Safety Verification of Stateful Networks

Sharon Shoham

TEL AVIV UNIVERSITY אוניברסיטת תל אביב

Collaborators

Kalev Alpernas
Mooly Sagiv

Roman Manevich

Yaron Velner

Aurojit Panda
Scott Shenker

TEL AVIV UNIVERSITY אוניברסיטת תל אביב

אוניברסיטת בן-גוריון בנגב

האוניברסיטה העברית בירושלים
THE HEBREW UNIVERSITY OF JERUSALEM

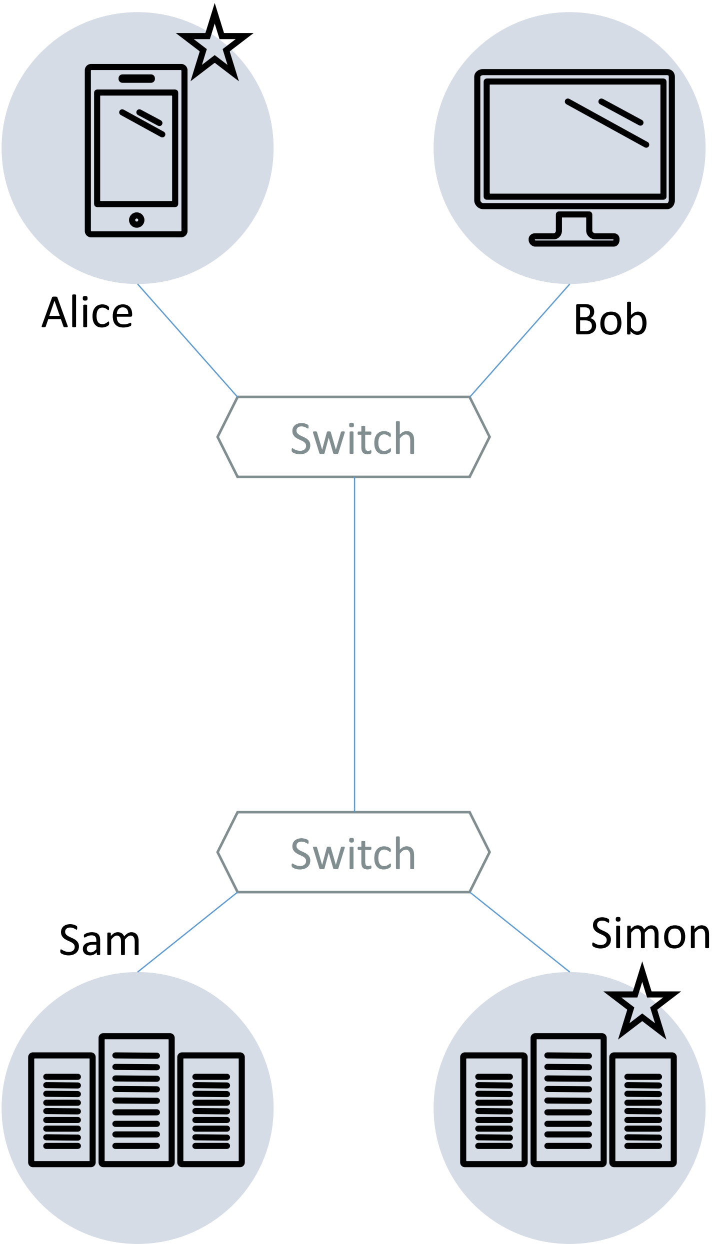
Berkeley
UNIVERSITY OF CALIFORNIA

Network Safety Verification

- Show that something bad cannot happen

- Isolation:
  - A packet of type **t** sent from host **A** never reaches host **B**

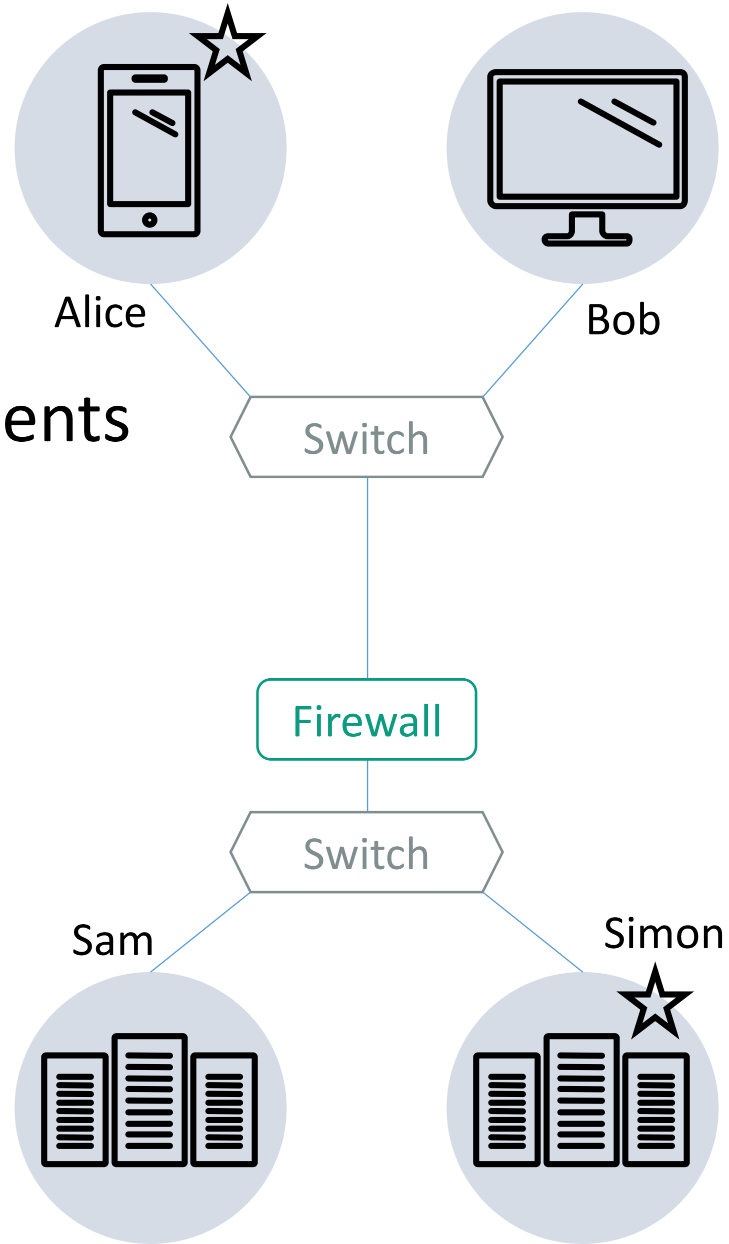
  - E.g., no packets from Simon to Bob



Stateful Networks

Middleboxes: Local functionality enhancements

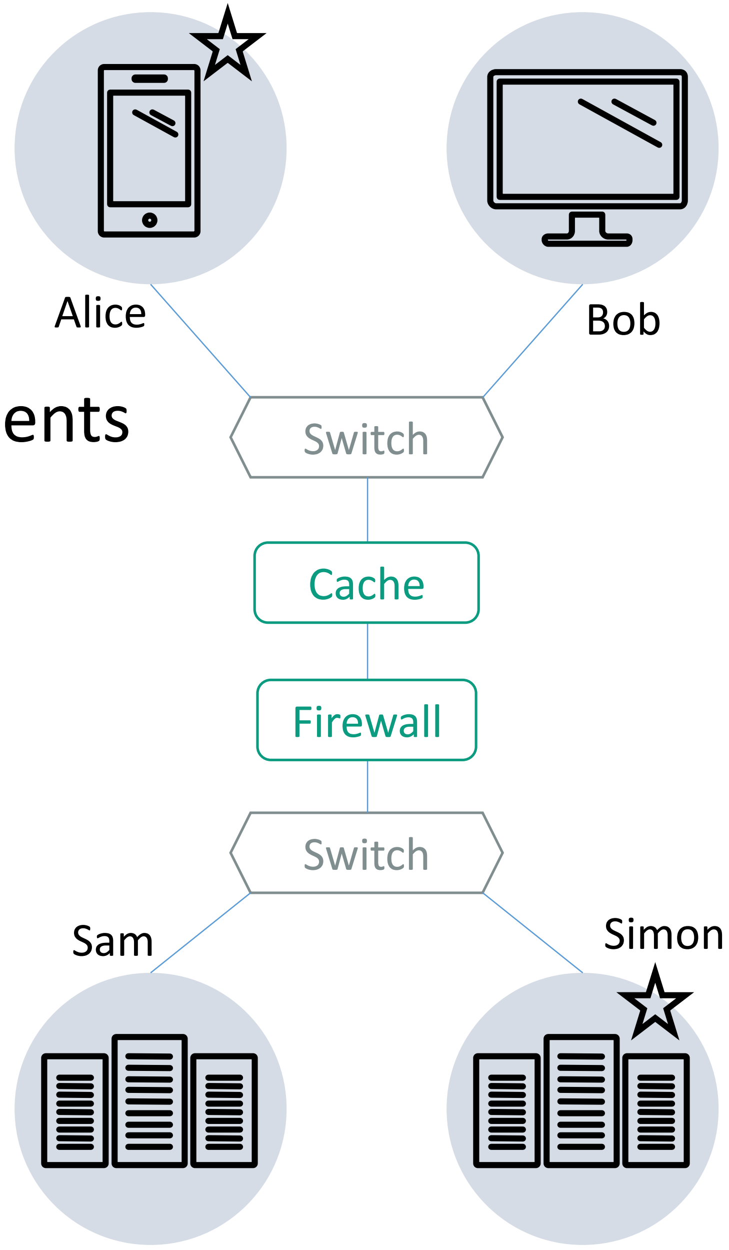
- Security (firewalls, IDSs,...)
- Performance (caches, load balancers,...)
- New functionality (proxies,...)



