OperationCheckpoint: SDN Application Control

Queen's University Belfast

CSIT CENTRE FOR SECURE INFORMATION TECHNOLOGIES

Workshop on Secure Network Protocols (NPSec'14)

19 October 2014

Sandra Scott-Hayward, Christopher Kane and Sakir Sezer

s.scott-hayward@qub.ac.uk

Centre for Secure Information Technologies (CSIT)

Est.2009, Based in The ECIT Institute

Initial funding over £30M

80 People

- Researchers
- Engineers
- Business Development

Largest UK University lab for cyber security technology research

GCHQ Academic Centre of Excellence

Industry Informed

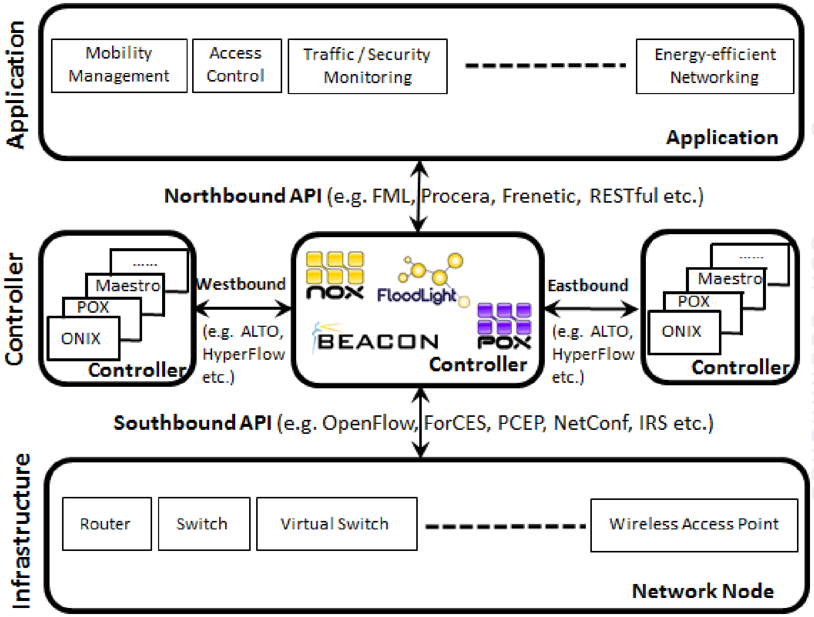
- Open Innovation Model

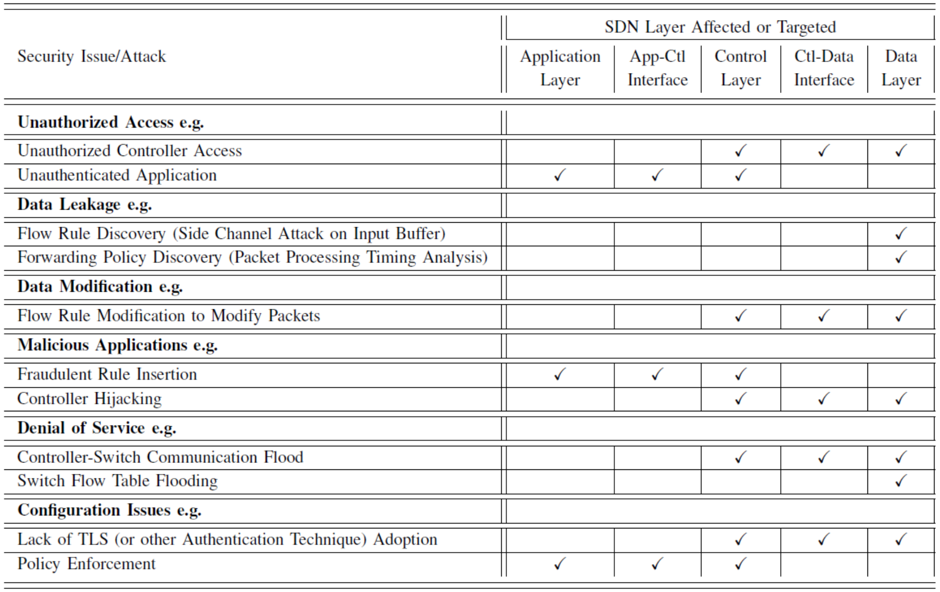
Strong international links

- ETRI, CyLab, GTRI, SRI International
- Cyber Security Technology Summit

SDN Research

Software Defined Networking …. and Security





Security Issue/Attack	SDN Layer Affected or Targeted				
	Application Layer	App-Ctl Interface	Control Layer	Ctl-Data Interface	Data Layer
Unauthorized Access e.g.					
Unauthorized Controller Access			✓	✓	✓
Unauthenticated Application	✓	✓	✓		
Data Leakage e.g.					
Flow Rule Discovery (Side Channel Attack on Input Buffer)					✓
Forwarding Policy Discovery (Packet Processing Timing Analysis)					✓
Data Modification e.g.					
Flow Rule Modification to Modify Packets			✓	✓	✓
Malicious Applications e.g.					
Fraudulent Rule Insertion	✓	✓	✓		
Controller Hijacking			✓	✓	✓
Denial of Service e.g.					
Controller-Switch Communication Flood			✓	✓	✓
Switch Flow Table Flooding					✓
Configuration Issues e.g.					
Lack of TLS (or other Authentication Technique) Adoption			✓	✓	✓
Policy Enforcement	✓	✓	✓		

