Access Control in Software-Defined Networking (SDN)

and an Implementation of SDN-based Network Access Control

Yuchen Wu
yuchen.wu@tuhh.de

Agenda

- Introduction
- Background
  - Software-Defined Networking (SDN)
  - OpenFlow
  - Access Control in SDN
- A Prototype of SDN-based Network Access Control
- Conclusion

Agenda

- Introduction
- Background
    - Software-Defined Networking (SDN)
    - OpenFlow
    - Access Control in SDN
- A Prototype of SDN-based Network Access Control
- Conclusion

Old Fashion: Conventional Network

Proprietary

Closed

No Abstraction

Inability to scale

Manual / SNMP

Mature **Stable**

Slow Evolution

New Trend: SDN

Abstracted

Open-Sourced

Easy to scale

Programmable Control

Continuing Trend

Known & Unknown Security Risks

Fast Evolution

How to Manage a Network

- Conventional Network
  - Network Access Control (NAC)
  - Authorization: Access Control
- Mitigation to SDN
  - NAC
  - Access Control for Applications

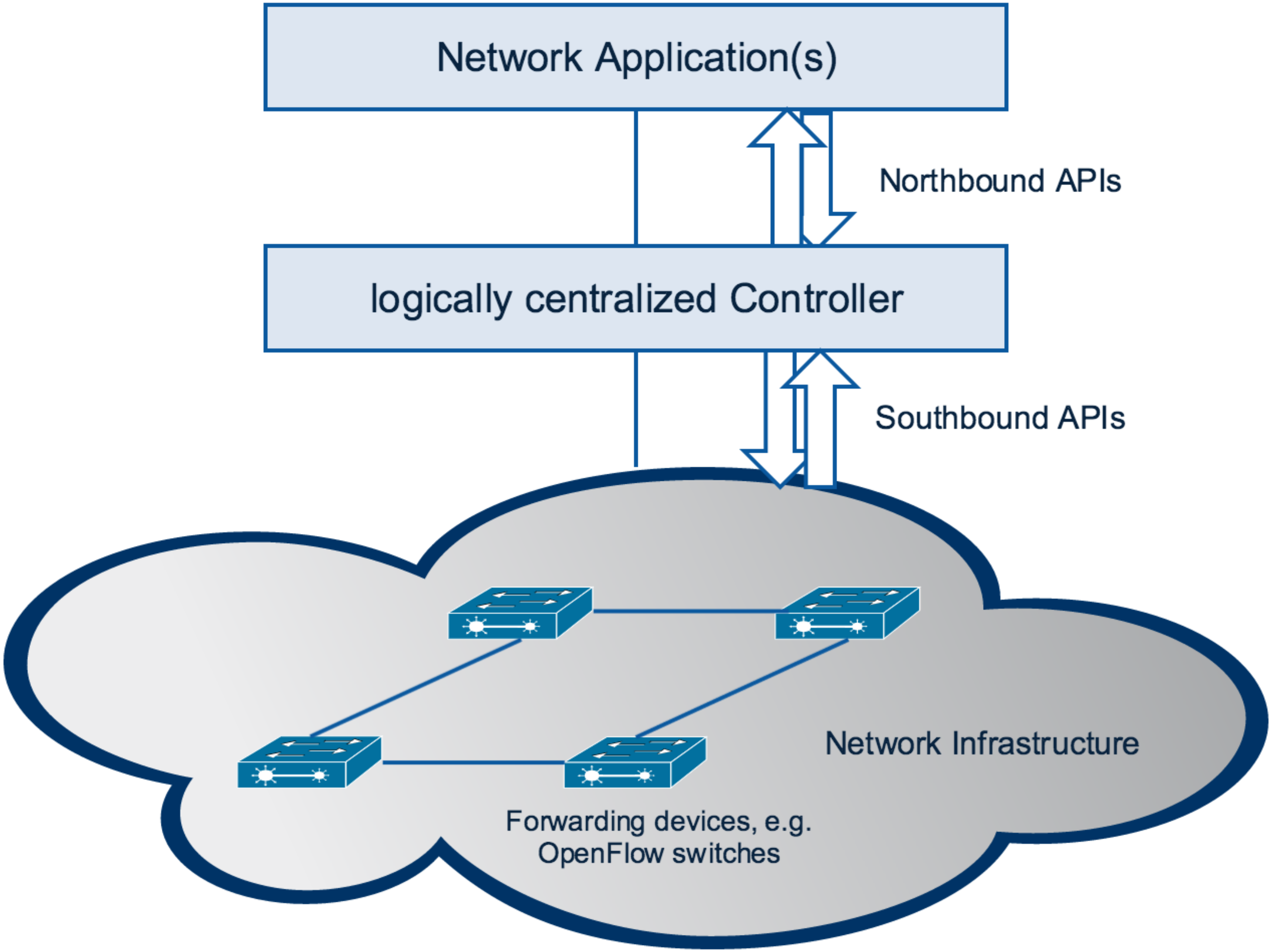
Agenda

- Introduction
- Background
  - Software-Defined Networking (SDN)
  - OpenFlow
  - Access Control in SDN
- A Prototype of SDN-based Network Access Control
- Conclusion

