A Formal Access Control Model for SE-Floodlight Controller

Abdullah Al-Alaj¹, Ravi Sandhu¹ and Ram Krishnan²

¹Dept. of Computer Science
²Dept. of Electrical and Computer Engineering
¹,²Institute for Cyber Security
¹,²Center for Security and Privacy Enhanced Cloud Computing (C-SPECC)
University of Texas at San Antonio, TX 78249

SDN-NFV Security 2019
Dallas, Texas, USA, March 27, 2019

World-Leading Research with Real-World Impact!
Introduction

• Software Defined Networks (SDN)
• Floodlight
• SE-Floodlight
• SDN Enabler.
Application Authorization in SDN

REST Calls

SDN Controller

OpenFlow

Switch

Data plane

World-Leading Research with Real-World Impact!
Components of the Formal Access Control Model

- Basic components
  - Apps (A),
  - Roles (R),
  - Data Exchange Operations (DXOP),
  - Types of DXOPs
• Two types:
  • Local OpenFlow apps
  • Remote OpenFlow apps
Roles in SE-Floodlight

• Two main purposes:
  • App permission authorization
  • Flow rule conflict resolution.
Data Exchange Operations (DXOP) and Types of DXOPs

Data Exchange Operations (DXOP)  

flow rule  
- add  
- modify  
- modify_strict  
- delete  
- delete_strict

queries for statistics  
- flows  
- ports  
- Tables  
- Queues  
- etc

Operation types (T)  

- ‘Flow rule mod’  
  - OFPT_FLOW_MOD

- ‘Switch stats request’  
  - OFPT_STATS_REQUEST

World-Leading Research with Real-World Impact!
## Types of Data Exchange Operations

<table>
<thead>
<tr>
<th>Type ID</th>
<th>Type of Data Exchange Operation</th>
<th>Minimum Authorization Role</th>
<th>Open Flow Message Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>Flow removal messages</td>
<td>APP</td>
<td>OFPT_FLOW_REMOVED</td>
</tr>
<tr>
<td>t2</td>
<td>Flow error reply</td>
<td>APP</td>
<td>OFPT_ERROR</td>
</tr>
<tr>
<td>t3</td>
<td>Echo requests</td>
<td>APP</td>
<td>OFPT_ECHO_REQUEST</td>
</tr>
<tr>
<td>t4</td>
<td>Echo replies</td>
<td>APP</td>
<td>OFPT_ECHO_REPLY</td>
</tr>
<tr>
<td>t5</td>
<td>Barrier requests</td>
<td>APP</td>
<td>OFPT_BARRIER_REQUEST</td>
</tr>
<tr>
<td>t6</td>
<td>Barrier replies</td>
<td>APP</td>
<td>OFPT_BARRIER_REPLY</td>
</tr>
<tr>
<td>t7</td>
<td>Switch get config</td>
<td>APP</td>
<td>OFPT_GET_CONFIG_REQUEST</td>
</tr>
<tr>
<td>t8</td>
<td>Switch config reply</td>
<td>APP</td>
<td>OFPT_GET_CONFIG_REPLY</td>
</tr>
<tr>
<td>t9</td>
<td>Switch stats request</td>
<td>APP</td>
<td>OFPT_STATS_REQUEST</td>
</tr>
<tr>
<td>t10</td>
<td>Switch stats report</td>
<td>APP</td>
<td>OFPT_STATS_REPLY</td>
</tr>
<tr>
<td>t11</td>
<td>Packet-In return</td>
<td>APP</td>
<td>OFPT_PACKET_IN</td>
</tr>
<tr>
<td>t12</td>
<td>Flow rule mod</td>
<td>APP</td>
<td>OFPT_FLOW_MOD</td>
</tr>
<tr>
<td>t13</td>
<td>Packet-Out</td>
<td>SEC</td>
<td>OFPT_PACKET_OUT</td>
</tr>
<tr>
<td>t14</td>
<td>Vendor actions</td>
<td>ADMIN</td>
<td>OFPT_VENDOR</td>
</tr>
<tr>
<td>t15</td>
<td>Vendor features</td>
<td>ADMIN</td>
<td>OFPT_FEATURES</td>
</tr>
<tr>
<td>t16</td>
<td>Switch port status</td>
<td>ADMIN</td>
<td>OFPT_PORT_STATUS</td>
</tr>
<tr>
<td>t17</td>
<td>Switch port mod</td>
<td>ADMIN</td>
<td>OFPT_PORT_MOD</td>
</tr>
<tr>
<td>t18</td>
<td>Switch set config</td>
<td>ADMIN</td>
<td>OFPT_SET_CONFIG</td>
</tr>
</tbody>
</table>

Porras, P. A et al. 2015. Securing the Software Defined Network Control Layer. In NDSS.  

