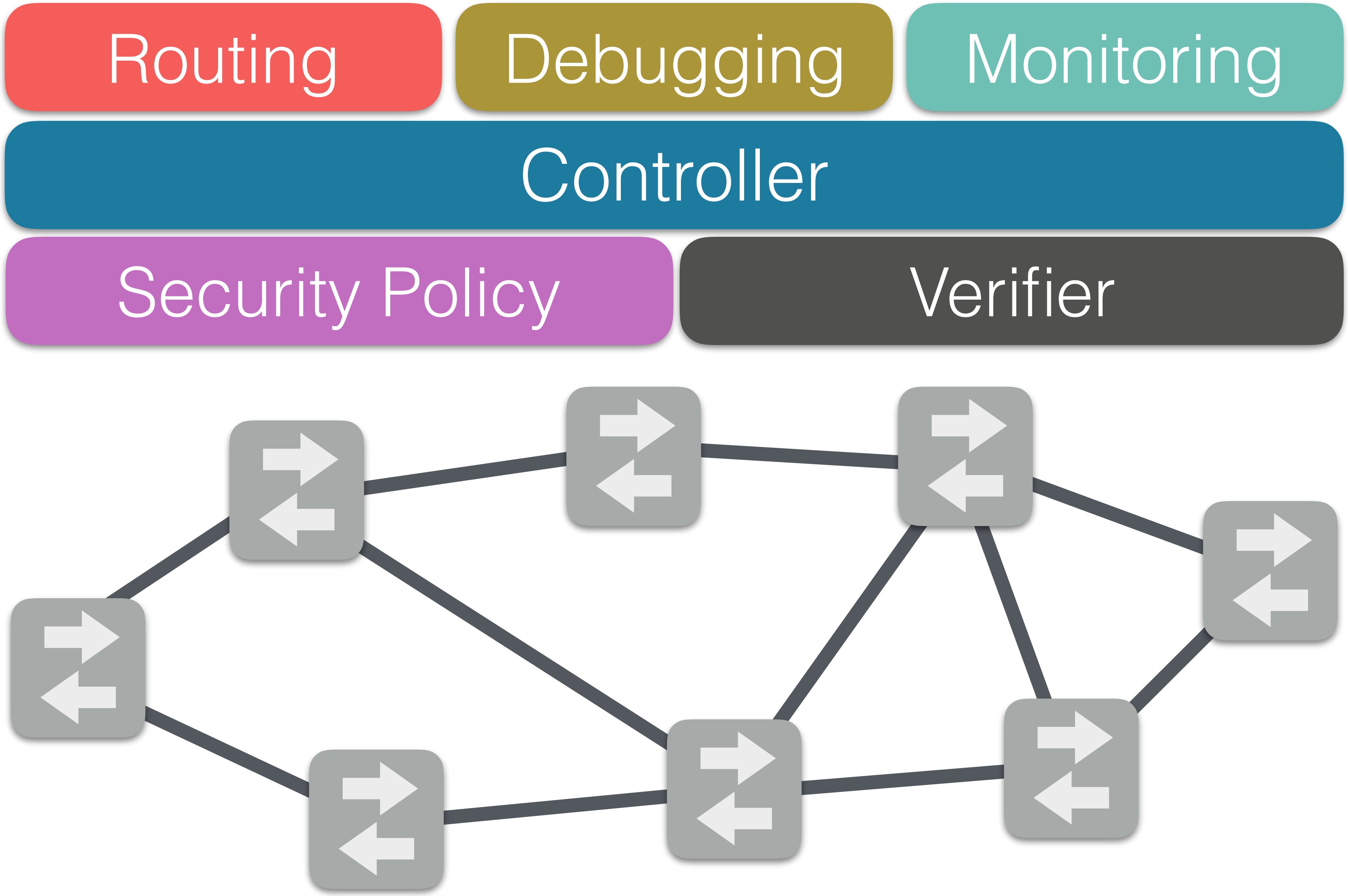
$\square(f=v_1)$ ✗



	$f=v_1$	$f=v_1$	$f=v_2$	$f=v_3$	$f=v_1$	$f=v_1$

Time

Examples



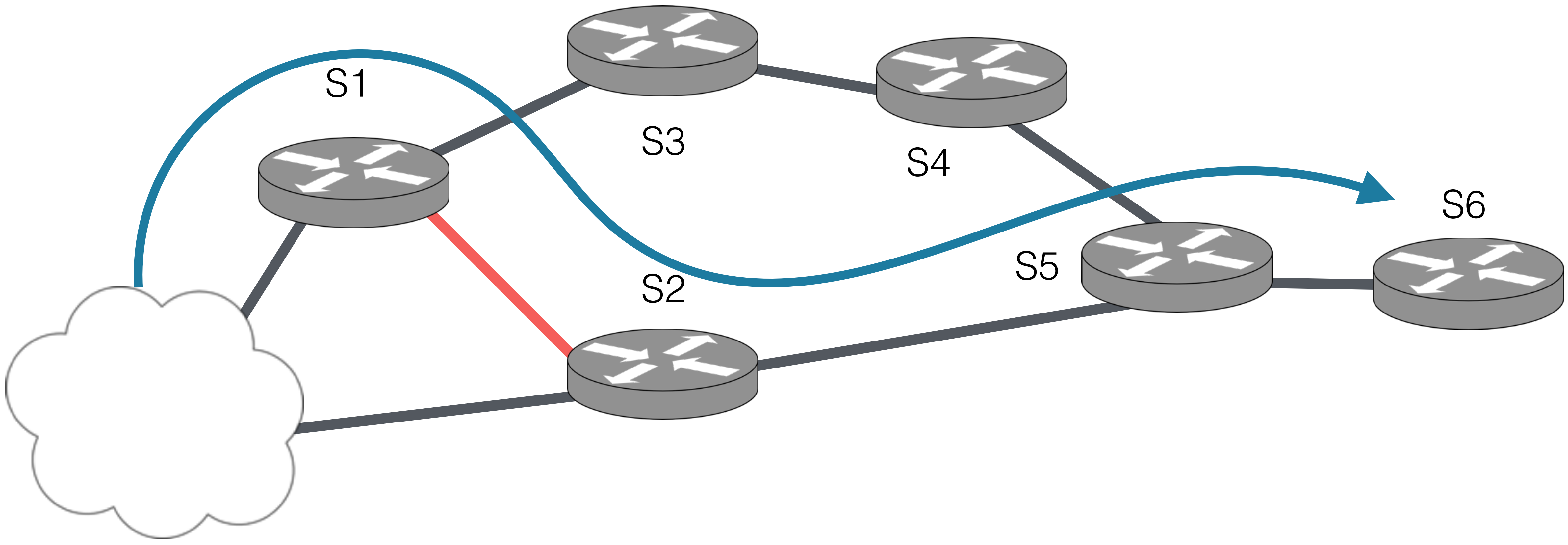
Routing Debugging Monitoring

Controller

Security Policy Verifier

Example: Debugging/Monitoring

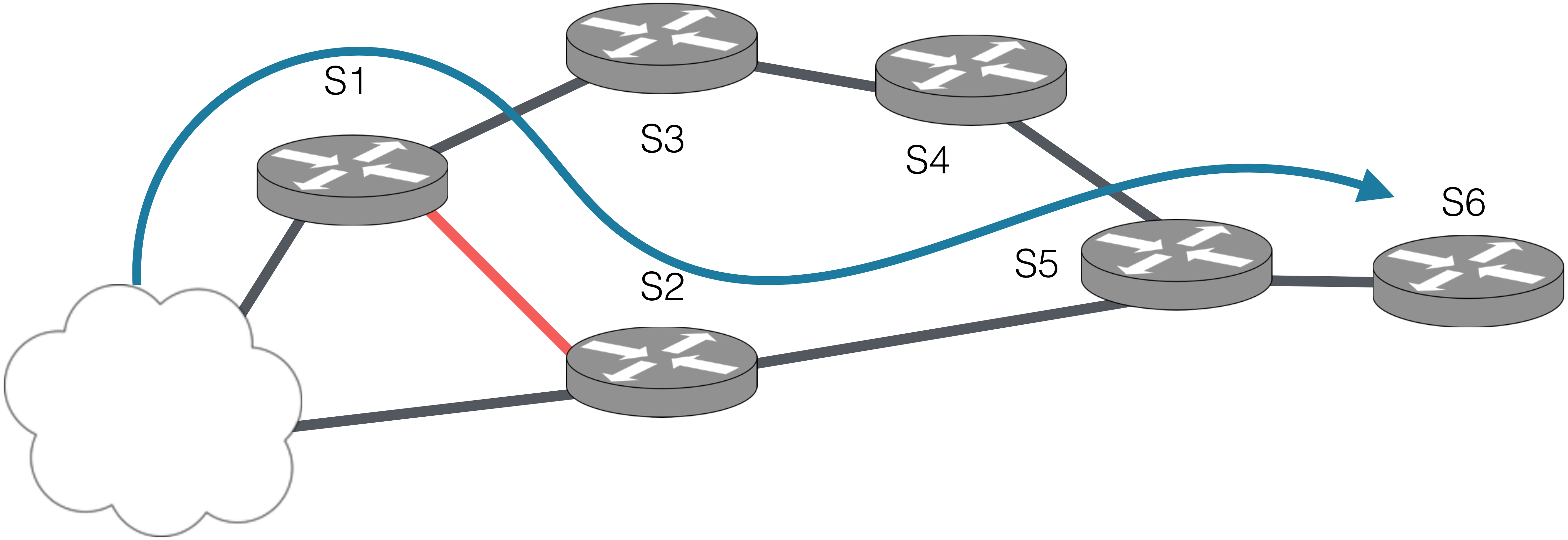
Determine flows utilizing a congested link



Example: Debugging/Monitoring

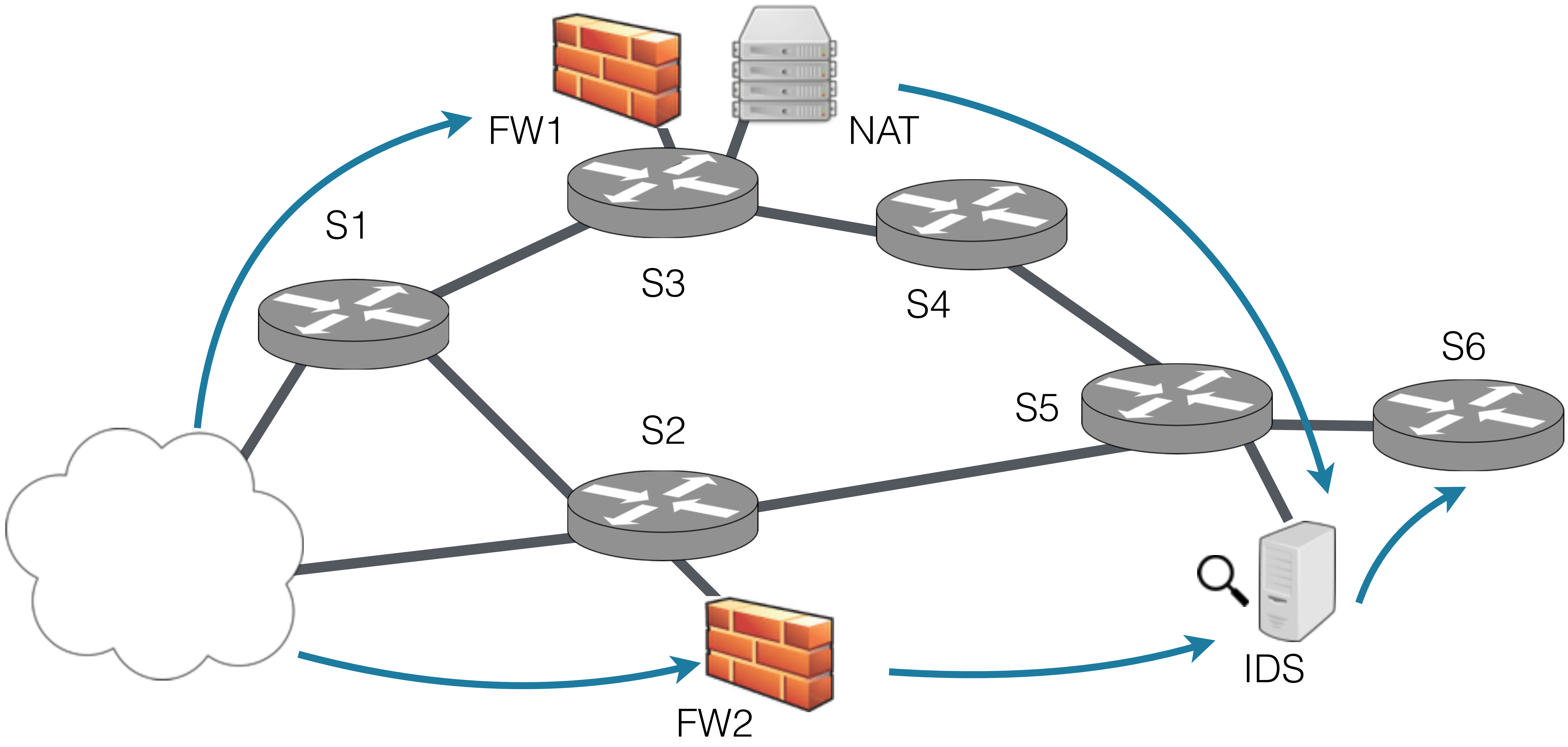
Determine flows utilizing a congested link

$$\texttt{pol} + \texttt{sw=S6}\cdot\Diamond(\texttt{sw=S2}\cdot\bigcirc(\texttt{sw=S1}))\cdot\texttt{pt}\leftarrow\texttt{ctrl}$$