Stateful Networks

Middleboxes: Local functionality enhancements

- Security (firewalls, IDSs,…)
- Performance (caches, load balancers,…)
- New functionality (proxies,…)



Safety with Middleboxes

- For stateless networks
  - Safety can be checked by tracing the forwarding graph
- Middleboxes make everything harder
  - Rewrite packet headers
  - Behave differently over time – need to reason about history
    - Forwarding of a packet depends on previous packets
    - E.g. cache

Challenges in Stateful Network Verification

- Source code complexity
  - Bro Network Intrusion
    - 101,500 lines of C++, Python, Perl, Awk, Lex, Yacc
  - Snort IDS 220,000 C, …
  - Pfsense 476,438 locs of C,php,scripts,…
- Configuration errors
  - Do the topology and the middlebox configuration satisfy the safety property?
  - Major source of network failures [IMC:RJ13]

[IMC:RJ13] Potharaju R., Jain N. Demystifying the dark side of the middle: a field study of middlebox failures in datacenters. IMC 2013

Our approach for verifying stateful networks

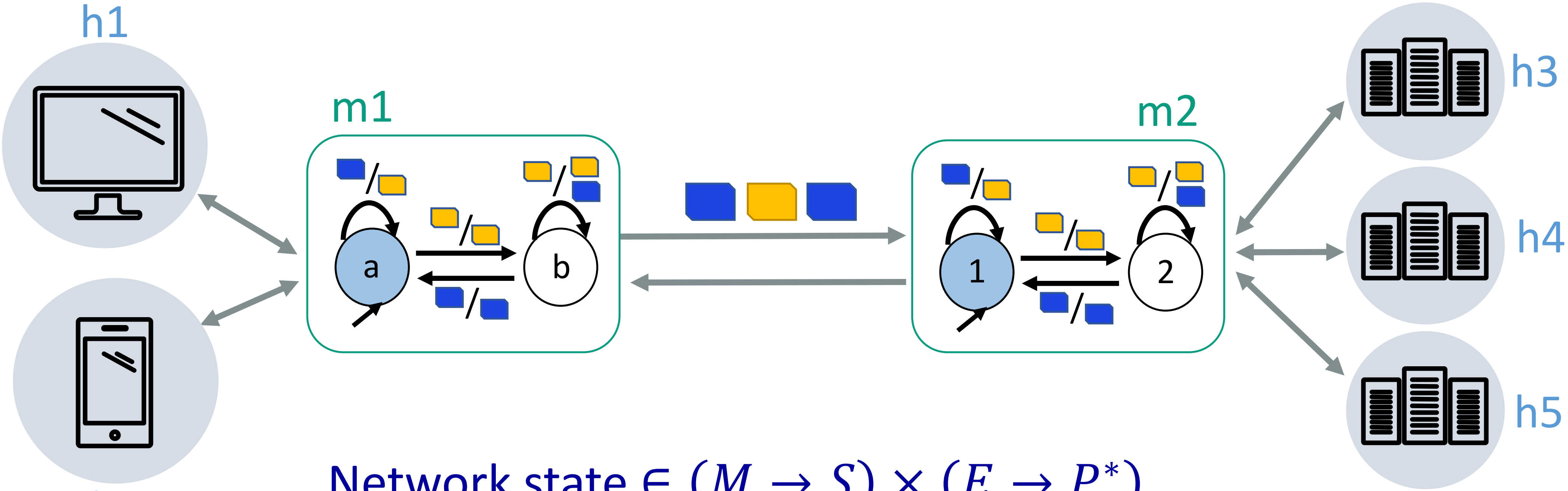
- Abstraction: over-approximate network behavior
  - If isolation is preserved by the over-approximation, the network is safe
  - If violation is detected, may be a false alarm

Abstractions for stateful networks

Abstracting middleboxes as finite state machines (FSM)	No need to analyze the code • Network safety is undecidable
Abstracting the order of packet arrival	Decidability [TACAS’16] • EXPSPACE-complete
Allowing each middlebox to nondeterministically revert to its initial state	Polynomial complexity for isolation • Abstract interpretation

[TACAS’16] Y. Velner, K. Alpernas, A. Panda, A. Rabinovich, M. Sagiv, S. Shenker, S. Shoham: Some Complexity Results for Stateful Network Verification

