I2NSF Registration Interface YANG Data Model

draft-ietf-i2nsf-registration-interface-dm-15

IETF 113, Vienna

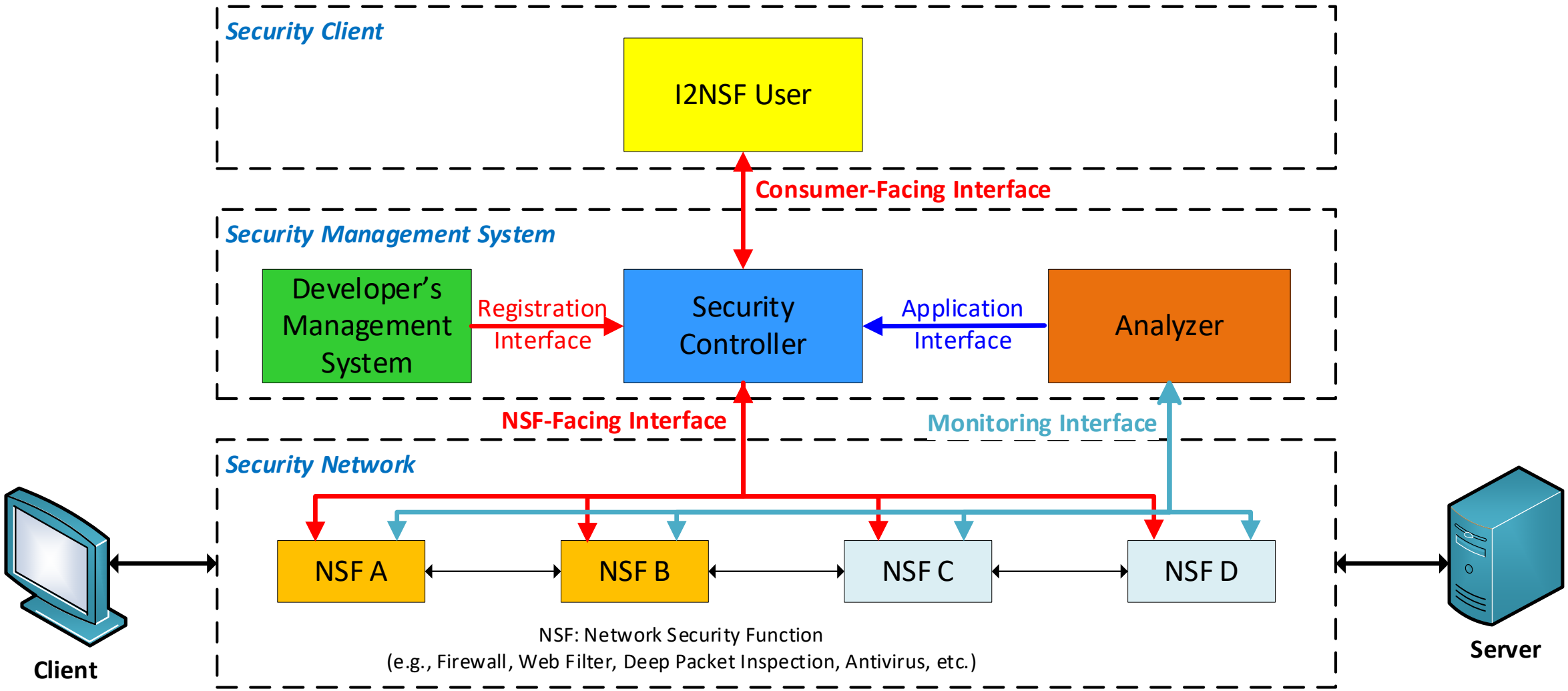
March 24th, 2022

Jaehoon (Paul) Jeong and Patrick Lingga

{pauljeong, patricklink}@skku.edu

Sungkyunkwan University

I2NSF Framework



I2NSF Framework (Registration Interface)

Objectives:

1. Registering the Capabilities of NSFs with an I2NSF Framework:
   - Developer's Management System (DMS) in an I2NSF Framework is typically run by an NSF vendor and uses Registration Interface to provide NSFs developed by the NSF vendor to Security Controller.
   - DMS registers NSFs and their capabilities with Security Controller in an I2NSF Framework through Registration Interface.

