IETF Hackathon: SRv6 VPN Configuration with NETCONF YANG

IETF 105
20-21 July 2019
Montreal

HACKATHON

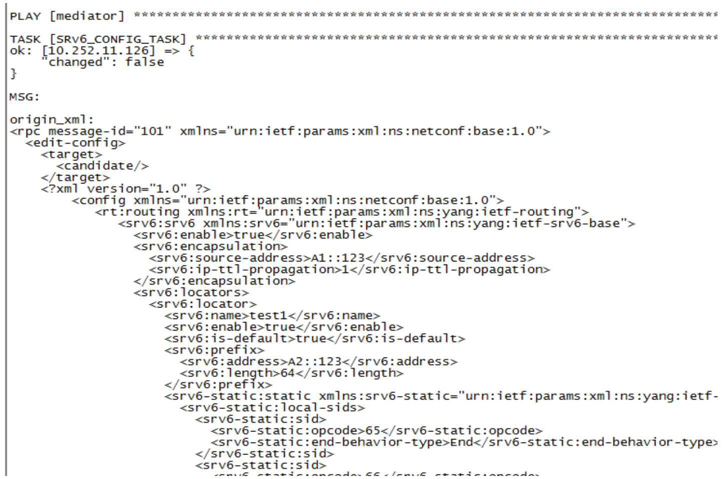
<IETF 105 / Montreal>

Hackathon Plan

- We have already deployed SRv6 VPN and config SRv6 VPN via CLI or Native SRv6 YANG model, Standard SRv6 YANG is under development by IETF.
- We are looking for the path to support the operator’s controller using IETF yang to interactive with vendor’s native yang that deployed on legacy device.
    - Use Ansible Playbook & IETF YANG data models to Configure SRv6 VPN;
    - Implement key features in IETF SRv6 model
    - Implement key function(s) including “Standard YANG(IETF) to Vendor Native YANG translation” plugin
- Important drafts：
    - draft-raza-spring-srv6-yang-00

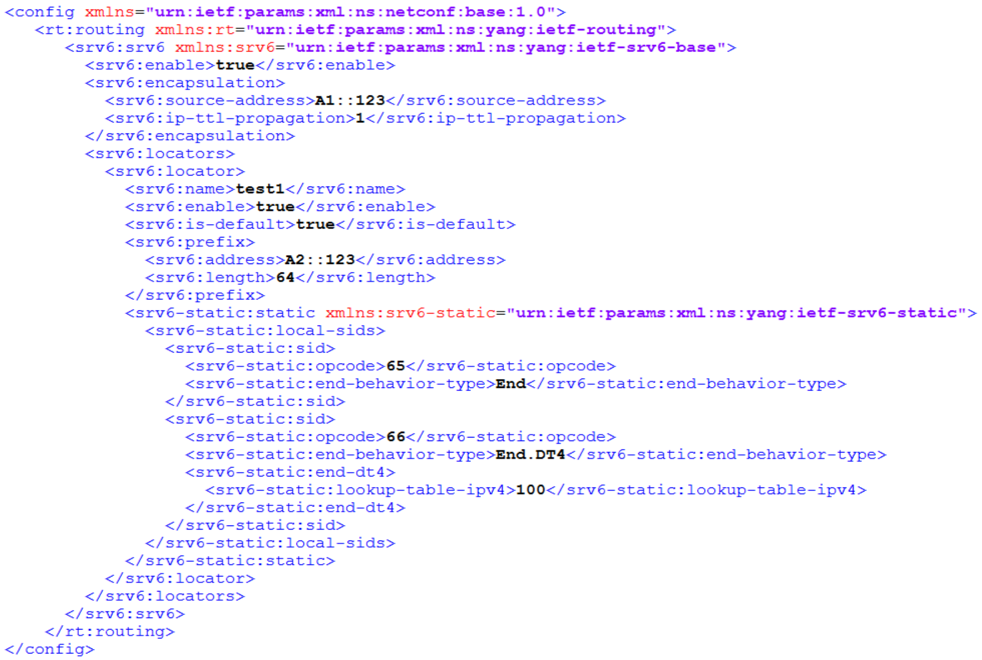
What Got Done

- Achievements
  - Implement Ansible APIs for 1) SRv6 Global Configuration; 2) ISIS SRv6 Configuration; 3) SRv6 VPN Configuration;

