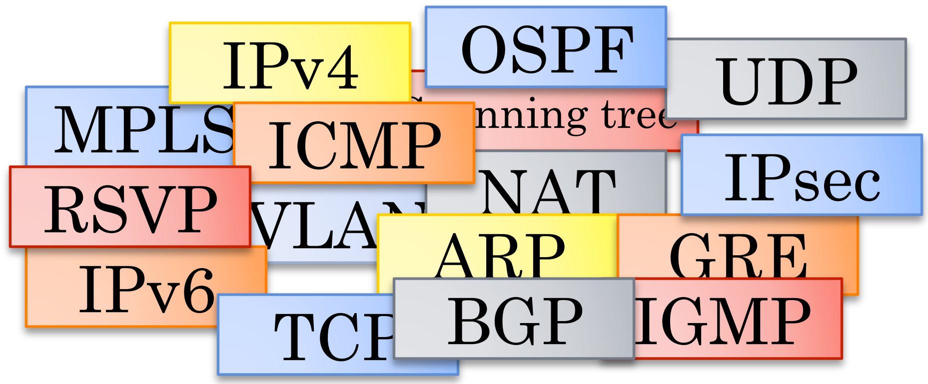
Header Space Analysis: Static Checking For Networks

Peyman Kazemian (Stanford University)
George Varghese (UCSD, Yahoo Labs)
Nick McKeown (Stanford University)

April 25th, 2012
NSDI 2012

TODAY…

- A typical network is a complex mix of protocols:



- Interact in complex ways.
- Cause unforeseen behavior.
- Hard to manage, understand and predict the behavior of networks.

TODAY…

- Even simple questions are hard to answer…
  - Can host A talk to host B?
  - What are all the packet headers from A that can reach B?
  - Are there any loops in the network?
  - Is Slice X isolated totally from Slice Y?
  - What will happen if I remove an entry from a router?

Header Space Analysis

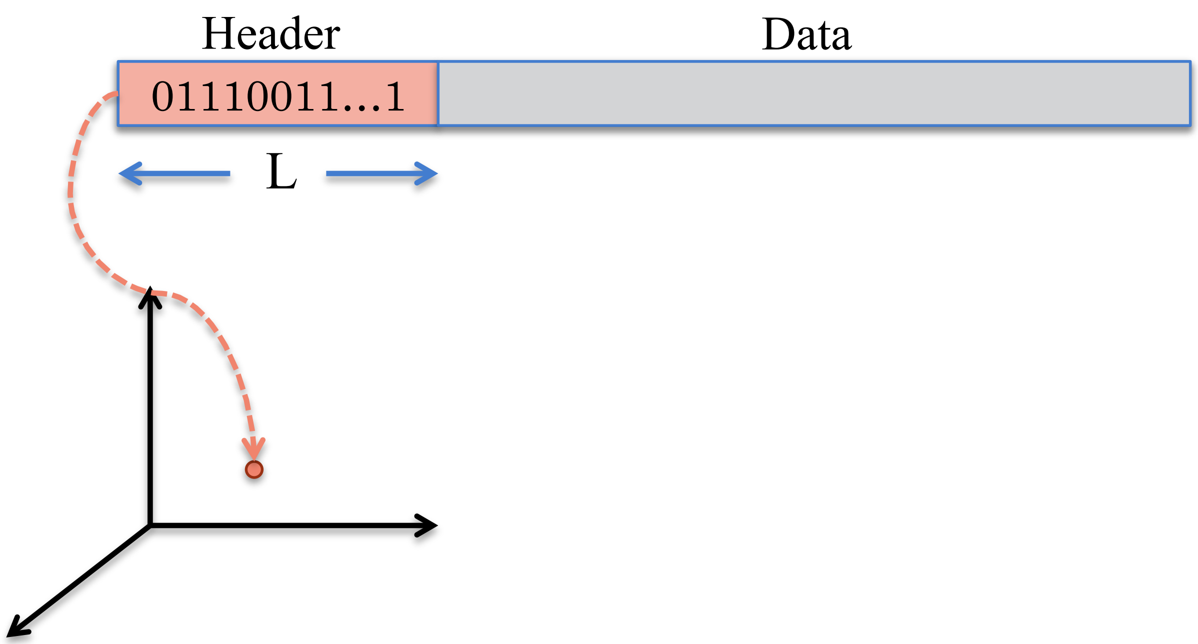
- A Powerful <u>General Foundation</u> that gives us
  - A unified view of almost all type of boxes.
  - A powerful interface for answering different questions about the network.

Header Space Framework

Simple Observation: a packet is a point in the space of possible headers and a box is a transformer on that space.