*World-Leading Research with Real-World Impact!*
Authentication & Authorization
Conceptual Authorization Model

Applications (A) <-> Roles (R) <-> Operation Types (T)

- AR (Applications to Roles)
- TR (Roles to Operation Types)
- one to many

N (Priority limit)

ADMIN, SEC, APP (Roles)

Data Exchange Operations (DXOP)

OT (Operation Types)
- Basic Sets and Functions:
  A: a finite set of OpenFlow apps.
  T: a finite set of types of data exchange operations.
  R = {ADMIN, SEC, APP}: a fixed set of three roles.
  >: a total order on R where ADMIN > SEC and SEC > APP.
  AR ⊆ A × R, a many-to-one relation, i.e., (a, r₁) ∈ AR ∧ (a, r₂) ∈ AR ⇒ r₁ = r₂, mapping each app to one role.
  TR ⊆ T × R, a many-to-one relation, i.e., (t, r₁) ∈ TR ∧ (t, r₂) ∈ TR ⇒ r₁ = r₂, mapping each operation type to one role.
  DXOP: a set of possible data exchange operations where each operation o ∈ DXOP contains a flow rule and a priority if o = 'add flow rule'.
  type: DXOP → T, a function specifying the type of each operation. Equivalently viewed as a many-to-one relation OT ⊆ DXOP × T, where (o, t₁) ∈ OT ∧ (o, t₂) ∈ OT ⇒ t₁ = t₂.

- Authorization Rule:
  Authorization_rule: A × DXOP → {T, F}, checks whether a ∈ A has the right to perform an operation o ∈ DXOP.
  Authorization_rule (a : A, o : DXOP) ≜ (∃ r₁, r₂ ∈ R: (a, r₁) ∈ AR ∧ (type(o), r₂) ∈ TR ∧ r₁ ≥ r₂).

Formal Authorization Model Definitions without Flow Rule Conflict Resolution.
Formal Model Definitions with Flow Rule Conflict Resolution.

- Basic Sets and Functions:
  All basic sets and functions from Table 2.
  \( FR \): a set of all possible flow rules where for each \( fr_i \in FR \) there should be a priority.
  \( priority\_limit: R \rightarrow \mathbb{N} \), the mapping of role to the highest priority an app in \( r \in R \) may assign to its flow rules, where \( priority\_limit(ADMIN) > priority\_limit(SEC) > priority\_limit(APP) \).
  \( S \): Set of switches in the network slice.
  \( FT: S \rightarrow 2^{FR} \), the set of flow rules currently in a switch’s flow table.
  \( rule: DXOP \rightarrow FR \), a function that returns the flow rule \( fr_c \in FR \) of an operation \( op \in DXOP \) given that \( type(op) = 'Flow Rule Mod' \).
  \( priority: FR \rightarrow \mathbb{N} \), the mapping of a flow rule \( fr_c \in FR \) to its priority.

\( RCA(fr_c: FR, pr_c: \mathbb{N}, s_t: S) \rightarrow \{Reject, Add, Exchange\} \), a function uses rule-based conflict analysis described in [16] that returns the result of a request to add of new flow rule \( fr_c \) into \( FT(s_t) \) submitted with priority \( pr_c \). ‘Reject’, ‘Add’, or ‘Exchange’ indicates whether \( fr_c \) is rejected, added without removing pre-existing rules, or exchanged with a conflicting flow rule \( fr_i \in FT(s_t) \), respectively.

- Authorization Rules:
  \( Authorization\_rule_{op=\text{add flow rule}}: A \times S \rightarrow \{T, F\} \), checks whether \( a \in A \) has the right to insert a flow rule \( rule(op) \) into \( FT(s_t) \in S \).
  \( Authorization\_rule_{op=\text{add flow rule}} (a: A, s_t: S) \equiv (\exists r_1, r_2 \in R \cdot (a, r_1) \in AR \land (type(op), r_2) \in TR \land r_1 \geq r_2) \land (RCA(rule(op), priority(rule(op))), s_t) \in \{Add, Exchange\} \).
  \( Authorization\_rule_{op \in DXOP=\text{add flow rule}}: A \times S \rightarrow \{T, F\} \), checks whether \( a \in A \) has the right to perform a non-flow-rule-insertion operation.
  \( Authorization\_rule_{op \in DXOP=\text{add flow rule}} (a: A, s_t: S) \equiv (\exists r_1, r_2 \in R \cdot (a, r_1) \in AR \land (type(op), r_2) \in TR \land r_1 \geq r_2) \).
RCA: Add case

- New flow rule
e.g., Priority = ?

- No Conflict

- Existing flow rule
e.g., Priority = ?