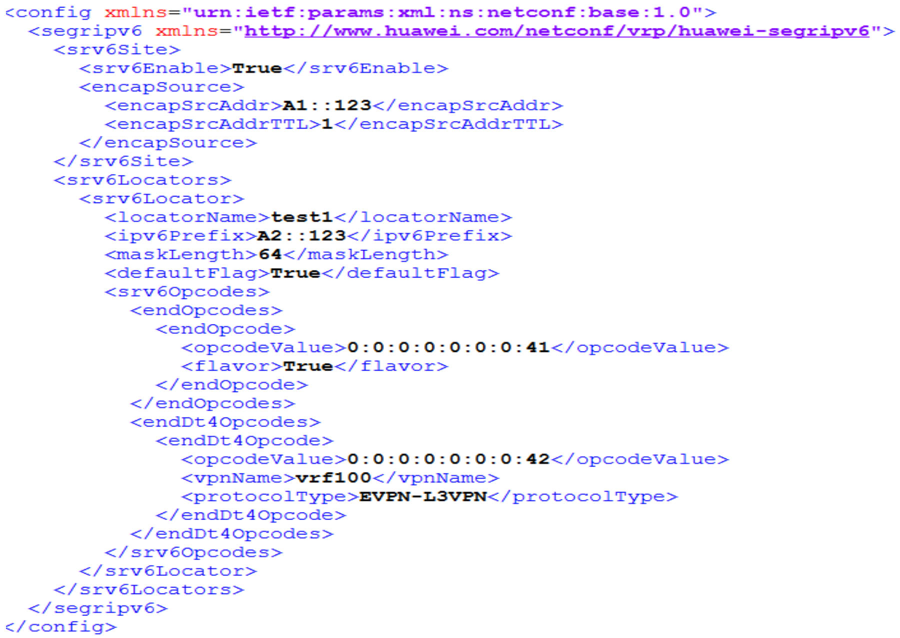


```
PLAY [mediator] *************************************************************************

TASK [SRv6_CONFIG_TASK] *****************************************************************
ok: [10.252.11.126] => {
    "changed": false
}

MSG:

origin_xml:
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <?xml version="1.0" ?>
        <config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
           <rt:routing xmlns:rt="urn:ietf:params:xml:ns:yang:ietf-routing">
              <srv6:srv6 xmlns:srv6="urn:ietf:params:xml:ns:yang:ietf-srv6-base">
                <srv6:enable>true</srv6:enable>
                <srv6:encapsulation>
                  <srv6:source-address>A1::123</srv6:source-address>
                  <srv6:ip-ttl-propagation>1</srv6:ip-ttl-propagation>
                </srv6:encapsulation>
                <srv6:locators>
                  <srv6:locator>
                    <srv6:name>test1</srv6:name>
                    <srv6:enable>true</srv6:enable>
                    <srv6:is-default>true</srv6:is-default>
                    <srv6:prefix>
                      <srv6:address>A2::123</srv6:address>
                      <srv6:length>64</srv6:length>
                    </srv6:prefix>
                    <srv6-static:static xmlns:srv6-static="urn:ietf:params:xml:ns:yang:ietf-
                      <srv6-static:local-sids>
                        <srv6-static:sid>
                          <srv6-static:opcode>65</srv6-static:opcode>
                          <srv6-static:end-behavior-type>End</srv6-static:end-behavior-type>
                        </srv6-static:sid>
                        <srv6-static:sid>
```

  - Extend pyangbind plugin to allow convert the IETF YANG model to the vendor's native YANG model, E.G:
    - Input ietf-srv6-base.yang → Output huawei-segripv6.yang