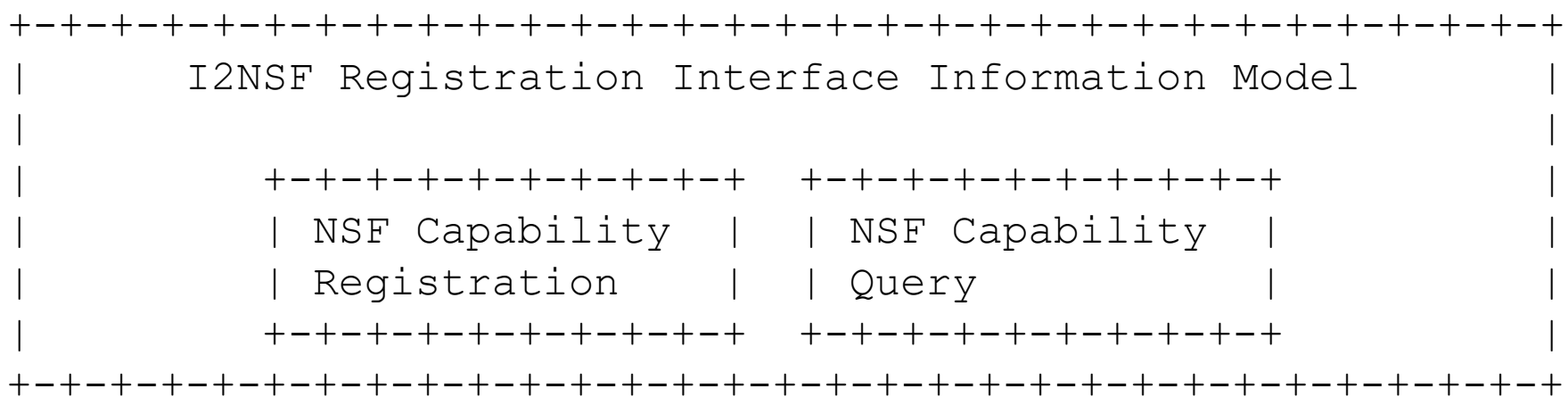
2. Updating the capabilities of registered NSFs:
   - After an NSF is registered with Security Controller, some modifications on the capability of the NSF MAY be required later.
   - In this case, DMS uses Registration Interface to update the capability of the NSF, and this update SHOULD be reflected in the catalog (or database) of NSFs.

3. Asking DMS about some required capabilities:
   - Security Controller might fail to find any NSFs having the required capabilities among the registered NSFs.
   - In this case, Security Controller needs to request DMS for additional NSF(s) that can provide the required security capabilities via Registration Interface.