Background: SDN

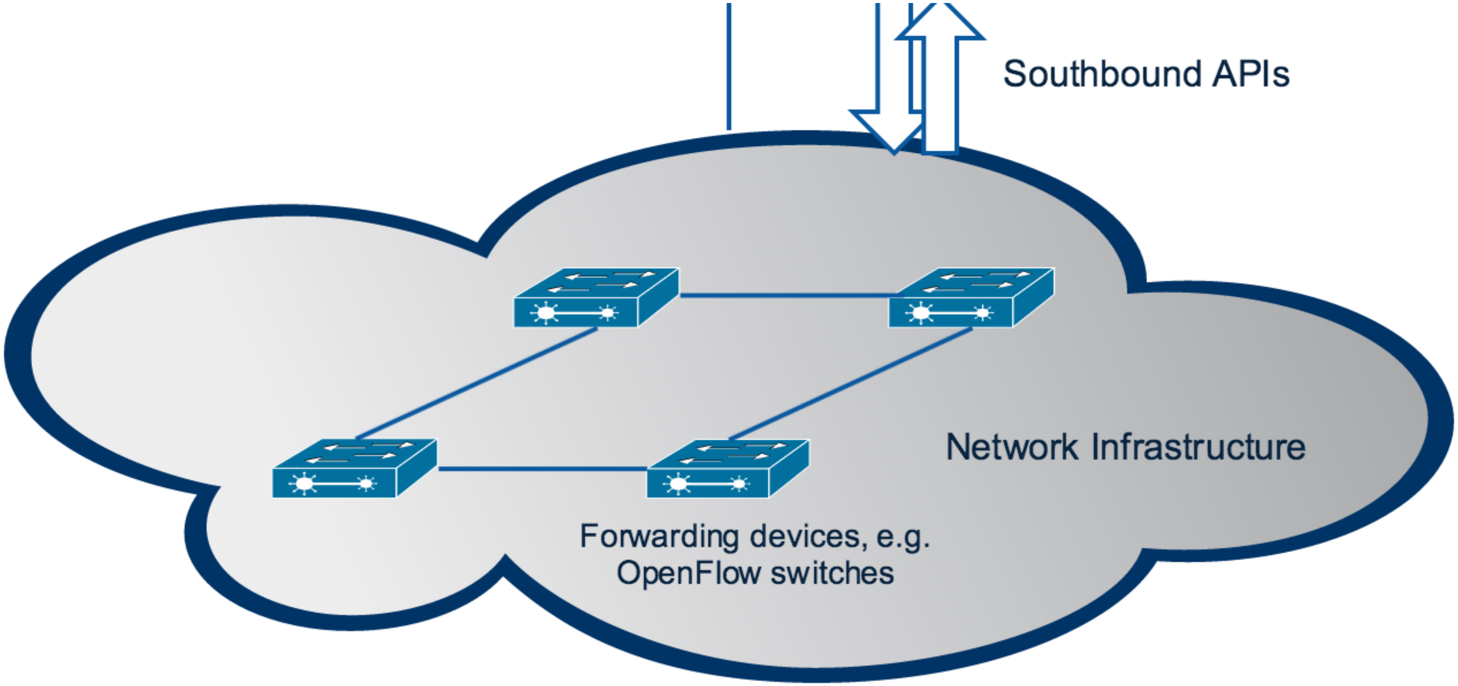
- Power of Abstraction
- ➢ Detachment of Control and Data Plane
- ➢ Centralized Control Logic
- ➢ Programmable Network
- ➢ Dumb Switches

Background: SDN

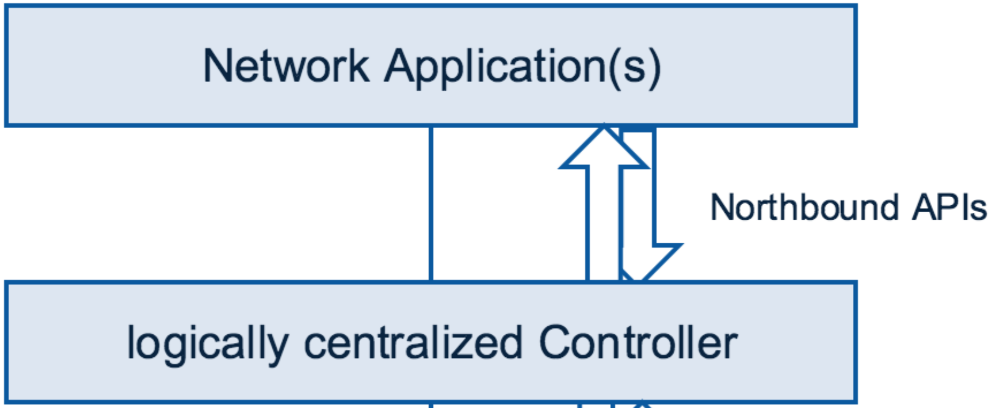


SDN Architecture

- Network Infrastructure
  - e.g. OpenFlow-enabled switches
  - Southbound API: OpenFlow

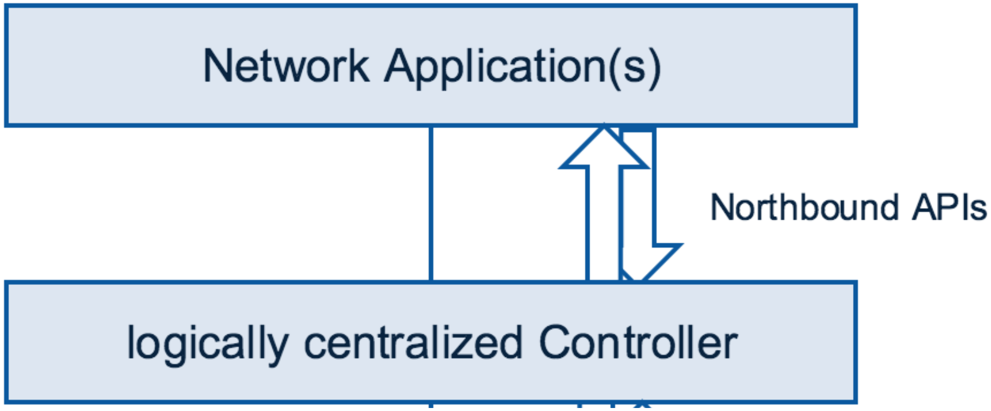


SDN Architecture (2)