Sezer, S., et al. “Are We Ready for SDN? Implementation Challenges for Software-Defined Networks” *IEEE Communications Magazine*, July 2013

Scott-Hayward, S., O’Callaghan, G. and Sezer, S. “SDN Security: A Survey” *IEEE SDN4FNS*, November 2013

Problem Description

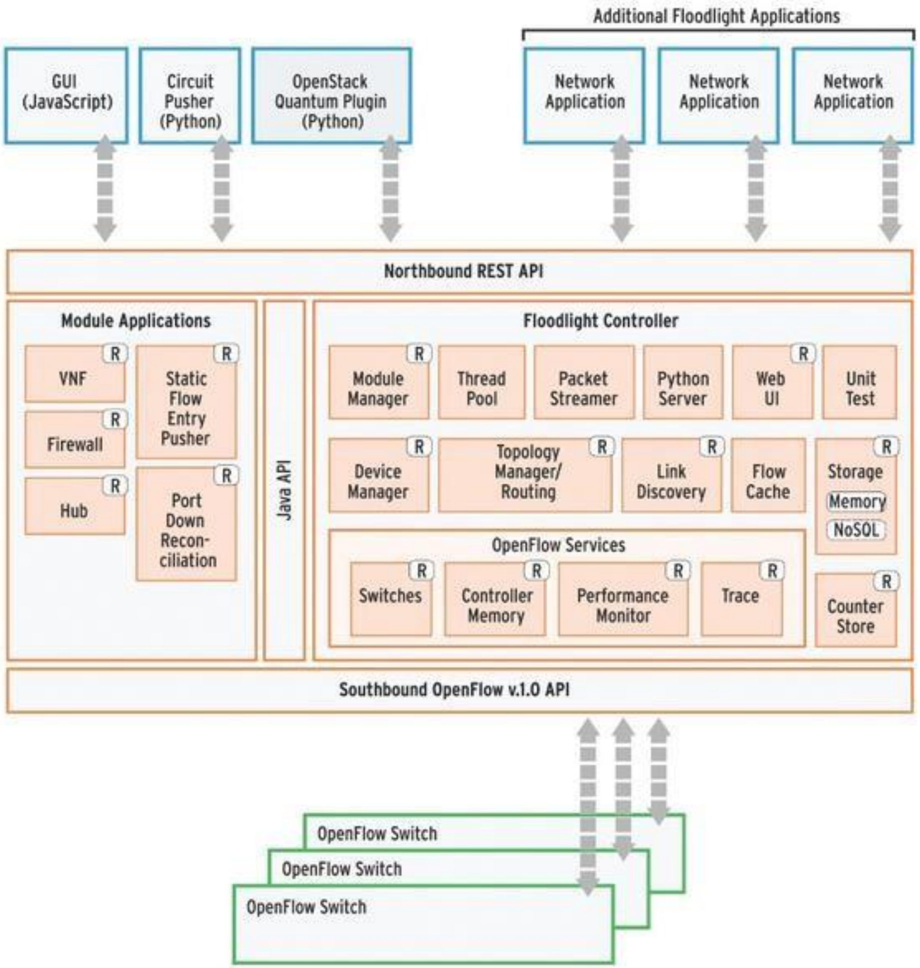
Fundamental security challenge is the ability for a malicious application to access network state information and manipulate network traffic for nefarious purposes.

Northbound Interface (NBI) Communication involves:

- Reading Network State
- Writing Network Policies

Objective: Protect against unauthorized control function access attempts

Floodlight Architecture



OpenFlow Controller Article, Floodlight Architecture and Relationships, http://www.admin-magazine.com/

CENTRE FOR SECURE INFORMATION TECHNOLOGIES

Problem Description

Weaknesses in current approach:

- No authentication of RESTful API commands
- No scheme to ensure rules installed do not overlap or interfere with one another
- Applications do not have to provide identity information
- No application regulation or behaviour inspection after installation

Potential Solutions:

