OFTEN Testing OpenFlow Networks

Maciej Kuźniar, Marco Canini*, Dejan Kostić

EPFL *TU Berlin / T-Labs
Software-Defined Networking (SDN)

Third-party program

OpenFlow
When Faults Happen

• Unreliable communication

• What is the root cause?
We need to test an integrated SDN network
Testing Integrated Network

OpenFlow program

Host A

Host B
Testing Integrated Network

Host A

Host B

OpenFlow program
Challenges

- Unmodified components
  - Switches as black boxes

- Correctness definition
  - Avoid false positives
NICE

Unmodified
OpenFlow
program

Environment model

Topology Description
Correctness Properties

Host A
Host B
State-Space Exploration

Define state and execute transitions
OFTEN

Test harness

OpenFlow program

Switch 1

Switch 2

Environment model

Host A

Host B

HOW?
Problem I: Transitions - Packes

Test harness

Switch Model

<table>
<thead>
<tr>
<th>Flow</th>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>MatchA</td>
<td>CTRL</td>
</tr>
<tr>
<td>MatchB</td>
<td>IN_PORT</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Problem I: Transitions – Control Messages
Problem II: Synchronizing states

Switch state

- Flow table
- Communication channels content
Problem III: Defining Correctness

- High-level correctness properties
  - Network-wide
  - User-defined

- Switch-level

Faulty controller

Faulty switch or model
Results

- 3 applications
  - MAC-learning switch
  - Energy-aware traffic engineering
  - Web server load balancer
- 3 switches
  - Reference Stanford implementation
  - Open vSwitch
  - HP ProCurve E5406zl
Results

Load-balancer issue
• Send to the controller – 0 bytes

Incorrect PacketIn message fields
Conclusions

OFTEN

• Testing of an integrated network
• Systematic state-space exploration
• Real unmodified switches

We need to test an integrated SDN network that contains all real components
Future Work

Integrated network testing
• Required functionalities
• Interface extensions