- Controller
  - Communicates with all network devices
  - Updates the network topology
  - Northbound API: REST API, Java/Python API
  - NOX, POX, Floodlight, Ryu, OpenDayLight

SDN Architecture (3)



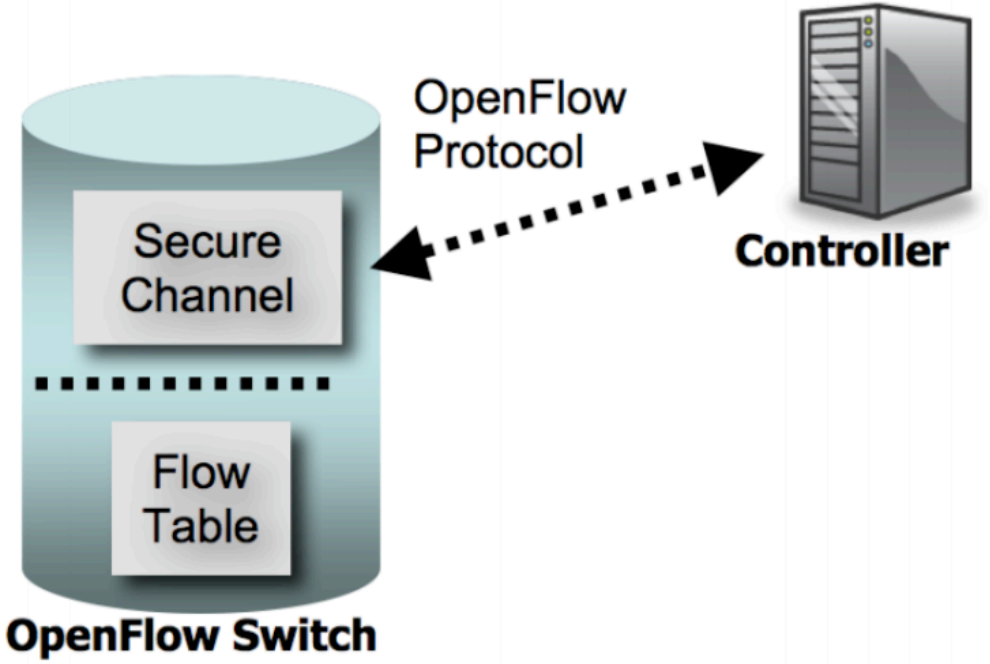
- Network Applications
  - All services, policies and features
  - Security, QoS, load balancing, etc.

Background: OpenFlow

- Communication Protocol – Southbound
  - for traffic between switches and the SDN controller
- Open-Sourced
  - Open Networking Foundation (ONF)
- First standardized and most dominant

OpenFlow (2)

- OpenFlow Switch
  - Flow Table
  - Secure Channel



OpenFlow (3)

- Flow Table
  - Header Fields

    IP src/dst, IP proto, (TCP/UDP src/dst port), etc.
  - Counters
  - Actions

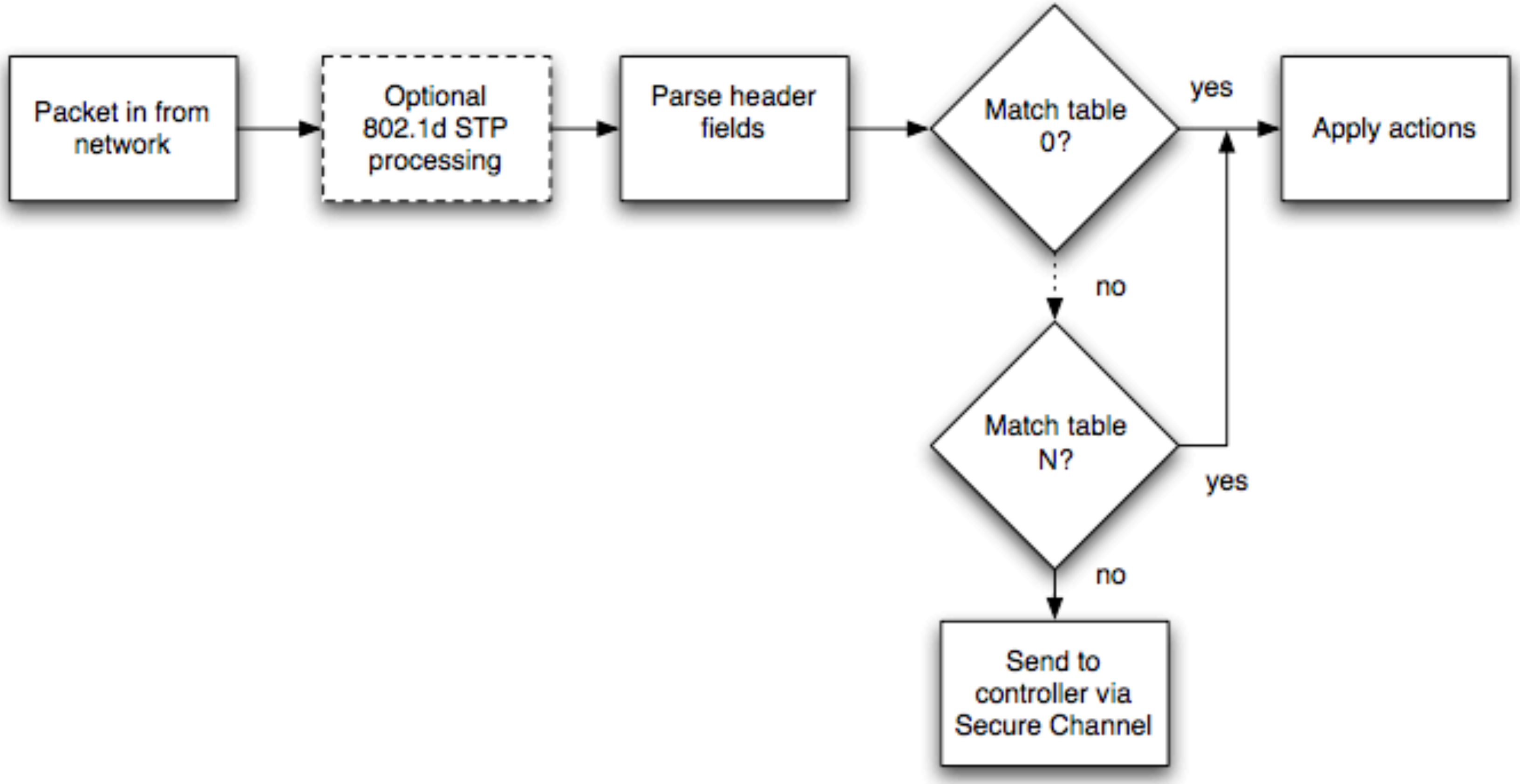
    Forwarding, Dropping, etc.