Example: Security

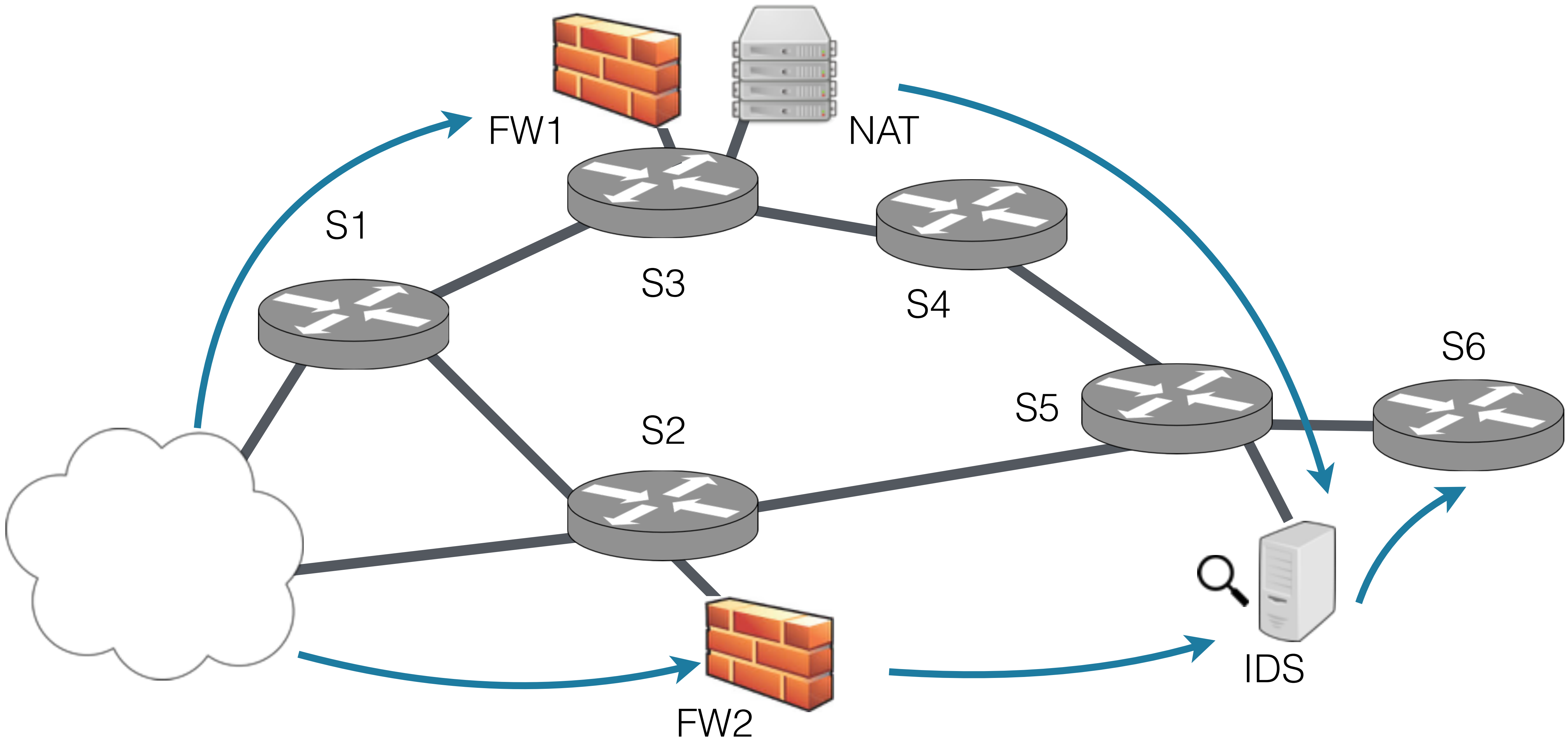
Ensure all traffic arriving at S6 went through a **FW** and **IDS**



Example: Security

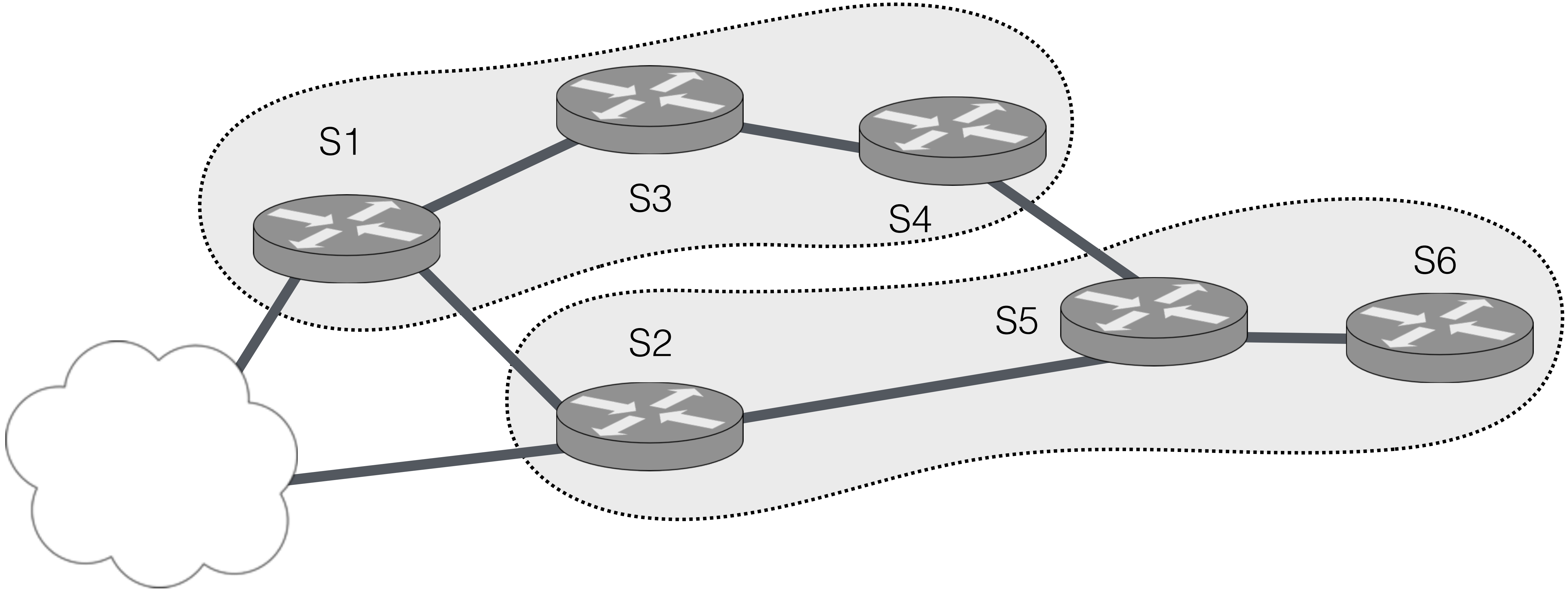
Ensure all traffic arriving at S6 went through a **FW** and **IDS**

`sw=S6 • ◇(sw=FW) • ◇(sw=IDS)`



Example: Isolation

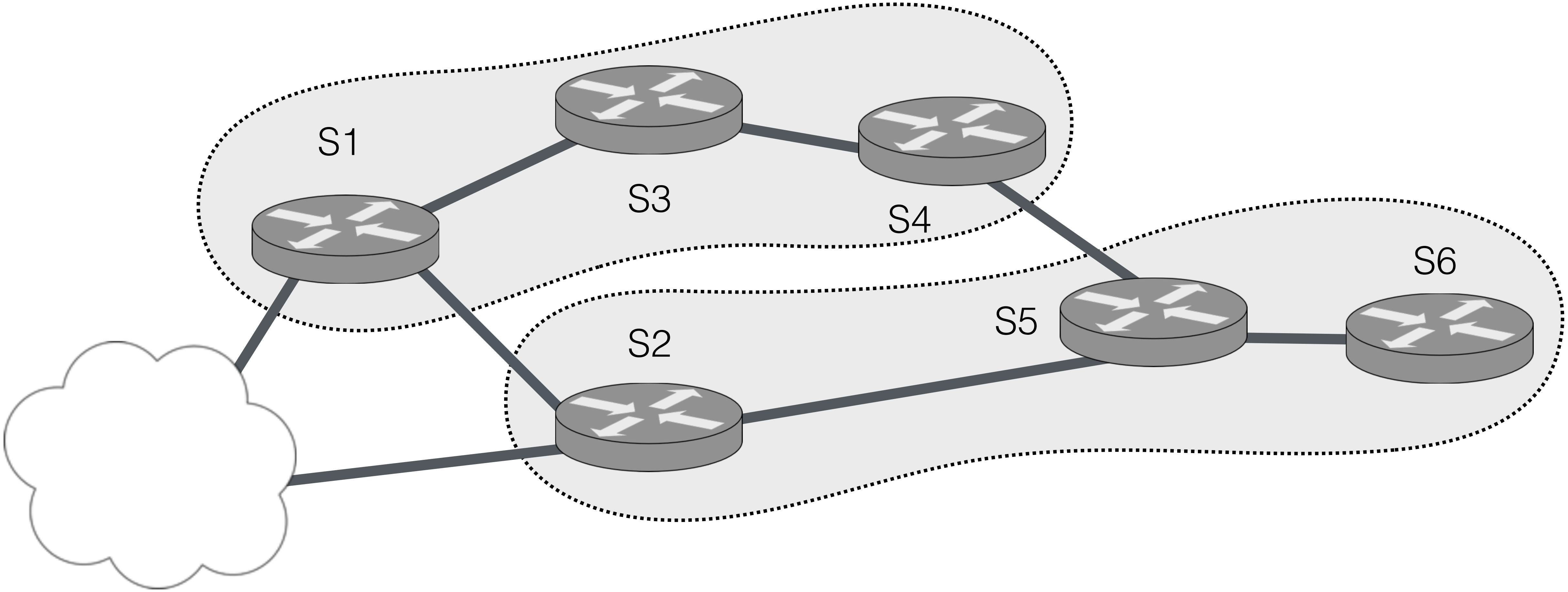
Enforce physical isolation of **S1**, **S3**, **S4** from **S2**, **S5**, **S6**



Example: Isolation

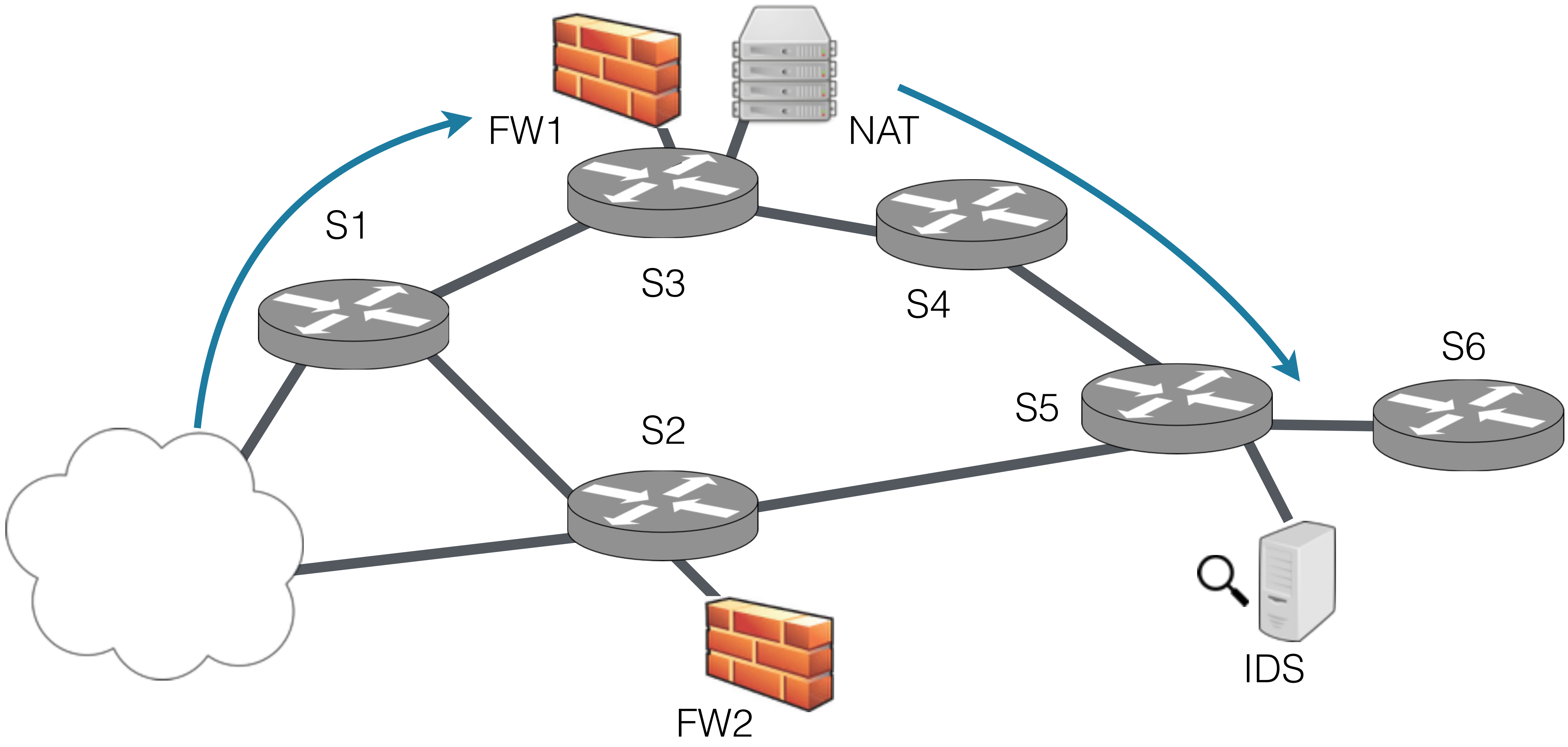
Enforce physical isolation of **S1**, **S3**, **S4** from **S2**, **S5**, **S6**

$$\mathtt{pol}\cdot(\Box(\mathtt{sw}=\mathtt{S}_1+\mathtt{sw}=\mathtt{S}_3+\mathtt{sw}=\mathtt{S}_4)\ +\ \Box(\mathtt{sw}=\mathtt{S}_2+\mathtt{sw}=\mathtt{S}_5+\mathtt{sw}=\mathtt{S}_6))$$



Example: Verification

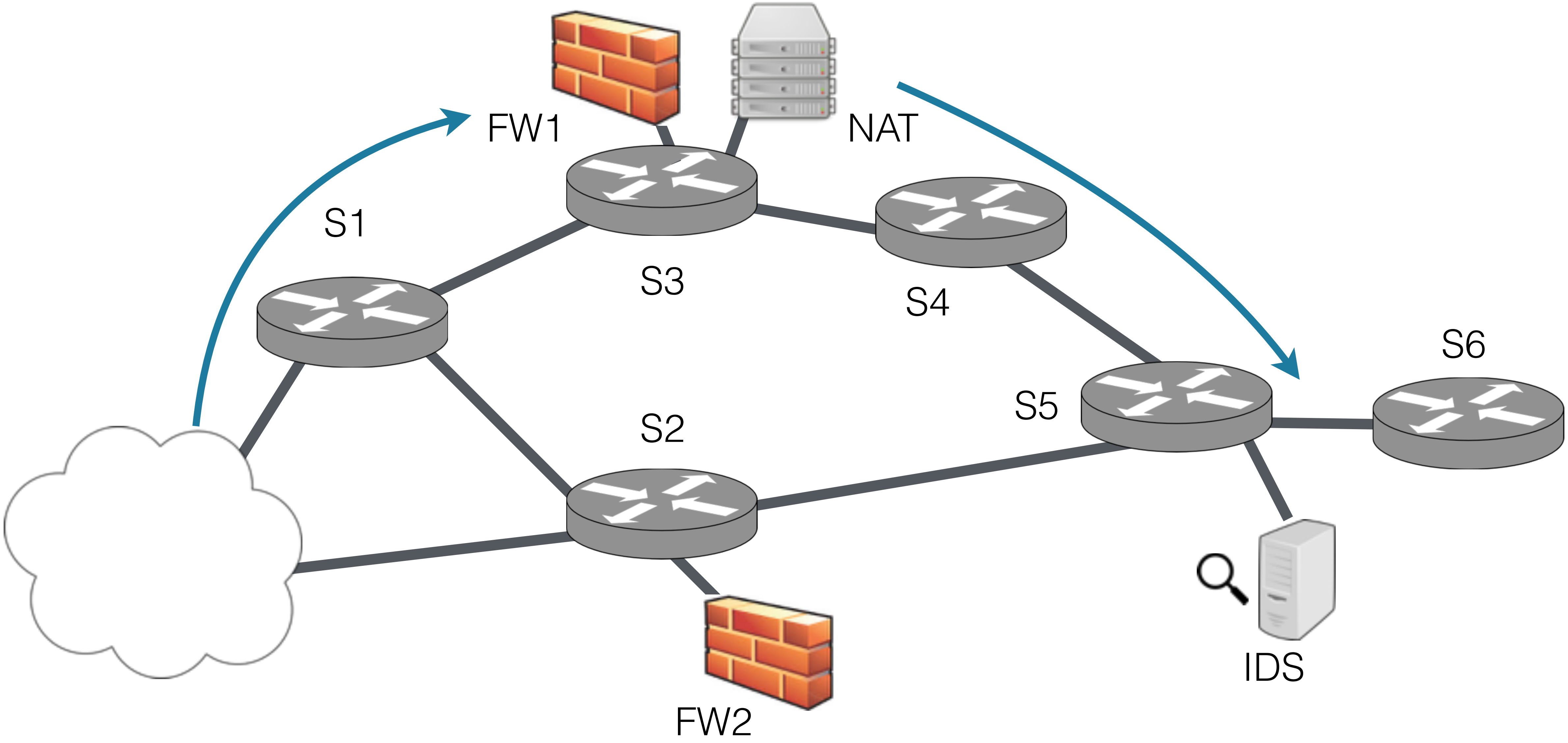
Does the NAT always modify the dst IP address correctly?



