Maple in OpenDaylight

github.com/maplesdn

Andreas Voellmy  Cody Doucette  Yao Xiao  Guanchen Zhang  Shigang Zhu

Problems with SDN Programming Practice

- **Declarative**: specify what, but not how
- **Low-level**: manage flow rule patterns, priorities, timeouts
- **Restrictive**: must use special programming language

To realize full benefits of SDN (for example, in cloud settings), we need better SDN programming abstractions and mechanisms.

Maple Overview

Maple is an SDN programming system that:

1. allows programmers to use standard languages to write centralized algorithms to determine network behavior
2. provides simplifying abstraction that alg. runs per packet
3. includes efficient mechanisms for computing flow tables, multicore scheduling, offloading work to switches

Algorithmic Policies

Algorithmic policies describe how a packet should be forwarded; not flow table rules!

\[ f: (\text{packet} \times \text{env}) \rightarrow \text{route} \]

Maple provides the abstraction that every packet is logically run through the algorithmic policy. Need to be careful in terms of latency, bandwidth, and computational capacity.

Flow Table Compilation

Maple observes dependency of \( f \) on packet data

Maple builds a partial decision tree for \( f \)

Maple compiles flow tables from trace trees

<table>
<thead>
<tr>
<th>Priority</th>
<th>Match</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>topDst == 22</td>
<td>drop</td>
</tr>
<tr>
<td>0</td>
<td>ethDst == 2</td>
<td>drop</td>
</tr>
<tr>
<td>0</td>
<td>ethDst == 4</td>
<td>port 30</td>
</tr>
</tbody>
</table>

Implementing Maple in ODL

As part of a course on cloud computing, we built:

- a Java library for Maple
- a Java adapter for OpenDaylight

With these tools, SDN programmers can now use Maple to write centralized algorithmic policies efficiently using OpenDaylight.

Adapters for other OpenFlow controller implementations can now also be written (Floodlight, OpenStack, …).