ISTANBUL TECHNICAL UNIVERSITY
BLG 632E - NEXT GENERATION WIRELESS NETWORKS
INSTRUCTOR: IRFAN ALI

STUDENT NAME: TUĞRUL YATAĞAN
STUDENT NUMBER: 504161551

HEADER-SPACE ANALYSIS PRESENTATION

June 7, 2017

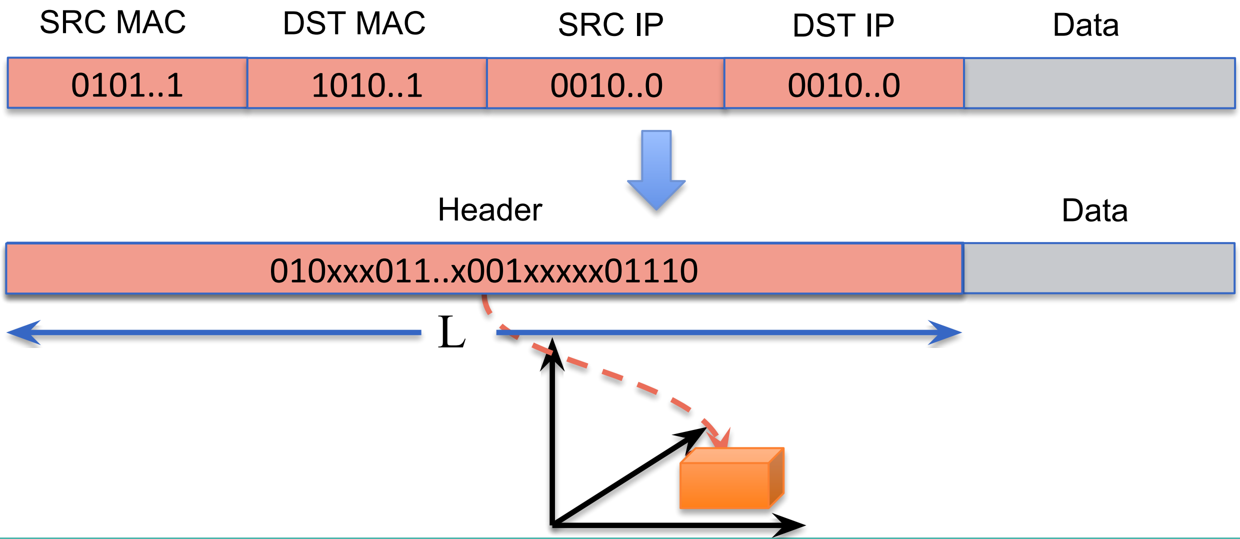
Data plane network verification

- Complex interaction;
    - Between multiple protocols on a switch/router.
    - Between state on different switches/routers.
- Network owner can’t…
    - Observe all state.
    - Control all state.

- 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?
- What happens if I remove this line in the config file?