Header Fields	Counters	Actions

Flow Table

OpenFlow (4)

- Matching



Background: Access Control in SDN

- Network Access Control (NAC)
  - Intuitive to OpenFlow-based SDN
  - Authentication
  - Dynamic NAC policy

Access Control in SDN

- Security Risks
  - Infected end-user systems
  - Malicious and erroneous applications
- Access Control for Applications
  - Role-based access control

Access Control for Applications

- SE-Floodlight
  - Three roles: ADMIN, SEC and APPLICATION
- PermOF
  - Four roles: read, notify real-time events, write and system permission
  - Isolates the Controller and applications
- Rosemary
  - Sandbox mechanism to quarantine specific applications
  - Check module intercepts privileged calls

Agenda

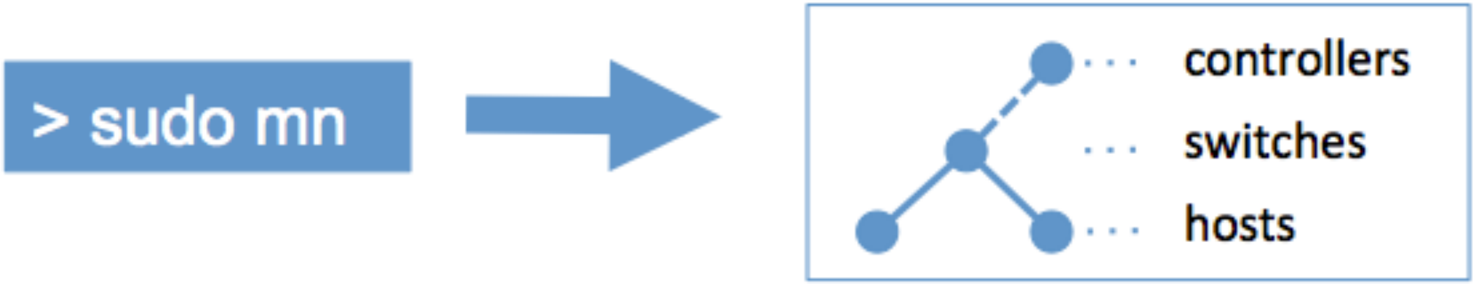
- Introduction
- Background
    - Software-Defined Networking (SDN)
    - OpenFlow
    - Access Control in SDN
- A Prototype of SDN-based Network Access Control
- Conclusion

Development Environment

- Mininet
  - Realistic virtual network emulator
  - Open-sourced, designed for SDN research at first
- Floodlight
  - OpenFlow controller
  - Java, open-sourced
  - Provides REST API

Mininet

- Download a Mininet VM or install it on your Linux VM
- Play it with Mininet CLI or its Python API
- Complete documentation
- Have look at Github



```
> sudo mn
```

