Path Splicing with Network Slicing

Nick Feamster
Murtaza Motiwala
Santosh Vempala
Goals of Routing

• Reachability

• Scalability

• **High Diversity:** expose paths to end hosts that do not share edges
  – Capacity
  – Fault tolerance

• **Low Stretch:** available paths should not be too circuitous

*Today’s routing protocols do not exploit the diversity of the underlying network graph*
Multipath: Promise and Problems

- **Bad**: If any link fails on both paths, $s$ is disconnected from $t$
- **Want**: End systems remain connected unless the underlying graph is disconnected
Path Splicing: Main Idea

Compute multiple forwarding trees per destination. Allow packets to switch slices midstream.

• **Step 1**: Run multiple instances of the routing protocol, each with slightly perturbed versions of the configuration
• **Step 2**: Allow traffic to switch between instances at any node in the protocol
Perturbations

- **Goal:** Each instance provides different paths
- **Mechanism:** Each edge is given a weight that is a slightly perturbed version of the original weight
  - Two schemes: Uniform and degree-based

```
    s -- 3 --> 4 -- 1.5 --> t
  3         3
```

```
    s -- 5  1.25 --> 1.5 --> t
  3         3.5
```

“Base” Graph

```
    s -- 3 --> 4 -- 1.5 --> t
  1.5
```

```
    s -- 5  1.25 --> 1.5 --> t
  1.5
```

Perturbed Graph
Network Slicing

• **Goal:** Allow multiple instances to co-exist
• **Mechanism:** Virtual forwarding tables
Path Splicing in Practice

- Packet has shim header with *routing bits*

<table>
<thead>
<tr>
<th>IP Header</th>
<th>011001100 ...</th>
<th>Transport Header</th>
<th>Payload</th>
</tr>
</thead>
</table>

- Routers use $lg(k)$ bits to index forwarding tables
  - Shift bits after inspection
  - Incremental deployment is trivial
  - Persistent loops cannot occur

- To access different (or multiple) paths, end systems simply change the forwarding bits
Reliability Approaches that of Underlying Graph

- GEANT (Real) and Sprint (Rocketfuel) topologies
- 1,000 trials
- \( p \) indicates probability edge was removed from base graph

![Graph showing reliability approaches]

- Reliability approaches optimal
- Average stretch is only 1.3
Design and Implementation

- Underway: Click and Quagga on PL-VINI

- Alternative: forwarding tables in separate slices
Open Questions

• Can the end hosts react fast enough to recover from failures?
  – How does the end system find the alternate path?

• What about arbitrary topologies?
High Points

• **Simple:** Routing bits provide access to different paths through the network

• **Scalable:** Exponential increase in available paths, linear increase in state

• **Stable:** Fast recovery does not require fast routing protocols

• No modifications to existing routing protocols
Some Possible Applications

- Fast recovery from poorly performing paths
- Static-routed core: Convergence no longer as important as it once was…
- Fast data transfer with easy multi-path
- Extensions to BGP
- Spatial diversity in wireless networks
- Security applications