Controller-agnostic SDN Debugging

Ram Durairajan*, Joel Sommers^, Paul Barford*

*University of Wisconsin - Madison    ^Colgate University
Motivation

- Debugging SDN applications is hard
- “Runs as designed” may be insufficient
- Deployments must cope with wide range of operating conditions
- How can we answer the following question:

  Will my SDN app run as designed when deployed in a live setting?

  Our solution: OFF!
Design goals of OFF

• Controller-agnostic debugging and test environment for SDN developers
• Default debugging options
  • Stepping, breakpoints, watch variables, etc.
• Comprehensive testing for SDN applications
  • Packet replay, packet tracing, visualization, alerts, etc.
• Tie unwanted network behavior to faulty controller logic in source code
• Simple, light-weight and no hardware support
• Facilitate transition to live environments
fs-sdn simulation engine

• Fast and Accurate SDN prototyping (Gupta et al., HotSDN 2013)
• Seamless transition of controllers to real deployments
• Based on fs simulator (Sommers et al., IEEE Infocom 2011)
  – Discrete event simulation techniques
  – Core abstraction is flowlets; high performance
  – Transparently incorporates POX components
• Significant extensions to support OFF

rkrish@cs.wisc.edu
OFFf architecture

OFFf Proxy

User Interface Wrapper

Debugger
Trace Replay
Diff Report Generator

fs-sdn Simulator

POX RYU OpenDaylight Trema FloodLight

rkrish@cs.wisc.edu
OFFf commands

• longlist and shortlist source code
• pretty print expressions
• hide and unhide frames
• interactive interpreter with all variables in scope
• track, watch, or unwatch variables
• edit source files during debugging
• enable or disable break points on the fly
• sticky mode to visualize code
OFFf additional features

• Trace packet through the network
  – Holistic view of flows, controller and switches
  – No additional hardware

• Replay packets later
  – No OFP modification

• Detect configuration changes
  – Topology changes
  – Rule/action changes
  – Performance variations
OFFf in action

• We demonstrate OFFf in three scenarios
  – Bad multi-app interaction
  – Incorrect ordering of updates
  – Unexpected rule expiration

• Goal: Identify logical bugs in the source code that lead to transient outages and losses
OFFf in action

• We demonstrate OFFf in three scenarios
  – Bad multi-app interaction
  – Incorrect ordering of updates
  – Unexpected rule expiration

• Goal: Identify logical bugs in the source code that lead to transient outages and losses
Bad multi-app interaction

Block: 10.0.0.1 to 10.0.0.4
Modify: From: 10.0.0.1
SrcIP: 10.0.0.2
Modify: To: 10.0.0.3
DstIP: 10.0.0.4
Allow: 10.0.0.2 to 10.0.0.3

10.0.0.1

Firewall

Routing App

POX Controller

Switch

Web server (10.0.0.4:80)

rkrish@cs.wisc.edu
Solution: Bad multi-app interaction

- Using Offf developer 2 can
  - collect network traces (T1)
  - prototype routing app using fs-sdn
  - collect traces again (T2)
  - runs diff reports (T1 and T2)
  - Rule set conflicts are found
  - Change and iterate
  - Verify firewall invariants

rkrish@cs.wisc.edu
Conclusion

• OFFf – a controller-agnostic debugging and test environment for SDN developers
• OFFf is simple, flexible, and light-weight
• We demonstrate OFFf using three scenarios
• Future work
  – Generation of regression tests, fuzz testing, etc.
Thank you!

Source Code
https://github.com/52-41-4d/fs-master

Questions?

rkrish@cs.wisc.edu