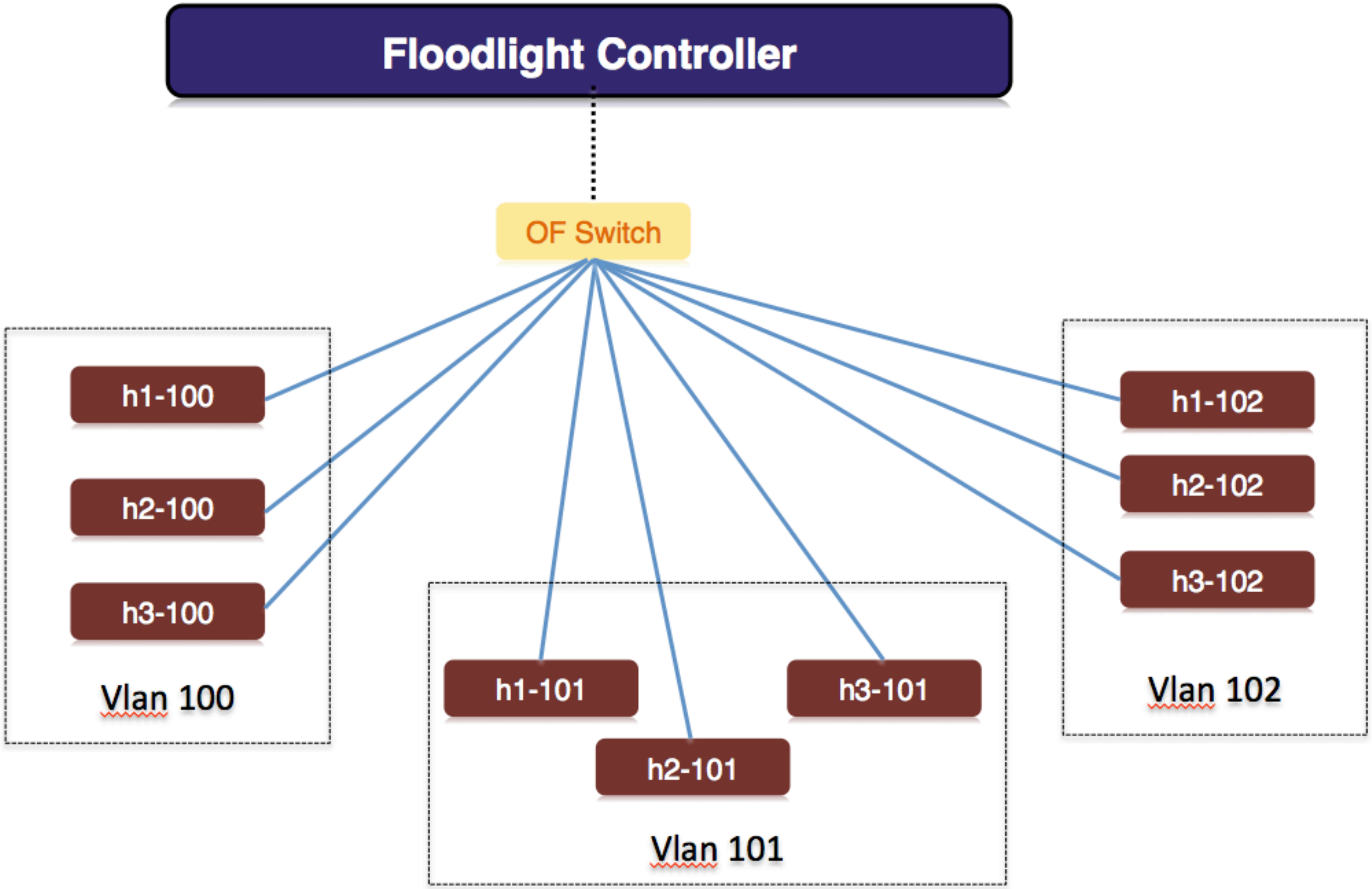
Floodlight

- Download Floodlight from its official website
- Tutorials step-by-step (both for developers and users)
- Modular designed
- Community and Github

Prototype Design

- Monitoring all devices adding events
- Recording a device and its connected switch as a pair [DPID, IP addr]
- Mapping access control rules into flows and insert flows to switch
- Inserting an existing rule returns error

