Revision Letter

I2NSF Consumer-Facing Interface YANG Data Model

- **Old Draft Name: draft-ietf-i2nsf-consumer-facing-interface-dm-06**
- **New Draft Name: draft-ietf-i2nsf- consumer-facing-interface-dm-07**

Jaehoon Paul Jeong

11/5/2019

Dear Jan Lindblad,

I sincerely appreciate your valuable comments on the Consumer-Facing Interface Data Model document. Your comments use a bold font and my answers use a regular font with the prefix [PAUL].

Thanks for your volunteering to improve our Consumer-Facing Interface YANG Module.

Could you propose a way to redesign ietf-i2nsf-cfi-policy.yang to befit NACM?

=> Yes, I redesigned ietf-i2nsf-cfi-policy.yang to befit NACM for network configuration access control in the Consumer-Facing Interface (CFI). I add a new section about NACM in this revision as follows:

NEW: Section 7. Network Configuration Access Control Model (NACM)

7. Network Configuration Access Control Model (NACM)

Network Configuration Access Control Model (NACM) provides a high-level overview of the access control with the following features [RFC8341]:

- o Independent control of action, data, and notification access is provided.
- o A simple and familiar set of datastore permissions is used.
- o Support for YANG security tagging allows default security modes to automatically exclude sensitive data.
- o Separate default access modes for read, write, and execute permissions are provided.
- o Access control rules are applied to configurable groups of users.

The data model for the I2NSF Consumer-Facing Interface provides NACM mechanisms and concepts to user-group and owners permissions. The NACM with the above features can be used to set up all the management access controls in the I2NSF high-level authorization view, and it may have a high impact on the optimization and performance.

Certainly. Find my proposed sketch for the module structure attached.

=> I redesigned ietf-i2nsf-cfi-policy.yang with your proposed sketch.

I think it is important for the adoption of this module that it is reasonably easy to implement it on top of existing NETCONF/RESTCONF/YANG servers. They all implement the NACM management access control mechanism today, so the ietf-i2nsf-cfi-policy module should build on that. Its therefore important to leverage the existing NACM mechanisms and concepts for groups, users, permissions.

=> I applied NACM to “user-group” and “owners-ref” in “endpoint groups” in the Consumer-Facing Interface.

NEW: Section 8. YANG Data Model of Consumer-Facing Interface

```
grouping user-group {
  description
  "The grouping for user-group entities, and
    contains information such as name & ip-address.";
  leaf-list name {
    type leafref {
      path /nacm:nacm/nacm:groups/nacm:group/nacm:user-name;
    }
    description
    "This represents the name of a user.";
  }
  uses ip-address-info;
}

...

grouping owners-ref {
  description
    "This grouping is for owners reference using Network configuration Access Control
Model (NACM).";
  leaf-list owners {
    type leafref {
      path /nacm:nacm/nacm:groups/nacm:group/nacm:name;
    }
    description
      "This leaf-list names the owner groups of the
      list instace it sits on. Only the owners and
      super users are authorized to modify the contents.";
  }
}
```

It would be technically possible to set up all the management access control rules needed to implement the I2NSF ideas by only creating rules in NACM. The NACM rules are massively more complex than the simple owner leaf proposed in your YANG module, however. From a usability perspective I think it makes good sense to keep the abstraction in ietf-i2nsf-cfi-policy and let the module implementor make sure this high-level authorization view is translated into NACM specifics.

=> I agree at your opinion of creating the I2NSF rules through the NACM. Especially, I apply NACM to user-group and owners permissions for the management access controls in the I2NSF high-level authorization view as follows:

NEW: Section 7. Network Configuration Access Control Model (NACM)

7. Network Configuration Access Control Model (NACM)

Network Configuration Access Control Model (NACM) provides a high-level overview of the access control with the following features [RFC8341]:

- o Independent control of action, data, and notification access is provided.
- o A simple and familiar set of datastore permissions is used.
- o Support for YANG security tagging allows default security modes to automatically exclude sensitive data.
- o Separate default access modes for read, write, and execute permissions are provided.
- o Access control rules are applied to configurable groups of users.