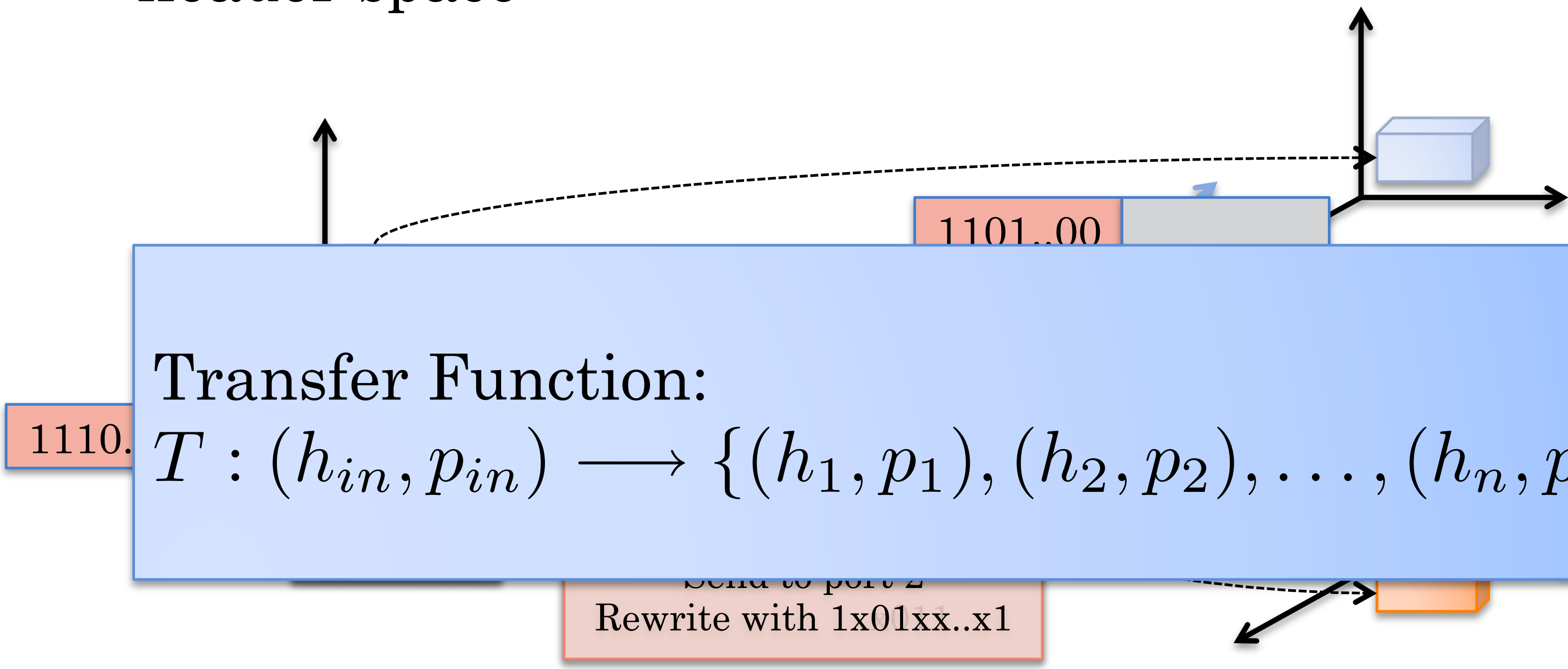
Header Space Framework

- Step 1 - Model packet header as a point in $\{0,1\}^L$ space – The Header Space



Header Space Framework

- Step 2 – Model all networking boxes as transformer of header space

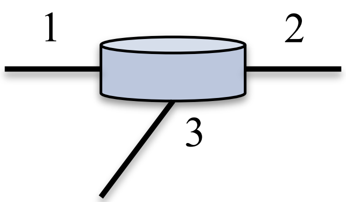


Transfer Function:

$$T : (h_{in}, p_{in}) \longrightarrow \{(h_1, p_1), (h_2, p_2), \ldots, (h_n, p_n)\}$$

Header Space Framework

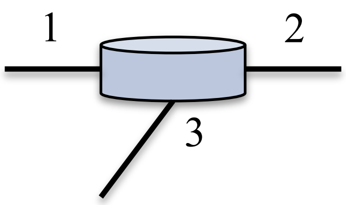
- Example: Transfer Function of an IPv4 Router
  - 172.24.74.0   255.255.255.0   Port1
  - 172.24.128.0   255.255.255.0   Port2
  - 171.67.0.0   255.255.0.0   Port3



$$T(h, p) = \begin{cases} (h,1) & \text{if } dst\_ip(h) = 172.24.74.x \\ (h,2) & \text{if } dst\_ip(h) = 172.24.128.x \\ (h,3) & \text{if } dst\_ip(h) = 171.67.x.x \end{cases}$$