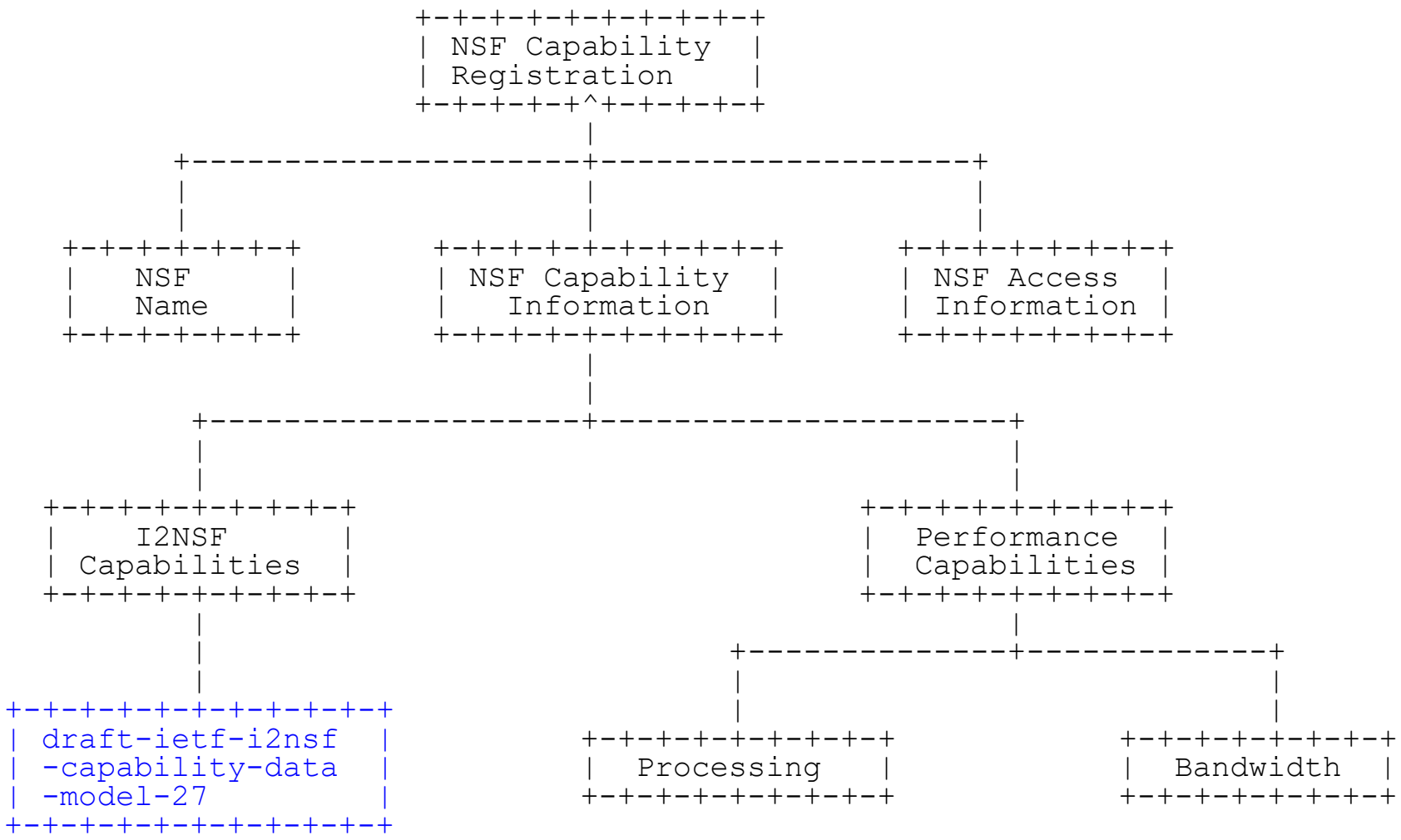
Registration Interface

To accomplish the objectives, the Registration YANG data model provides two sub-models:

1. NSF Capability Registration → Registering and Updating NSFs' capabilities
2. NSF Capability Query → Query for required capabilities.