Concrete Network ≈ Communicating FSMs

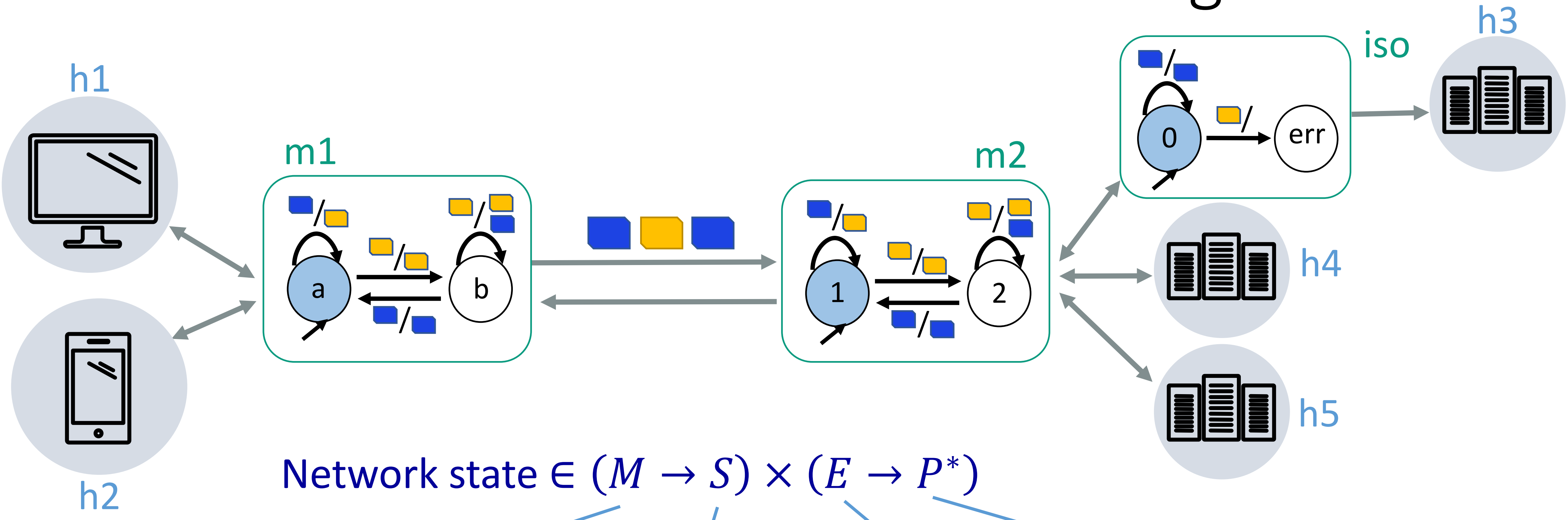


Network state $\in (M \rightarrow S) \times (E \rightarrow P^*)$

mboxes   mbox states   channels   packets $\in$ Hosts $\times$ Hosts $\times$ T

<src, dst, tpe>

Concrete Network ≈ Communicating FSMs



Network state $\in (M \rightarrow S) \times (E \rightarrow P^*)$

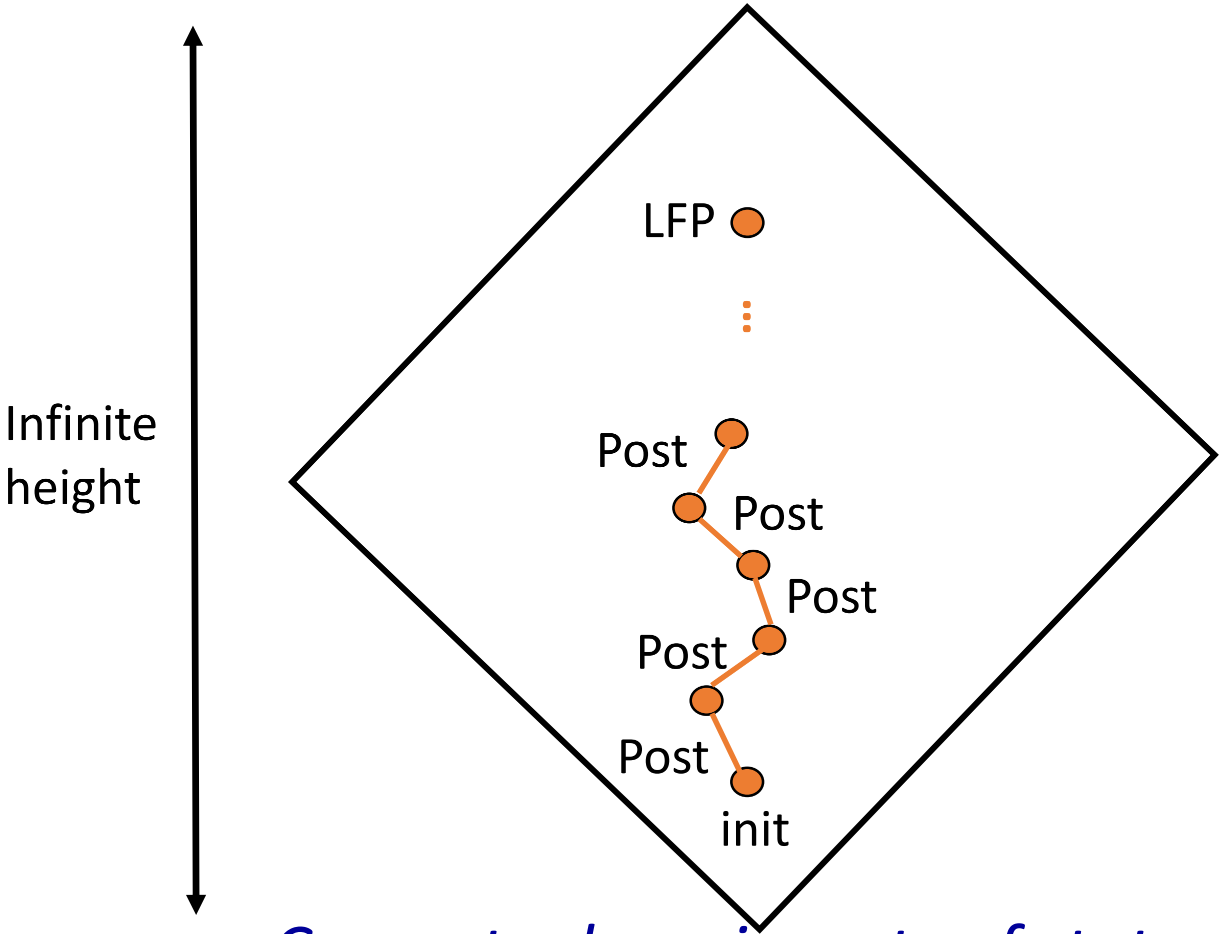
mboxes

mbox states

channels

packets ∈ Hosts × Hosts × T
<src, dst, tpe>

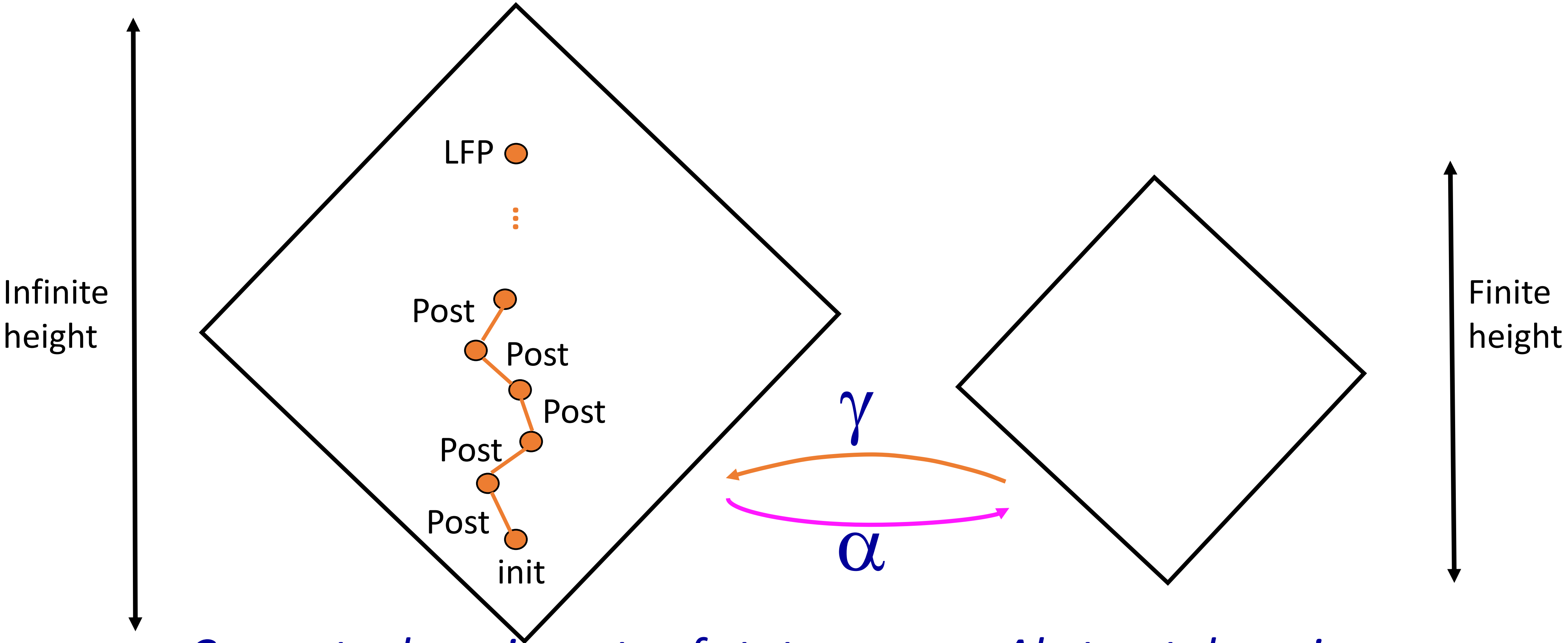
Concrete Interpretation



Concrete domain: sets of states

$$C = \mathbb{P}\big((M \to S) \times (E \to P^*)\big)$$

Abstract Interpretation + *Finite* Abstraction

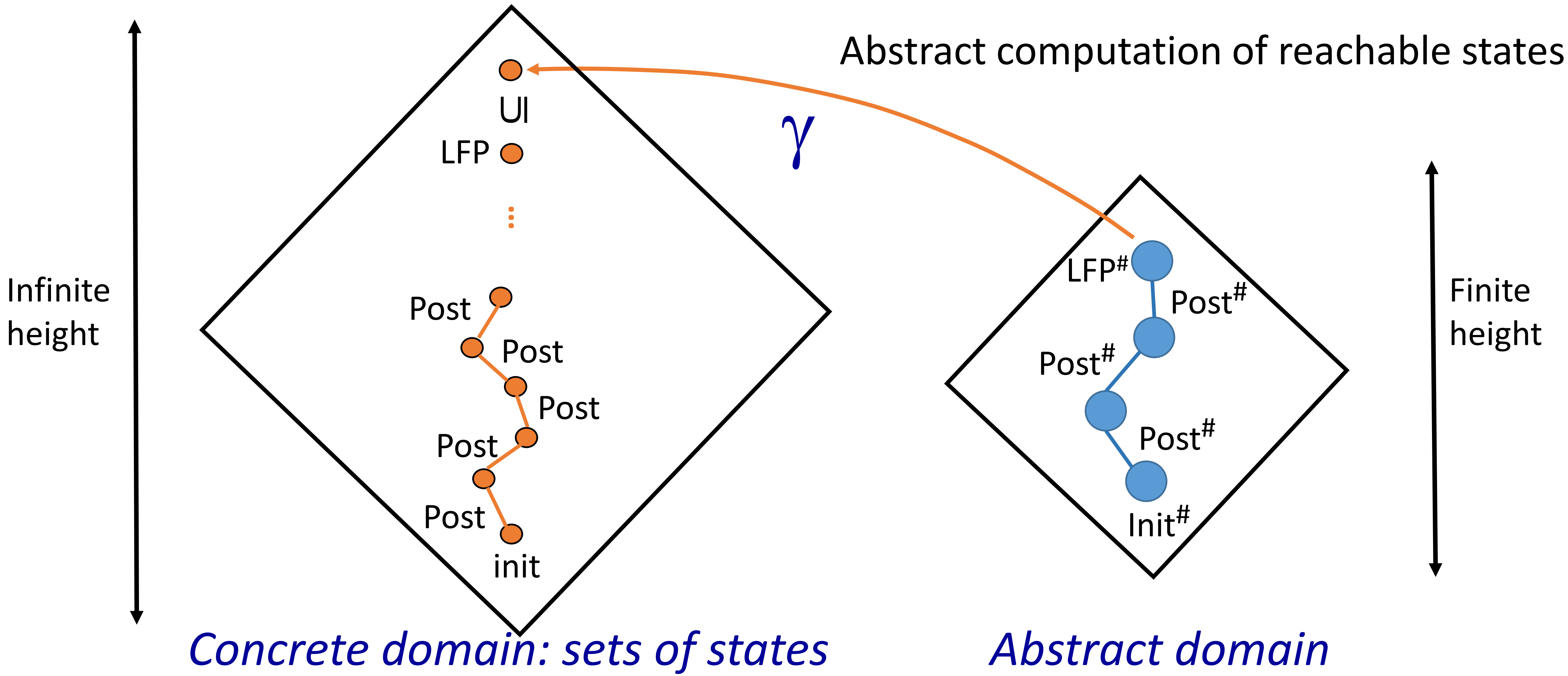


Concrete domain: sets of states

$$C = \mathbb{P}\big((M \to S) \times (E \to P^*)\big)$$

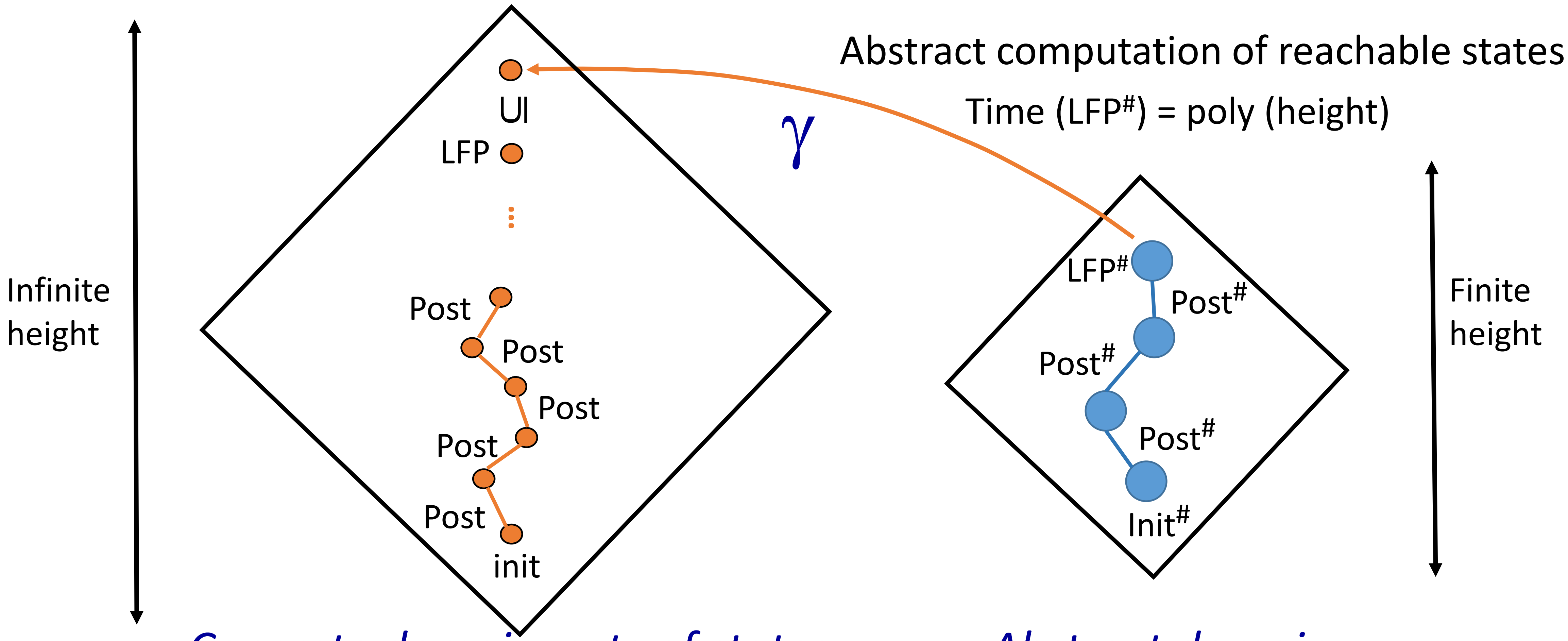
Abstract domain

Abstract Interpretation + *Finite* Abstraction



$$C = \mathbb{P}\big((M \to S) \times (E \to P^*)\big)$$

Abstract Interpretation + *Finite* Abstraction



Concrete domain: sets of states

$$C = \mathbb{P}\big((M \to S) \times (E \to P^*)\big)$$

Abstract domain

Network Abstractions

(0) Concrete domain

mboxes   mbox states   channels   packets

$$C = \mathbb{P}\big((M \to S) \times (E \to P^*)\big)$$

Network Abstractions

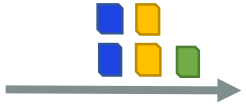
(0) Concrete domain

mboxes   mbox states   channels   packets

$$C = \mathbb{P}\big((M \to S) \times (E \to P^*)\big)$$

(1) Unordered channels

- Channels as multisets of packets



$$\mathbb{P}\big((M \to S) \times (E \to \underbrace{P \to \mathbb{N})\big)}_{\text{Infinite height}}$$

Network Abstractions

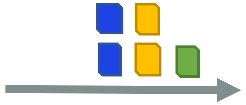
(0) Concrete domain

mboxes mbox states channels packets

$$C = \mathbb{P}\big((M \to S) \times (E \to P^*)\big)$$

(1) Unordered channels

- Channels as multisets of packets



$$\mathbb{P}\big((M \to S) \times (E \to \underbrace{P \to \mathbb{N}}_{\text{Infinite height}})\big)$$