Header Space Framework

- Example: Transfer Function of an IPv4 Router
  - 172.24.74.0 255.255.255.0 Port1
  - 172.24.128.0 255.255.255.0 Port2
  - 171.67.0.0 255.255.0.0 Port3

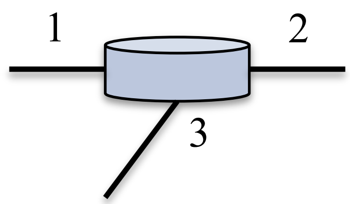


$$T(h, p) = \begin{cases} (dec\_ttl(h),1) & \text{if } dst\_ip(h) = 172.24.74.x \\ (dec\_ttl(h),2) & \text{if } dst\_ip(h) = 172.24.128.x \\ (dec\_ttl(h),3) & \text{if } dst\_ip(h) = 171.67.x.x \end{cases}$$

Header Space Framework

- Example: Transfer Function of an IPv4 Router

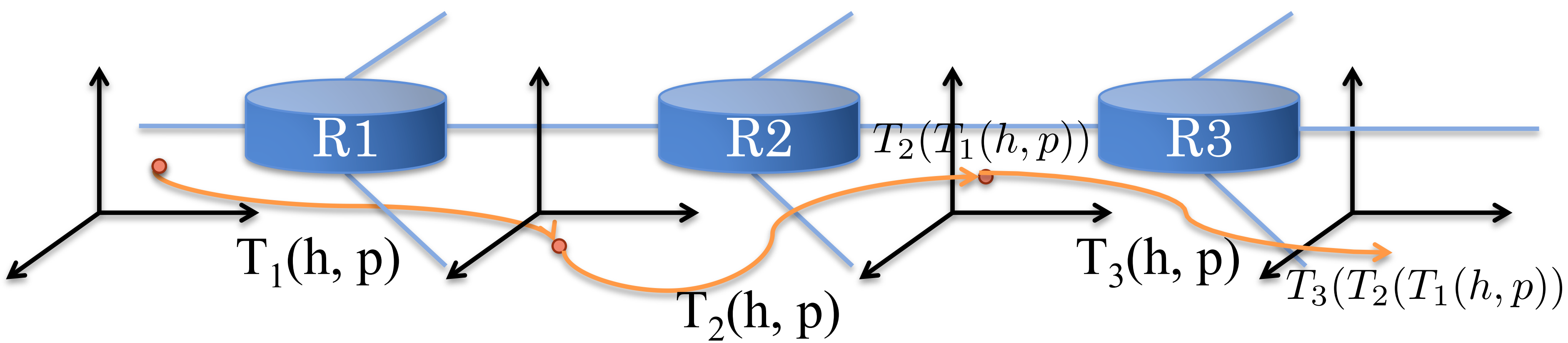
  - 172.24.74.0 255.255.255.0 Port1
  - 172.24.128.0 255.255.255.0 Port2
  - 171.67.0.0 255.255.0.0 Port3



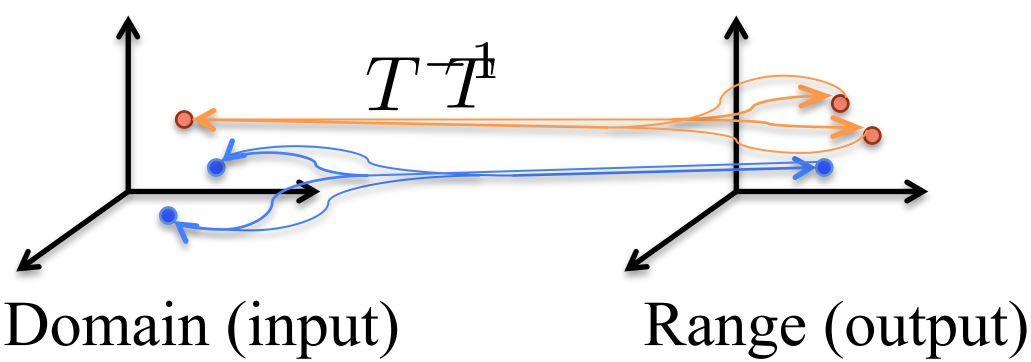
$$T(h, p) = \begin{cases} (rw\_mac(dec\_ttl(h), next\_mac)\,,\, 1) & \text{if } dst\_ip(h) = 172.24.74.x \\ (rw\_mac(dec\_ttl(h), next\_mac)\,,\, 2) & \text{if } dst\_ip(h) = 172.24.128.x \\ (rw\_mac(dec\_ttl(h), next\_mac)\,,\, 3) & \text{if } dst\_ip(h) = 171.67.x.x \end{cases}$$

Header Space Framework

- Properties of transfer functions
  - Composable: $T_3(T_2(T_1(h,p)))$



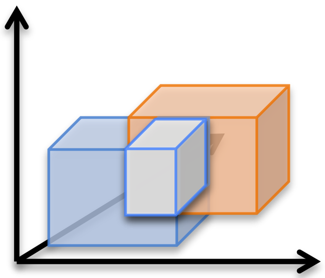
  - Invertible:



Header Space Framework

- Step 3 - Develop an algebra to work on these spaces.
- Every object in Header Space, can be described by union of Wildcard Expressions.