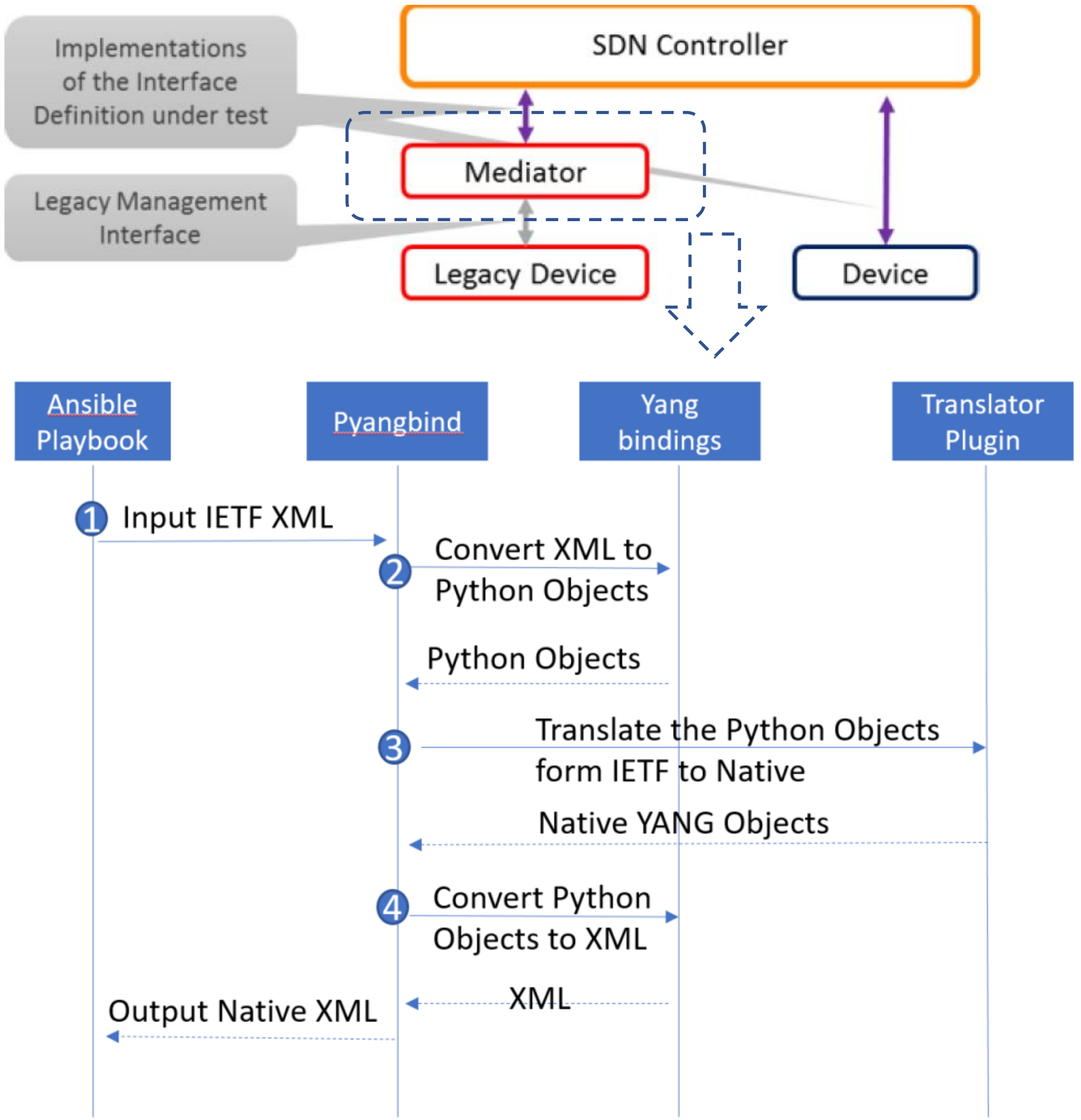
```
<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <rt:routing xmlns:rt="urn:ietf:params:xml:ns:yang:ietf-routing">
      <srv6:srv6 xmlns:srv6="urn:ietf:params:xml:ns:yang:ietf-srv6-base">
        <srv6:enable>true</srv6:enable>
        <srv6:encapsulation>
          <srv6:source-address>A1::123</srv6:source-address>
          <srv6:ip-ttl-propagation>1</srv6:ip-ttl-propagation>
        </srv6:encapsulation>
        <srv6:locators>
          <srv6:locator>
            <srv6:name>test1</srv6:name>
            <srv6:enable>true</srv6:enable>
            <srv6:is-default>true</srv6:is-default>
            <srv6:prefix>
              <srv6:address>A2::123</srv6:address>
              <srv6:length>64</srv6:length>
            </srv6:prefix>
            <srv6-static:static xmlns:srv6-static="urn:ietf:params:xml:ns:yang:ietf-srv6-static">
              <srv6-static:local-sids>
                <srv6-static:sid>
                  <srv6-static:opcode>65</srv6-static:opcode>
                  <srv6-static:end-behavior-type>End</srv6-static:end-behavior-type>
                </srv6-static:sid>
                <srv6-static:sid>
                  <srv6-static:opcode>66</srv6-static:opcode>
                  <srv6-static:end-behavior-type>End.DT4</srv6-static:end-behavior-type>
                  <srv6-static:end-dt4>
                    <srv6-static:lookup-table-ipv4>100</srv6-static:lookup-table-ipv4>
                  </srv6-static:end-dt4>
                </srv6-static:sid>
              </srv6-static:local-sids>
            </srv6-static:static>
          </srv6:locator>
        </srv6:locators>
      </srv6:srv6>
    </rt:routing>
</config>
```



```
<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <segripv6 xmlns="http://www.huawei.com/netconf/vrp/huawei-segripv6">
    <srv6Site>
      <srv6Enable>True</srv6Enable>
      <encapSource>
        <encapSrcAddr>A1::123</encapSrcAddr>
        <encapSrcAddrTTL>1</encapSrcAddrTTL>
      </encapSource>
    </srv6Site>
    <srv6Locators>
      <srv6Locator>
        <locatorName>test1</locatorName>
        <ipv6Prefix>A2::123</ipv6Prefix>
        <maskLength>64</maskLength>
        <defaultFlag>True</defaultFlag>
        <srv6Opcodes>
          <endOpcodes>
            <endOpcode>
              <opcodeValue>0:0:0:0:0:0:0:41</opcodeValue>
              <flavor>True</flavor>
            </endOpcode>
          </endOpcodes>
          <endDt4Opcodes>
            <endDt4Opcode>
              <opcodeValue>0:0:0:0:0:0:0:42</opcodeValue>
              <vpnName>vrf100</vpnName>
              <protocolType>EVPN-L3VPN</protocolType>
            </endDt4Opcode>
          </endDt4Opcodes>
        </srv6Opcodes>
      </srv6Locator>
    </srv6Locators>
  </segripv6>
</config>
```