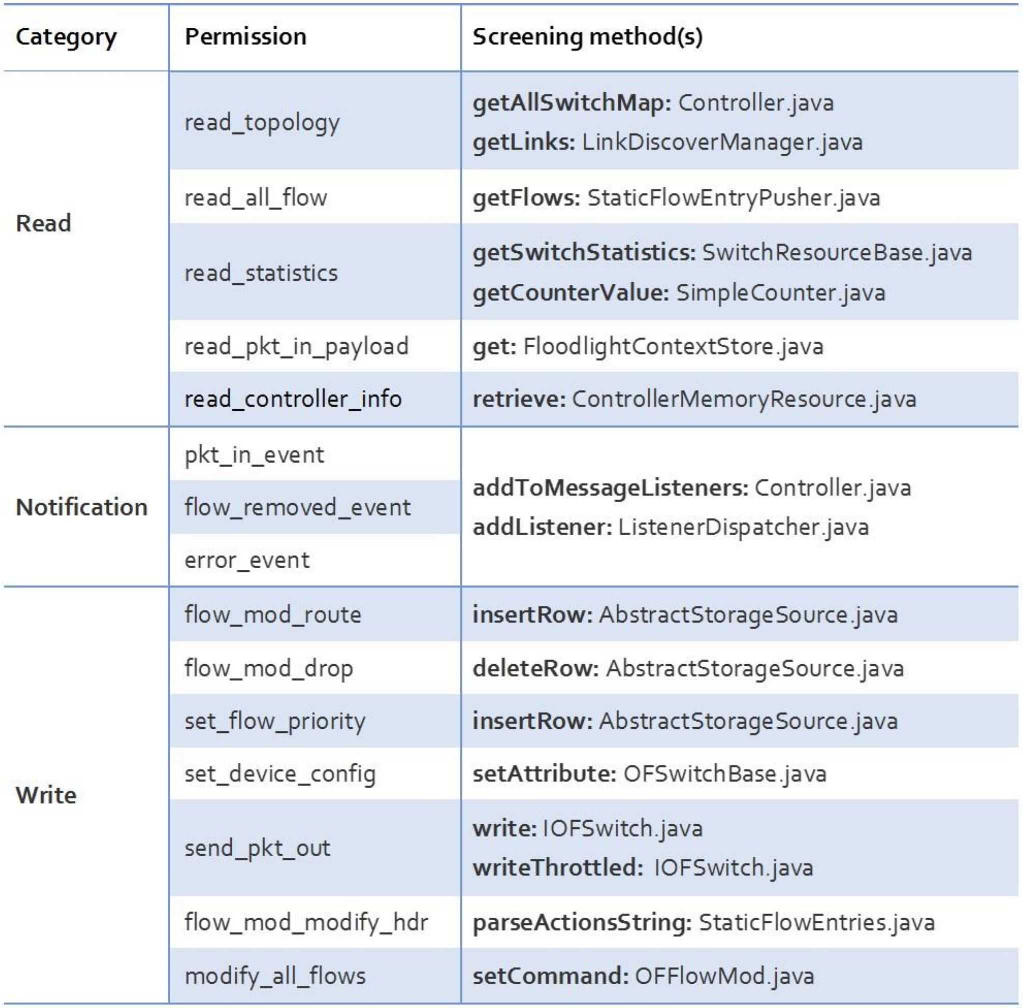
- Rule conflict detection and correction
- Application identification and priority enforcement
- Malicious activity detection and mitigation

System Design

System Attributes:

1. Define a complete set of permissions
2. Provide a secure storage structure for saving unique application IDs mapped to the set of permissions granted to that application
3. Provide a means for the network administrator/operator to add/remove application permissions (by its unique ID)
4. Provide a REST call for applications to query the controller and discover their assigned permissions
5. Secure the methods, in the Floodlight controller, that carry out the functions described by each of the permissions in the permission set
6. Log all unauthorized operation attempts to a log file for auditing purposes

Permissions Categorization



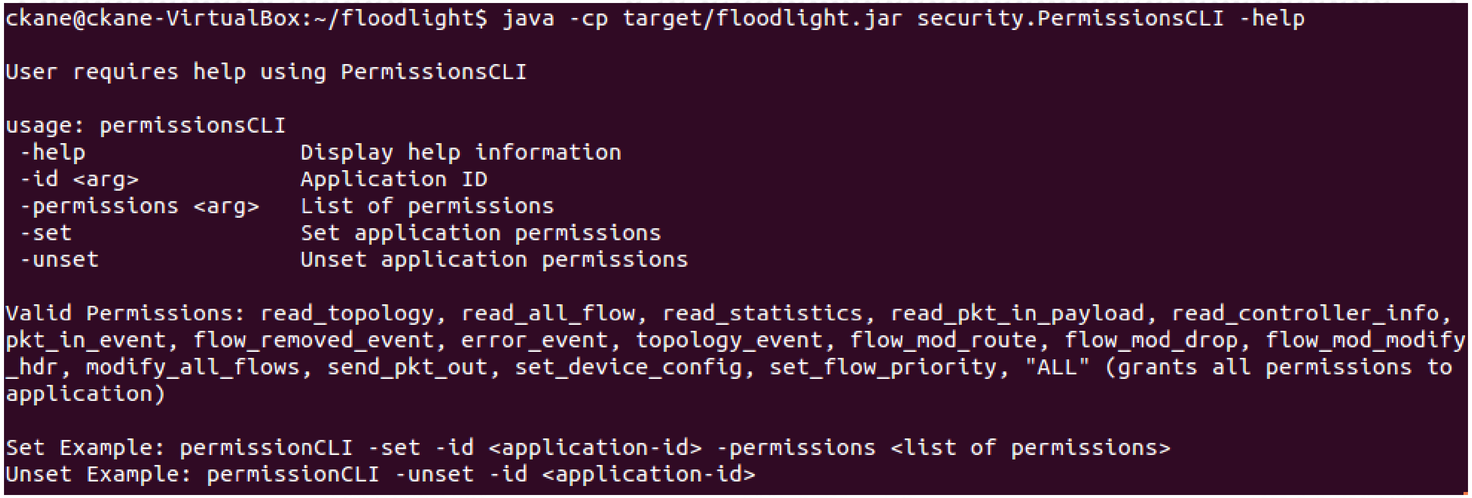
Category	Permission	Screening method(s)
Read	read_topology	**getAllSwitchMap:** Controller.java **getLinks:** LinkDiscoverManager.java
	read_all_flow	**getFlows:** StaticFlowEntryPusher.java
	read_statistics	**getSwitchStatistics:** SwitchResourceBase.java **getCounterValue:** SimpleCounter.java
	read_pkt_in_payload	**get:** FloodlightContextStore.java
	read_controller_info	**retrieve:** ControllerMemoryResource.java
Notification	pkt_in_event	**addToMessageListeners:** Controller.java **addListener:** ListenerDispatcher.java
	flow_removed_event	
	error_event	
Write	flow_mod_route	**insertRow:** AbstractStorageSource.java
	flow_mod_drop	**deleteRow:** AbstractStorageSource.java
	set_flow_priority	**insertRow:** AbstractStorageSource.java
	set_device_config	**setAttribute:** OFSwitchBase.java
	send_pkt_out	**write:** IOFSwitch.java **writeThrottled:** IOFSwitch.java
	flow_mod_modify_hdr	**parseActionsString:** StaticFlowEntries.java
	modify_all_flows	**setCommand:** OFFlowMod.java