Example: Verification

Does the NAT always modify the dst IP address correctly?

$$\texttt{pol} \equiv \texttt{pol}\cdot(\texttt{(dst=10.0.0.17)}\ \mathcal{S}\ \texttt{(sw=NAT)})$$



Questions

Reasoning

- How **expressive** is Temporal NetKAT?
- When are two programs **equivalent**?

Compilation

- How to **compile** Temporal NetKAT to switch rules?
- Can we **scale** compilation to realistic topologies/policies?

Reasoning

Equational Theory

Kleene Algebra Axioms

Idempotent Semiring Laws

$(p+q)r \equiv pr+qr$ $\qquad p+p \equiv p$

$p+q \equiv q+p$ $\qquad 1p \equiv p1 \equiv p$

$p+0 \equiv p$ $\qquad p0 \equiv 0p \equiv 0$

$p(q+r) \equiv pq+pr$ $\qquad p(qr) \equiv (pq)r$

$p+(q+r) \equiv (p+q)+r$

*Axioms for **

$p^* \equiv 1+pp^*$ $\qquad q+px \leq x \Rightarrow p^*q \leq x$

$p^* \equiv 1+p^*p$ $\qquad q+px \leq x \Rightarrow p^*q \leq x$

Boolean Algebra Axioms

$aa \equiv a$

$a \cdot \neg a \equiv 0$

$a + 1 \equiv a$

$a + \neg a \equiv 1$

$(p + q)r \equiv pr + qr$

$a + bc \equiv (a + b)(a + c)$

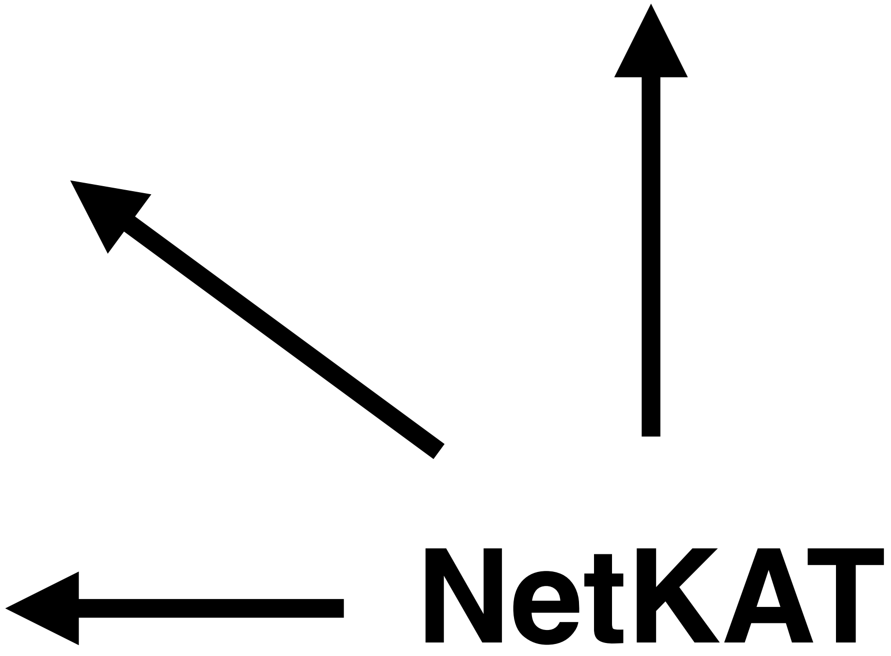
Packet Axioms

$\sum (f = v) \equiv 1$

$(f = v)\cdot(f' = v') \equiv 0$

$(f \leftarrow v)\cdot(f = v) \equiv f \leftarrow n$

$(f \leftarrow v)\cdot(f' = v') \equiv (f' = v')\cdot(f \leftarrow v)$



NetKAT

Equational Theory

Kleene Algebra Axioms

Idempotent Semiring Laws

$$(p+q)r \equiv pr+qr \qquad p+p \equiv p$$

$$p+q \equiv q+p \qquad 1p \equiv p1 \equiv p$$

$$p+0 \equiv p \qquad p0 \equiv 0p \equiv 0$$

$$p(q+r) \equiv pq+pr \qquad p(qr) \equiv (pq)r$$

$$p+(q+r) \equiv (p+q)+r$$

*Axioms for **

$$p^* \equiv 1+pp^* \qquad q+px \leq x \Rightarrow p^*q \leq x$$

$$p^* \equiv 1+p^*p \qquad q+px \leq x \Rightarrow p^*q \leq x$$

Packet Axioms

$$\sum (f = v) \equiv 1$$

$$(f = v)\cdot(f' = v') \equiv 0$$

$$(f \leftarrow v)\cdot(f = v) \equiv f \leftarrow n$$

$$(f \leftarrow v)\cdot(f' = v') \equiv (f' = v')\cdot(f \leftarrow v)$$

Boolean Algebra Axioms

$$aa \equiv a$$

$$a\cdot\neg a \equiv 0$$

$$a + 1 \equiv a$$

$$a + \neg a \equiv 1$$

$$(p + q)r \equiv pr + qr$$

$$a + bc \equiv (a + b)(a + c)$$

LTL$_f$ Axioms

$$\bullet 1 \equiv 1$$

$$\bigcirc(a+b) \equiv \bigcirc a + \bigcirc b$$

$$\bigcirc(a\cdot b) \equiv \bigcirc a \cdot \bigcirc b$$

$$(a \mathbin{S} b) \equiv b + a\cdot\bigcirc(a \mathbin{S} b)$$

$$\neg(a \mathbin{S} b) \equiv (\neg b) \mathbin{B} (\neg a\cdot\neg b)$$

$$\Box a \leq \Diamond(\mathrm{start}\cdot a)$$

$$(a \leq \bullet a\cdot b) \Rightarrow (a \leq \Box a)$$

Equational Theory

Kleene Algebra Axioms

Idempotent Semiring Laws

$(p+q)r \equiv pr+qr \qquad p+p \equiv p$

$p+q \equiv q+p \qquad 1p \equiv p1 \equiv p$

$p+0 \equiv p \qquad p0 \equiv 0p \equiv 0$

$p(q+r) \equiv pq+pr \qquad p(qr) \equiv (pq)r$

$p+(q+r) \equiv (p+q)+r$

*Axioms for **

$p^* \equiv 1+pp^* \qquad q+px \leq x \Rightarrow p^*q \leq x$

$p^* \equiv 1+p^*p \qquad q+px \leq x \Rightarrow p^*q \leq x$

Packet Axioms

$\sum (f = v) \equiv 1$

$(f = v)\cdot(f' = v') \equiv 0$

$(f \leftarrow v)\cdot(f = v) \equiv f \leftarrow n$

$(f \leftarrow v)\cdot(f' = v') \equiv (f' = v')\cdot(f \leftarrow v)$

Boolean Algebra Axioms

$aa \equiv a$

$a\cdot\neg a \equiv 0$

$a + 1 \equiv a$

$a + \neg a \equiv 1$

$(p + q)r \equiv pr + qr$

$a + bc \equiv (a + b)(a + c)$

LTL$_f$ Axioms

$\bullet 1 \equiv 1$

$\bigcirc(a+b) \equiv \bigcirc a + \bigcirc b$

$\bigcirc(a\cdot b) \equiv \bigcirc a \cdot \bigcirc b$

$(a \mathbin{S} b) \equiv b + a\cdot\bigcirc(a \mathbin{S} b)$

$\neg(a \mathbin{S} b) \equiv (\neg b) \mathbin{B} (\neg a\cdot\neg b)$

$\Box a \leq \Diamond(\mathit{start}\cdot a)$

$(a \leq \bullet a\cdot b) \Rightarrow (a \leq \Box a)$

Step Axiom

$(f \leftarrow v)\cdot\bigcirc a \equiv a\cdot(f \leftarrow v)$

Metatheory

NetKAT:

Soundness: If $\vdash p \equiv q$, then $[\![p]\!] = [\![q]\!]$

Completeness: If $[\![p]\!] = [\![q]\!]$, then $\vdash p \equiv q$

Temporal NetKAT:

Soundness: If $\vdash p \equiv q$, then $[\![p]\!] = [\![q]\!]$

Completeness: If $[\![\mathrm{start} \cdot p]\!] = [\![\mathrm{start} \cdot q]\!]$, then $\vdash \mathrm{start} \cdot p \equiv \mathrm{start} \cdot q$

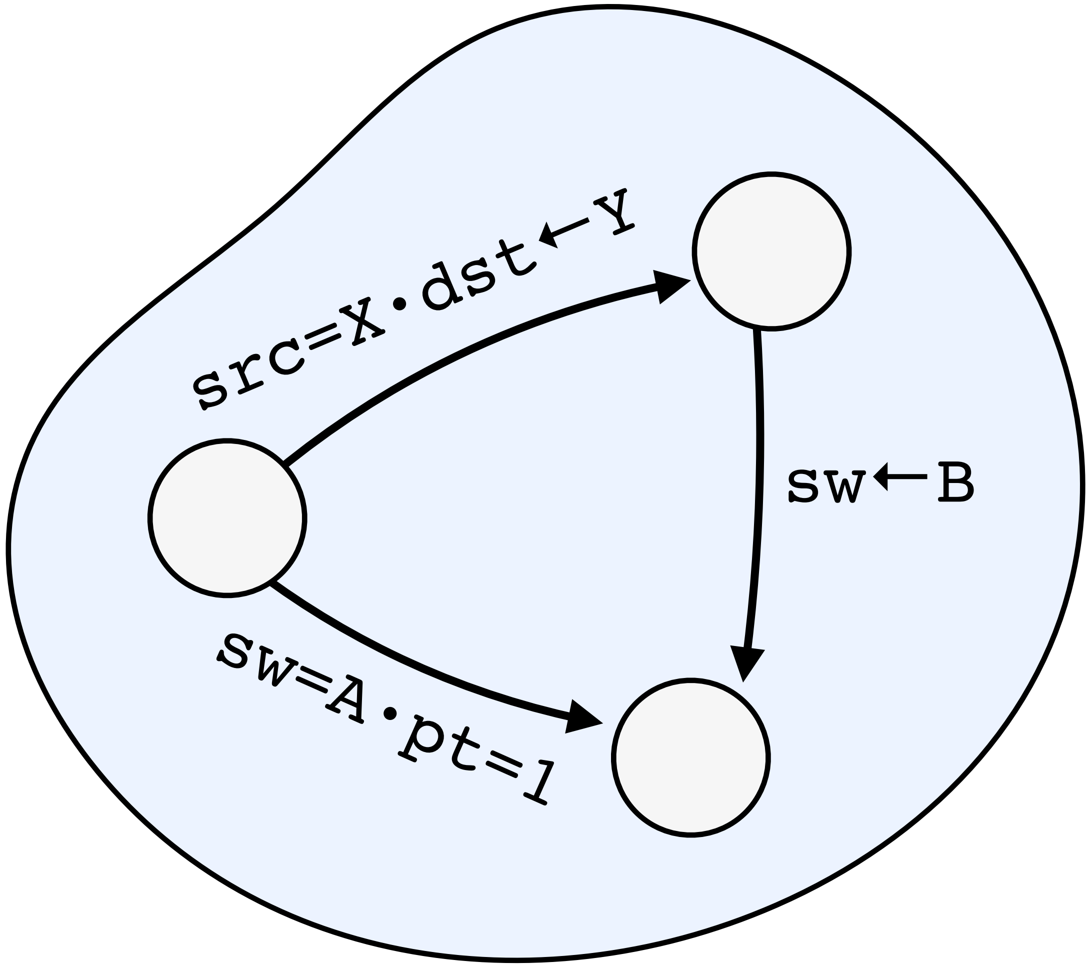
- Completeness for network-wide policies
- Normalization reduces Temporal NetKAT terms to NetKAT
- Interesting induction invariant — see the paper!