> Safety verification is decidable [TACAS'16]
> - Reduction to/from Petri Net coverability
> - EXPSPACE complexity

[TACAS'16] Y. Velner, K. Alpernas, A. Panda, A. Rabinovich, M. Sagiv, S. Shenker, S. Shoham: Some Complexity Results for Stateful Network Verification

Network Abstractions

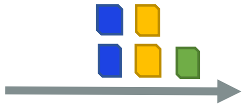
(0) Concrete domain

mboxes mbox states channels packets

$$C = \mathbb{P}\big((M \to S) \times (E \to P^*)\big)$$

(1) Unordered channels

- Channels as multisets of packets



$$\mathbb{P}\big((M \to S) \times (E \to P \to \mathbb{N})\big)$$

(2) Counter abstraction on channels

- Channels as sets of packets

$$\mathbb{P}\big((M \to S) \times (E \to \mathbb{P}(P))\big)$$

Network Abstractions

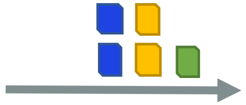
(0) Concrete domain

mboxes mbox states channels packets

$$C = \mathbb{P}\big((M \to S) \times (E \to P^*)\big)$$

(1) Unordered channels

- Channels as multisets of packets



$$\mathbb{P}\big((M \to S) \times (E \to P \to \mathbb{N})\big)$$

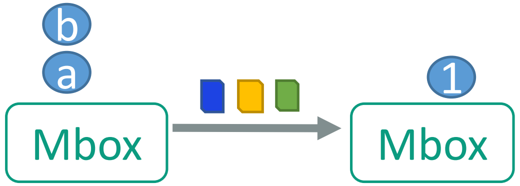
(2) Counter abstraction on channels

- Channels as sets of packets

$$\mathbb{P}\big((M \to S) \times (E \to \mathbb{P}(P))\big)$$

(3) Cartesian Abstraction

- No correlations between mboxes, channels, packets



$$\mathcal{A} = \big(M \to \mathbb{P}(S)\big) \times \big(E \to \mathbb{P}(P)\big)$$

Network Abstractions

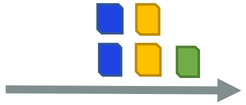
(0) Concrete domain

mboxes   mbox states   channels   packets

$$C = \mathbb{P}\big((M \to S) \times (E \to P^*)\big)$$

(1) Unordered channels

- Channels as multisets of packets



$$\mathbb{P}\big((M \to S) \times (E \to P \to \mathbb{N})\big)$$

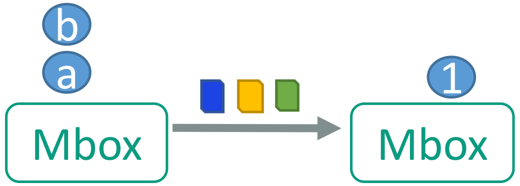
(2) Counter abstraction on channels

- Channels as sets of packets

$$\mathbb{P}\big((M \to S) \times (E \to \mathbb{P}(P))\big)$$

(3) Cartesian Abstraction

- No correlations between mboxes, channels, packets



$$\mathcal{A} = \big(M \to \mathbb{P}(S)\big) \times \big(E \to \mathbb{P}(P)\big)$$

