YANG – Role-based security of Instances

…and multi tenant capability

Markus Seehofer
Stephan Kehrer Hirschmann Automation and Control GmbH

July 2022

Motivation

- Discussion in TSN profiles have shown the need for YANG modules that allow
  - role-based security of instances
  - multi tenant capability
- The realization for both of these requirements is profile specific
- As discussed in YANGsters on April 26, 2022, IEEE 802.1 base standards should not try to cover these profile specific requirements directly but provide the capabilities for a profile to use standard mechanisms, e.g., NACM (RFC 8341), to achieve them
- To help achieve these goals, this presentation
  - Provides an overview over the possibilities NACM offers for role-based security
  - How this can be used to provide multi tenant capabilities
  - An example, how a YANG module of a base standard could be modeled to allow NACM to be used to achieve both requirements

NACM Overview

- The IETF Network Access Control Model provides a simple and easy-to-configure access control rule framework

- NETCONF and RESTCONF enabled devices can transparently make use of NACM.
  No need to configure NACM for each protocol

- NACM assigns users to groups and assigns them to rules-lists and corresponding rules

- NACM allows very granular configuration:
  Per module, protocol-operation, data note and notification access

- By using XPATH expressions for data notes, access can be controlled on indexed instance data too

- In case of misconfigurations the use of a recovery session can bypass the existing NACM rules without explicitly deleting them

Access Control

NACM Features

The NACM model allows configuration to control:

- Module Access
  - Permission to access for definitions in a specific YANG module, identified by its name
- Protocol Operation Access
  - Permission to invoke specific protocol operations
- Datastore/Datanode Access
  - Permission to read and/or alter specific data nodes within any datastore
- Outgoing Notifications
  - Permission to receive specific notification event types

Groups and Users

NACM Groups and Users

- NACM is group-based and these groups and group membership lists are maintained in the NACM configuration
- A group contains zero or more users
- A user can be a member of several groups
- A user is derived from the transport layer during session establishment
- User authentication itself is not handled by NACM, but by other processes depending on how the user connects

Rule-Lists and Rules

- The NACM model defines ‘rule-lists’ and corresponding ‘rule’ entries for access control
- Rule lists are checked in the order they were created (‘ordered-by-user’)
- Rule lists consist of a name, a list of assigned groups and a list of ‘rule’ entries
- Rules are also checked in the order they were created (‘ordered-by-user’)
- Rules are processed in order, until a rule that matches the requested access operation is found, if no rule matches, NACM default settings apply (‘read’, ‘write’, and ‘exec’ can be configured)
- Rules consist of the following:
  - **name**
    - The name of the rule
  - **module-name**
    - Controls access for definitions in a specific YANG module, identified by its name. The wildcard ‘*’ allows access to all modules

Rule-Lists and Rules

- **rule-type**
  - Configures access by either
    - **protocol-operation**
      - controls access for a specific protocol operation, identified by its YANG module and name
    - **notification**
      - controls access for a specific notification event type, identified by its YANG module and name
    - **data-node**
      - controls access for a specific data node and its descendants, identified by its path location within the conceptual XML document for the data node

Rule-Lists and Rules

- **access-operation**
  - The type of operations matched by this rule. Can be multiple e.g.: create and delete
    - **‘*’**
      - allow all operations
    - **create**
      - Any protocol operation that creates a new data node.
    - **delete**
      - Any protocol operation that removes a data node.
    - **exec**
      - Execution access to the specified protocol operation.
    - **read**
      - Any protocol operation or notification that returns the value of a data node.
    - **update**
      - Any protocol operation that alters an existing data node.
- **action**
  - The action to take when this rule is matched, either
    - ‘permit’
    - or
    - ‘deny’
- **comment**
  - An arbitrary text describing the purpose of this rule

Requirements on base YANG models

- If instance-based security is required, a hierarchy must allow instantiation at each level where security restrictions apply
- ‘Actions’ must be used, if instance-based security is required
- ‘RPCs’ can be used, if no instance-based security is required