Application Permissions

Application Permissions Management:

Unique ID is key to access LinkedHashMap structure storing application permissions (encrypted and serialized)

Application Permissions Interrogation:



```
ckane@ckane-VirtualBox:~/floodlight$ java -cp target/floodlight.jar security.PermissionsCLI -help

User requires help using PermissionsCLI

usage: permissionsCLI
 -help                   Display help information
 -id <arg>               Application ID
 -permissions <arg>      List of permissions
 -set                    Set application permissions
 -unset                  Unset application permissions

Valid Permissions: read_topology, read_all_flow, read_statistics, read_pkt_in_payload, read_controller_info,
pkt_in_event, flow_removed_event, error_event, topology_event, flow_mod_route, flow_mod_drop, flow_mod_modify
_hdr, modify_all_flows, send_pkt_out, set_device_config, set_flow_priority, "ALL" (grants all permissions to
application)

Set Example: permissionCLI -set -id <application-id> -permissions <list of permissions>
Unset Example: permissionCLI -unset -id <application-id>
```

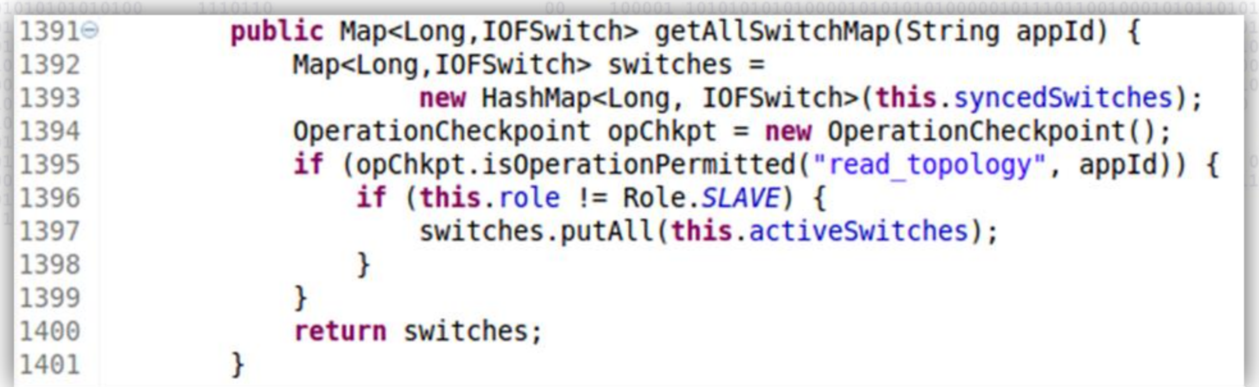
Application Permissions Querying:

REST URI: */wm/security/<id>/permissions/json*

Operation Checkpoint

Operation Checkpoint:

Floodlight Method *getAllSwitchMap* has been modified to incorporate the new security mechanism



