Towards A Self-Stabilizing, Plug & Play, In-Band SDN Control Network
Liron Schiff  Stefan Schmid  Marco Canini

Main Contributions

Medieval: A Plug & Play Distributed SDN Control Plane
- Flexible controller membership (additions, removals, failures)
- Automatic topology, controller and switch discovery & management
- Supports ONIX, ElastiCon, Beehive, STN, and more.

Provable Self-Stabilization
- Self-Stabilization: Important concept in fault-tolerant distributed systems: Converge to “good state” from an arbitrary initial state.
- Medieval tolerates failures and delays: low re-convergence times

Background + Model

Network Managed by Multiple Controllers

“Good network state” :=
- Every switch is connected to a controller.
- Controllers can communicate and make joint decisions.

Why in-band? No reason to build, operate, and ensure the reliability of a separate out-of-band network. Also, out-of-band networks are typically underprovisioned and have limited redundancy.

The Medieval Approach

- Controllers aim to continuously grow their management regions...
- … and “conquer” unmanaged switches.
- Management with two spanning trees:
  1. Per-region spanning tree (bidirectional, owned by controller)
  2. Network-wide spanning tree (to connect controllers)

Switch Structure

- Control traffic is sent in-band.
- The switch identifies and forward control traffic to its control module.

Controller to Controller Connectivity

Controller uses a managed switch, R, to detect and establish connection to a new switch S.

Failure Resiliently

- Example:
- Goal: Network should return to a good state.

The Protocol

Switch States

- Controllers “conquer” switches adjacent to their regions of control and build a spanning tree for controller-to-switch connectivity.

Switch States

1. Broadcast
2. Any controller can respond

Unmanaged

Session established

Managed

1. Controller traffic is passed through
2. Other controllers are blocked

No keep-alive timeout

Evaluation

Prototype Implementation
- Emulator in Java
- OpenFlow switches and controllers: lightweight threads
- Links modelled by message queues
- Fat-tree topology (k=4), 1-8 controllers
- Measured time to manage switches

<table>
<thead>
<tr>
<th># ctrls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time (ms)</td>
<td>9382</td>
<td>6983</td>
<td>6150</td>
<td>4224</td>
<td>6035</td>
<td>5104</td>
<td>3704</td>
<td>3680</td>
</tr>
</tbody>
</table>

Formal Analysis

Theorem 1. Medieval is self-stabilizing: Given any initial configuration and set of controllers, Medieval will establish a communication network between controllers and any physically connected network.

Related Work