Recall: height($\mathbb{P}$(X)) = |X|

Time(LFP$^{\#}$) = poly(|M|, |S|, |E|, |P|)

Network Abstractions

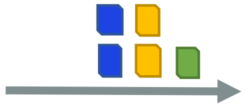
mboxes mbox states channels packets

(0) Concrete domain

$$C = \mathbb{P}\big((M \to S) \times (E \to P^*)\big)$$

(1) Unordered channels

- Channels as multisets of packets



$$\mathbb{P}\big((M \to S) \times (E \to P \to \mathbb{N})\big)$$

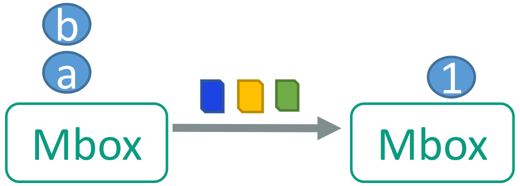
(2) Counter abstraction on channels

- Channels as sets of packets

$$\mathbb{P}\big((M \to S) \times (E \to \mathbb{P}(P))\big)$$

(3) Cartesian Abstraction

- No correlations between mboxes, channels, packets

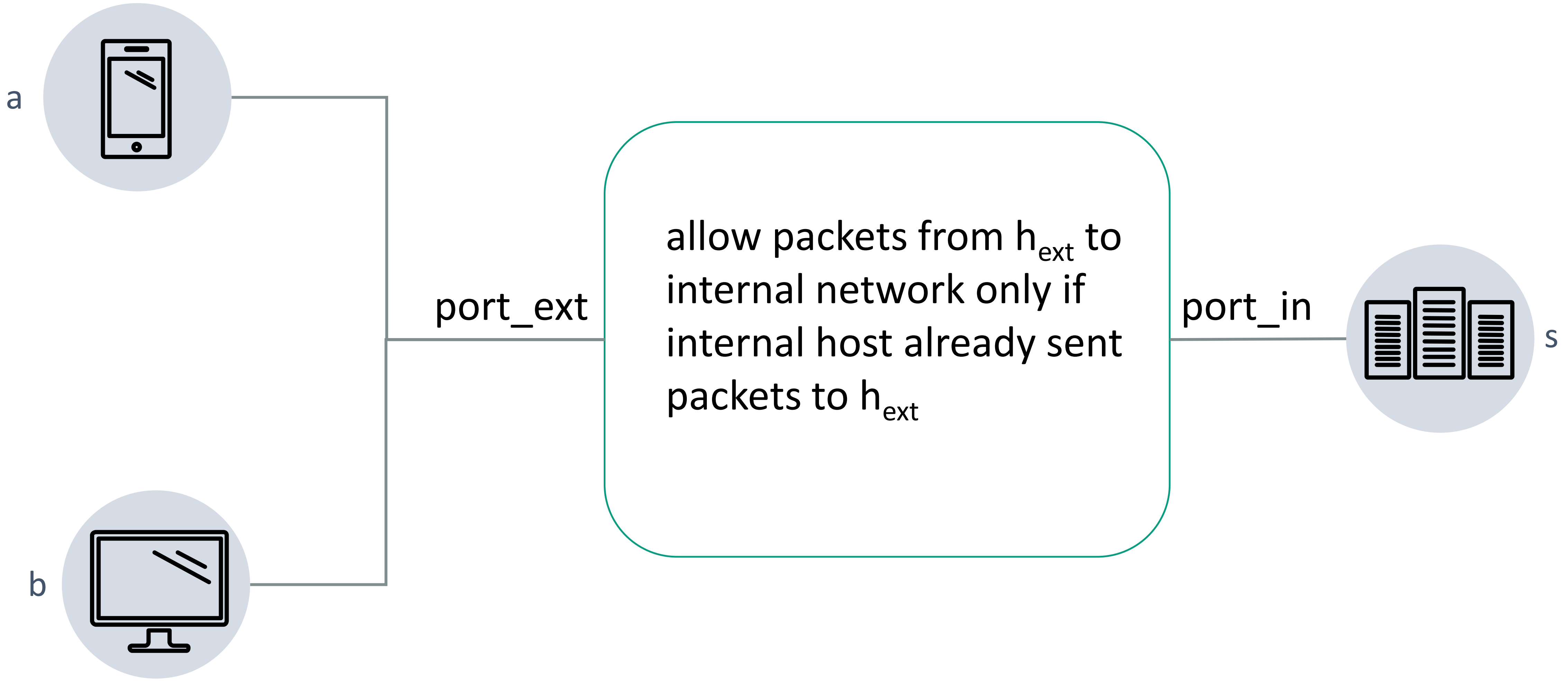


$$\mathcal{A} = \big(M \to \mathbb{P}(S)\big) \times \big(E \to \mathbb{P}(P)\big)$$

Unfortunately
|S| = exp (|Hosts|)

Time(LFP$^{\#}$) = poly(|M|, |S|, |E|, |P|)

Example: Hole-Punching Firewall



Example: Firewall

AMDL: Abstract MBox Def. Lang.

- Similar to [SIGCOMM'16]

- States ≈ n-ary relations
- Topology agnostic
- Encode FSM compactly
  - For fixed topology finite state

```
hole_punching_firewall =
  port_in ? <src,dst,tpe> =>
    trusted(dst) := true;
    port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
    src in trusted =>
      port_in ! <src,dst,tpe>
```

[SIGCOMM'16] A. Sivaraman, A. Cheung, M. Budiu, C. Kim, M. Alizadeh, H. Balakrishnan, G. Varghese, N. McKeown, S. Licking: **Packet Transactions: High-Level Programming for Line-Rate**

Example: Firewall

AMDL: Abstract MBox Def. Lang.

- Similar to [SIGCOMM'16]

- States ≈ n-ary relations
- Topology agnostic
- Encode FSM compactly
  - For fixed topology finite state

```
hole_punching_firewall =
  port_in ? <src,dst,tpe> =>
    trusted(dst) := true;
    port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
    src in trusted =>
      port_in ! <src,dst,tpe>
```

[SIGCOMM'16] A. Sivaraman, A. Cheung, M. Budiu, C. Kim, M. Alizadeh, H. Balakrishnan, G. Varghese, N. McKeown, S. Licking: **Packet Transactions: High-Level Programming for Line-Rate**

Example: Firewall

AMDL: Abstract MBox Def. Lang.

- Similar to [SIGCOMM'16]

- States ≈ n-ary relations
- Topology agnostic
- Encode FSM compactly
  - For fixed topology finite state

```
hole_punching_firewall =
  port_in ? <src,dst,tpe> =>
    trusted(dst) := true;
    port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
    src in trusted =>
      port_in ! <src,dst,tpe>
```

[SIGCOMM'16] A. Sivaraman, A. Cheung, M. Budiu, C. Kim, M. Alizadeh, H. Balakrishnan, G. Varghese, N. McKeown, S. Licking: **Packet Transactions: High-Level Programming for Line-Rate**

Example: Firewall

AMDL: Abstract MBox Def. Lang.

- Similar to [SIGCOMM'16]

- States ≈ n-ary relations
- Topology agnostic
- Encode FSM compactly
  - For fixed topology finite state

```
hole_punching_firewall =
  port_in ? <src,dst,tpe> =>
    trusted(dst) := true;
    port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
    src in trusted =>
      port_in ! <src,dst,tpe>
```

[SIGCOMM'16] A. Sivaraman, A. Cheung, M. Budiu, C. Kim, M. Alizadeh, H. Balakrishnan, G. Varghese, N. McKeown, S. Licking: **Packet Transactions: High-Level Programming for Line-Rate**

Example: Firewall

AMDL: Abs

- Similar to [S

trusted: $\mathbb{P}(\text{Hosts})$

$|S| = 2^{|\text{Hosts}|}$

- States ≈ n-ary relations
- Topology agnostic
- Encode FSM compactly
  - For fixed topology finite state

```
hole_punching_firewall =
  port_in ? <src,dst,tpe> =>
    trusted(dst) := true;
    port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
    src in trusted =>
      port_in ! <src,dst,tpe>
```

[SIGCOMM'16] A. Sivaraman, A. Cheung, M. Budiu, C. Kim, M. Alizadeh, H. Balakrishnan, G. Varghese, N. McKeown, S. Licking: **Packet Transactions: High-Level Programming for Line-Rate**