Translator Function and Work Flow

- Translate Function:
  - Input: XML instance that modelled using IETF YANG
  - Output: XML instance that modelled using Native YANG
- The plugin yang2yang-translate flow is as shown below:
  1. Input XML instance to be translated. This XML modelled using “ietf-srv6-base”
  2. Use Pyangbind to Parse and instantiate the Python Objects (YANG bindings) based on “ietf-srv6-base”.
  3. Call the Translator Plugin.
     - traverse this Python Obj and create new Python Obj based on “huawei-segripv6”
  4. Use Pyangbind to convert native based Python Obj to XML instance.


Implementations of the Interface Definition under test
SDN Controller
Mediator
Legacy Management Interface
Legacy Device
Device
Ansible Playbook
Pyangbind
Yang bindings
Translator Plugin
1 Input IETF XML
2 Convert XML to Python Objects
Python Objects
3 Translate the Python Objects form IETF to Native
Native YANG Objects
4 Convert Python Objects to XML
XML
Output Native XML

What We Learned

- IETF SRv6 YANG can be deployed to support SRv6 service delivery
  - We using Ansible Playbook to orchestrate multitask including 1) SRv6 Global YANG 2) ISIS SRv6 YANG 3) SRv6 VPN YANG.
- But there is a problem that vendor's Legacy Device may only support Native YANG, and operator's Controller may want to deploy IETF Standard YANG as common interface to inactive with multiple vendors.
  - To address this problem, we developed a plugin to support to translate standard model to native model.
- It is feasible to translate standard model to native model;
- Solicited a lot of valuable use cases such as:
  - IETF YANG ←→ CLI;

The Crews

- Champion(s)
  - Guangying Zheng <zhengguangying@huawei.com>
- Members:
  - Zhibo Hu;
  - Rohit R Ranade (remote);
  - Gang Yan;
  - Chong Feng;
  - Zitao Wang
  - Qin Wu