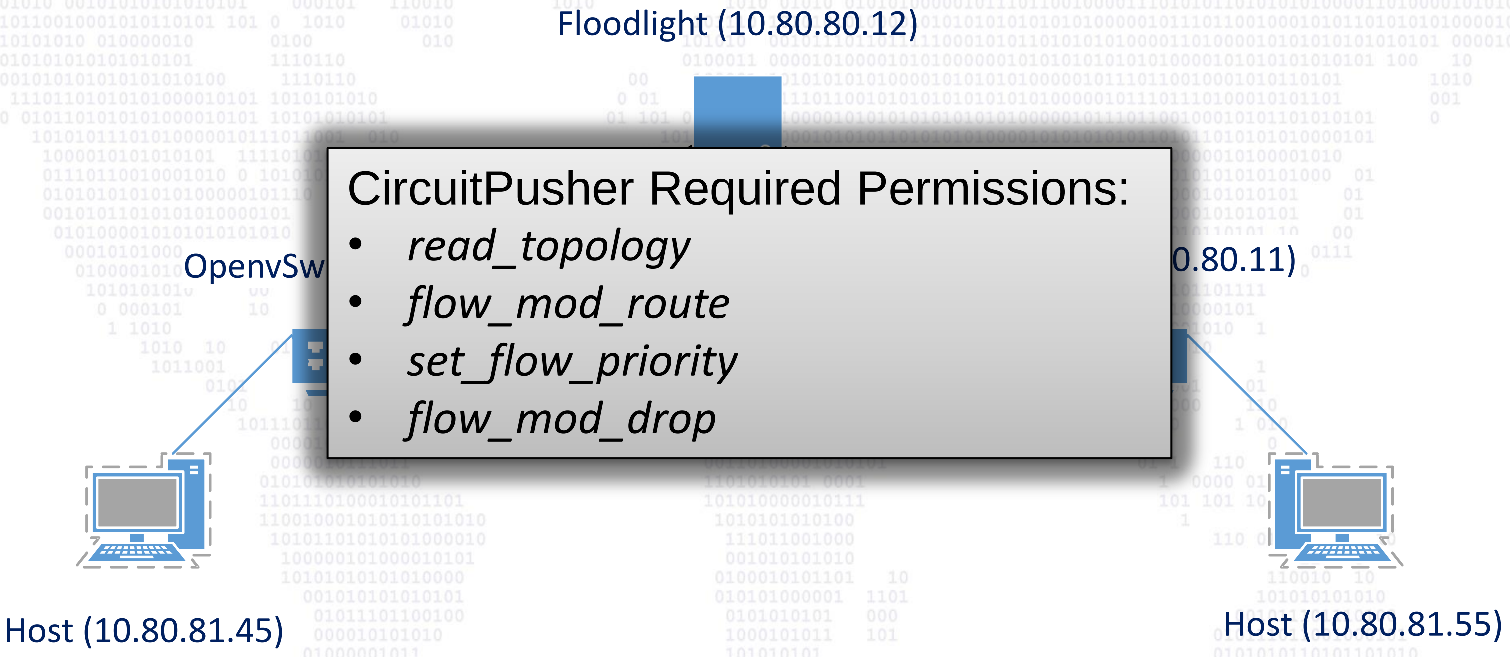
```java
    public Map<Long,IOFSwitch> getAllSwitchMap(String appId) {
        Map<Long,IOFSwitch> switches =
                new HashMap<Long, IOFSwitch>(this.syncedSwitches);
        OperationCheckpoint opChkpt = new OperationCheckpoint();
        if (opChkpt.isOperationPermitted("read_topology", appId)) {
            if (this.role != Role.SLAVE) {
                switches.putAll(this.activeSwitches);
            }
        }
        return switches;
    }
```

Unauthorized Operations Log:

<date><time><applicationID><deniedpermission>

CircuitPusher Example (1/5)

CircuitPusher …*“utilizes Floodlight rest APIs to create a bidirectional circuit, i.e. permanent flow entry, on all switches in route between two devices based on IP addresses with specified priority”*



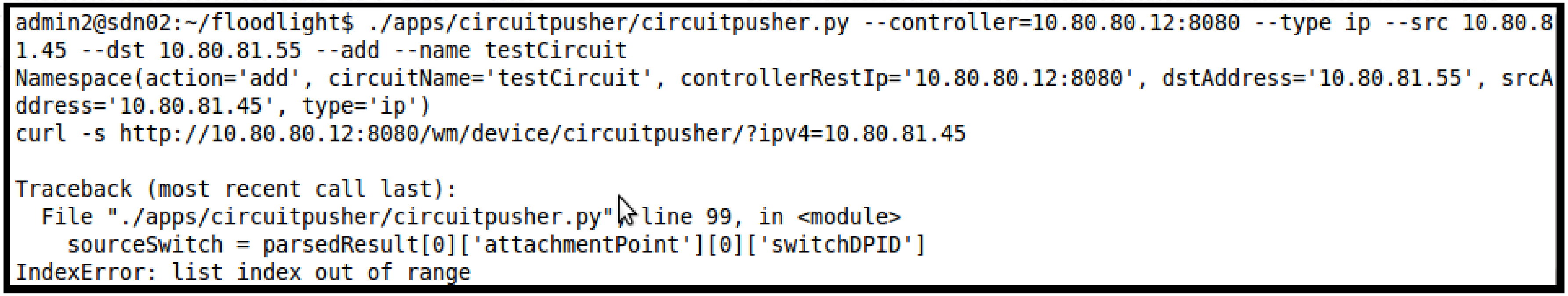
CircuitPusher Required Permissions:

- *read_topology*
- *flow_mod_route*
- *set_flow_priority*
- *flow_mod_drop*

Floodlight Controller, User Documentation -> REST Applications -> Circuit Pusher, 19 Nov 2012, bit.ly/1qZ1Rjk/

