

[Review]
Declarative Routing:
 Extensible Routing
 with Declarative Overlays

Loo, Hellerstein, Stoica, Ramakrishnan
 SIGCOMM 2005

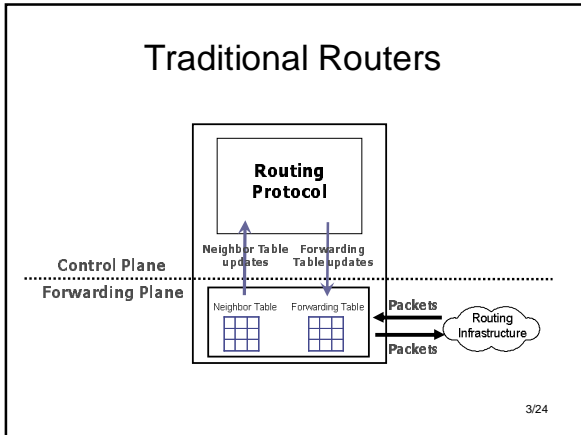
Presentation by Joe Kopena
 Many slides courtesy of Boon Loo
 2007/02/15

1/24

What's Up

- Today: Introduction to Declarative Routing
 - Goals, framework, syntax, semantics, basic networking protocols
- Tuesday: Declarative network overlays
 - Larger protocols in declarative form---DHTs, multicast, etc

2/24



Router Elements: Control

- Control Plane typically:
 - Sends and receives beacons for neighborhood detection
 - Advertises local paths to other nodes
 - Computes new paths based on detected links and received advertisements
 - Configures the forwarding plane based on those paths

4/24

Router Elements: Forwarding

- Forwarding Plane typically:
 - Receives incoming packets
 - Looks them up in paths given by control plane
 - Sends them back along the link toward the destination
- Some variations in the interaction between the two planes, e.g. in reactive protocols
 - Incoming packet requests trigger control traffic and decision making

5/24

Routing...

- Planes typically implemented in low level software or hardware
- Most of the field very focused on unicast, explicit addressing, and optimizing latency, hops, or throughput over overhead traffic
- Not well suited for cross-layer design
- Difficult to develop & test new approaches
- Extremely difficult to deploy

6/24

New Routing Approaches

- Many areas addressing these problems:
 - Modular and extensible routers
 - Active networks
 - Overlay networks

7/24

Modular & Extensible Routers

- Idea: Make it easier to construct & deploy new routers, enable code-sharing
- Still faces deployment challenges
- Not necessarily well suited for rapid or cross-layer design

8/24

Active Networks

- Idea: Enable packets to carry routing intelligence with them as general code
- Major security considerations
- Efficiency and reliability concerns
- Still writing programs rather than working with data & intent

9/24

Overlay Networks

- Idea: Add new functionality over existing infrastructure via application-layer routing
- Lots of work to develop & get right
- Still writing programs

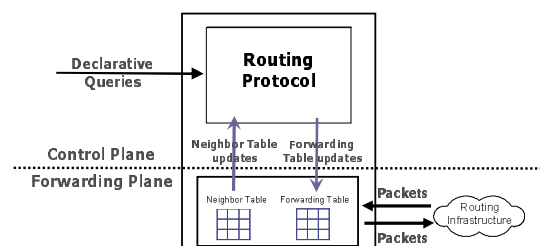
10/24

Declarative Routing

- Basic premise: Replace the control plane with a general query/rule processor
- Express routing protocols via rules manipulating network and application data
- Goal: Enable routing protocol development much like SQL enables rapid database application development
- Goals: Easy deployment & flexibility to run many protocols simultaneously

11/24

Declarative Routers



12/24

Declarative Routing Logic

- Specify routing logic using a declarative, Datalog-based rule/query language
- DataLog syntax:

$\langle \text{result} \rangle \leftarrow \langle \text{condition1} \rangle, \langle \text{condition2} \rangle, \dots, \langle \text{conditionN} \rangle$

Head Body

- Conditions are predicates or tables containing basic network state data and knowledge derived as head of rules

13/24

Example: Reachability

- On a single host w/ global knowledge, can easily compute reachability in Datalog:

R1: $\text{reachable}(S,D) \leftarrow \text{link}(S,D)$

R2: $\text{reachable}(S,D) \leftarrow \text{link}(S,Z), \text{reachable}(Z,D)$

"For all nodes S,D and Z,
If there is a link from S to Z, AND Z can reach D, then S can reach D".

