Enabling Layer-2 Path Tracing in SDN through Context Encoding

Hui Zhang  Cristian Lumezanu  Junghwan Rhee  Nipun Arora  Qiang Xu  Geoff Jiang

NEC Labs America
Path Tracing in SDN

1. Correctness: Passively monitor the real packets

2. Scalability: Minimal operational, configuration cost

1. Correctness: Trackdown the real path in the data plane
Challenges from Accuracy and Scalability

Correctness vs. Complexity

Scalability vs. Flexibility

Correctness vs. Complexity

- Flow tables @switches
- Configuration @controller

Scalability vs. Flexibility

- Probe packets
- Chosen paths
- Real packets
- Real paths
Reducing Tracing Space

- # of paths is small
  - Monitoring pathlets

- Tagging packets to differentiate paths
  - Precise calling context encoding (PCCE) is utilized
    - Using K bits to differentiate $>2^K$ paths
Precise Calling Context Encoding

1. for s in switches:
   PP[s] = calculatedPossiblePaths(s, A)

2. for s in switches:
   for l = <p, s> in getIncomingEdges(s):
     if l is the first:
       continue
     else:
       PN[l] = PP[s]

Precise Calling Context Encoding

1. for s in switches:
   \[PP[s] = \text{calculatedPossiblePaths}(s, A)\]

2. for s in switches:
   for \(l = <p, s>\) in getIncomingEdges(s):
     if \(l\) is the first:
       continue
     else:
       \[PN[l] = PP[s]\]

3. for \(p\) in DepthFirstSearchPaths(G):
   \[ID[p] = \text{sum}(PN[l]\text{ for } l\text{ in } p)\]

Precise Calling Context Encoding

1. for s in switches:
   \[ PP[s] = \text{calculatedPossiblePaths}(s, A) \]

2. for s in switches:
   for \( l = \langle p, s \rangle \) in getIncomingEdges(s):
     if \( l \) is the first:
       continue
     else:
       \[ PN[l] = PP[s] \]

3. for \( p \) in DepthFirstSearchPaths(G):
   \[ ID[p] = \text{sum}(PN[l] \text{ for } l \text{ in } p) \]

Precise Calling Context Encoding

1. for s in switches:
   PP[s] = calculatedPossiblePaths(s, A)

2. for s in switches:
   for l = <p, s> in getIncomingEdges(s):
      if l is the first:
         continue
      else:
         PN[l] = PP[s]

3. for p in DepthFirstSearchPaths(G):
   ID[p] = sum(PN[l] for l in p)

Precise Calling Context Encoding

1. for s in switches:
   \[ PP[s] = \text{calculatedPossiblePaths}(s, A) \]

2. for s in switches:
   for \( l = \langle p, s \rangle \) in getIncomingEdges(s):
     if \( l \) is the first:
       continue
     else:
       \[ PN[l] = PP[s] \]

3. for p in DepthFirstSearchPaths(G):
   \[ ID[p] = \text{sum}(PN[l] \text{ for } l \text{ in } p) \]

Codebook of Path IDs
Codebook of Path IDs

<table>
<thead>
<tr>
<th>ID</th>
<th>dst</th>
<th>path</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>G</td>
<td>ABDEG</td>
</tr>
<tr>
<td>2</td>
<td>G</td>
<td>ABDFG</td>
</tr>
<tr>
<td>1</td>
<td>G</td>
<td>ACDEG</td>
</tr>
<tr>
<td>3</td>
<td>G</td>
<td>ACDFG</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>
Codebook of Path IDs

- `ID` + `dst` can uniquely identify a path
  - As long as `ID`'s can be imprinted into packets, the paths can be determined

<table>
<thead>
<tr>
<th>ID</th>
<th>dst</th>
<th>path</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>G</td>
<td>ABDEG</td>
</tr>
<tr>
<td>2</td>
<td>G</td>
<td>ABDFG</td>
</tr>
<tr>
<td>1</td>
<td>G</td>
<td>ACDEG</td>
</tr>
<tr>
<td>3</td>
<td>G</td>
<td>ACDFG</td>
</tr>
<tr>
<td>0</td>
<td>B</td>
<td>GEDB</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[6\]
Where to Store Encoded Path IDs

Enabling Actions of “ID=ID+X”

Directly assign values instead of increment
Enabling Actions of “ID=ID+X”

Branch enumerating ingress ports and paths

- Port ‘a’: port with no path
- Port ‘b’: port with one path
- Port ‘c’: port with multiple paths
PathletTracer: Path Tracing System
Conclusions

• A scalable data-plane path tracing application for SDN networks
  – Accuracy: data plane tracing
    • Tagging flows with path identifiers
  – Scalability: calling context encoding
    • Tracing more than $2^K$ paths by re-using K bits in packet headers
Thank You! Questions?