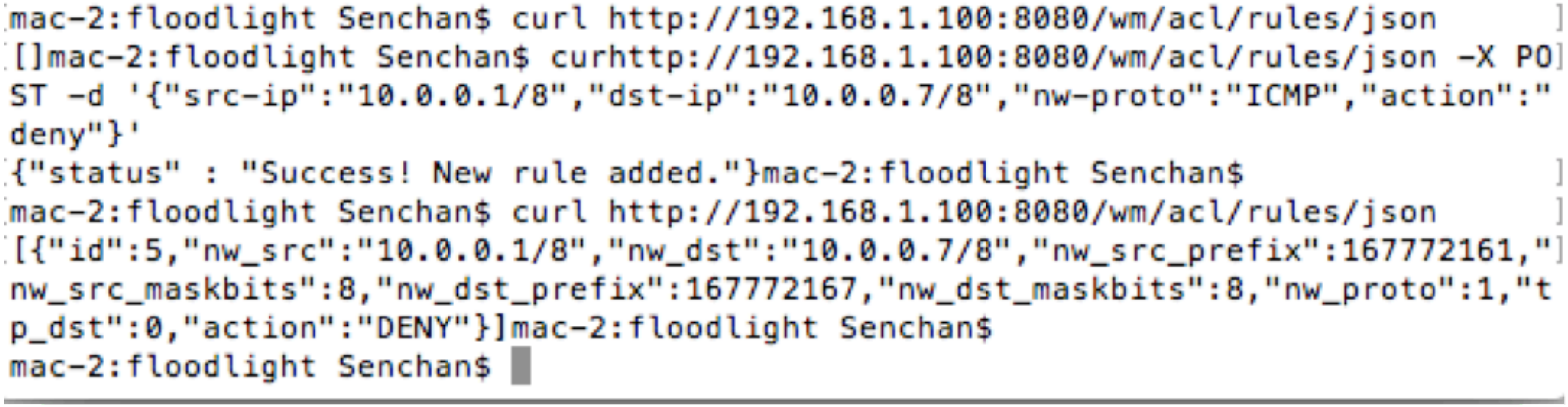
Network Topology



Adding a Rule

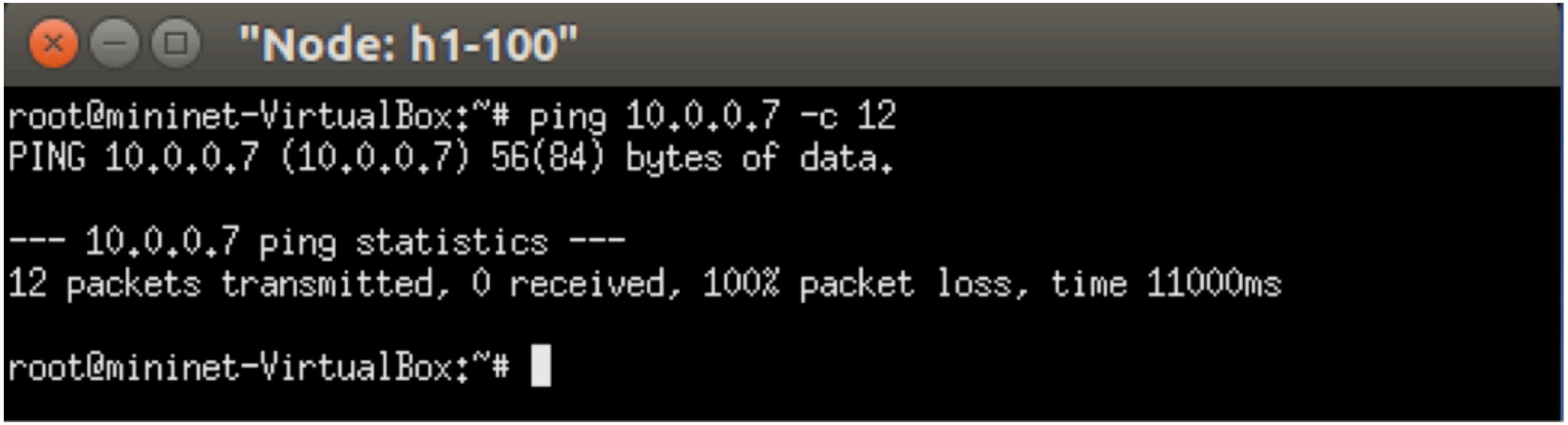
- ICMP Traffics from h3-100[10.0.0.1/8] to h1-100[10.0.0.7/8] is denied

- curl [controller address:port]/wm/acl/rules/json -X POST -d ’"src-ip":"[source ip addresss]","dst- ip":"[destination ip address]","nw-proto":"[protocol:ICMP/TCP/UDP]","action":"[deny/allow]"’;



```
mac-2:floodlight Senchan$ curl http://192.168.1.100:8080/wm/acl/rules/json
[]mac-2:floodlight Senchan$ curhttp://192.168.1.100:8080/wm/acl/rules/json -X PO
ST -d '{"src-ip":"10.0.0.1/8","dst-ip":"10.0.0.7/8","nw-proto":"ICMP","action":"
deny"}'
{"status" : "Success! New rule added."}mac-2:floodlight Senchan$
mac-2:floodlight Senchan$ curl http://192.168.1.100:8080/wm/acl/rules/json
[{"id":5,"nw_src":"10.0.0.1/8","nw_dst":"10.0.0.7/8","nw_src_prefix":167772161,"
nw_src_maskbits":8,"nw_dst_prefix":167772167,"nw_dst_maskbits":8,"nw_proto":1,"t
p_dst":0,"action":"DENY"}]mac-2:floodlight Senchan$
mac-2:floodlight Senchan$
```

Adding a Rule (2)



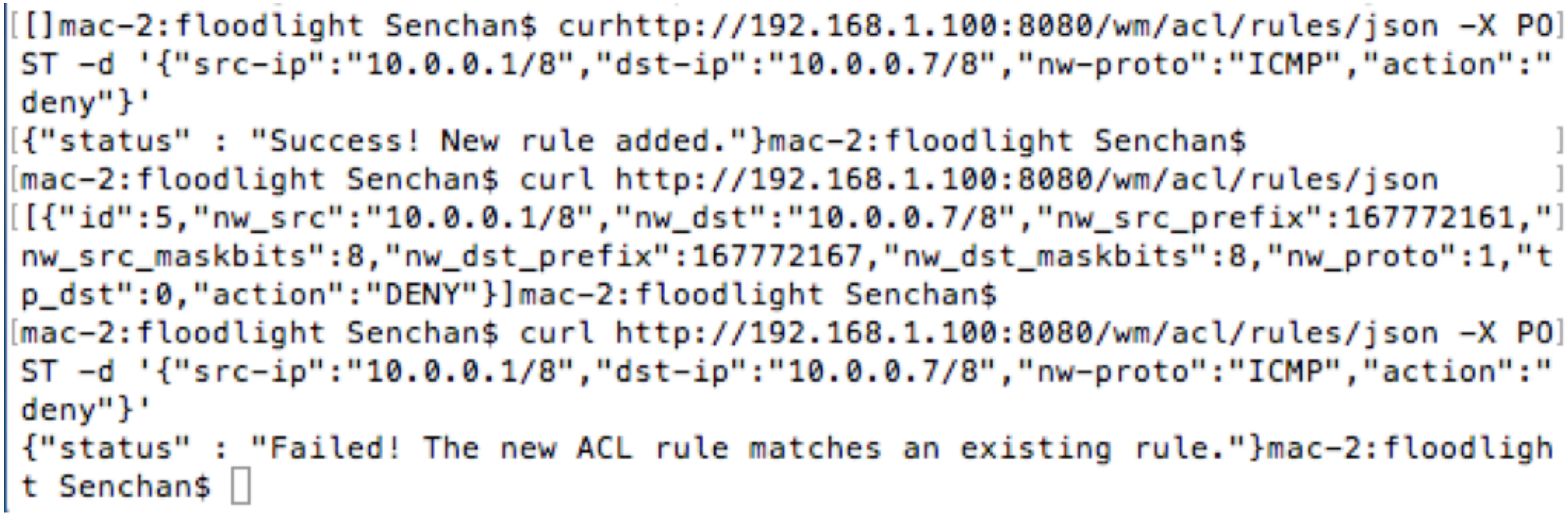
```
root@mininet-VirtualBox:~# ping 10.0.0.7 -c 12
PING 10.0.0.7 (10.0.0.7) 56(84) bytes of data.

--- 10.0.0.7 ping statistics ---
12 packets transmitted, 0 received, 100% packet loss, time 11000ms

root@mininet-VirtualBox:~# 
```