- We want to perform the following set operations on wildcard expressions:
  - Intersection
  - Complementation
  - Difference



Header Space Framework

- Finding Intersection:
  - Bit by bit intersect using intersection table:
    - Example: $10xx \cap 1xx0 = 10x0$
    - If result has any 'z', then intersection is empty:
    - Example: $10xx \cap 0xx1 = z0x1 = \phi$

b_i \ b_j	0	1	x
0	0	z	0
1	z	1	1
x	0	1	x

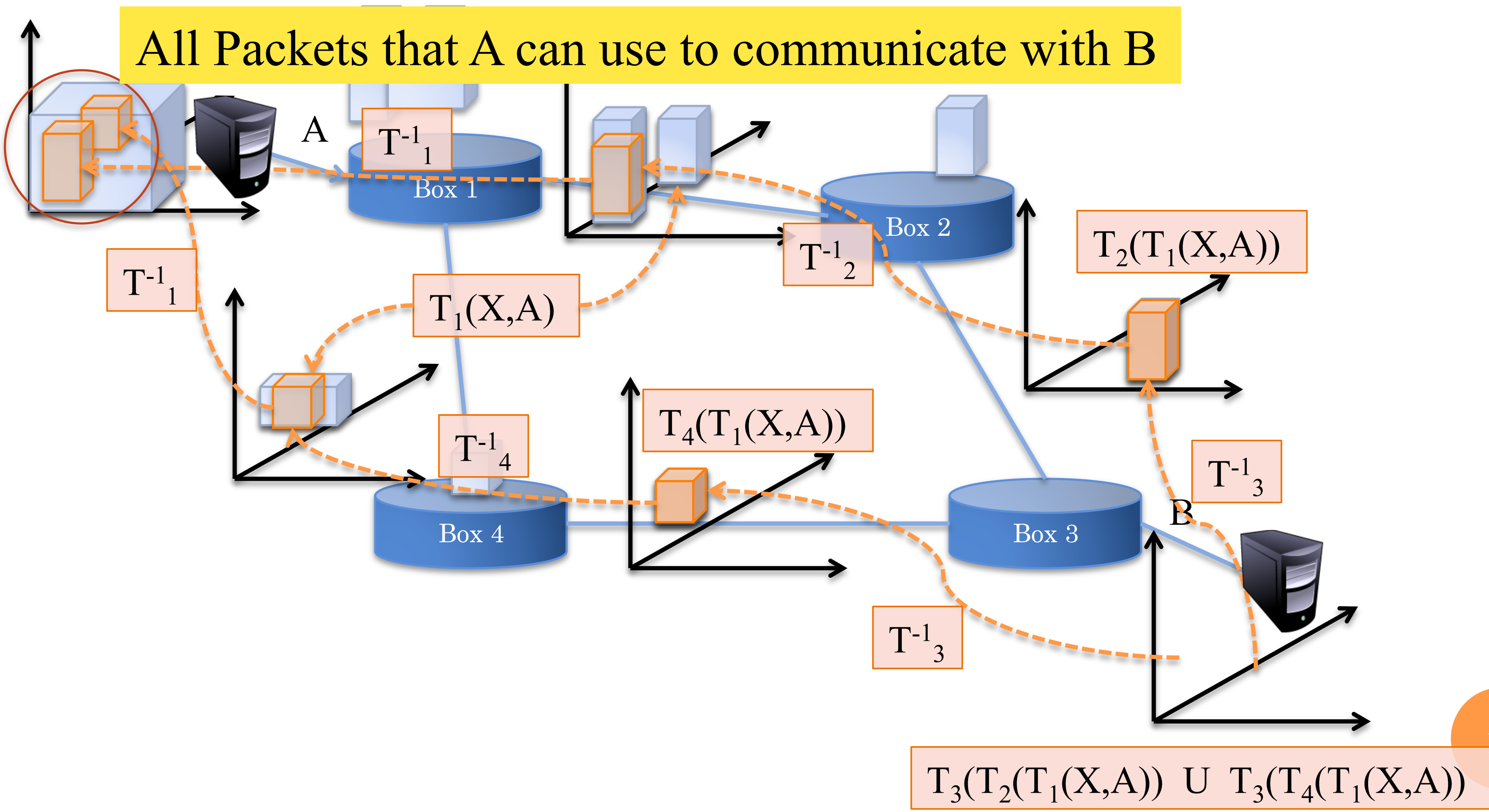
- See the paper for how to find complement and difference.

