Where is the Debugger for my Software-Defined Network?

Nikhil Handigol, Brandon Heller, Vimalkumar Jeyakumar,
David Mazières, Nick McKeown

Stanford University
Bug story: incomplete handover
Bug story: incomplete handover

A

Switch X

WiFi AP Y

WiFi AP Z

B
Bug story: incomplete handover
Bug story: incomplete handover

A

Switch X

WiFi AP Y

WiFi AP Z

B
Debugging SDNs

• Bugs can be anywhere in the SDN stack
  – Hardware, control plane logic, race conditions
• Switch state might change rapidly
• Bugs might show up rarely
How can we exploit the SDN architecture to systematically track down the root cause of bugs?
ndb: Network Debugger

Goal

- Capture and reconstruct the sequence of events leading to the errant behavior

Allow users to define a Network Breakpoint

- A (header, switch) filter to identify the errant behavior

Produce a Packet Backtrace

- Path taken by the packet
- State of the flow table at each switch
Debugging software programs

**Function A()**:  
\[ i = ...; \ j = ...; \ u = B(i, \ j) \]

**Function B(x, y)**:  
\[ k = ...; \ v = C(x, \ k) \]

**Function C(x, y)**:  
\[ ... \ w = \text{abort}() \]

**Breakpoint**  
“line 25, \ w = \text{abort}()”

**Backtrace**  
File “A”, line 10, **Function A()**  
File “B”, line 43, **Function B()**  
File “C”, line 21, **Function C()**
Debugging networks

**Breakpoint**
"ICMP packets A→B, arriving at X, but not Z"

**Backtrace**

```
Switch X: {
  inport: p0,
  outports: [p1]
  mods: [...]  
  matched flow: 23 [...]  
  matched table version: 3
}

Switch Y: {
  inport: p1,
  outports: [p3]
  mods: ...
  ...
}
```
Using ndb to debug common issues

Reachability
- Symptom: A is not able to talk to B
- Breakpoint: “Packet A->B, not reaching B”

Isolation
- Symptom: A is talking to B, but it shouldn’t
- Breakpoint: “Packet A->B, reaching B”

Race conditions
- Symptom: Flow entries not reaching on time
- Breakpoint: “Packet-in at switch S, port P”
So, how does ndb work?
Control Plane

Flow Table State Recorder

Breakpoint
Switch = S
IP src = A, IP dst = B
TCP Port = 22
Breakpoint
Switch = S
IP src = A, IP dst = B
TCP Port = 22
Control Plane

Flow Table State Recorder

Breakpoint
Switch = S
IP src = A, IP dst = B
TCP Port = 22

Postcard Collector
**Breakpoint**

Switch = S
IP src = A, IP dst = B
TCP Port = 22

Flow Table State Recorder

Control Plane
Breakpoint
Switch = S
IP src = A, IP dst = B
TCP Port = 22
Breakpoint
Switch = S
IP src = A, IP dst = B
TCP Port = 22
Breakpoint
Switch = S
IP src = A, IP dst = B
TCP Port = 22
Who benefits

Network developers
  – Programmers debugging control programs

Network operators
  – Find policy errors
  – Send error report to switch vendor
  – Send error report to control program vendor
Performance and scalability

Control channel

- Negligible overhead
- No postcards
- Extra flow-mods

Postcards in the datapath

- Single collector server for the entire Stanford backbone
- Selective postcard generation to reduce overhead
- Parallelize postcard collection
Status

First working prototype of ndb
  – Works without change to switches or controller

Code undergoing heavy churn
  – Will be made available once stable
Summary

- **ndb**: Network Breakpoint + Packet Backtrace
- Systematically track down root cause of bugs
- Practical and deployable today