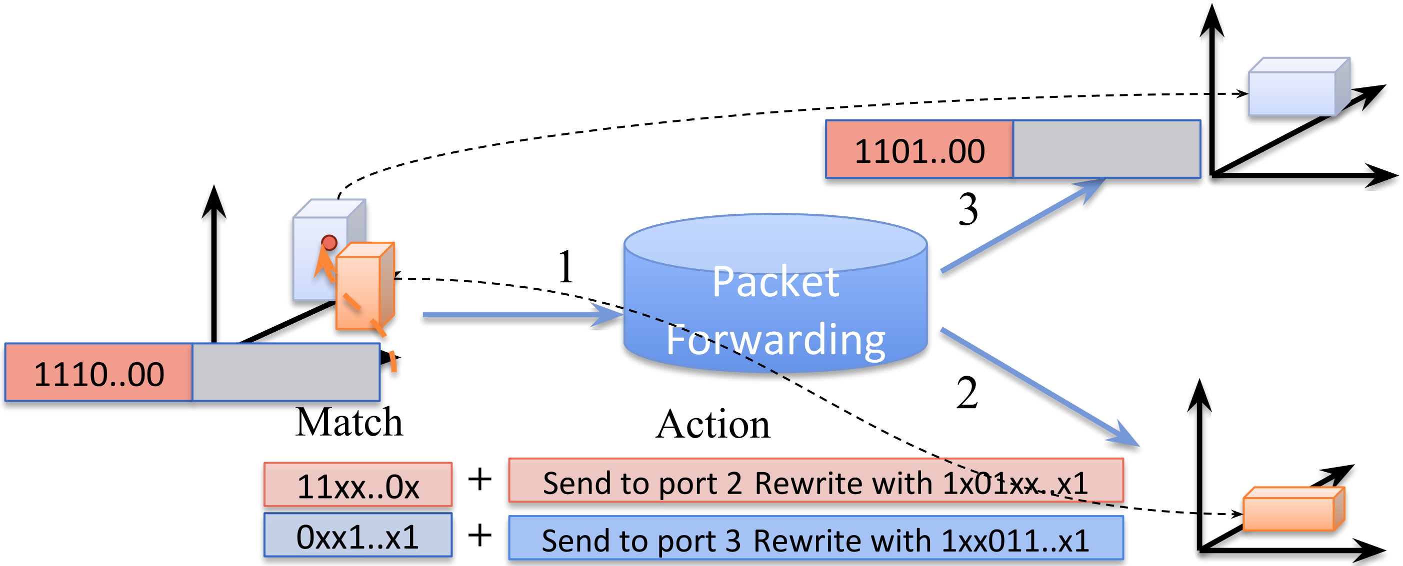
Header Space Framework - 1

- Ignore protocol dependent meaning of header bits and see it as a flat sequence of 0s and 1s.
- Model a packet as a point in $\{0,1\}^L$ space – The Header Space



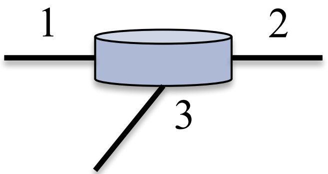
Header Space Framework - 2

- Model all networking boxes as transformers of header space
- Transfer Function: $T:(h,p)\rightarrow\{(h_1,p_1),\ldots,(h_n,p_n)\}$
- Every region of Header Space, can be described by union of Wildcard



Transfer Function Examples

- IPv4 Router
  - 172.24.74.x Port1
  - 172.24.128.x Port2
  - 171.67.x.x Port3



Forwarding Behavior

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

Forwarding Behavior + Time to Live (TTL)

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

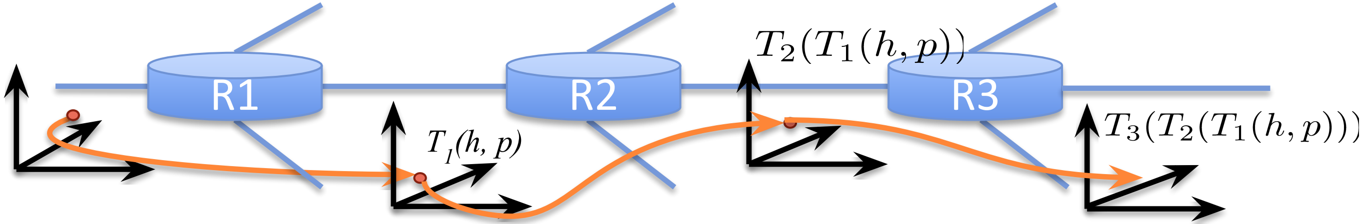
Transfer Function Other Examples

- Rewrite: rewrite bits 0-2 with value 101
  - (h & 000111…) | 101000…
- Encapsulation: encap packet in a 1010 header.
  - (h >> 4) | 1010….
- Decapsulation: decap 1010xxx… packets
  - (h << 4) | 000…xxxx
- TTL Decrement:
  - if ttl(h) == 0: Drop
  - if ttl(h) > 0: h – 0…000000010…0
- Load Balancing:
  - LB(h,p) = {(h,P1),…(h,Pn)}

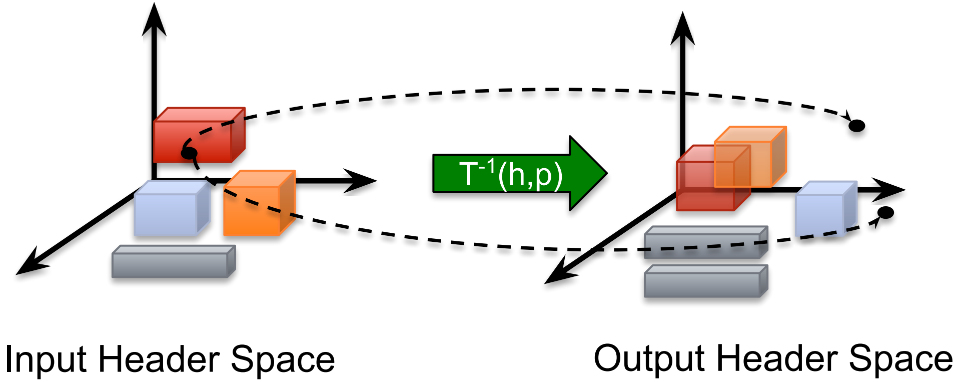
Transfer Function

- By composing transfer functions, we can find the end to end behavior of networks.