Use Cases of Header Space Framework

These are only some example use cases that we developed so far…

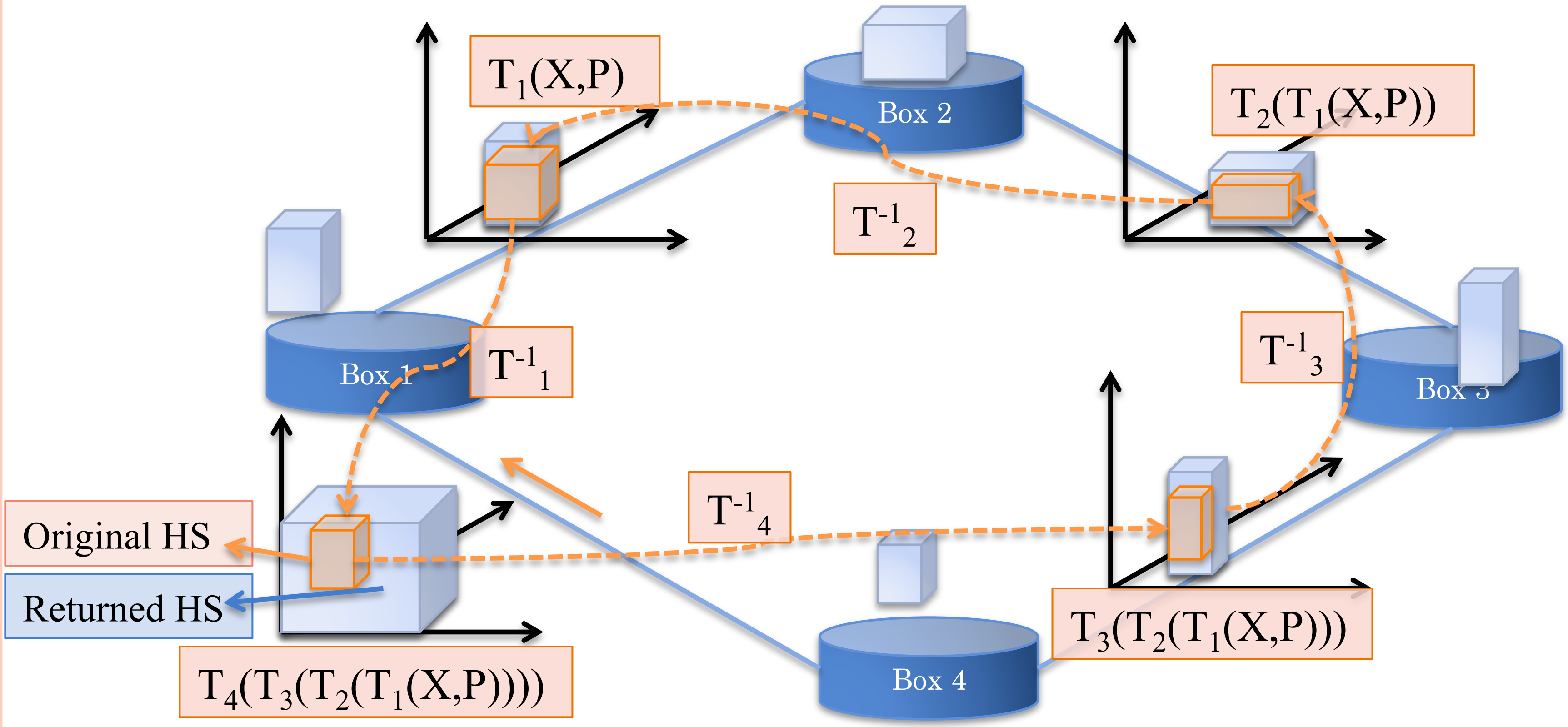
USE CASES

- Can host A talk to B?



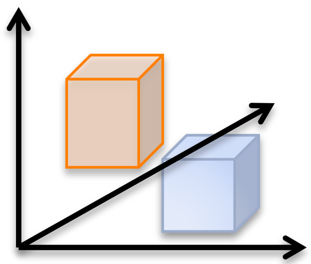
Use Cases

- Is there a loop in the network?
  - Inject an all-x text packet from every switch-port
  - Follow the packet until it comes back to injection port

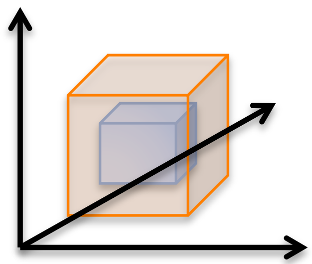


Use Cases

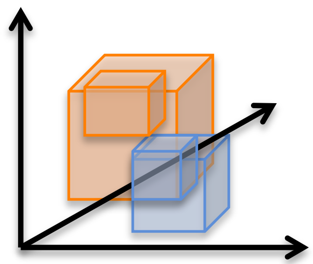
- Is the loop infinite?