Compilation

Compilation

A Fast Compiler for NetKAT [Smolka et al '15]

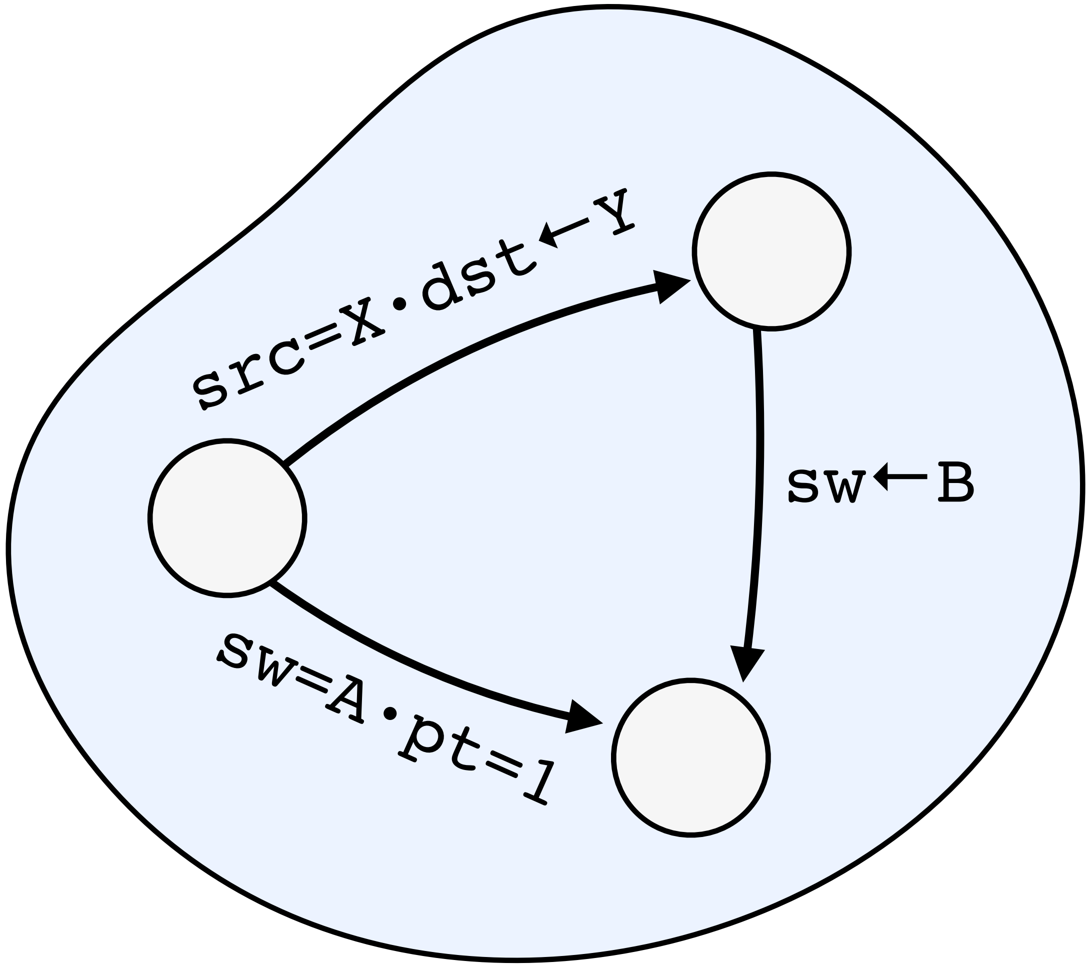
- Symbolic NetKAT automata
- Based on FDDs a variant of BDDs
- Tags the packet with the state of the automaton



Compilation

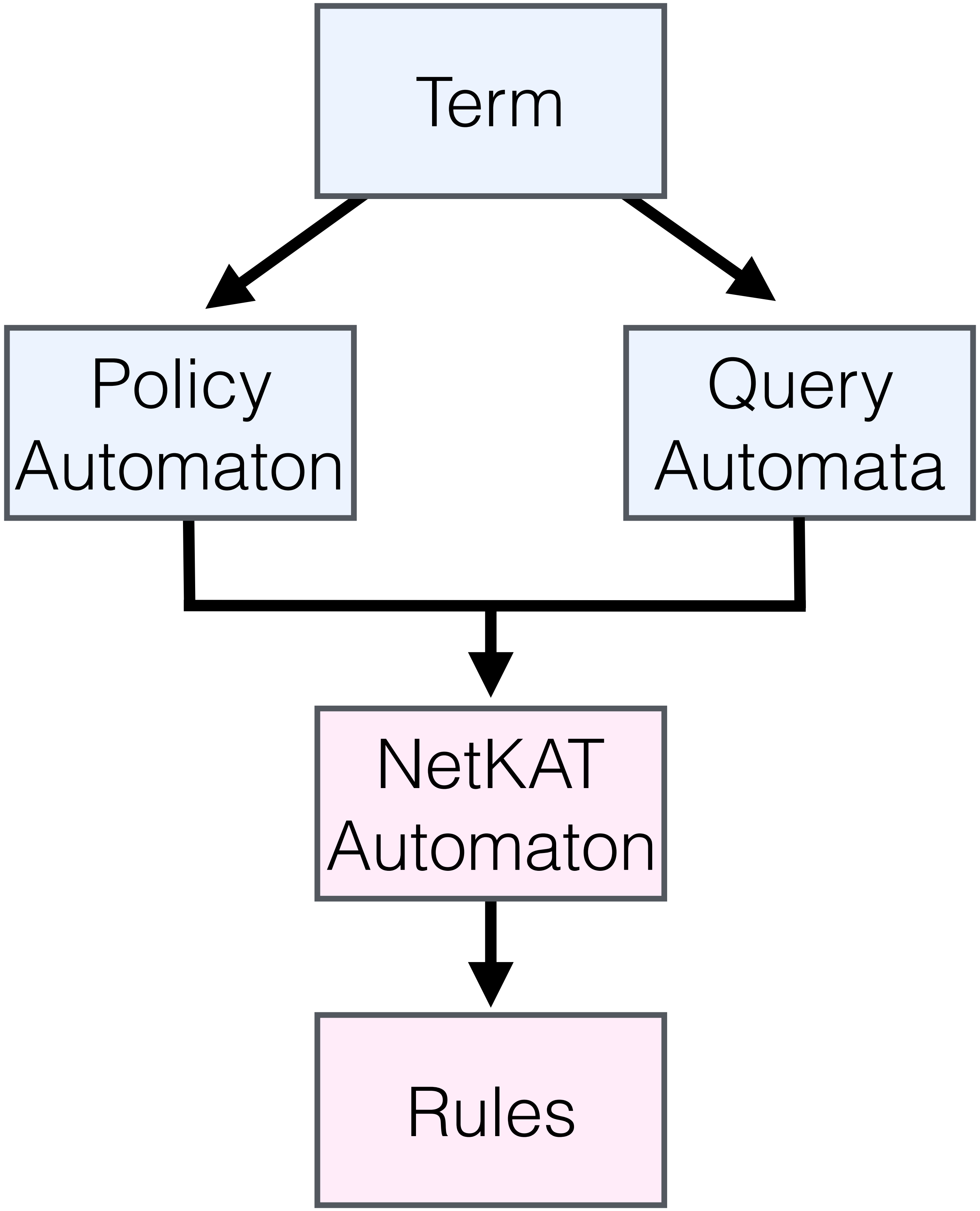
A Fast Compiler for NetKAT [Smolka et al ‘15]

- Symbolic NetKAT automata
- Based on FDDs a variant of BDDs
- Tags the packet with the state of the automaton



High-level Idea:
Reuse tagging mechanism for tracking history as well

Compilation

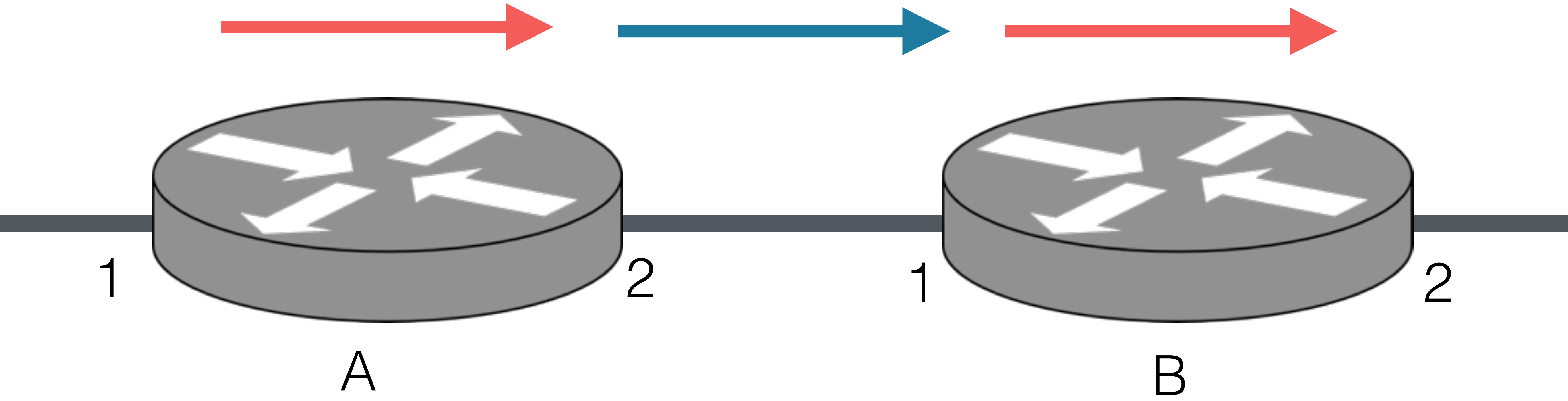


Compilation: Example

$$\mathtt{pol}_{\mathtt{A}} = (\mathtt{sw{=}A} \cdot \mathtt{pt{=}1}) \cdot (\mathtt{pt} \leftarrow \mathtt{2})$$

$$\mathtt{link} = (\mathtt{sw} \leftarrow \mathtt{B}) \cdot (\mathtt{pt} \leftarrow \mathtt{1})$$

$$\mathtt{pol}_{\mathtt{B}} = (\mathtt{sw{=}B} \cdot \mathtt{pt{=}1}) \cdot (\mathtt{pt} \leftarrow \mathtt{2})$$



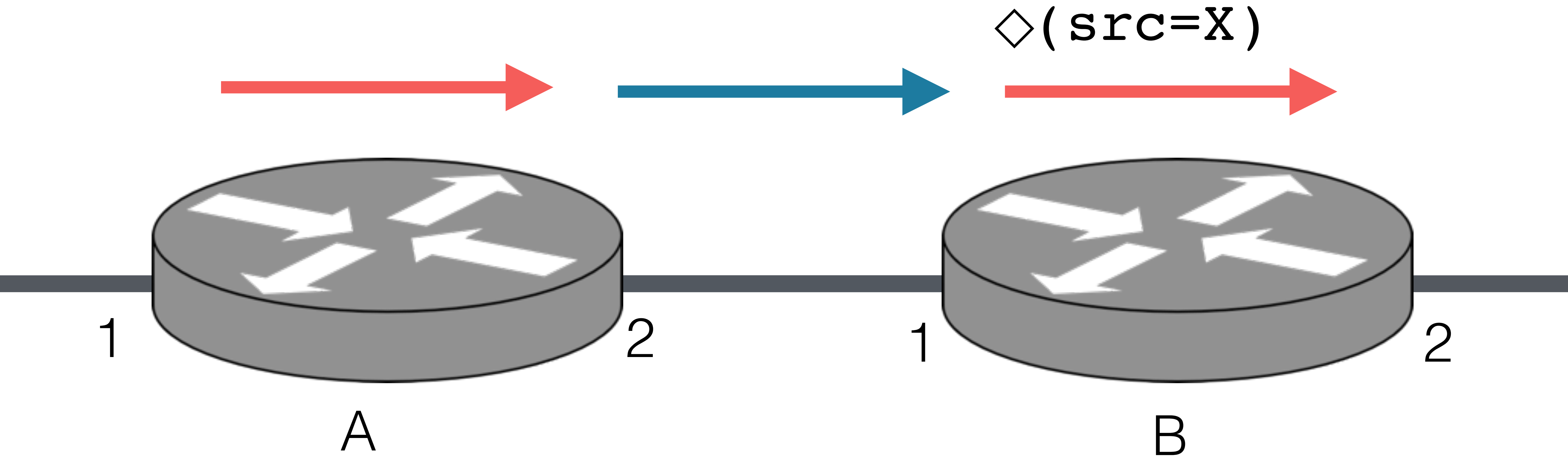
Compilation: Example

$$\mathtt{pol}_{A} = (\mathtt{sw{=}A} \cdot \mathtt{pt{=}1}) \cdot (\mathtt{pt} \leftarrow \mathtt{2})$$

$$\mathtt{link} = (\mathtt{sw} \leftarrow \mathtt{B}) \cdot (\mathtt{pt} \leftarrow \mathtt{1})$$

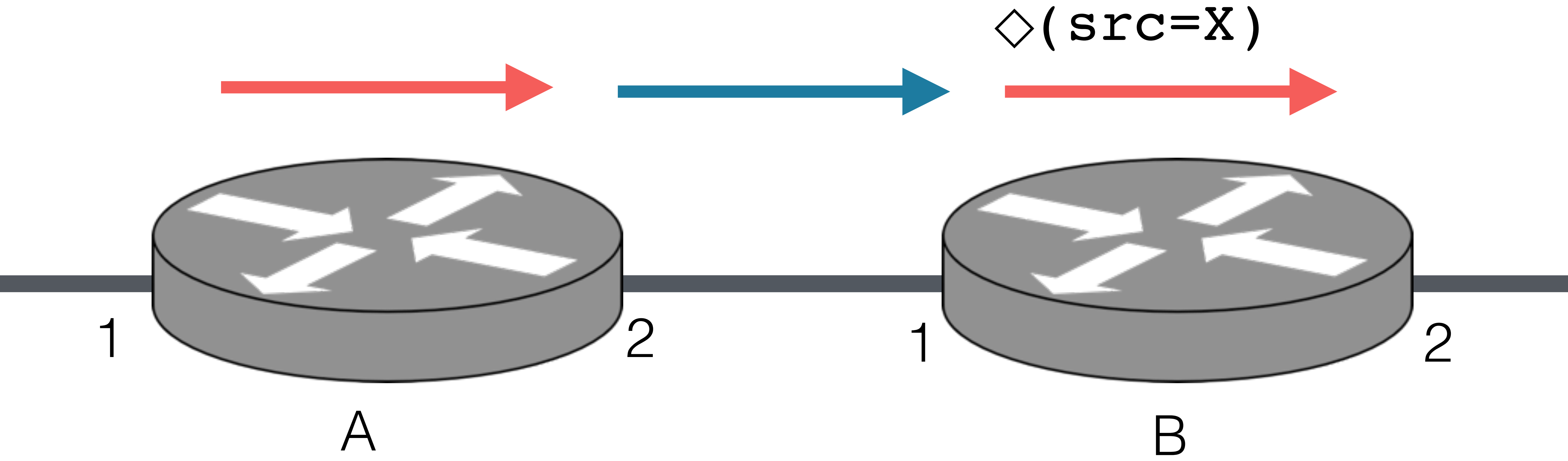
$$\mathtt{pol}_{B} = (\mathtt{sw{=}B} \cdot \mathtt{pt{=}1}) \cdot (\mathtt{pt} \leftarrow \mathtt{2})$$

$$\mathtt{pol} = \mathtt{pol}_{A} \cdot \mathtt{link} \cdot \Diamond(\mathtt{src{=}X}) \cdot \mathtt{pol}_{B}$$



Compilation: Example

$$\texttt{pol}_{A} \cdot \texttt{link} \cdot \Diamond(\texttt{src=X}) \cdot \texttt{pol}_{B}$$



Compilation: Example

$\texttt{pol}_{\texttt{A}} \cdot \texttt{link} \cdot \Diamond(\texttt{src=X}) \cdot \texttt{pol}_{\texttt{B}}$