The data model for the I2NSF Consumer-Facing Interface provides NACM mechanisms and concepts to user-group and owners permissions. The NACM with the above features can be used to set up all the management access controls in the I2NSF high-level authorization view, and it may have a high impact on the optimization and performance.

NEW: Section 8. YANG Data Model of Consumer-Facing Interface

```
grouping user-group {
  description
  "The grouping for user-group entities, and
    contains information such as name & ip-address.";
  leaf-list name {
    type leafref {
      path /nacm:nacm/nacm:groups/nacm:group/nacm:user-name;
    }
    description
    "This represents the name of a user.";
  }
  uses ip-address-info;
}

...
```

```
grouping owners-ref {
  description
    "This grouping is for owners reference using Network configuration Access Control
Model (NACM).";
  leaf-list owners {
    type leafref {
      path /nacm:nacm/nacm:groups/nacm:group/nacm:name;
    }
    description
      "This leaf-list names the owner groups of the
      list instace it sits on. Only the owners and
      super users are authorized to modify the contents.";
  }
}
```

In order to make this feasible, I changed the owner string leaf into a leafref pointer to NACM groups and removed the modules separate identities for permissions. Lets adopt the NACM counterparts instead.

=> According to your comments, I changed the owner string leaf into a leafref pointer to NACM groups and removed the module's separate identities for permissions as follows:

NEW: Section 8. YANG Data Model of Consumer-Facing Interface

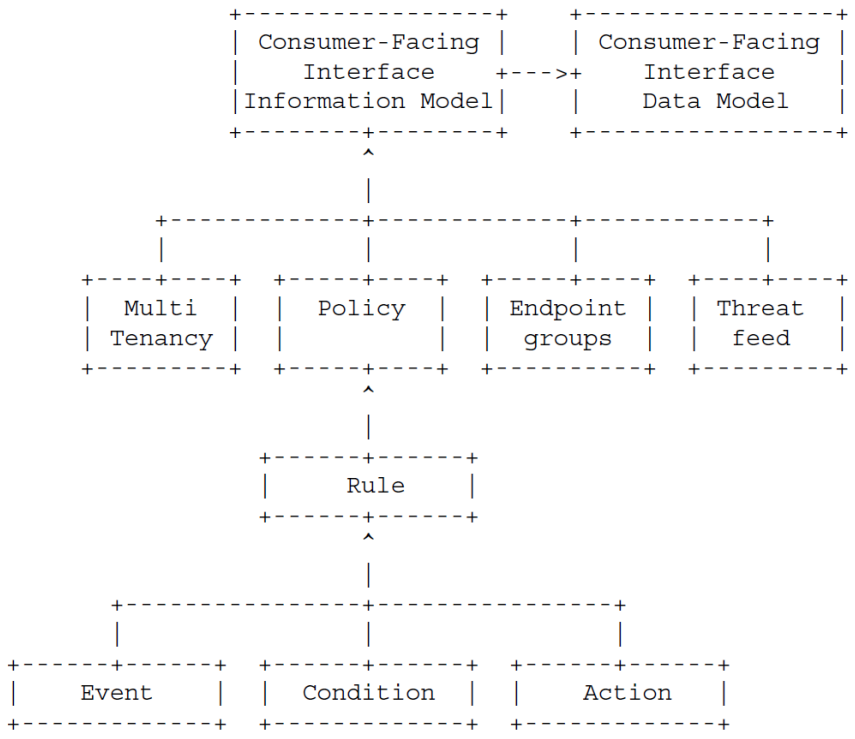
```
grouping user-group {
  description
  "The grouping for user-group entities, and
    contains information such as name & ip-address.";
  leaf-list name {
    type leafref {
      path /nacm:nacm/nacm:groups/nacm:group/nacm:user-name;
    }
    description
    "This represents the name of a user.";
  }
  uses ip-address-info;
}

...

grouping owners-ref {
  description
    "This grouping is for owners reference using Network configuration Access Control
Model (NACM).";
  leaf-list owners {
    type leafref {
      path /nacm:nacm/nacm:groups/nacm:group/nacm:name;
    }
    description
```

```
      "This leaf-list names the owner groups of the
      list instace it sits on. Only the owners and
      super users are authorized to modify the contents.";
  }
}
```