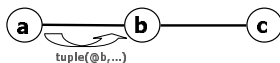
14/24

Network Datalog

- Don't have global data on a router
 - Have to exchange state
- Introduce operators to indicate data placement on neighboring hosts
 - Restrict to neighbors to limit communication

$\text{reachable}(@S,D) \leftarrow \text{link}(@S,Z), \text{reachable}(@Z,D)$

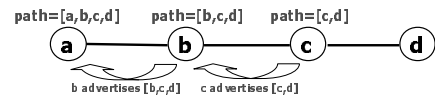
Data placement induces communication



15/24

Path Vector Protocol

- Each node transmits advertisements to its neighbors of all known paths
- Each node receives advertisements, adds itself to paths, and forwards to neighbors



16/24

Network Datalog Path Vector

- Input: Link knowledge from router framework (e.g. receiving periodic pings)
- Output: Paths in the network, i.e. results to push to forwarding plane or return to host

R1: $\text{path}(@S,D,P) \leftarrow \text{link}(@S,D), P=(S,D)$

R2: $\text{path}(@S,D,P) \leftarrow \text{link}(@Z,S), \text{path}(@Z,D,P_2), P=S \cdot P_2$

Query: $\text{path}(@S,D,P)$ Add S to front of P₂

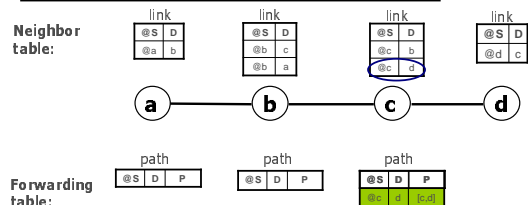
17/24

Path Vector Execution

R1: $\text{path}(@S,D,P) \leftarrow \text{link}(@S,D), P=(S,D)$

R2: $\text{path}(@S,D,P) \leftarrow \text{link}(@Z,S), \text{path}(@Z,D,P_2), P=S \cdot P_2$

Query: $\text{path}(@a,d,P)$



18/24

Path Vector Execution

R1: $\text{path}(@S,D,P) \leftarrow \text{link}(@S,D), P=(S,D).$

R2: $\text{path}(@S,D,P) \leftarrow \text{link}(@Z,S), \text{path}(@Z,D,P_2), P=S \bullet P_2.$

Query: $\text{path}(@a,d,P)$ Matching variable $Z = \text{"Join"}$ 

Neighbor table: **Communication patterns are identical to those in the actual path vector protocol**



$\text{path}(@a,d,[a,b,c,d])$

$\text{path}(@b,d,[b,c,d])$

path

Forwarding table:

@S	D	PP	@S	D	PP	@S	D	P
a	b	[a,b,c,d]	b	c	[b,c,d]	c	d	[c,d]

19/24

All Pairs Best Path

R1: $\text{path}(@S,D,P,C) \leftarrow \text{link}(@S,D,C), P=(S,D).$

R2: $\text{path}(@S,D,P,C) \leftarrow \text{link}(@S,Z,C_1), \text{path}(@Z,D,P_2,C_2), C=C_1+C_2, P=S \bullet P_2.$

R3: $\text{bestPathCost}(@S,D,\min\langle C \rangle) \leftarrow \text{path}(@S,D,Z,C).$

R4: $\text{bestPath}(@S,D,Z,C) \leftarrow \text{bestPathCost}(@S,D,C), \text{path}(@S,D,P,C).$

Query: $\text{bestPath}(@S,D,P,C)$

20/24

Customizable All Pairs Best Path

R1: $\text{path}(@S,D,P,C) \leftarrow \text{link}(@S,D,C), P=(S,D).$

R2: $\text{path}(@S,D,P,C) \leftarrow \text{link}(@S,Z,C_1), \text{path}(@Z,D,P_2,C_2), C=FN(C_1,C_2), P=S \bullet P_2.$

R3: $\text{bestPathCost}(@S,D,AGG\langle C \rangle) \leftarrow \text{path}(@S,D,Z,C).$

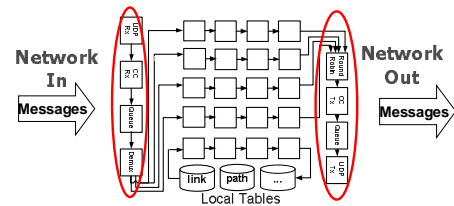
R4: $\text{bestPath}(@S,D,Z,C) \leftarrow \text{bestPathCost}(@S,D,C), \text{path}(@S,D,P,C).$

Query: $\text{bestPath}(@S,D,P,C)$

Customizing C, AGG and FN: lowest RTT, lowest loss rate, highest capacity, *best-k*

21/24

Query Processing Engine



22/24

Questions/Concerns

- How is truth maintenance performed?
- Still limits to application layer reasoning, e.g. path planning.
- Performance? How to aggregate control traffic, e.g. path advertisements?
- Late-binding & source routing policies?

23/24

Other Interesting Possibilities

- Verification of protocols?
- Compilation of code or even hardware for extreme high-performance?
- Tracing/explanatory answers for diagnosing network behavior
 - “Why are you routing there?”

24/24