Middlebox-level Abstraction

$$\text{Time}(\text{LFP}^{\#}) = \text{poly}(|M|, |S|, |E|, |P|)$$

- Problem: Middlebox state space exponential in number of hosts

Middlebox-level Abstraction

$$\mathrm{Time}(\mathrm{LFP}^{\#}) = \mathrm{poly}(|M|, |S|, |E|, |P|)$$

- Problem: Middlebox state space exponential in number of hosts
- Solution: apply Cartesian abstraction
  - Ignore some correlations *within* a middlebox state

Middlebox-level Abstraction

$$\text{Time}(\text{LFP}^{\#}) = \text{poly}(|M|, |S|, |E|, |P|)$$

- Problem: Middlebox state space exponential in number of hosts
- Solution: apply Cartesian abstraction
  - Ignore some correlations *within* a middlebox state
  - **How to decompose a state into sub-states?**

Packet state

- Alternative (isomorphic) state representation
- Depends on AMDL: the restricted way in which middleboxes query and update their state

Packet state example

(src, dst, type) ↦ `{queries which hold}`

```
(1,_,_) ↦ {}
(2,_,_) ↦ {}
```

```
hole_punching_firewall =      // hosts ∈ {1, 2}
  port_in ? <src,dst,tpe> =>
     trusted(dst) := true; port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
     srcT: src in trusted => port_in ! <src,dst,tpe>
```

Query name

Query

Packet state example

(src, dst, type) ↦ `{queries which hold}`

```
(1,_,_) ↦ {}
(2,_,_) ↦ {}
```

(1,_,_)

```
hole_punching_firewall =      // hosts ∈ {1, 2}
  port_in ? <src,dst,tpe> =>
     trusted(dst) := true; port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
     srcT: src in trusted => port_in ! <src,dst,tpe>
```

Query name

Query

Packet state example

(src, dst, type) ↦ `{queries which hold}`

```
(1,_,_) ↦ {}
(2,_,_) ↦ {}
```

(1,_,_)

```
hole_punching_firewall =      // hosts ∈ {1, 2}
  port_in ? <src,dst,tpe> =>
     trusted(dst) := true; port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
     srcT: src in trusted => port_in ! <src,dst,tpe>
```

Query name

Query

Packet state example

(src, dst, type) ↦ `{queries which hold}`

```
(1,_,_) ↦ {}
(2,_,_) ↦ {}
```

(_,1,_)

```
hole_punching_firewall =      // hosts ∈ {1, 2}
  port_in ? <src,dst,tpe> =>
     trusted(dst) := true; port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
     srcT: src in trusted => port_in ! <src,dst,tpe>
```

Query name

Query

Packet state example

(src, dst, type) ↦ `{queries which hold}`

```
(1,_,_) ↦ {}
(2,_,_) ↦ {}
```

```
(1,_,_) ↦ {srcT}
(2,_,_) ↦ {}
```

(_,1,_)

```
hole_punching_firewall =      // hosts ∈ {1, 2}
  port_in ? <src,dst,tpe> =>
     trusted(dst) := true; port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
     srcT: src in trusted => port_in ! <src,dst,tpe>
```

Query name

Query

Packet state example

(src, dst, type) ↦ **{queries which hold}**

```
(1,_,_) ↦ {}
(2,_,_) ↦ {}
```

```
(1,_,_) ↦ {srcT}
(2,_,_) ↦ {}
```

```
hole_punching_firewall =       // hosts ∈ {1, 2}
  port_in ? <src,dst,tpe> =>
     trusted(dst) := true; port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
     srcT: src in trusted => port_in ! <src,dst,tpe>
```

(1,_,_)

Query name

Query

Packet state example

(src, dst, type) ↦ `{queries which hold}`

```
(1,_,_) ↦ {}
(2,_,_) ↦ {}
```

```
(1,_,_) ↦ {srcT}
(2,_,_) ↦ {}
```

(1,_,_)

```
hole_punching_firewall =      // hosts ∈ {1, 2}
  port_in ? <src,dst,tpe> =>
     trusted(dst) := true; port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
     srcT: src in trusted => port_in ! <src,dst,tpe>
```

Query name

Query

Cartesian packet state example

```
(1,_,_) ↦ {} {srcT}
(2,_,_) ↦ {} {srcT}
```

```
hole_punching_firewall =      // hosts ∈ {1, 2}
  port_in ? <src,dst,tpe> =>
     trusted(dst) := true; port_ext ! <src,dst,tpe>
  |
  port_ext ? <src,dst,tpe> =>
     srcT: src in trusted => port_in ! <src,dst,tpe>
```

Query name

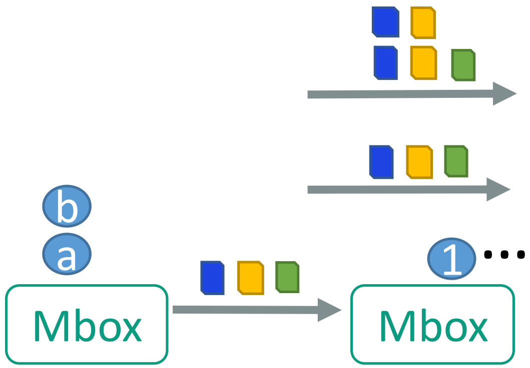
Query

Summary: Network Abstractions

(1) Unordered channels

(2) Counter abstraction on channels

(3) Network-level Cartesian Abstraction

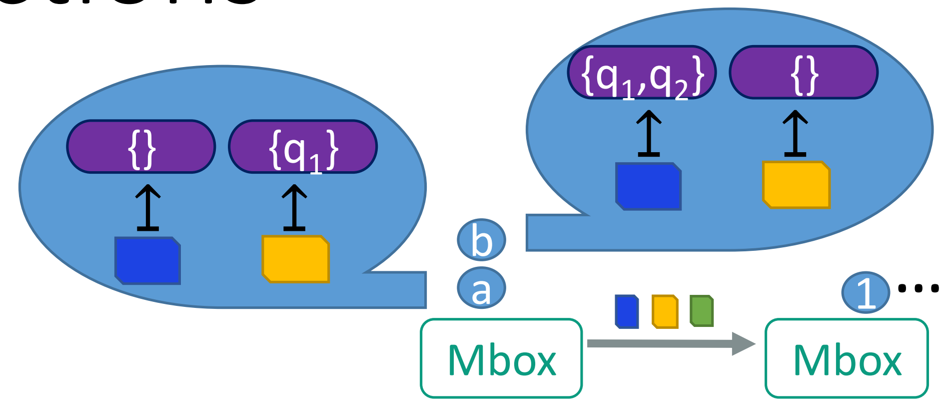


Summary: Network Abstractions

(1) Unordered channels

(2) Counter abstraction on channels

(3) Network-level Cartesian Abstraction



Summary: Network Abstractions

(1) Unordered channels

(2) Counter abstraction on channels

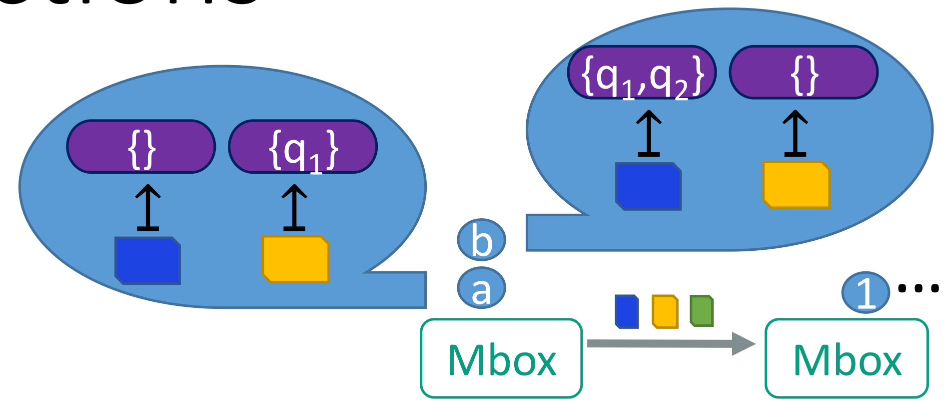
(3) Network-level Cartesian Abstraction

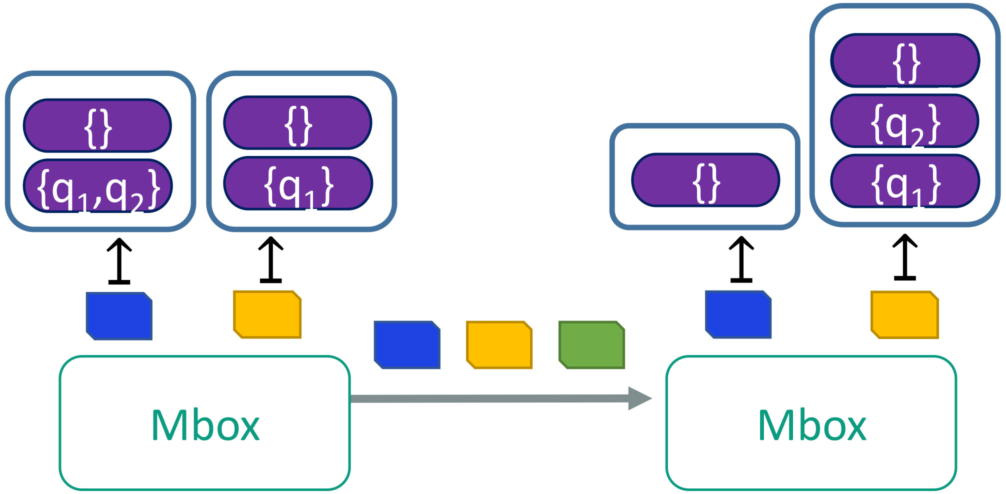
(4) Middlebox-level Cartesian abstraction