Add
RCA : Reject case

Example

New flow rule
e.g., Priority = 50

Reject

Conflict

Existing flow rule
e.g., Priority = 100
RCA : Exchange case

New flow rule
  e.g., Priority = 100

Conflict

Existing flow rule
  e.g., Priority = 50

Exchange
## Administrative Model

<table>
<thead>
<tr>
<th>Function</th>
<th>Condition</th>
<th>Update</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addApp(a)</code></td>
<td>( a \notin A )</td>
<td>( A' = A \cup {a} )</td>
</tr>
<tr>
<td><code>deleteApp(a)</code></td>
<td>( a \in A \land (a,r) \in AR )</td>
<td>( AR' = AR \setminus {(a,r)} ), ( A' = A \setminus {a} )</td>
</tr>
<tr>
<td><code>addType(t)</code></td>
<td>( t \notin T )</td>
<td>( T' = T \cup {t} )</td>
</tr>
<tr>
<td><code>deleteType(t)</code></td>
<td>( t \in T \land (o,t) \in OT \land (t,r) \in TR )</td>
<td>( OT' = OT \setminus {\forall (o,t) \in OT} ), ( TR' = TR \setminus {(t,r)}, T' = T \setminus {t} )</td>
</tr>
<tr>
<td><code>addRole(r)</code></td>
<td>( r \notin R )</td>
<td>( R' = R \cup {r} )</td>
</tr>
<tr>
<td><code>deleteRole(r)</code></td>
<td>( r \in R \land (a,r) \in AR \land (t,r) \in TR )</td>
<td>( AR' = AR \setminus {(a,r)} ), ( TR' = TR \setminus {(t,r)}, R' = R \setminus {r} )</td>
</tr>
<tr>
<td><code>assignApp(a,r)</code></td>
<td>( a \in A \land r \in R \land (a,r) \notin AR )</td>
<td>( AR' = AR \cup {(r,a)} )</td>
</tr>
<tr>
<td><code>revokeApp(a,r)</code></td>
<td>( a \in A \land r \in R \land (a,r) \in AR )</td>
<td>( AR' = AR \setminus {(a,r)} )</td>
</tr>
<tr>
<td><code>assignType(t,r)</code></td>
<td>( t \in T \land r \in R \land (t,r) \notin TR )</td>
<td>( TR' = TR \cup {(t,r)} )</td>
</tr>
<tr>
<td><code>revokeType(t,r)</code></td>
<td>( t \in T \land r \in R \land (t,r) \in TR )</td>
<td>( TR' = TR \setminus {(t,r)} )</td>
</tr>
<tr>
<td><code>assignOp(o,t)</code></td>
<td>( o \in DXOP \land t \in T \land (o,t) \notin OT )</td>
<td>( OT' = OT \cup {(o,t)} )</td>
</tr>
<tr>
<td><code>revokeOp(o,t)</code></td>
<td>( o \in DXOP \land t \in T \land (o,t) \in OT )</td>
<td>( OT' = OT \setminus {(o,t)} )</td>
</tr>
</tbody>
</table>
A = \{LS, LB, NIP, FW, OC\},
R = \{APP, SEC, ADMIN\} with a total order \succ on R, as defined in Table 2,
T = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}, t_{15}, t_{16}, t_{17}, t_{18}\}, as labeled in Table 1,
AR = \{(LS, APP), (LB, APP), (NIP, SEC), (FW, SEC), (OC, ADMIN)\},
TR = \{(t_i, APP), (t_{13}, SEC), (t_j, ADMIN)\mid (t_i \in T \mid 1 \leq i \leq 12, t_j \in T \mid 14 \leq j \leq 18)\},
DXOP = \{\text{`add flow rule'}, \text{`packet in'}, \text{`flow stats'}, \text{`packet out'}\},
Type(`add flow rule') = `Flow rule mod', Type(`packet in') = `Packet – In return',
Type(`flow stats') = `Switch stats request' = `Switch stats report', Type(`packet out') = `Packet – Out',
AuthorizationRule(LS, `add flow rule') = true, AuthorizationRule(LB, `add flow rule') = true,
AuthorizationRule(FW, `add flow rule') = true,
AuthorizationRule(LS, `packet in') = true, AuthorizationRule(LB, `packet in') = true, AuthorizationRule(NIP, `packet in') = true,
AuthorizationRule(FW, `packet in') = true AuthorizationRule(OC, `packet in') = true,
Refined Role Hierarchy

Admin
Vendor actions, Vendor features

Switch_Config.
Switch port mod, Switch set config

Switch_Diagnostics
Switch get config, Switch config reply
Switch port status

Traffic_Monitoring
(e.g., IDS)
Packet-In return

Traffic_Eng.
(e.g., LS, LB)
Flow rule mod

Stats_Collecting
(e.g., Billing)
Switch stats request
Switch stats report

Logging
Flow removal messages
Flow error reply

Syncing
Barrier requests, Barrier replies

Connection_Tracking
Echo requests, Echo replies

World-Leading Research with Real-World Impact!
• A formal authorization model for SDN apps.
• An administration model.
• A configuration of the formal model in a use case scenario of five apps.
• A refined Role hierarchy.

• Some future goals:
  • Extension of SE-Floodlight access control model to cover all controller resources.
  • An access control model following the NIST RBAC concept.
  • Fine-grained access control using ABAC within a holistic view to SDN resources.
Thank you! Questions?

abdullah.al-alaj@utsa.edu