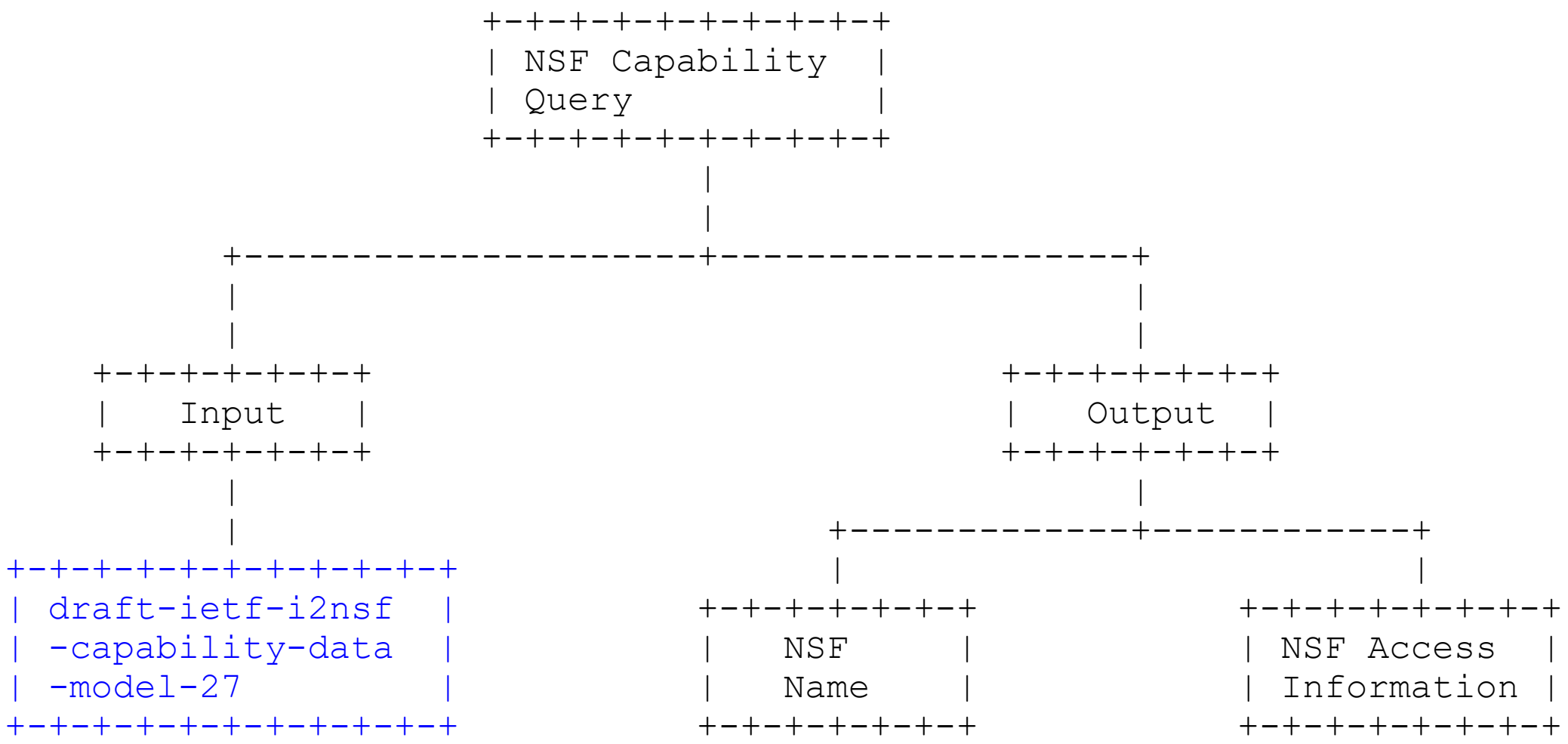
NSF Capability Registration Model



NSF Capability Registration YANG Tree

```
module: ietf-i2nsf-reg-interface
  +--rw nsf-registrations
     +--rw nsf-information* [nsf-name]
        +--rw nsf-name                string
        +--rw nsf-capability-info
        |  +--rw security-capability
        |  |  +--uses ietf-i2nsf-capability -> import from draft-ietf-i2nsf-capability-data-model-27
        |  +--rw performance-capability
        |     +--rw processing
        |     |  +--rw processing-average?   uint16
        |     |  +--rw processing-peak?      uint16
        |     +--rw bandwidth
        |        +--rw outbound
        |        |  +--rw outbound-average?   uint32
        |        |  +--rw outbound-peak?      uint32
        |        +--rw inbound
        |           +--rw inbound-average?   uint32
        |           +--rw inbound-peak?      uint32
        +--rw nsf-access-info
           +--rw ip?     inet:ip-address-no-zone
           +--rw port?   inet:port-number
```

NSF Capability Query Model



NSF Capability Query YANG Tree

```
module: ietf-i2nsf-reg-interface
rpcs:
    +---x nsf-capability-query
       +---w input
       |  +---w query-nsf-capability
       |  |  +--uses ietf-i2nsf-capability -> import from draft-ietf-i2nsf-capability-data-model-27
       +--ro output
          +--ro nsf-access-info
             +--ro nsf-name?   string
             +--ro ip?         inet:ip-address-no-zone
             +--ro port?       inet:port-number
```

XML Example

```
<nsf-registrations xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-reg-interface" xmlns:nsfcap="urn:ietf:params:xml:ns:yang:ietf-
i2nsf-capability">
  <nsf-information>
    <nsf-name>firewall</nsf-name>
    <nsf-capability-info>
      <security-capability>
        <condition-capabilities>
          <generic-nsf-capabilities>
            <ipv6-capability>nsfcap:next-header</ipv6-capability>
            <ipv6-capability>nsfcap:source-address</ipv6-capability>
            <ipv6-capability>nsfcap:destination-address</ipv6-capability>
            <tcp-capability>nsfcap:source-port-number</tcp-capability>
            <tcp-capability>nsfcap:destination-port-number</tcp-capability>
          </generic-nsf-capabilities>
        </condition-capabilities>
        <action-capabilities>
          <ingress-action-capability>nsfcap:pass</ingress-action-capability>
          <ingress-action-capability>nsfcap:drop</ingress-action-capability>
          <egress-action-capability>nsfcap:pass</egress-action-capability>
          <egress-action-capability>nsfcap:drop</egress-action-capability>
        </action-capabilities>
      </security-capability>
      <performance-capability>
        <processing>
          <processing-average>2</processing-average>
          <processing-peak>4</processing-peak>
        </processing>
        <bandwidth>
          <outbound>
            <outbound-average>1000</outbound-average>
            <outbound-peak>5000</outbound-peak>
          </outbound>
          <inbound>
           <inbound-average>1000</inbound-average>
           <inbound-peak>5000</inbound-peak>
          </inbound>
        </bandwidth>
      </performance-capability>
    </nsf-capability-info>
    <nsf-access-info>
      <ip>2001:db8:0:1::11</ip>
      <port>49152</port>
    </nsf-access-info>
  </nsf-information>
</nsf-registrations>
```

Conclusion

- The Registration Interface has a short YANG data model as most contents are imported from the Capability YANG data model - draft-ietf-i2nsf-capability-data-model.

- Most changes in the Registration Interface document is the examples following the changes in the Capability YANG data model document.

- The Registration Interface provides additional information of performance capabilities and access information of the NSFs.