Rule Confliction

- Add a conflicting rule, return error

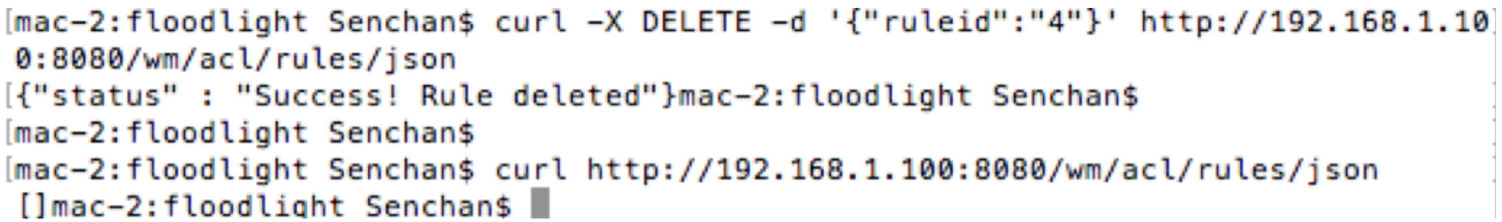


```
[]mac-2:floodlight Senchan$ curhttp://192.168.1.100:8080/wm/acl/rules/json -X PO
ST -d '{"src-ip":"10.0.0.1/8","dst-ip":"10.0.0.7/8","nw-proto":"ICMP","action":"
deny"}'
{"status" : "Success! New rule added."}mac-2:floodlight Senchan$
mac-2:floodlight Senchan$ curl http://192.168.1.100:8080/wm/acl/rules/json
[{"id":5,"nw_src":"10.0.0.1/8","nw_dst":"10.0.0.7/8","nw_src_prefix":167772161,"
nw_src_maskbits":8,"nw_dst_prefix":167772167,"nw_dst_maskbits":8,"nw_proto":1,"t
p_dst":0,"action":"DENY"}]mac-2:floodlight Senchan$
mac-2:floodlight Senchan$ curl http://192.168.1.100:8080/wm/acl/rules/json -X PO
ST -d '{"src-ip":"10.0.0.1/8","dst-ip":"10.0.0.7/8","nw-proto":"ICMP","action":"
deny"}'
{"status" : "Failed! The new ACL rule matches an existing rule."}mac-2:floodligh
t Senchan$ 
```