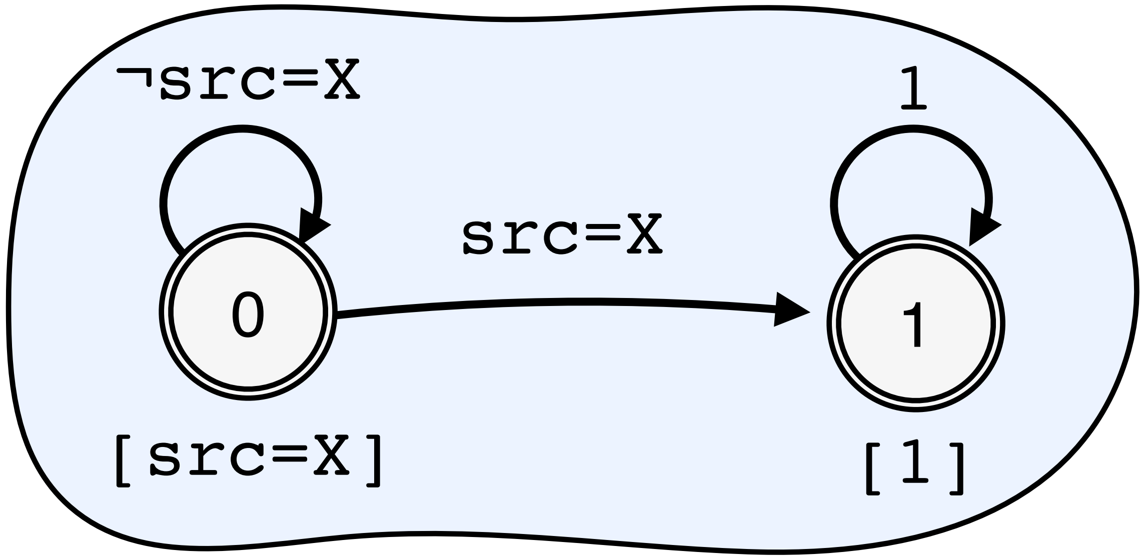
Compilation: Example

$$\begin{tikzcd}
\texttt{pol}_{A} \cdot \texttt{link} \cdot \Diamond(\texttt{src=X}) \cdot \texttt{pol}_{B} \arrow[d, "\textit{abstract predicate}"] \\
\texttt{pol}_{A} \cdot \texttt{link} \cdot \alpha \cdot \texttt{pol}_{B}
\end{tikzcd}$$

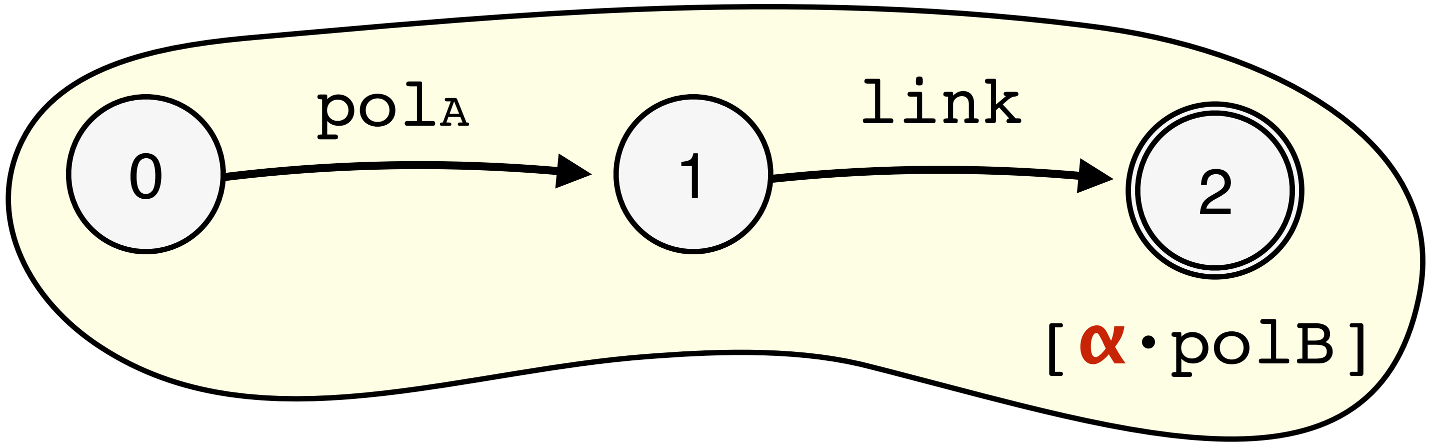
Compilation: Example

polA·link·α·polB

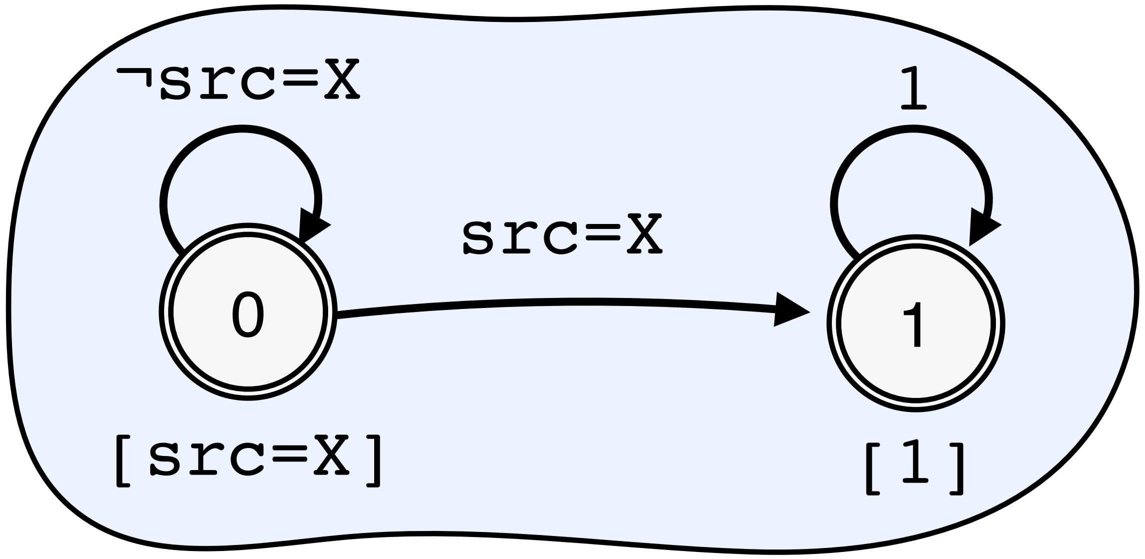
Query Automaton (α)



Policy Automaton

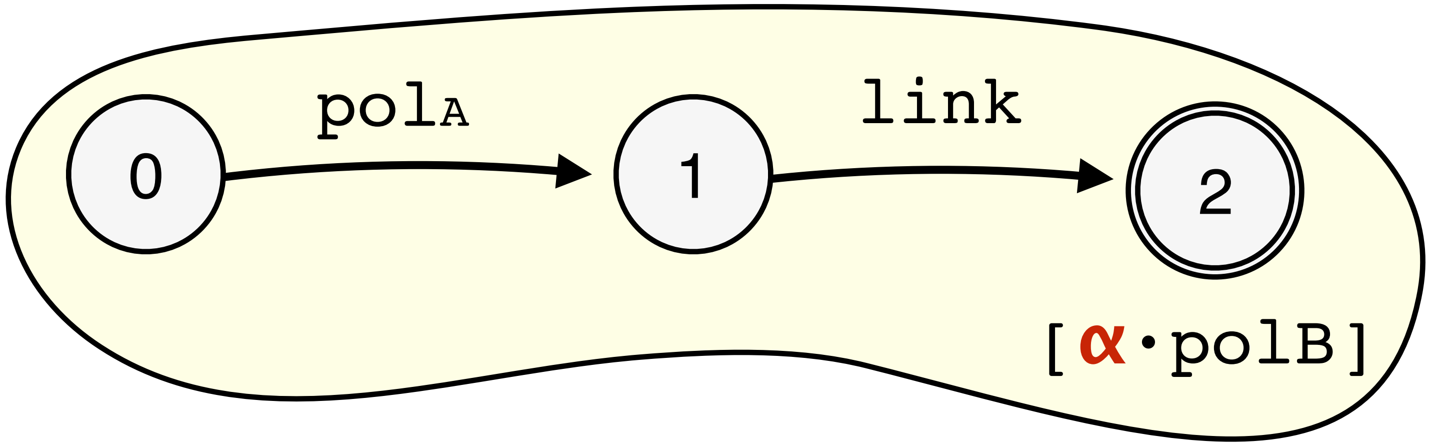


Query Automaton (α)



