AEGIS : An Automated Permission Generation and Verification System for SDNs

ACM SIGCOMM 2018 Workshop on SecSoN

Heedo Kang, Seungwon Shin, Vinod Yegneswaran*, Shalini Ghosh*, Phillip Porras*

KAIST, SRI International*
Contents

1. Background

2. Motivation & Challenge

3. AEGIS Design
   • Static Engine
   • Dynamic Engine

4. Evaluation

5. Conclusion
Background

Software Defined Networking (SDN)?

- Network decoupling
  - Network control and forwarding functions

- Programmable network
  - Flexible and dynamic network control
  - Innovative network service

- Potential abuse
  - SDN controller API can be abused by SDN app
  - Entire resources can be manipulated
Abusing SDN controller API

Background

Existing SDN permission systems

• SE-Floodlight
  • Role based access control (for only Data-Plane related resources)

• SDNShield
  • Permission & policy based access control (for only Data-Plane related resources)

• Security-Mode ONOS
  • Permission based access control (for all resources)
1. Background

2. Motivation & Challenge

3. AEGIS Design
   • Static Engine
   • Dynamic Engine

4. Evaluation

5. Conclusion
Motivation

1. Automation deficiency

- To build SDN permission system..
  
  (i) Analyze what resources (assets) should be protected
  
  (ii) Inspect what resources are accessed by each APIs
  
  (iii) Design permission model
  
  (iv) Implement permission system

Example of human error existed in Security-Mode ONOS
Motivation

2. Portability deficiency

- Procedure for building SDN permission system
  - Too complicated task
  - Error prone

- Existing SDN permission systems
  - Tightly coupled with SDN controller implementation
    - e.g. SE-Floodlight (Floodlight), Security-Mode ONOS (ONOS)
  - Cannot be ported to any other controller
Motivation

3. Flexibility deficiency

- Different security requirements

  Bob (Network operator)
  Our network needs fine-grained access control over only topology resource.

  Alice (Network operator)
  Our network needs fine-grained access control over all resources

- Existing SDN permission systems
  - Permission model is fixed
Challenges

• **Ultimate goal**
  - Suggest new automated permission generation and verification system for SDN

• **Summary of challenges**
  - **Automation**
    - Automatically generate permission model for SDN controller
  - **Portability**
    - Independently designed and implemented from specific SDN controller implementation
  - **Flexibility**
    - Provide way to flexibly generate permission model
Contents

1. Background

2. Motivation & Challenge

3. AEGIS Design
   • Static Engine
   • Dynamic Engine

4. Evaluation

5. Conclusion
AEGIS Design

- Overview

- **Static Engine** (execute before run-time)
  - Automatically generates permission model
  - Various NLP techniques

- **Dynamic Engine** (execute on run-time)
  - Verifies if application has right permissions to execute API
  - Hooking & Code injection technique
AEGIS Design

• **Static Engine**

  • Consists of seven modules
    - API Document Parser
    - Preprocessor
    - Semantic Role Labeler
    - Intermediate processor
    - Dependency Analyzer
    - SDN Asset Map Generator
    - API-Permission Mapping Constructor

  • Takes controller API document & permission model policy as inputs

  • Generates permission model as output
AEGIS Design

- API document Parser

  - Extract following features from API document
    - Package path
    - Class name
    - API name
    - API description

API document = org.onosproject.net.flow.FlowRuleService.getFlowRuleCount

Description = Returns the number of flow rules in the system.

ONOS controller API document

SDN controller API document
AEGIS Design

- Preprocessor
  - Replace all uppercase letters with lowercase letters
  - Remove special characters
  - Inject fake subject
  - Converge entity n-grams into one word
  - Change verb into three types of action word
    - e.g)
      - obtain, fetch, get, find, check ...... -> read
      - Send, create, remove, add, unregister ......-> write
      - Invoke, activate, stop, perform ......-> execute

Returns the number of flow rules in the system.

It read the number of flow_rule in the system.
AEGIS Design

● Semantic Role Labeler

- Classifies description into semantic constituents
  - Object contains resources that API access

- Investigates classified object
  - Starts with to-infinitive or gerund?