Deleting a Rule

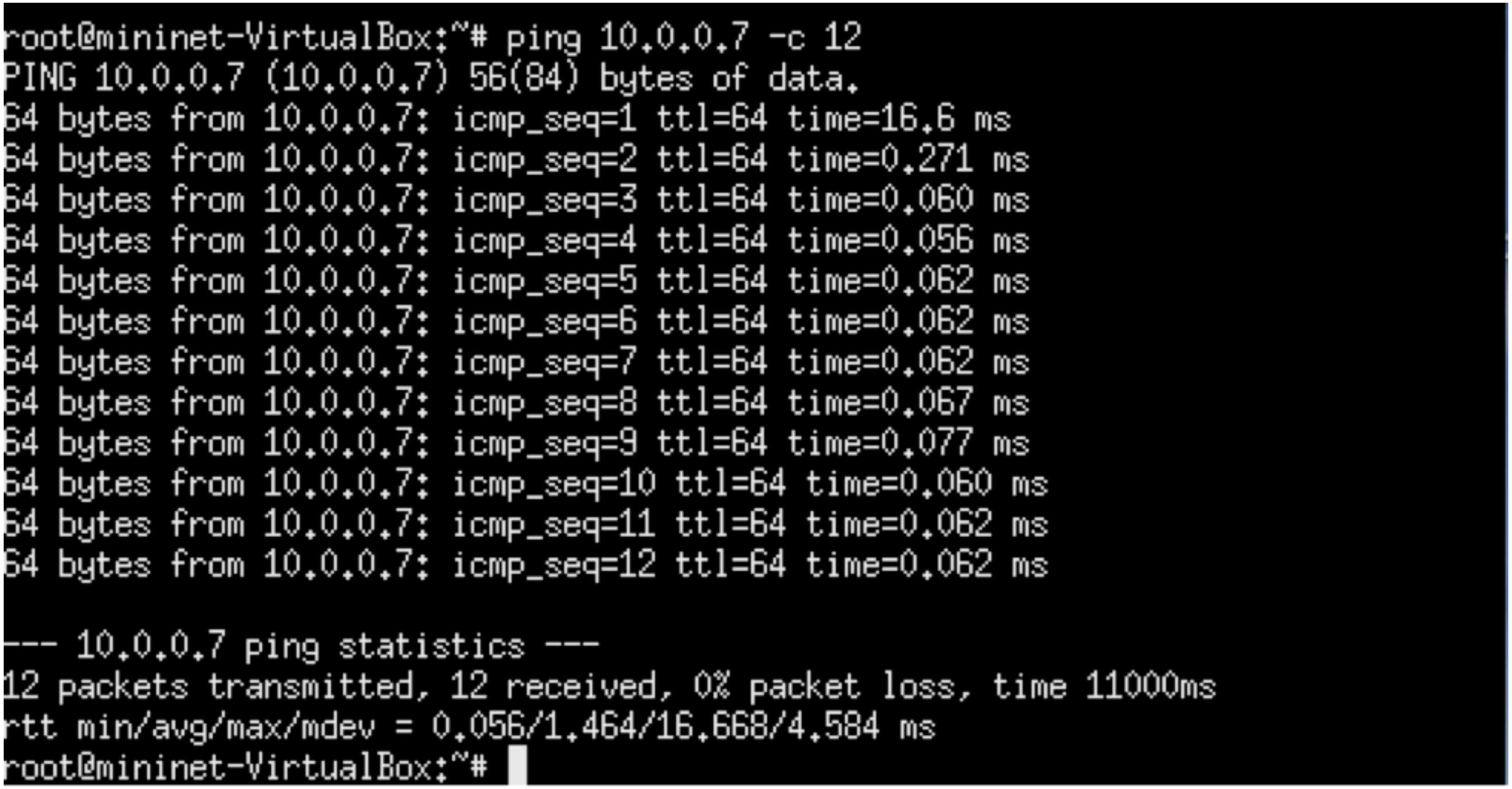
- Delete the rule just added

➢ curl [controller address:port]/wm/acl/rules/json -X POST -d ’"src-ip":"[source ip addressss]","dst- ip":"[destination ip address]","nw-proto":"[protocol:ICMP/TCP/UDP]","action":"[deny/allow]"’;



```
[mac-2:floodlight Senchan$ curl -X DELETE -d '{"ruleid":"4"}' http://192.168.1.10
 0:8080/wm/acl/rules/json
[{"status" : "Success! Rule deleted"}mac-2:floodlight Senchan$
[mac-2:floodlight Senchan$
[mac-2:floodlight Senchan$ curl http://192.168.1.100:8080/wm/acl/rules/json
 []mac-2:floodlight Senchan$
```

Deleting a Rule (2)



```
root@mininet-VirtualBox:~# ping 10.0.0.7 -c 12
PING 10.0.0.7 (10.0.0.7) 56(84) bytes of data.
64 bytes from 10.0.0.7: icmp_seq=1 ttl=64 time=16.6 ms
64 bytes from 10.0.0.7: icmp_seq=2 ttl=64 time=0.271 ms
64 bytes from 10.0.0.7: icmp_seq=3 ttl=64 time=0.060 ms
64 bytes from 10.0.0.7: icmp_seq=4 ttl=64 time=0.056 ms
64 bytes from 10.0.0.7: icmp_seq=5 ttl=64 time=0.062 ms
64 bytes from 10.0.0.7: icmp_seq=6 ttl=64 time=0.062 ms
64 bytes from 10.0.0.7: icmp_seq=7 ttl=64 time=0.062 ms
64 bytes from 10.0.0.7: icmp_seq=8 ttl=64 time=0.067 ms
64 bytes from 10.0.0.7: icmp_seq=9 ttl=64 time=0.077 ms
64 bytes from 10.0.0.7: icmp_seq=10 ttl=64 time=0.060 ms
64 bytes from 10.0.0.7: icmp_seq=11 ttl=64 time=0.062 ms
64 bytes from 10.0.0.7: icmp_seq=12 ttl=64 time=0.062 ms

--- 10.0.0.7 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11000ms
rtt min/avg/max/mdev = 0.056/1.464/16.668/4.584 ms
root@mininet-VirtualBox:~# 
```

Conclusion

- Background theory on SDN and OpenFlow
- SDN-based Network Acess Control (NAC)
- Access Control for Applications
- Prototype of NAC module in Floodlight Controller

Experience on how-to-learn

- Nick Feamster's Cousera course
- Read OpenFlow's whitepaper
- SDN Reading list
- Deploy the environment at the beginning!
- Documentation, Community, Github, StackOverFlow

Questions?

Thank You!

Literature

- O. N. Foundation: Software-defined networking: The new norm for networks
- N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner: Openflow: Enabling innovation in campus networks
- P. A. Porras, S. Cheung, M. W. Fong, K. Skinner, and V. Yegneswaran: Securing the software defined network control layer
- S. Shin, Y. Song, T. Lee, S. Lee, J. Chung, P. Porras, V. Yegneswaran, J. Noh, and B. B. Kang: A robust, secure, and high-performance network operating system
- X.Wen, Y.Chen, C.Hu, C.Shi, and Y.Wang: Towards a secure controller platform for openflow applications
- O. S. Consortium *et al. :* Openflow switch specification version 1.0.0

Literature (2)

- B.Jäger, C.Röpke, I.Adam, and T.Holz: Multi-layer access control for sdn-based telco clouds
- B. Lantz, B. Heller, and N. McKeown: A network in a laptop: Rapid prototyping for software-defined networks
- F. Project: Floodlight controller