Operation Checkpoint: SDN Application Control

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
Software Defined Networking … and Security

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

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 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

<table>
<thead>
<tr>
<th>Category</th>
<th>Permission</th>
<th>Screening method(s)</th>
</tr>
</thead>
</table>
| **Read**     | read_topology            | getAllSwitchMap: Controller.java  
gerLinks: 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                                           |
|              | flow_removed_event       | addListener: ListenerDispatcher.java                                              |
|              | 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 Management:

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

Application Permissions Interrogation:

Application Permissions Querying:

REST URI: /wm/security/<id>/permissions/json
Operation Checkpoint:

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

Unauthorized Operations Log:

`<date><time><applicationID><deniedpermission>`
**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`
With no permissions granted to `circuitpusher`, the attempt to add a bidirectional circuit fails in an attempt to retrieve switch details:

```bash
admin2@sdn02:/floodlight$ ./apps/circuitpusher/circuitpusher.py --controller=10.80.80.12:8080 --type ip --src 10.80.81.45 --dst 10.80.81.55 --add --name testCircuit Namespace(action='add', circuitName='testCircuit', controllerRestIp='10.80.80.12:8080', dstAddress='10.80.81.55', srcAddress='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
After the `read_topology` permission is added, the initial commands of the application complete successfully:

```bash
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 10.80.81.45 --dst 10.80.81.55 --add --name testCircuit
Namespace(action='add', circuitName='testCircuit', controllerRestIp='10.80.80.12:8080', dstAddress='10.80.81.55', srcAddress='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
Once the remaining permissions are added (\texttt{flow\_mod\_route} and \texttt{set\_flow\_priority}), the circuit is installed correctly with flow rules installed at the switches:

```
admin2@sdn02:~$ 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:2
  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:~$ ```
The log file holds the record of the unauthorized circuitpusher access attempts:
\textit{OperationCheckpoint} introduces limited latency to the Floodlight Controller:

<table>
<thead>
<tr>
<th></th>
<th>Avg.</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution Time ((\mu)s) without \textit{OperationCheckpoint}</td>
<td>5.625</td>
<td>2.955</td>
</tr>
<tr>
<td>Execution Time ((\mu)s) with \textit{OperationCheckpoint}</td>
<td>372.750</td>
<td>103.191</td>
</tr>
<tr>
<td>Latency ((\mu)s)</td>
<td>367.125</td>
<td>102.437</td>
</tr>
</tbody>
</table>
Related Work


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!

Questions?

s.scott-hayward@qub.ac.uk
Thought leader in Secure Information Technology Research
Network of Commercial & Research partnerships
Portfolio of successful Technology Transfer