  ➢ Re-classifies object sentence

  eg.) It attempts to assign leadership for a topic to a specified node

  (S It) (V attempts) (O to assign leadership for a topic to a specified node)

  Re-classify

  (S It) (V assign) (O leadership for a topic to a specified node)
AEGIS Design

• Intermediate processor

• Tags Part of speeches (POS)
  - e.g. ) ~ (NN/ flow_rule) (IN/ in) ~

• Removes determiner words
  - e.g. ) the number of ~

• Converts word to stem of the word
  - e.g. ) ~ devices

the number of flow_rule in the system

Intermediate processor

(NN/ number) (IN/ of) (NN/ flow_rule) (IN/ in) (NN/ system)
AEGIS Design

• Dependency Analyzer

• Analyzes relationships between each word
  - Dependency parsing
    - root (Root-0, number-1)
    - case(flow_rule-3, of-2)
    - nmod:of(number-1, flow_rule-3)
    - case(system-5, in-4)
    - nmod:in(number-1, system-5)
  - Extract set of nominal modifier(nmod) relation
  - Generates asset-linked list
    - Based on predefined rules
    - Tag API path & action

Example:

(read)

```
(read)
```

Asset linked-list

```
(system)
```

```
(flow_rule)
```

```
(number)
```

```
(read ,
org.onosproject.net.flow.FlowRuleService.
getFlowRuleCount)
```

```
(Dependency
Analyzer)
```

```
(root (Root-0, number-1)
  case(flow_rule-3, of-2)
  nmod:of(number-1, flow_rule-3)
  case(system-5, in-4)
  nmod:in(number-1, system-5))
```
AEGIS Design

- **SDN Asset Map Generator**
  - Integrates all asset-linked list
  - Flexible permission model generation
    - Pruning map based on permission model policy
      - e.g.) Remove STATSTIC node and move tags to PORT node

- **API-permission Mapping Constructor**
  - Creates permission type
    - By concatenating node name from each starting node to root node and action word
  - Maps each generated permission type to API path
### AEGIS Design

- **Permission model**

<table>
<thead>
<tr>
<th>AEGIS permission token</th>
<th>Accessible API</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEVICE_WRITE</td>
<td>DeviceAdminService.removeDevice(DeviceId)</td>
</tr>
<tr>
<td>DEVICE_PORT_WRITE</td>
<td>DeviceAdminService.changePortState(DeviceId, PortNumber, boolean)</td>
</tr>
<tr>
<td>DEVICE_READ</td>
<td>DeviceService.isAvailable(DeviceId)</td>
</tr>
<tr>
<td></td>
<td>DeviceService.getDevices(Device.Type)</td>
</tr>
<tr>
<td></td>
<td>DeviceService.getDevices()</td>
</tr>
<tr>
<td></td>
<td>DeviceService.getDevice(DeviceId)</td>
</tr>
<tr>
<td>DEVICE_PORT_READ</td>
<td>DeviceService.getPort(DeviceId, PortNumber)</td>
</tr>
<tr>
<td></td>
<td>DeviceService.getPorts(DeviceId)</td>
</tr>
<tr>
<td>DEVICE_PORT_STATISTIC_READ</td>
<td>DeviceService.getPortDeltaStatistics(DeviceId)</td>
</tr>
<tr>
<td></td>
<td>DeviceService.getPortDeltaStatisticsForPort(DeviceId, PortNumber)</td>
</tr>
<tr>
<td></td>
<td>DeviceService.getPortStatistics(DeviceId)</td>
</tr>
<tr>
<td></td>
<td>DeviceService.getStatisticsForPort(DeviceId, PortNumber)</td>
</tr>
<tr>
<td>DEVICE_TIMESTAMP_READ</td>
<td>DeviceClockService.getTimestamp(DeviceId)</td>
</tr>
<tr>
<td></td>
<td>DeviceClockService.isTimestampAvailable(DeviceId)</td>
</tr>
<tr>
<td>DEVICE_COUNT_READ</td>
<td>DeviceService.getCount()</td>
</tr>
<tr>
<td>DEVICE_MASTERSHIP_ROLE_READ</td>
<td>DeviceService.getRole(DeviceId)</td>
</tr>
</tbody>
</table>

Example of ONOS API – permission mappings
AEGIS Design

- **Dynamic Engine**
  - Consists of four modules
    - API Hooker
    - Permission Enforcer
    - Permission Checker
    - Injector
  - Takes permission model and invoked API information as inputs
  - Generates and injects security exception code as output
AEGIS Design

- **API Hooker**
  - Sniffs all of Northbound-API calls
    - By using hooking technique

- **Permission Enforcer**
  - Enforces permission reviewing process
  - Grant(store) declared permissions

- **Permission checker**
  - Checks if application has right permissions
  - Makes decision

- **Injector**
  - Injects code that generates security exception
    - By using code injection technique
1. Background

2. Motivation & Challenge

3. AEGIS Design
   • Static Engine
   • Dynamic Engine

4. Evaluation

5. Conclusion
Evaluation

● Completeness

• How many SDN API descriptions can be covered by AEGIS?