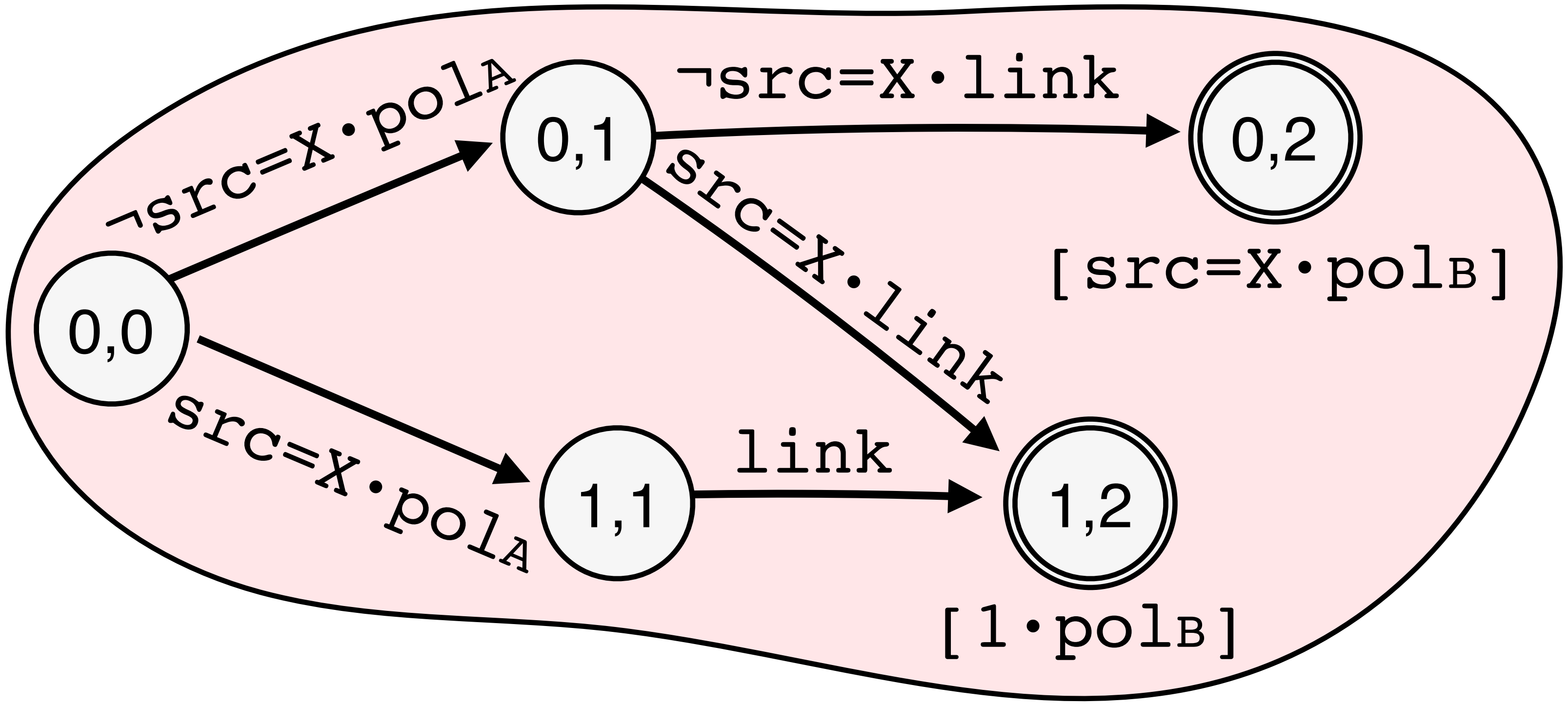
∩

Policy Automaton

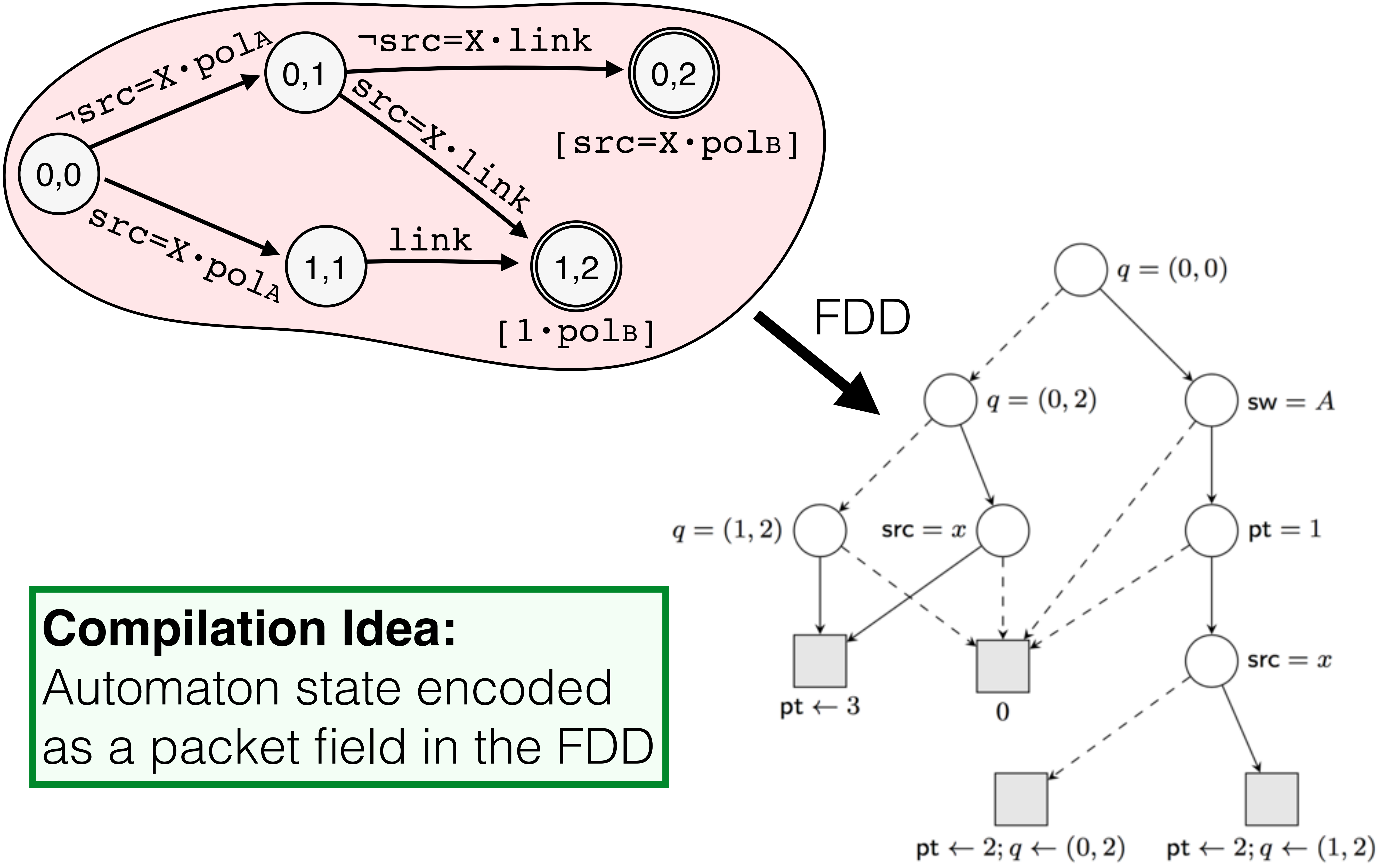


=

Product Automaton

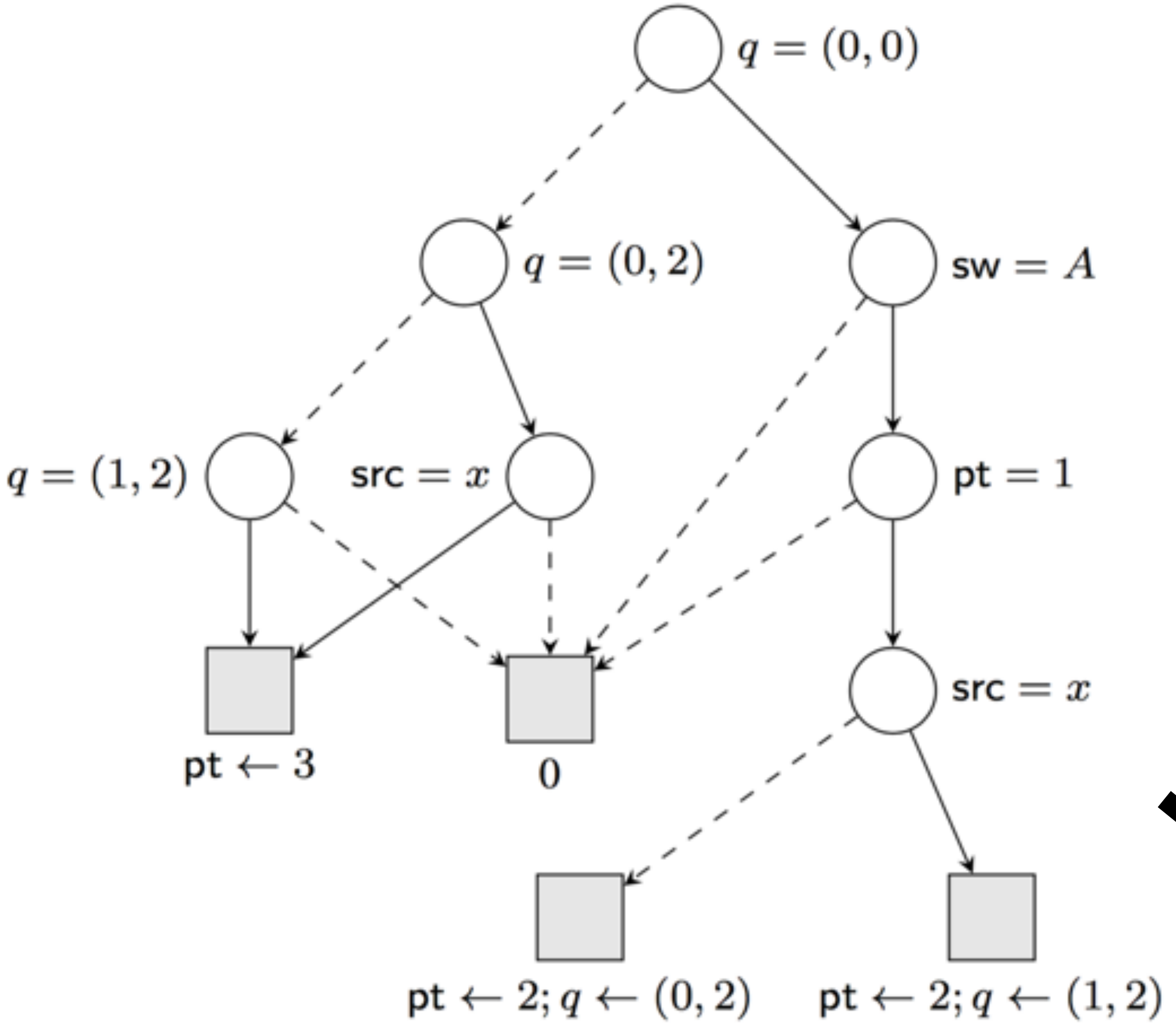


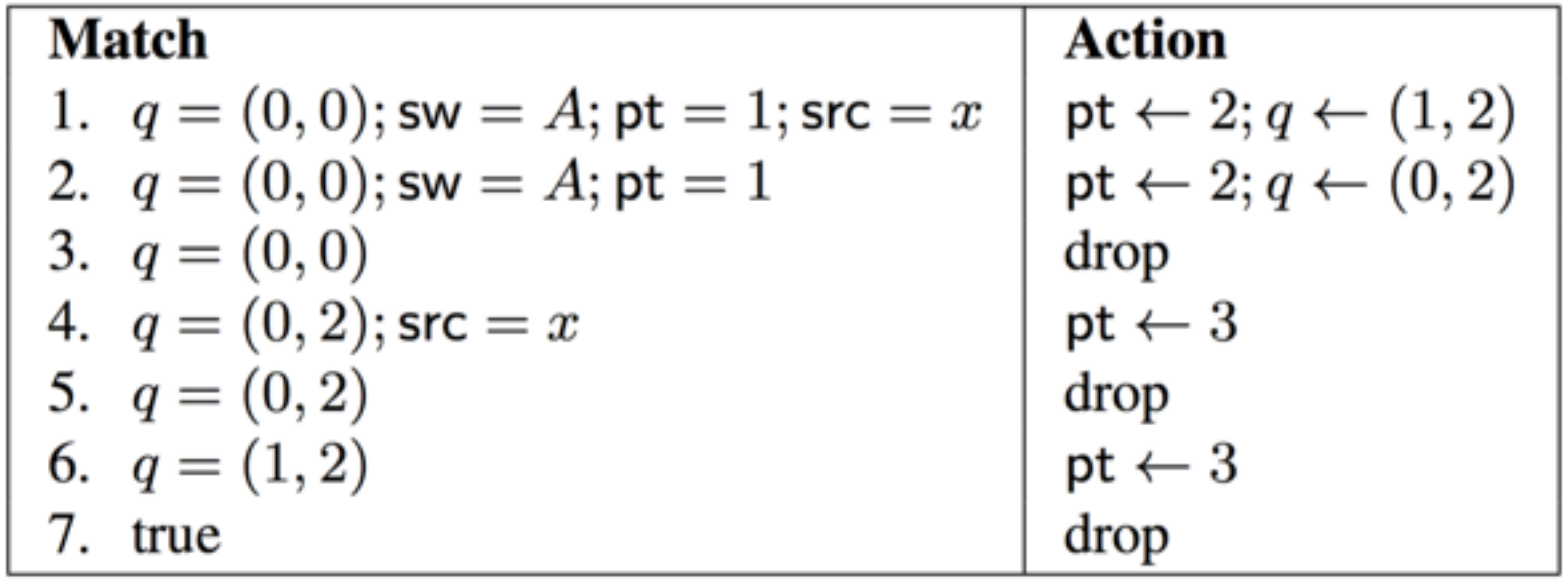
Compilation: Example



Compilation Idea:
Automaton state encoded as a packet field in the FDD

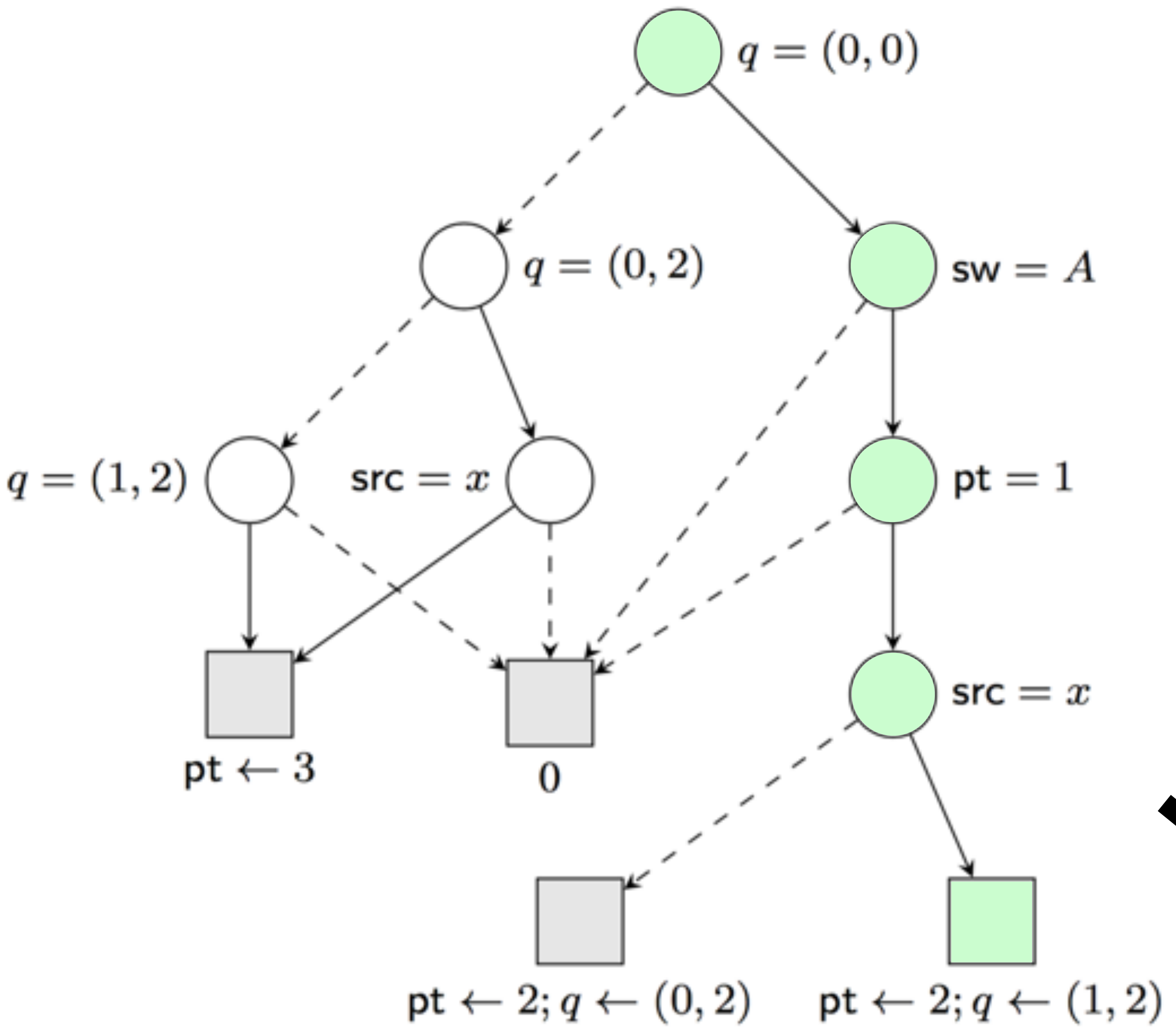
Compilation: Example

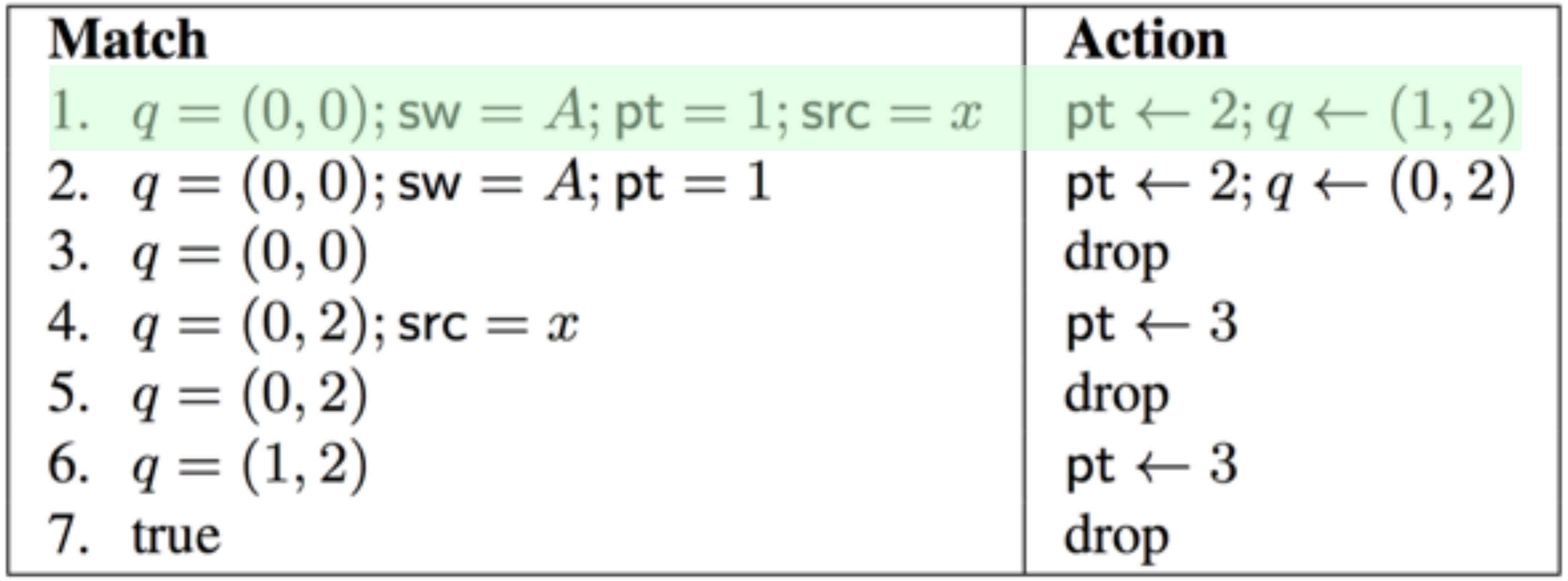




Match	**Action**
1. $q = (0,0)$; sw $= A$; pt $= 1$; src $= x$	pt $\leftarrow 2$; $q \leftarrow (1,2)$
2. $q = (0,0)$; sw $= A$; pt $= 1$	pt $\leftarrow 2$; $q \leftarrow (0,2)$
3. $q = (0,0)$	drop
4. $q = (0,2)$; src $= x$	pt $\leftarrow 3$
5. $q = (0,2)$	drop
6. $q = (1,2)$	pt $\leftarrow 3$
7. true	drop