Finite Loop



Infinite Loop



?

Use Cases

- Are two slices isolated?

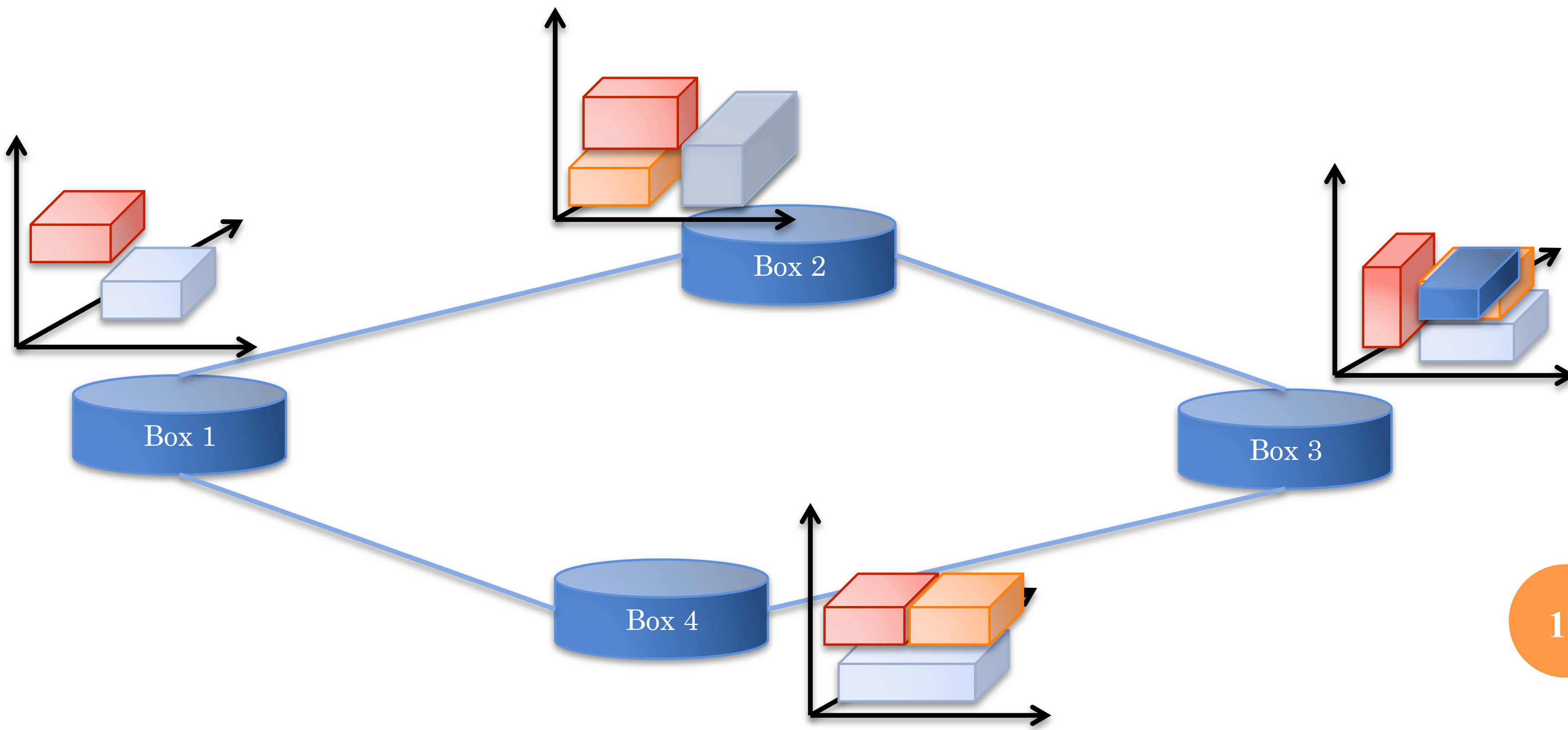
- What do we mean by slice?
  - Fixed Slices: VLAN slices
  - Programmable Slices: slices created by FlowVisor

- Why do we care about isolation?
  - Banks: for added security.
  - Healthcare: to comply with HIPAA.
  - GENI: to isolate different experiments running on the same network.

USE CASES

- Are two slices isolated?
  - 1) slice definitions don't intersect.
  - 2) packets do not leak.

Solution: Apply header space reservation of each slice to the slice's transfer function and check intersection of the result with other slices' reservations



Complexity

- Run time

  Reachability: $O(dR^2)$

  Loop Detection: $O(dPR^2)$

  - R: maximum number of rules per box.
  - d: diameter of network.
  - P: number of ports to be tested

  Slice Isolation Test: $O(NW^2)$

  - W: number of wildcard expressions in definition of a slice.
  - N: number of slices in the network.