$$T_3(T_2(T_1(h,p)))$$

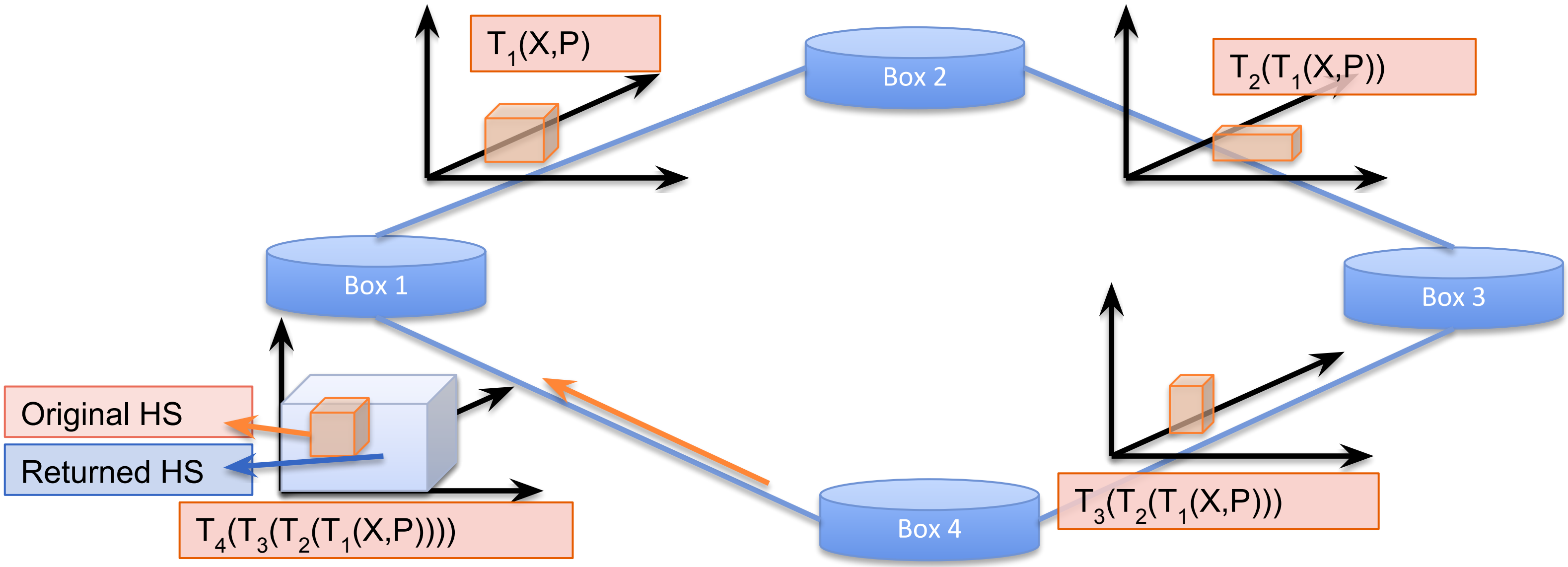


- Inverting transfer functions, gives all possible input packets that can generate an output packet.



Algorithms

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



- Also, finding reachability, checking isolation of slices, etc.

Implementations

- Agent sits between the control plane and the switches, and it uses the Header Space Analysis algorithm that can be used to check a rule update against a single policy within very short time (50-500 us)
- There are several well known HSA implementations;
    - Hassel
        - IP Table compression
        - Lazy subtraction
        - Dead object deletion
        - Lazy TE evaluation
    - Net-Plumber
        - Instead of recomputing all the transformations each time the network changes, it incrementally updates only the portions of those transfer function results affected by the change.

Thank you for listening!

References:

- Mohammad Alizadeh, MIT 6.888 Lecture 16: Header Space Analysis
- Nick Feamster, Data-Plane Verification: Header Space Analysis
- P. Kazemian, M. Chang, H. Zeng, G. Varghese, N. McKeown, S. Whyte, Real time network policy checking using header space analysis
- Ian F. Akyildiz, A roadmap for traffic engineering in SDN-OpenFlow networks