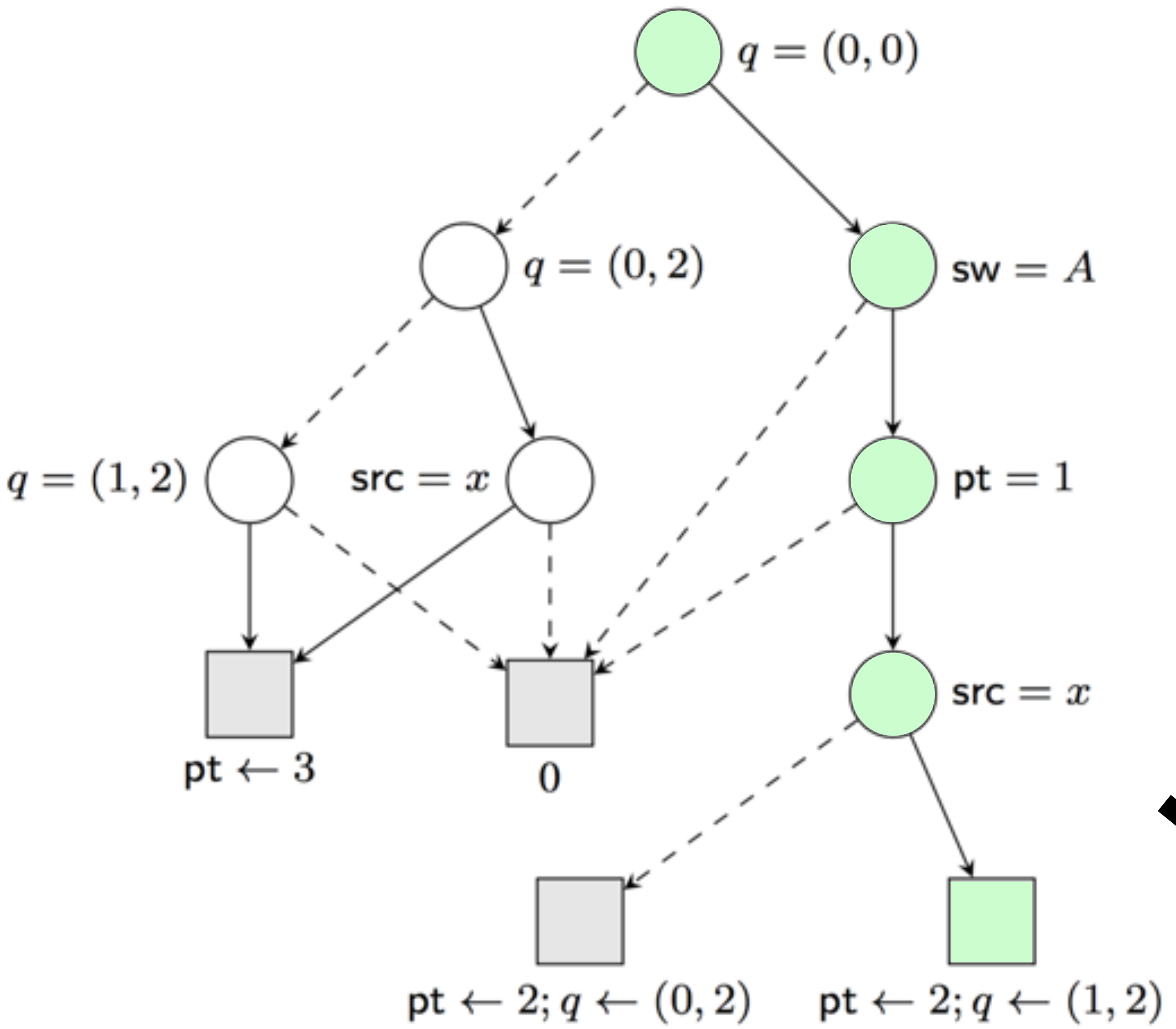
Compilation: Example



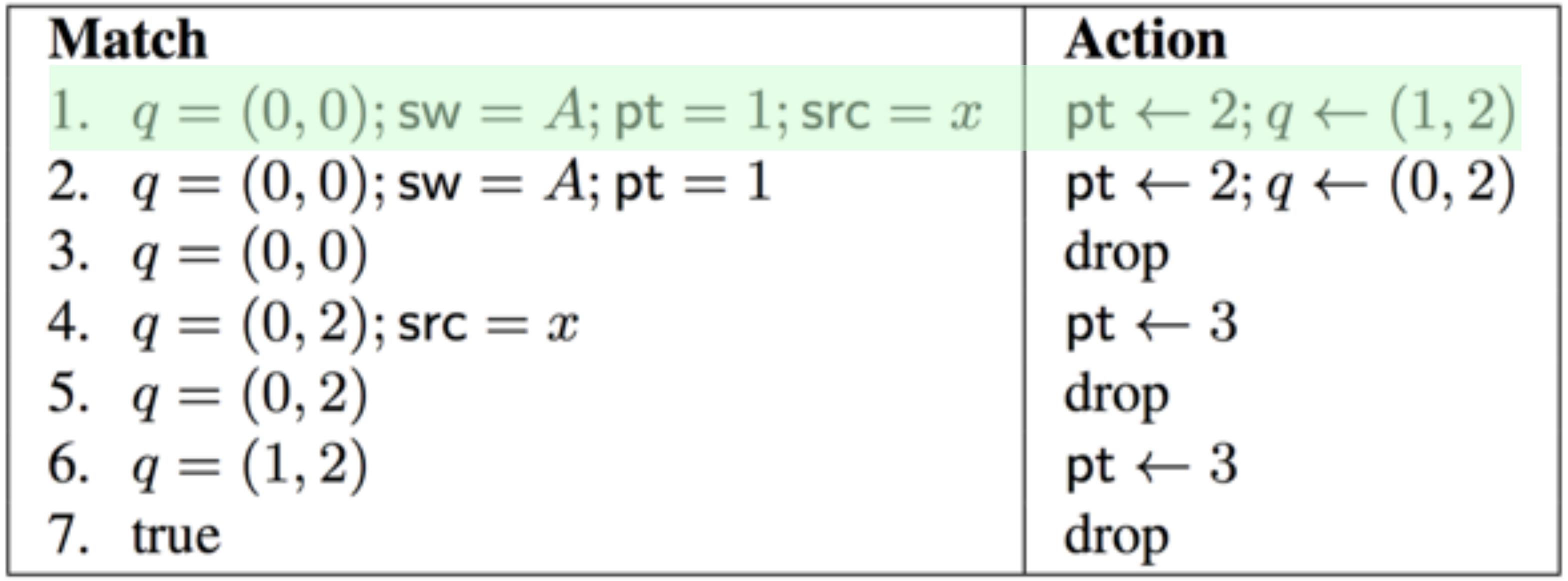


Match	**Action**
1. $q = (0,0)$; sw $= A$; pt $= 1$; src $= x$	pt $\leftarrow 2; q \leftarrow (1,2)$
2. $q = (0,0)$; sw $= A$; pt $= 1$	pt $\leftarrow 2; q \leftarrow (0,2)$
3. $q = (0,0)$	drop
4. $q = (0,2)$; src $= x$	pt $\leftarrow 3$
5. $q = (0,2)$	drop
6. $q = (1,2)$	pt $\leftarrow 3$
7. true	drop

Compilation: Example



See the paper for additional optimizations!



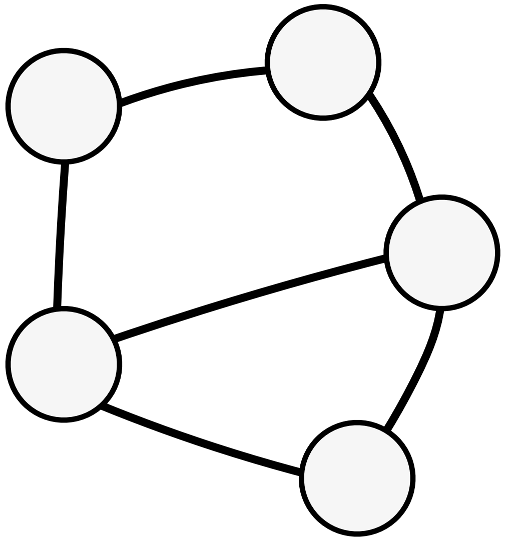
Match	Action
1. $q = (0,0)$; sw $= A$; pt $= 1$; src $= x$	pt $\leftarrow 2; q \leftarrow (1,2)$
2. $q = (0,0)$; sw $= A$; pt $= 1$	pt $\leftarrow 2; q \leftarrow (0,2)$
3. $q = (0,0)$	drop
4. $q = (0,2)$; src $= x$	pt $\leftarrow 3$
5. $q = (0,2)$	drop
6. $q = (1,2)$	pt $\leftarrow 3$
7. true	drop

Evaluation

Compiler Evaluation

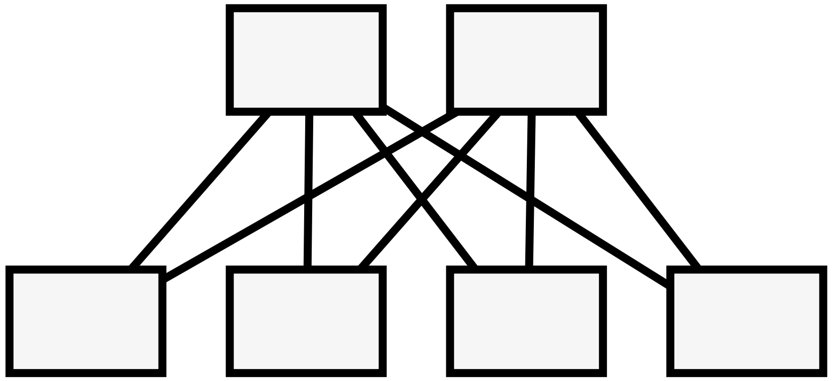
Topology Zoo

- Over 250 real topologies
- Shortest path routing



Stanford Campus Network

- Mid-sized campus network
- 16 core backbone routers
- Rich, non-uniform routing policy



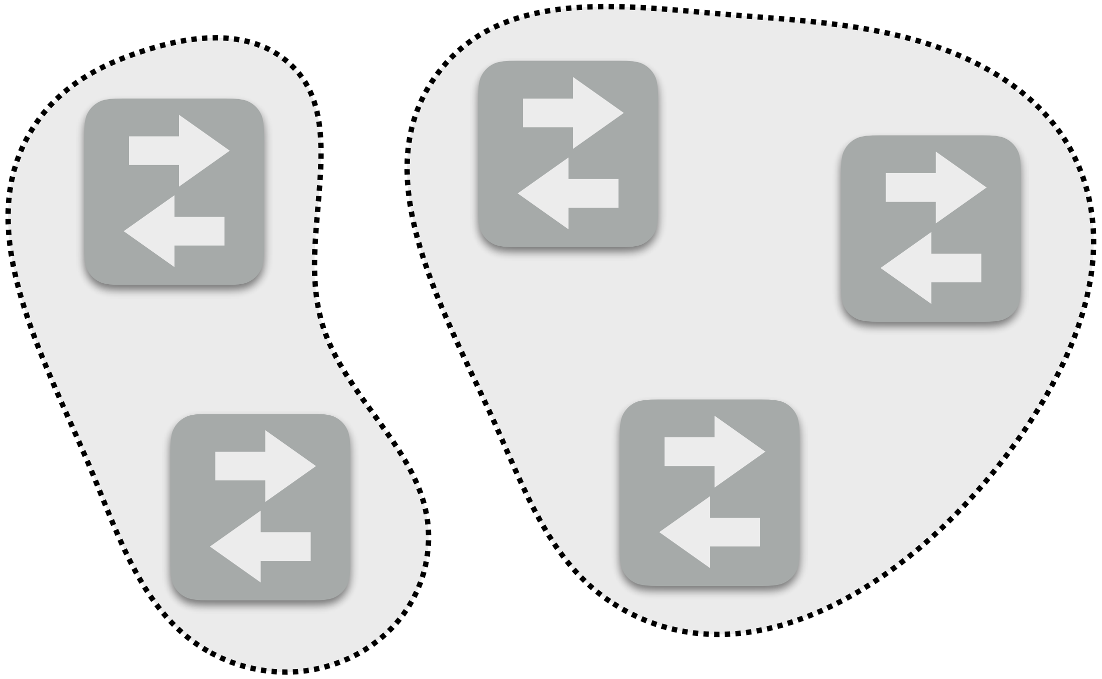
Compiler Evaluation

Baseline:

- Routing only

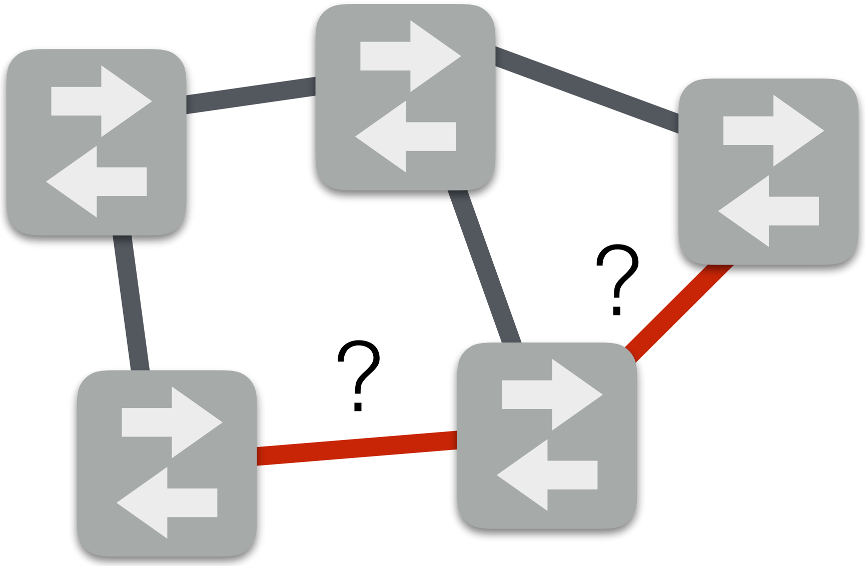
Security:

- Enforce physical isolation
- Enforce logical isolation



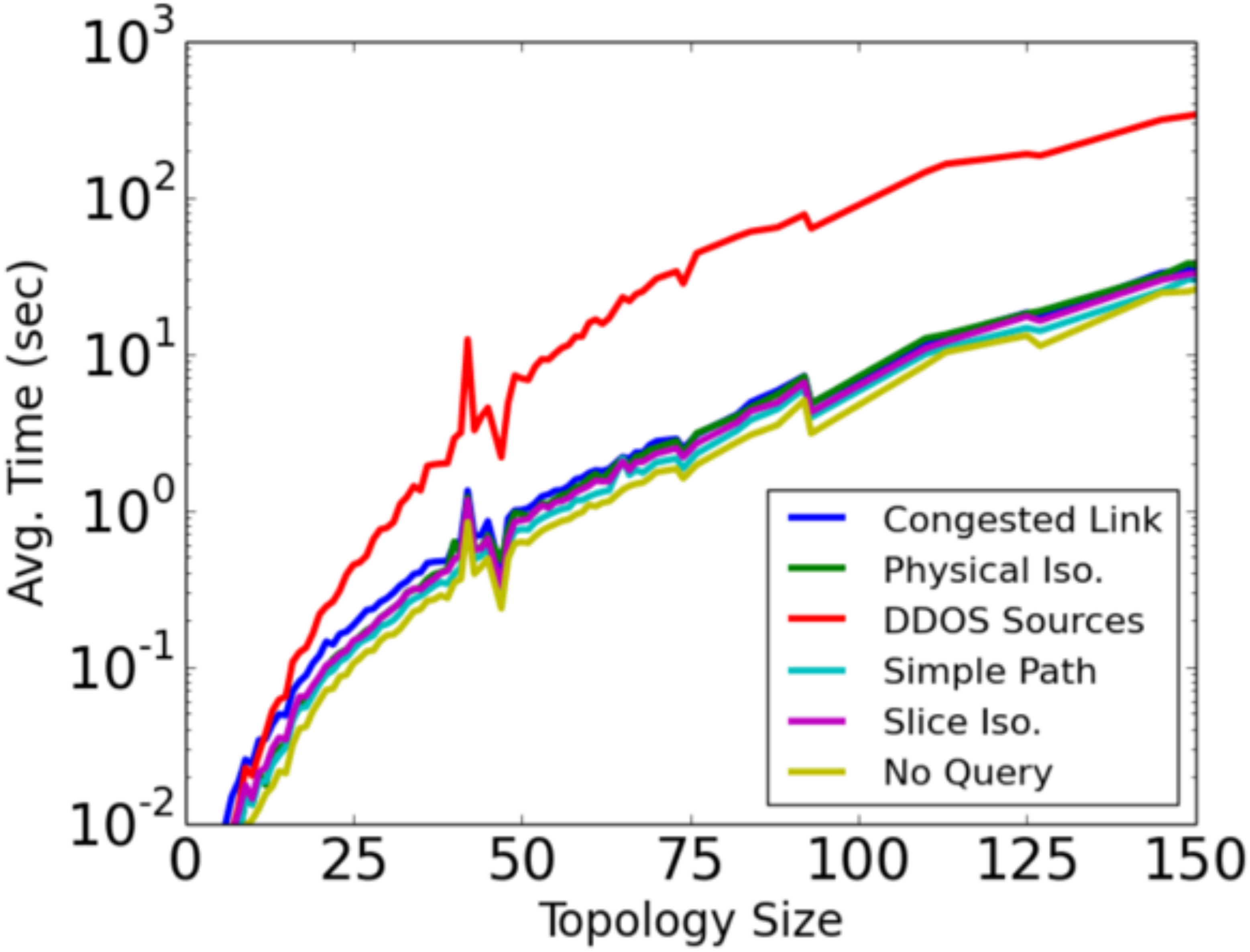
Debugging/Monitoring:

- Congested Link
- Simple path
- Port Matrix
- DDOS sources



Topology Zoo

Compilation Time



Most policies have very little overhead

~12 min worst case

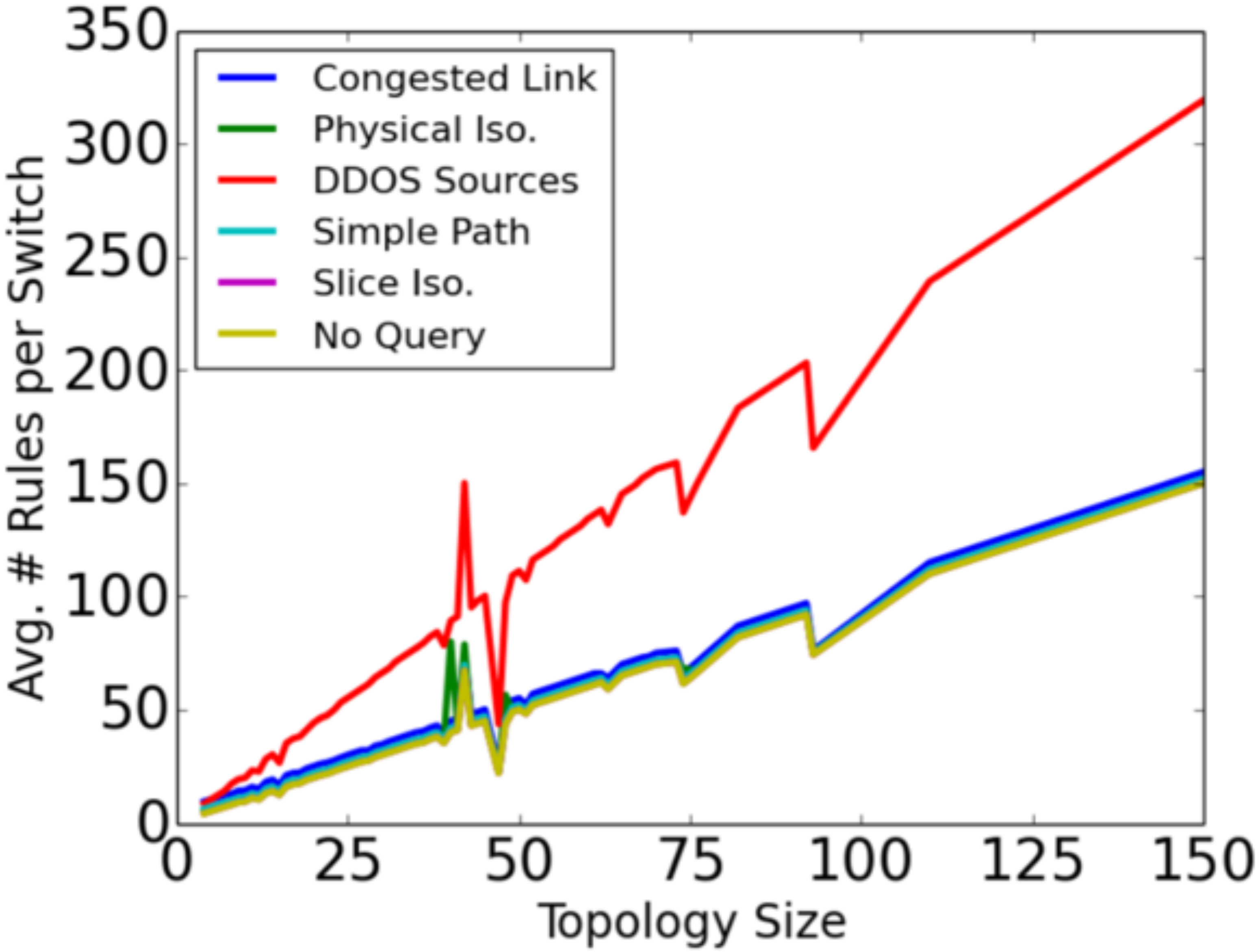
Main limiting factor: number of queries

Topology Zoo

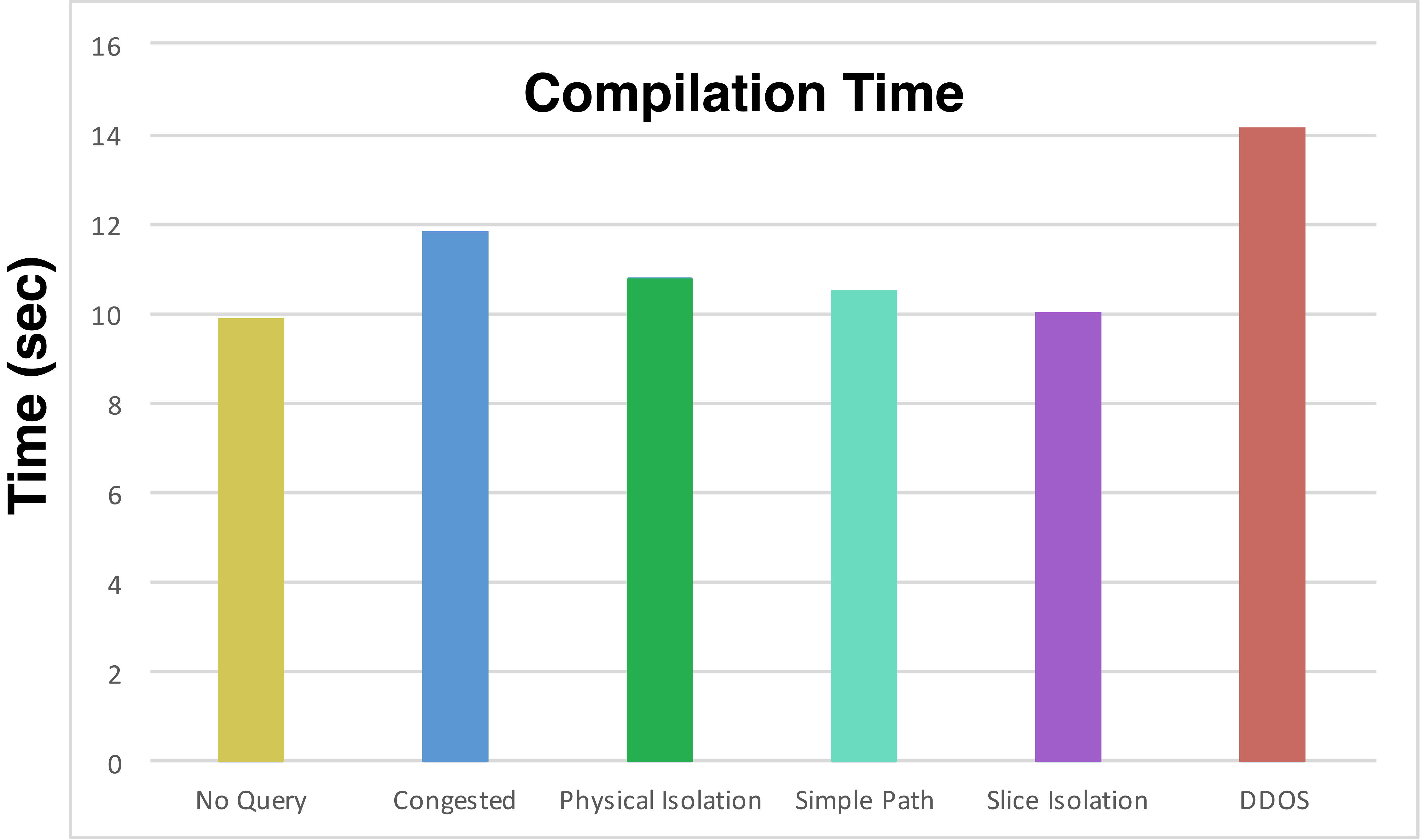
Often near minimal rule overhead

~2x increase with DDOS query

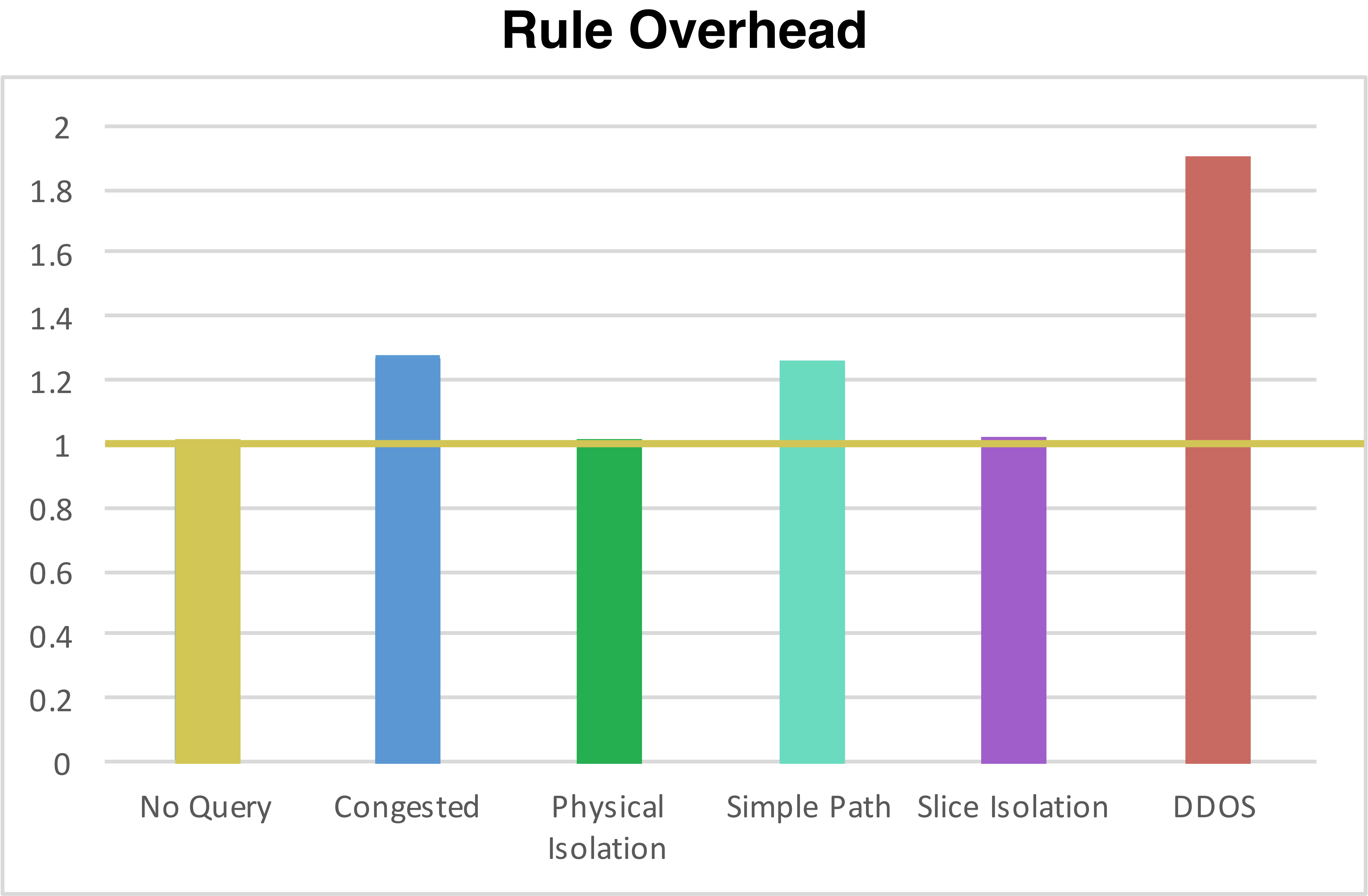
Number of Rules



Stanford Network



Stanford Network



Conclusions

Language

- Extension of NetKAT with **queries over packet history**
- Useful in a variety of network **applications**

Theory

- **Soundness** and **completeness** for network-wide programs
- New proof technique for completeness

Compiler

- Inspired by structure of the completeness proof
- **Scales** to many real network topologies/policies