  See paper for more details.

HEADER SPACE FRAMEWORK

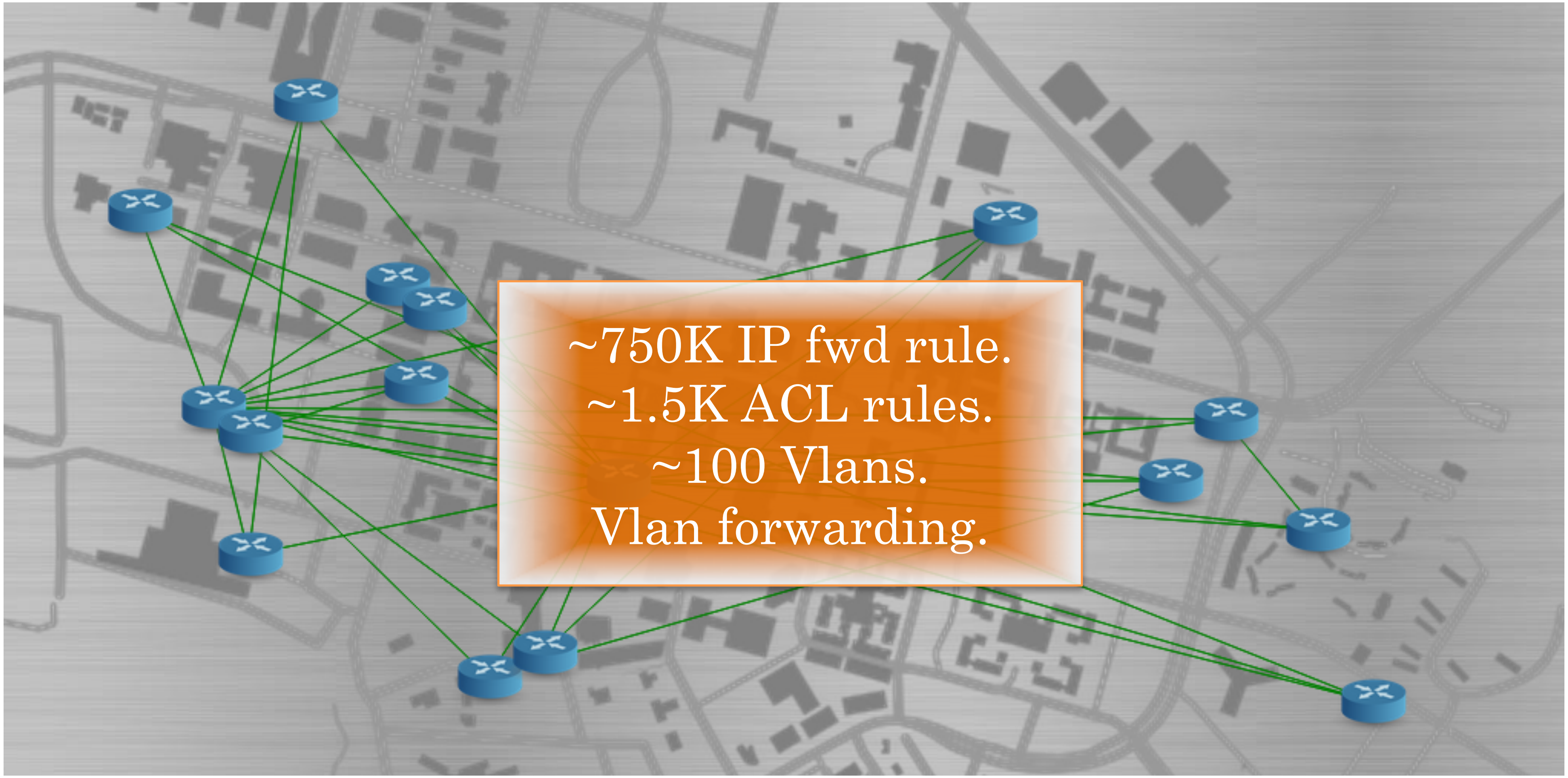
- A Powerful General Foundation that gives us
  - A unified view of almost all type of boxes.
    - Transfer Function.
  - A powerful interface for answering different questions about the network.
    - $T(h,p)$ and $T^{-1}(h,p)$
    - Set operations on Header Space