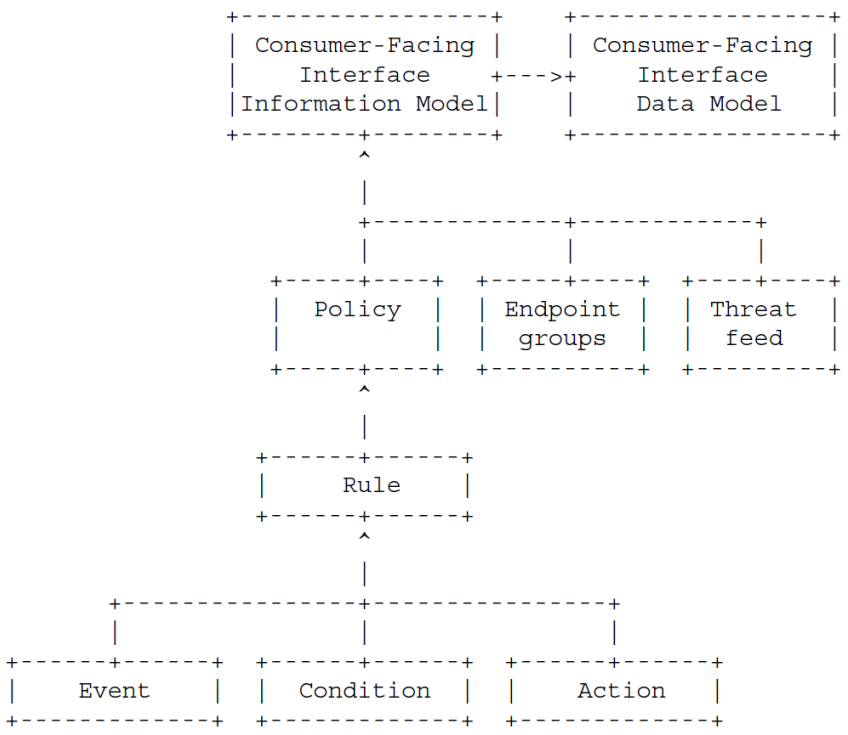
The structure of the rules was very flat, i.e. the domains, tenants, policies and rules were mostly side by side, not reflecting their logical hierarchy in the YANG. This would make the number of NACM rules to control access to each individual item very high. By arranging them in a tree structure, I believe the number of NACM rules can be kept to a minimum. NACM rules may have a high impact on server performance, so its important to not have excessive amounts of them.

=> Since domains and tenants can be expressed by letting a user group include the information of a domain and a tenant, this revision removes the Multi-Tenancy having domain and tenant. As a result, the YANG tree structure for a policy rule becomes hierarchical rather than flat.

OLD: Figure 1: Diagram for High-level Abstraction of Consumer-Facing Interface



NEW: Figure 1: Diagram for High-level Abstraction of Consumer-Facing Interface



I created a hierarchy with domains on top, each domain containing zero or more tenants, each with zero or more policies that in turn consist of zero or more rules. At each level it is possible to list owners in the form of NACM groups. The module implementor would then have to translate these owner references to actual NACM rules.

=> According your guideline, as shown in the above figure, I created a hierarchy for a policy rule such that the domain and tenant can be expressed by a user group based on NACM. For example, a tenant called “eng-it” in a domain called “example.com” can be represented as “example.com-- eng-it”. For a policy rule, a traffic source target (denoted as “src-target”) can be set to “example.com-- eng-it” in our CFI data model, and can be parsed into the domain “example.com” and the tenant “eng-it”.

Here is an example sketch configuration and the resulting NACM rules (in CLI style syntax for readability):

i2nsf-cfi domains domain example.com

owners [example.com--eng-it]

tenants tenant dev

policies policy team-black

owners [example.com--dev]

rules rule 2

!

rules rule allow-malware-sites

owners [example.com--dev]

This is supposed to mean that members of the example.com--eng-it group have full ownership of everything in the example.com domain. Within this domain, there is a tenant called dev, with a policy called team-black. That policy is owned by example.com--dev. This means this policy may be updated by members in example.com--dev and example.com--eng-it. Within the policy there are two rules ("2" and "allow-malware-sites"). The "allow-malware-sites" rule has the example.com--dev group listed as owner; this is superfluous. In this example, the rules are otherwise empty.