CircuitPusher Example (2/5)

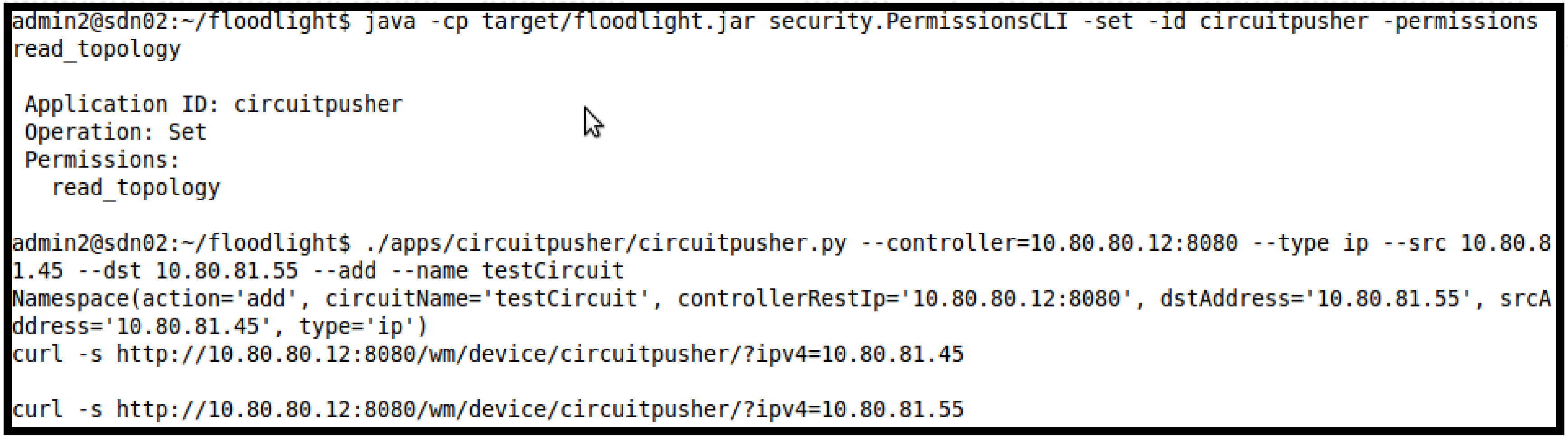
With no permissions granted to *circuitpusher*, the attempt to add a bidirectional circuit fails in an attempt to retrieve switch details:



```
admin2@sdn02:~/floodlight$ ./apps/circuitpusher/circuitpusher.py --controller=10.80.80.12:8080 --type ip --src 10.80.8
1.45 --dst 10.80.81.55 --add --name testCircuit
Namespace(action='add', circuitName='testCircuit', controllerRestIp='10.80.80.12:8080', dstAddress='10.80.81.55', srcA
ddress='10.80.81.45', type='ip')
curl -s http://10.80.80.12:8080/wm/device/circuitpusher/?ipv4=10.80.81.45

Traceback (most recent call last):
  File "./apps/circuitpusher/circuitpusher.py", line 99, in <module>
    sourceSwitch = parsedResult[0]['attachmentPoint'][0]['switchDPID']
IndexError: list index out of range
```

CircuitPusher Example (3/5)

After the *read_topology* permission is added, the initial commands of the application complete successfully:

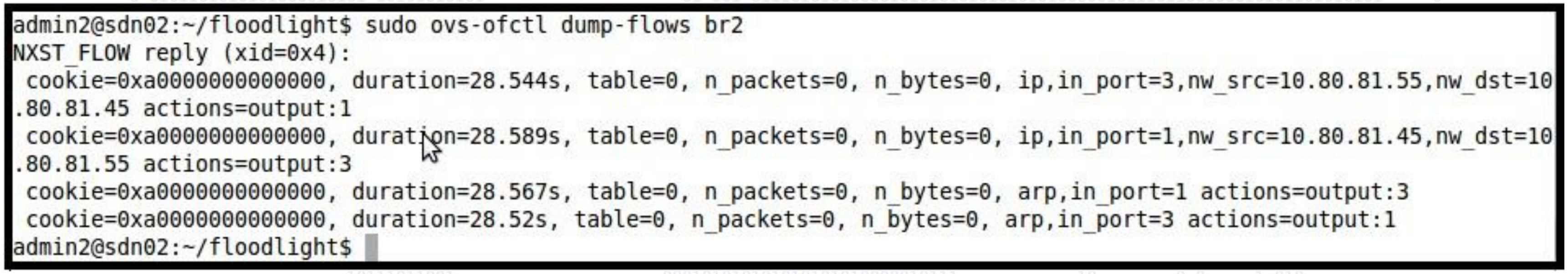


```
admin2@sdn02:~/floodlight$ java -cp target/floodlight.jar security.PermissionsCLI -set -id circuitpusher -permissions
read_topology

 Application ID: circuitpusher
 Operation: Set
 Permissions:
   read_topology

admin2@sdn02:~/floodlight$ ./apps/circuitpusher/circuitpusher.py --controller=10.80.80.12:8080 --type ip --src 10.80.8
1.45 --dst 10.80.81.55 --add --name testCircuit
Namespace(action='add', circuitName='testCircuit', controllerRestIp='10.80.80.12:8080', dstAddress='10.80.81.55', srcA
ddress='10.80.81.45', type='ip')
curl -s http://10.80.80.12:8080/wm/device/circuitpusher/?ipv4=10.80.81.45

curl -s http://10.80.80.12:8080/wm/device/circuitpusher/?ipv4=10.80.81.55
```