- No correlations between packet states
- But **keep** correlations between **queries**

$$\mathcal{A} = \left(M \to P \to \mathbb{P}(\mathbb{P}(Q))\right) \times \left(E \to \mathbb{P}(P)\right)$$

Time($LFP^{\#}$) = poly(|M|, |P|, $2^{|Q|}$, |E|)





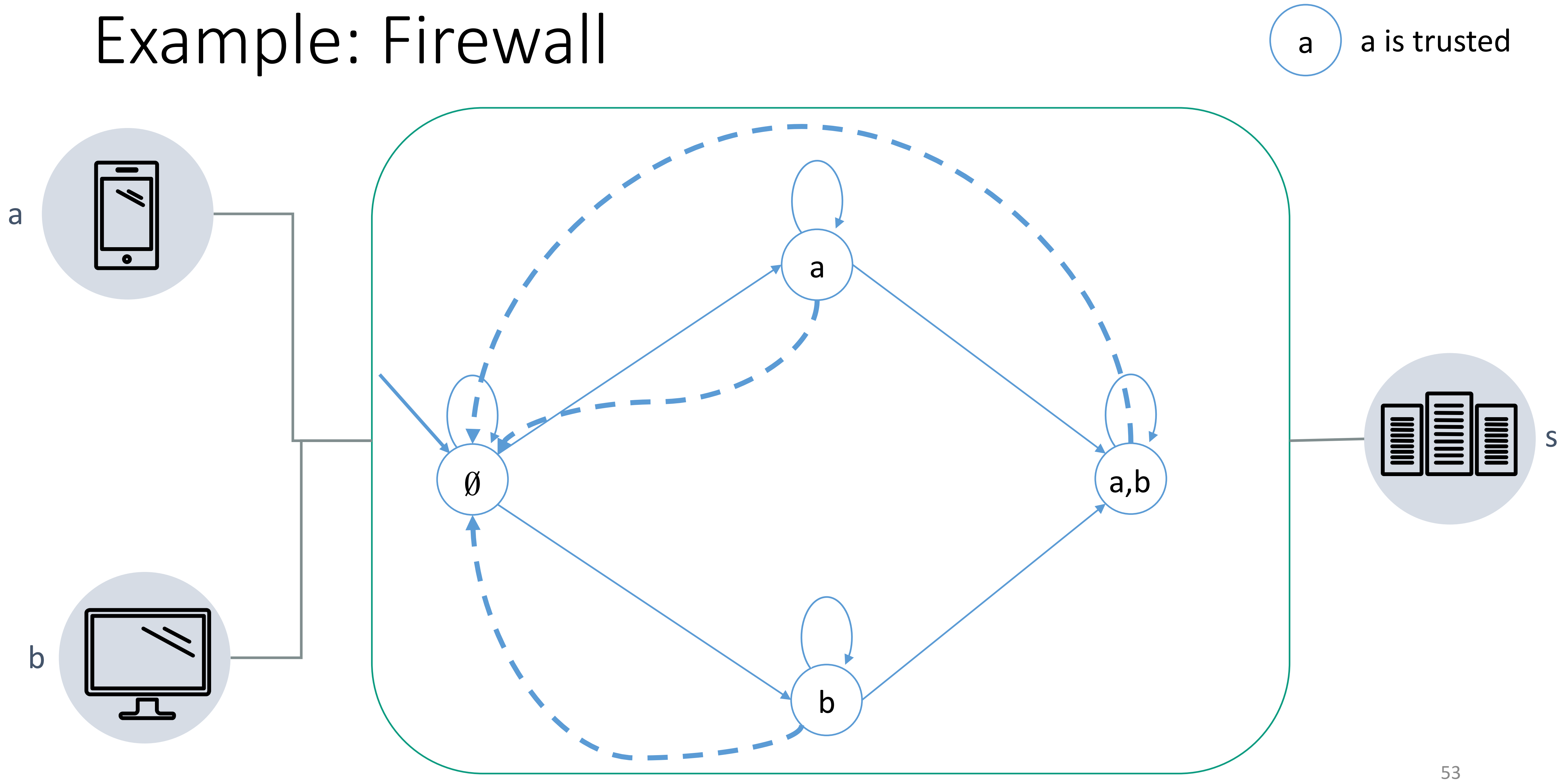
When is this precise?

Reverting Middlebox Abstraction

Cartesian abstraction ≈ Reverting middlebox abstraction

- Let middleboxes independently revert to their initial state

Example: Firewall

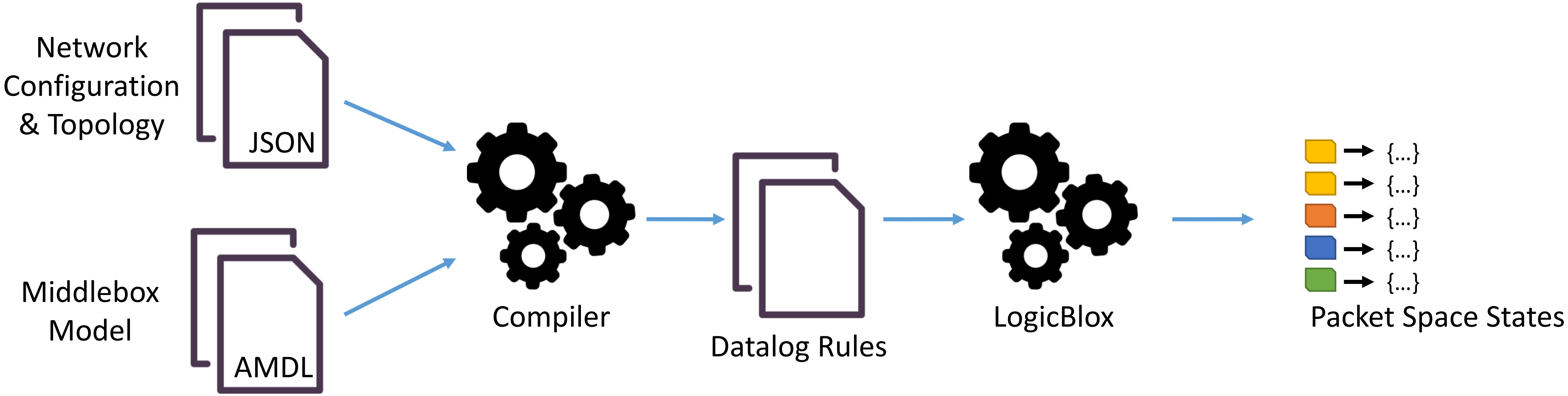


Reverting Middlebox Abstraction

Theorem: If the network is correct in the presence of packet reordering and middlebox reverts then our analysis is precise.

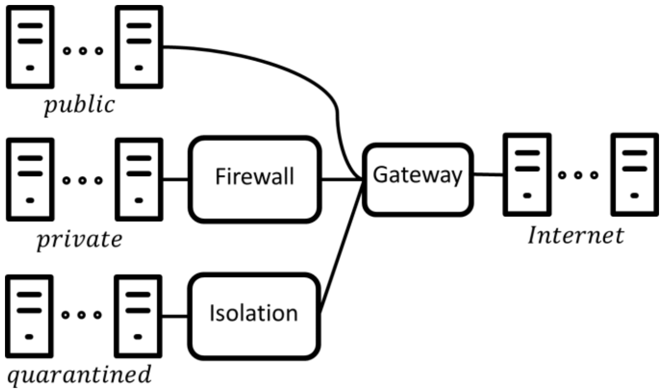
- Common wisdom: Network resets make verification harder
  - Reachability for Petri nets with resets is undecidable
- But: Simplifies the task of automatic verification of networks
  - The analysis is precise for isolation
  - No false alarms