Example base YANG module

```
module: example-uni
  +--rw c-uni
     +--rw domain* [domain-id]
        +--rw domain-id    string
        +--rw cuc* [cuc-id]
           +--rw cuc-id    string
           +--rw stream* [stream-id]
              +--rw stream-id      uint8
              +--rw stream-name?   string
```

Example intended behavior

Admin User MUST:

- create, update and delete domains ("**d1**")
- create, update and delete CUCs ("**cuc1**", "**cuc2**“, and "**cuc3**")

Operator Users ("**op_cuc1**", "**op_cuc2**“, and "**op_cuc3**") MUST:

- only create, modify and delete streams for their own CUC for a specific domain
- not delete or modify each other's streams
- not delete or create the top-level container ("c-uni")
- not delete or create the domains ("**d1**")
- not delete or create the CUCs (" **cuc1**", " **cuc2**“, and " **cuc3**")

Example config domain & cucs

```
<config>
  <eu:c-uni xmlns:eu="urn:example-uni">
   <eu:domain>
    <eu:domain-id>d1</eu:domain-id>
    <eu:cuc>
     <eu:cuc-id>cuc1</eu:cuc-id>
    </eu:cuc>
    <eu:cuc>
     <eu:cuc-id>cuc2</eu:cuc-id>
    </eu:cuc>
    <eu:cuc>
     <eu:cuc-id>cuc3</eu:cuc-id>
    </eu:cuc>
   </eu:domain>
  </eu:c-uni>
</config>
```

Example default NACM configuration

```
<config>
  <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm“>
    <enable-nacm>true</enable-nacm>
    <read-default>permit</read-default>
    <write-default>deny</write-default>
    <exec-default>permit</exec-default>
  </nacm>
</config>
```

Example group - user assignment

```
<config>
  <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
    <groups>
      <group>
        <name>g_cuc1</name>
        <user-name>op_cuc1</user-name>
      </group>
      <group>
        <name>g_cuc2</name>
        <user-name>op_cuc2</user-name>
      </group>
      <group>
        <name>g_cuc3</name>
        <user-name>op_cuc3</user-name>
      </group>
    </groups>
  </nacm>
</config>
```

Example rule-list / rule [cuc1]

```
<rule-list>
  <name>rl_cuc1</name>
  <group>g_cuc1</group>
  <rule>
    <name>r1</name>
    <module-name>example-uni</module-name>
    <path xmlns:eu="urn:example-uni">/eu:c-uni/eu:domain[domain-id='d1']/eu:cuc[cuc-id!='cuc1']</path>
    <access-operations>create update delete exec</access-operations>
    <action>deny</action>
    <comment>Deny all other CUCs which are not ‘cuc1’</comment>
  </rule>
  <rule>
    <name>r2</name>
    <module-name>example-uni</module-name>
    <path xmlns:eu="urn:example-uni">/eu:c-uni/eu:domain[domain-id='d1']/eu:cuc[cuc-id='cuc1']</path>
    <access-operations>delete</access-operations>
    <action>permit</action>
    <comment>Permit ‘cuc1’ to delete own streams</comment>
  </rule>
```

Example rule-list / rule [cuc1]

```
  <rule>
    <name>r3</name>
    <module-name>example-uni</module-name>
    <path xmlns:eu="urn:example-uni">/eu:c-uni</path>
    <access-operations>delete</access-operations>
    <action>deny</action>
    <comment>Deny all to delete domains and/or CUCs</comment>
  </rule>
  <rule>
    <name>r4</name>
    <module-name>example-uni</module-name>
    <path xmlns:eu="urn:example-uni">/eu:c-uni</path>
    <access-operations>*</access-operations>
    <action>permit</action>
    <comment>Allow all operations (catch all)</comment>
  </rule>
</rule-list>
```

Conclusions

- A hierarchical structure of the base YANG module is necessary to allow modelling role-based security of instances and multi tenant capability, using NACM

- NACM can be used to achieve most of the intended behavior stated on slide 11

- For some intended restrictions, e.g., preventing a CUC from creating a new domain, it is currently unclear to the authors of this presentation how they could be achieved

- The structure of the base YANG module suggested on slide 10 might need to be modified to allow a decoupling of CUCs from domains

Thank you

Thank you!