<table>
<thead>
<tr>
<th>Controller</th>
<th># of total APIs</th>
<th># of covered APIs</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>ONOS</td>
<td>355</td>
<td>348</td>
<td>98%</td>
</tr>
<tr>
<td>Floodlight</td>
<td>198</td>
<td>186</td>
<td>94%</td>
</tr>
<tr>
<td>POX</td>
<td>14</td>
<td>14</td>
<td>100%</td>
</tr>
<tr>
<td>Total</td>
<td>567</td>
<td>548</td>
<td>96.6%</td>
</tr>
</tbody>
</table>

• Failure case examples
  • A builder for the creation of local persistent maps backed by disk
  • Removes all links between between the specified src and dst connection point
## Evaluation

- **Soundness**

- How much accurately does AEGIS extract?
  - No ground truth
  - Survey of 20 SDN experts
  - Randomly select 30 Northbound-API descriptions from ONOS, Floodlight and POX controller

<table>
<thead>
<tr>
<th>Question</th>
<th># of positive responses</th>
<th># of negative responses</th>
<th>Correctness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action word &amp; resources</td>
<td>583</td>
<td>17</td>
<td>97.2%</td>
</tr>
<tr>
<td>Relation</td>
<td>574</td>
<td>23</td>
<td>95.7%</td>
</tr>
</tbody>
</table>
Evaluation

- Use case

- Can AEGIS invalidate attack scenario that is valid on existing permission system?

**App description**: This is a DDOS prevention application that detects DDOS attack and disable a network port relaying attack traffic.

**Security-Mode ONOS**

- Grant
- DEVICE_WRITE permission

**Accessible API list**
- ChangePortState()
- RemoveDevices()
- ...

**ONOS Controller**

Before
1. DefaultDevice{id=0000000000000001, type=SWITCH, manufacturer=Nicira, Inc, hwVersion=Open vSwitch, swVersion=2.0.2, serialNumber=None, driver-ovs}
2. DefaultDevice{id=0000000000000002, type=SWITCH, manufacturer=Nicira, Inc, hwVersion=Open vSwitch, swVersion=2.0.2, serialNumber=None, driver-ovs}

After

Remove all device information

Bob (Network operator)

Download

SDN Application
# Evaluation

- Use case

<table>
<thead>
<tr>
<th>Security-Mode ONOS permission token</th>
<th>AEGIS permission token</th>
<th>Accessible API</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEVICE_WRITE</td>
<td>DEVICE_WRITE</td>
<td>DeviceAdminService.removeDevice(DeviceId)</td>
</tr>
<tr>
<td>DEVICE_READ</td>
<td>DEVICE_PORT_WRITE</td>
<td>DeviceAdminService.changePortState(DeviceId, PortNumber, boolean)</td>
</tr>
</tbody>
</table>

Device: DEVICE

- DEVICE_READ
  - DEVICE_PORT_READ: DeviceService.getPort(ConnectPoint)
  - DEVICE_PORT_STATISTIC_READ: DeviceService.getPortDeltaStatistics(DeviceId)
  - DEVICE_TIMESTAMP_READ: DeviceClockService.getTimestamp(DeviceId)
  - DEVICE_COUNT_READ: DeviceService.getCount()
  - DEVICE_MASTERSHIP_ROLE_READ: DeviceService.getRole(DeviceId)

Accessible DEVICE resource related APIs of ONOS with each permission token in Security-Mode ONOS and AEGIS

Attack scenario result with AEGIS

```
2017-04-22 00:19:03,568 | ERROR | 1 for user karaf | onos-app-attack |
| 180 - org.onosproject.onos-app-attack - 1.5.0 | [org.onosproject.attack.AppComponent(128)] The activate method has thrown an exception java.lang.SecurityException
at org.onosproject.net.device.impl.DeviceManager.removeDevice(DeviceManager.java:246)
at org.onosproject.attack.AttackNode.attack(AttackComponent.java:76)
at org.onosproject.attack.AttackComponent.activate(AttackComponent.java:58)
```
Conclusion

- Address some deficiencies of existing SDN permission system

- Propose AEGIS
  - Automatically and flexibly generates SDN permission model
  - Verifies permissions of SDN app in separated process from SDN controller

- Implement prototype

- Evaluate its completeness and soundness & demonstrate its usecase
Q & A