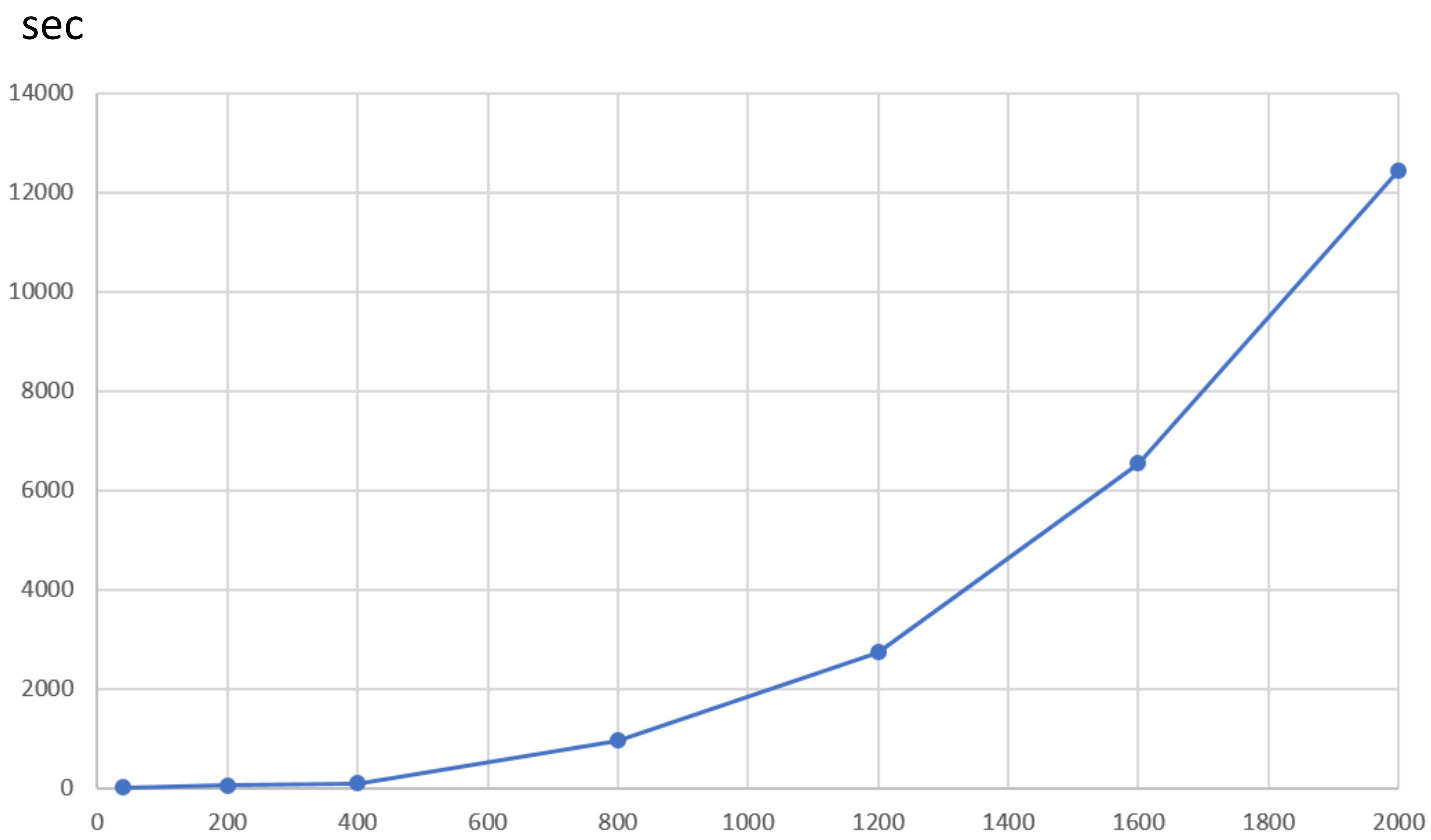
Initial Experimental Results



Scalability Testing - Hosts

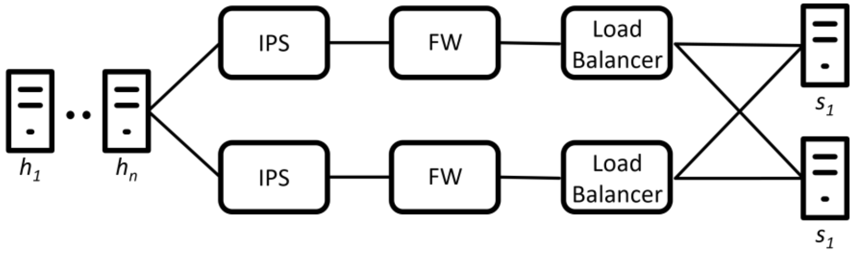
- Enterprise network with 3 subnets
  - Each with a different security policy
- Isolation between *quarantined* and *Internet*

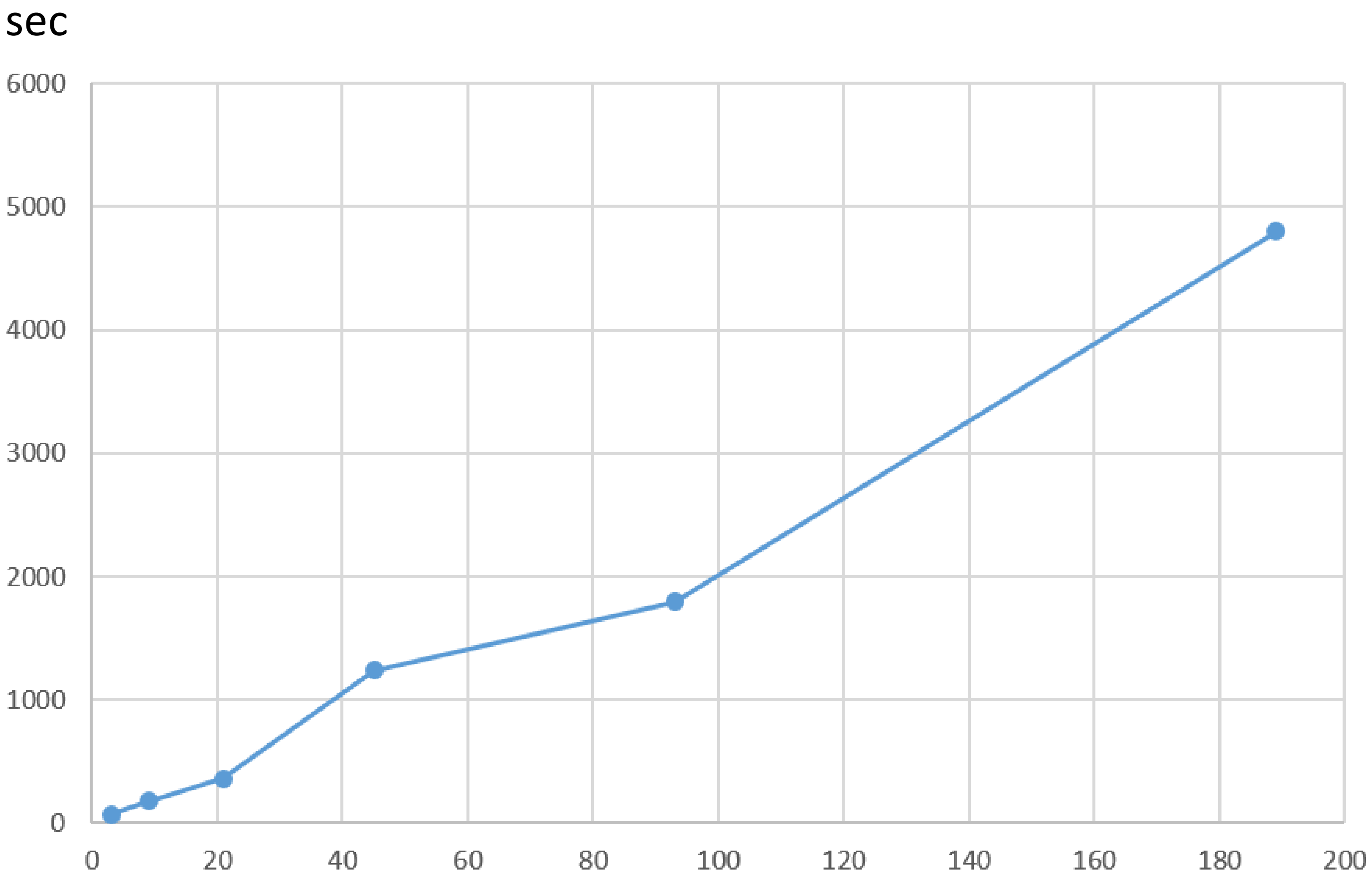




Scalability Testing - Middleboxes

- Servers with parallel middlebox chains
- Scaled the number of chains
- Isolation – packets from h_1 never reach bottom flow





Summary

- Abstract interpretation of stateful networks
    - Unordered + Counter + Cartesian X2
- AMDL – Abstract Middlebox Definition Language
- Packet effect semantics for middelboxes
    - Enables middlebox-level Cartesian abstraction
- Precise for unordered channels + reverting middleboxes

Further Work

- Correlated middlebox states
- Temporal properties
- Parameterized case

Seeking students and postdocs

erc Supervised Verification of Infinite-State Systems