However, *ovs-ofctl dump-flows <dpid>* shows switch flow table empty

CircuitPusher Example (4/5)

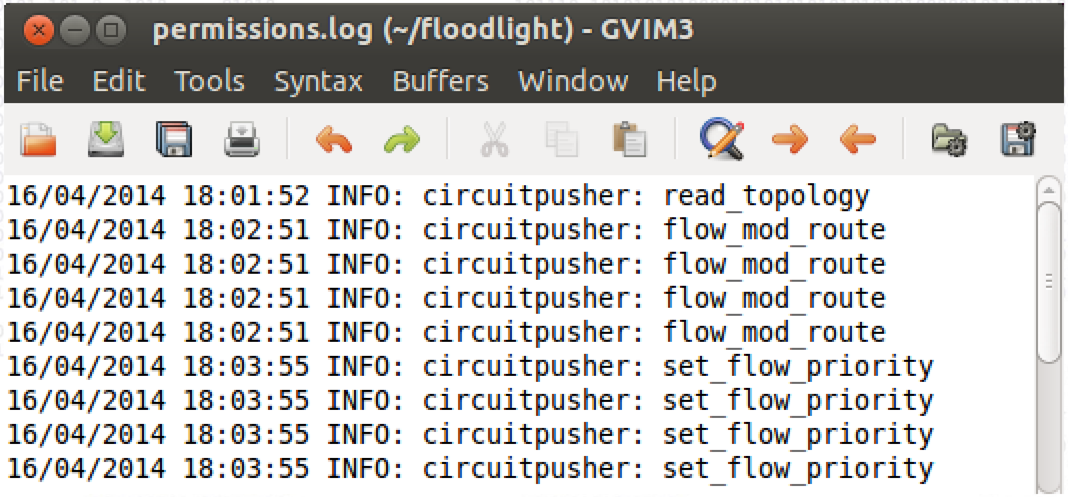
Once the remaining permissions are added (*flow_mod_route* and *set_flow_priority*), the circuit is installed correctly with flow rules installed at the switches:



```
admin2@sdn02:~/floodlight$ sudo ovs-ofctl dump-flows br2
NXST_FLOW reply (xid=0x4):
 cookie=0xa0000000000000, duration=28.544s, table=0, n_packets=0, n_bytes=0, ip,in_port=3,nw_src=10.80.81.55,nw_dst=10
.80.81.45 actions=output:1
 cookie=0xa0000000000000, duration=28.589s, table=0, n_packets=0, n_bytes=0, ip,in_port=1,nw_src=10.80.81.45,nw_dst=10
.80.81.55 actions=output:3
 cookie=0xa0000000000000, duration=28.567s, table=0, n_packets=0, n_bytes=0, arp,in_port=1 actions=output:3
 cookie=0xa0000000000000, duration=28.52s, table=0, n_packets=0, n_bytes=0, arp,in_port=3 actions=output:1
admin2@sdn02:~/floodlight$ 
```

CircuitPusher Example (5/5)

The log file holds the record of the unauthorized *circuitpusher* access attempts:



permissions.log (~/floodlight) - GVIM3
File Edit Tools Syntax Buffers Window Help

```
16/04/2014 18:01:52 INFO: circuitpusher: read_topology
16/04/2014 18:02:51 INFO: circuitpusher: flow_mod_route
16/04/2014 18:02:51 INFO: circuitpusher: flow_mod_route
16/04/2014 18:02:51 INFO: circuitpusher: flow_mod_route
16/04/2014 18:02:51 INFO: circuitpusher: flow_mod_route
16/04/2014 18:03:55 INFO: circuitpusher: set_flow_priority
16/04/2014 18:03:55 INFO: circuitpusher: set_flow_priority
16/04/2014 18:03:55 INFO: circuitpusher: set_flow_priority
16/04/2014 18:03:55 INFO: circuitpusher: set_flow_priority
```

Performance

OperationCheckpoint introduces limited latency to the Floodlight Controller:

	Avg.	Std. Dev.
Execution Time *(μs)* without *OperationCheckpoint*	5.625	2.955
Execution Time *(μs)* with *OperationCheckpoint*	372.750	103.191
Latency *(μs)*	367.125	102.437

Related Work

PermOF

X. Wen, Y. Chen, C. Hu, C. Shi, and Y. Wang, “Towards a secure controller platform for openflow applications,” in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in SDN*. ACM, 2013, pp. 171–172.

FortNOX

P. Porras, S. Shin, V. Yegneswaran, M. Fong, M. Tyson, and G. Gu, “A security enforcement kernel for OpenFlow networks,” in *Proceedings of the 1st workshop on Hot topics in SDN*. ACM, 2012, pp. 121–126.