=> In our new CFI data model, a policy can have multiple rules with the information of a domain and a tenant.

NEW: 4. Information Model for Policy

```
+--rw i2nsf-cfi-policy* [policy-name]
   +--rw policy-name          string
   |  +--rw rule* [rule-name]
   +--rw endpoint-group
   +--rw threat-prevention
```

Figure 2: Policy YANG Data Tree

In order for existing NC/RC/YANG servers to enforce the above, the ietf-i2nsf-cfi-policy module implementation would need to translate the intent above to NACM rules like the ones below. In this example, the implementation created a rule to allow members of the dev and eng-it groups within the example.com org to see the example.com domain and everything within it. Next there is a rule to allow members of the example.com dev group to update the policy named team-black within the dev tenant. Finally, there is a rule to allow the eng-it group members to update anything within the example.com domain. The default nacm policy per statement in the YANG is to deny anyone else to see anything within the i2nsf domain.

nacm rule-list example.com

group [example.com--dev example.com--eng-it]

rule read-all

path /i2nsf-cfi/domains/domain[name=example.com]

access-operations read

action permit

!

!

nacm rule-list example.com--dev

group [example.com--dev]

rule 1

path /i2nsf-cfi/domains/domain[name=example.com]/tenants/tenant[name =dev]/policies/policy[name=team-black]

action permit

!

!

nacm rule-list example.com--eng-it

group [example.com--eng-it]

rule 1

path /i2nsf-cfi/domains/domain[name=example.com]

action permit

!

!

=> The revised CFI data model can accommodate multiple groups with user-group for a policy as follows:

NEW: 5.1. User Group

```
+--rw user-group* [name]
   +--rw name -> /../../nacm:group/nacm:user-name
   +--rw (match-type)?
      +--:(exact-match-ipv4)
      |  +--rw ip-address*            inet:ipv4-address
      +--:(exact-match-ipv6)
      |  +--rw ip-address*            inet:ipv4-address
      +--:(range-match-ipv4)
      |  +--rw range-ipv4-address*
                        [start-ipv4-address end-ipv4-address]
      |     +--rw start-ipv4-address    inet:ipv4-address
      |     +--rw end-ipv4-address      inet:ipv4-address
      +--:(range-match-ipv6)
         +--rw range-ipv6-address*
                        [start-ipv6-vaddress end-ipv6-address]
            +--rw start-ipv6-address    inet:ipv6-address
            +--rw end-ipv6-address      inet:ipv6-address
```

Figure 9: User Group YANG Data Tree

NACM also contains a mapping from user names to groups. Is this in line you're your expectations? Do we need additional infrastructure to control this mapping?

=> In our CFI data model, user-group can represent either a group or a user. If you think that we need to differentiate a user and a group in our revised data model, please let us know how to revise the YANG module of our latest version -07.

nacm groups group example.com--dev

user-name [jan vasilij]

!

nacm groups group example.com--eng-it

user-name [chris victor]

!

nacm groups group example.com--finance

user-name [clara sakura]

!

What do you think about this approach to the management access control? Im not sure I got the relations between domains, tenants, policies and rules as you want them. Are all these levels needed? Do you believe this is this is a workable approach to your vision?

=> Yes, as you may see it, your approach is workable, so I modified our CFI data model as much as possible according to your suggestions.

Please let me know if you would like me to take any further steps with this sketch. I should mention that I also have plenty of other comments on your updated module, but I want to get the access control approach resolved before looking at anything else.

=> If you have further comments, please let me know.

I am not aware of any particular party interested to implement our data model.

Then it is all the more important that the solution can be implemented on top of the existing servers out there without modifying them.

=> Yes, I totally agree with you.

Thanks.

Best Regards,

Paul

--

==========

Mr. Jaehoon (Paul) Jeong, Ph.D.

Associate Professor

Department of Software

Sungkyunkwan University

Office: +82-31-299-4957

Email: jaehoon.paul@gmail.com, pauljeong@skku.edu

Personal Homepage: http://iotlab.skku.edu/people-jaehoon-jeong.php