Implementation And Evaluation

Implementation

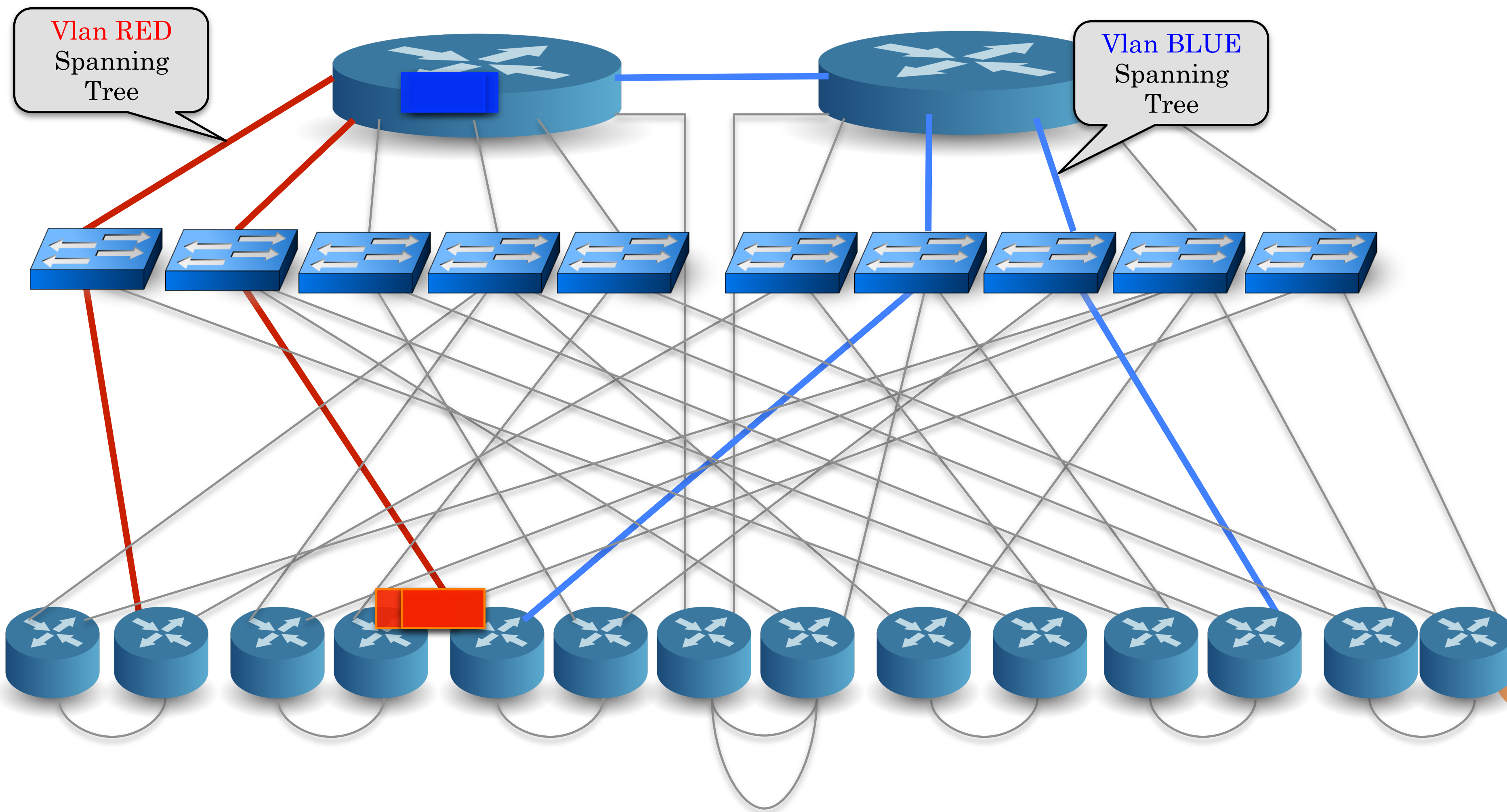
- Header Space Library (Hassel)
    - Written in Python
    - Implements Header Space Class
        - Set operations
    - Implements Transfer Function Class
        - T and T^{-1}
    - Implements Reachability, Loop Detection and Slice Isolation checks.
        - < 50 lines of code
    - Includes a Cisco IOS parser
        - Generates transfer function from output of IOS commands and config file.
        - Keeps the mapping from Transfer function rule to line number in config file.
    - Publicly available: git clone https://bitbucket.org/peymank/hassel-public.git

Stanford backbone network



STANFORD BACKBONE NETWORK

- Loop detection test – run time < 10 minutes on a single laptop.



Performance

Performance result for Stanford Backbone Network on a single machine: 4 core, 4GB RAM.

Generating TF Rules	~150 sec
Loop Detection Test (30 ports)	~560 sec
Average Per Port	~18 sec
Min Per Port	~ 8 sec
Max Per Port	~ 135 sec
Reachability Test (Avg)	~13 sec

Summary

- We Introduced Header Space Analysis:

  A Powerful General Foundation that gives us
  - A unified view of almost all type of boxes.
  - A powerful interface for answering different questions about the network.

- We showed that our initial Python-based implementation can scale to enterprise-size networks on a single laptop.

Thank You!

Questions?