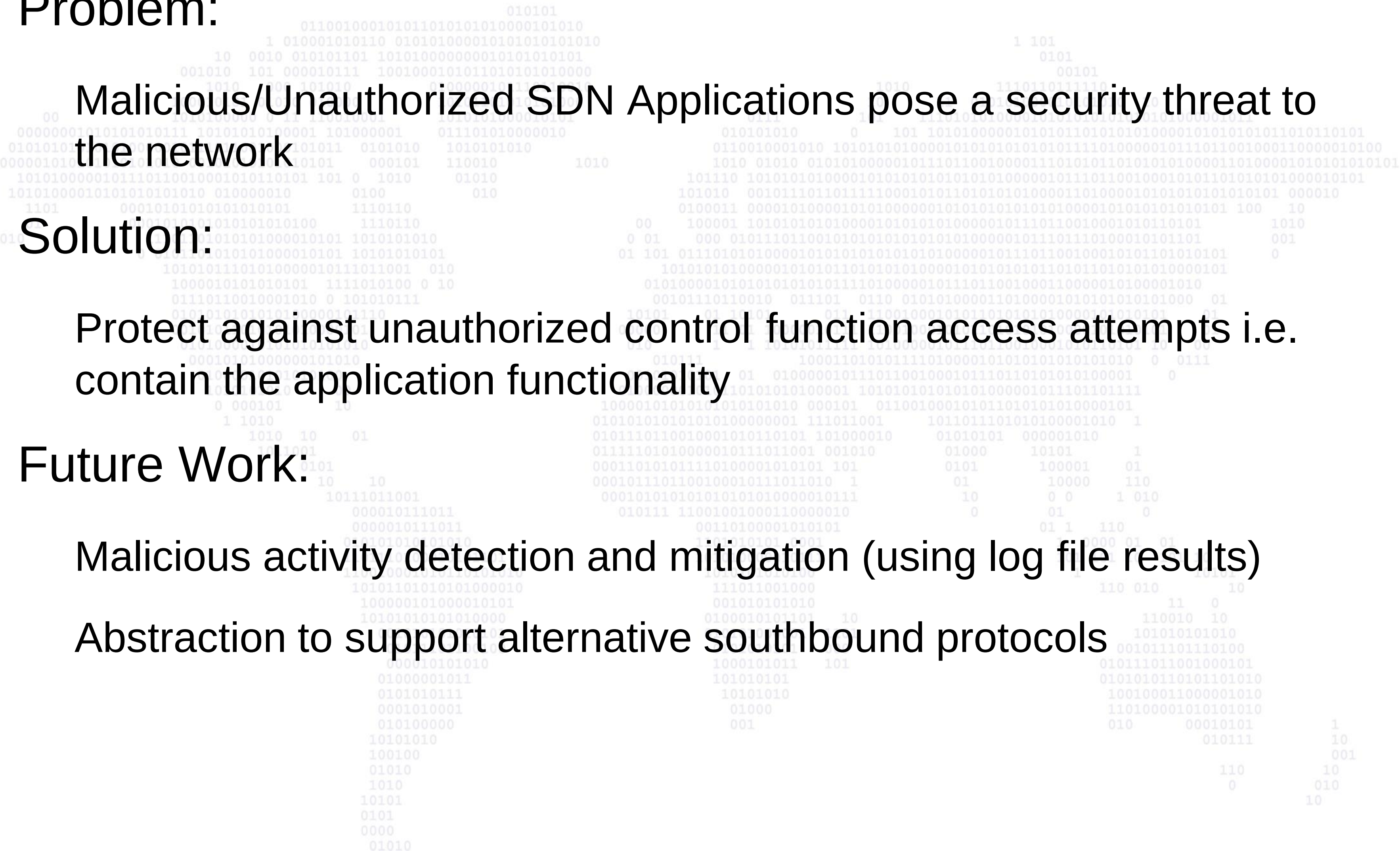
SE-Floodlight

“Security-Enhanced Floodlight.” [Online]. Available: 1drv.ms/1k2WDTC

ROSEMARY

S. Shin, Y. Song, T. Lee, S. Lee, J. Chung, P. Porras, V. Yegneswaran, J. Noh, and B. B. Kang, “Rosemary: A Robust, Secure, and High-Performance Network Operating System,” in *20th ACM Conference on Computer and Communications Security, To be Published*, November 2014

Conclusion



Problem:

Malicious/Unauthorized SDN Applications pose a security threat to the network

Solution:

Protect against unauthorized control function access attempts i.e. contain the application functionality

Future Work:

Malicious activity detection and mitigation (using log file results)

Abstraction to support alternative southbound protocols

Thank you!

Queen's University Belfast

CSIT CENTRE FOR SECURE INFORMATION TECHNOLOGIES

Questions?

s.scott-hayward@qub.ac.uk

CSIT: A Global Cyber Innovation Hub

CSIT
CENTRE FOR SECURE INFORMATION TECHNOLOGIES

Thought leader in Secure Information Technology Research
Network of Commercial & Research partnerships
Portfolio of successful Technology Transfer