

EVPN Service Automation with Anuta ATOM

Highly Scalable Vendor Agnostic Service Orchestration

Key Capabilities

- Supports IETF YANG & OpenConfig Models
- Ensures Service Compliance
- Automated delivery of Model-driven Service UI
- Supports brownfield network discovery
- Supports complete lifecycle management of services
- Dry-run mode to validate command generation
- Vendor Agnostic
- Integration into Internal & External IPAM, VLAN, RD & RT Manager
- Workflow Integration for Pre & Post Validations
- Quickly scale up to 1M+

With the increase in the adoption of cloud-based architectures and the proliferation of data centers closer to end-users, there is a growing need to simplify today's transport networks. EVPN or Ethernet VPN, with this ability and has given significant confidence to SPs, web scale companies, and large enterprises to take this route.

Over the years, transport networks have become complicated. With imposing demands of complex use cases, different overlay networks have come into play across access, core, and datacenters - L2VPN for metro network domain, L3VPN for core network domain, and VXLAN for datacenters. These disparate networks also require stitching to facilitate an end-to-end network infrastructure, adding to the existing complexity. While it has been instrumental in realizing the use cases, network management has taken a hit. With 5G and other evolving technologies, there has been an initiative to simplify the structure by adopting a single overlay technology such as EVPN.

Today, we are at the cusp of widespread EVPN deployment to the extent that it now serves as an underlay to SD-WAN technology in several implementations. It enables a single interface for enterprises by integrating L2 and L3 under a single technology, removing the need for multiple VPN protocols. For DCIs, EVPN also brings uniformity between WAN and data centers and enables scalable L2 or L3 connectivity with standard control and data plane between virtualized data centers. EVPN can play an essential role in IP networks used for mobile backhaul and aggregation in CSP networks. EVPN supports a unified IP Any haul (IP mobile transport) solution with VLL, VPLS, and L3VPN connectivity. Mobile RAN traffic can be transported directly to the EPC/cloud mobile gateway using a point-to-point layer 2 VPN (EVPN-VPWS). The unified solution can provide efficient local hand-off and low-latency using EVPN-VPLS for point-to-multipoint Layer 2 VPN connectivity and EVPN-IP VPN for Layer 3 VPN connectivity.

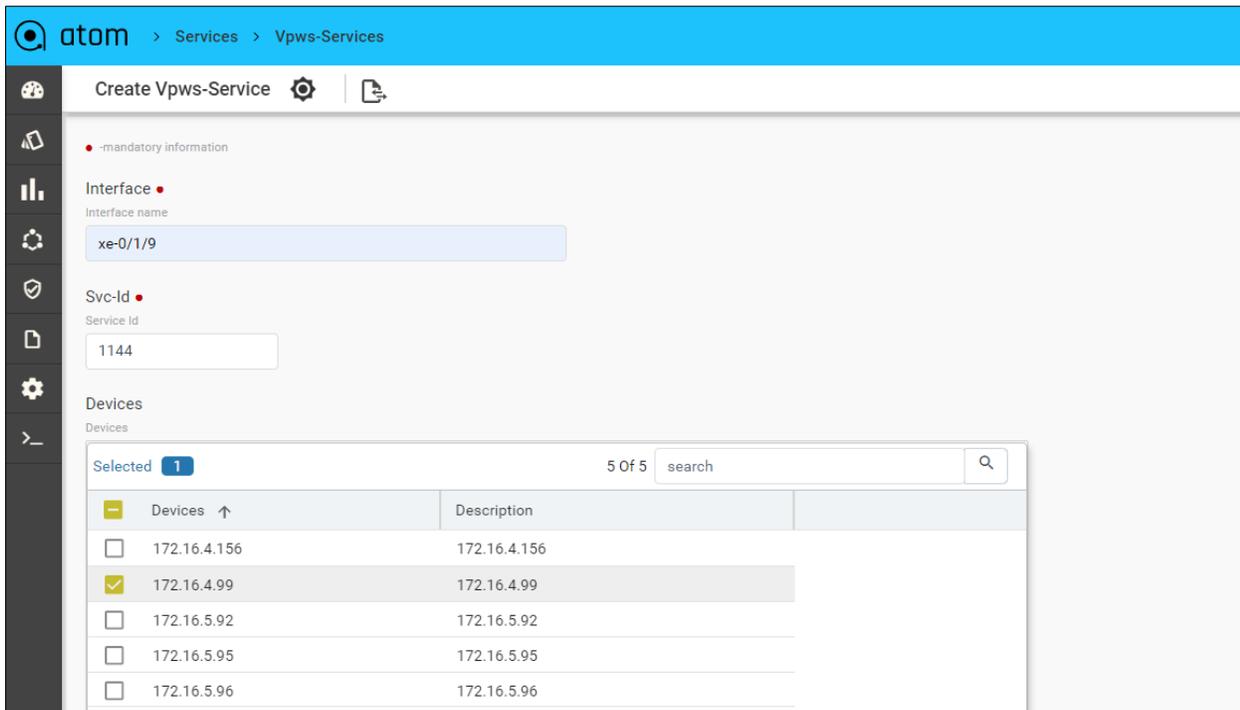
This journey of network simplification and transformation could turn unpleasant if the automation of EVPN is not done right.

Today Anuta ATOM offers out-of-box support to provision and maintain the lifecycle of EVPN VPWS and EVPN L3VPN services.

EVPN VPWS Automation

The model here follows a minimalistic approach and offers an entirely abstracted means to EVPN VPWS provisioning. As seen in the below screenshot, the only inputs taken include:

1. Interface name
2. Service-Id
3. Devices



The screenshot shows the ATOM web interface for creating a Vpws-Service. The breadcrumb navigation is 'atom > Services > Vpws-Services'. The main heading is 'Create Vpws-Service'. There are three sections for mandatory information:

- Interface:** Interface name is 'xe-0/1/9'.
- Svc-Id:** Service Id is '1144'.
- Devices:** A table with 5 rows. The second row is selected.

Devices	Description
<input type="checkbox"/> 172.16.4.156	172.16.4.156
<input checked="" type="checkbox"/> 172.16.4.99	172.16.4.99
<input type="checkbox"/> 172.16.5.92	172.16.5.92
<input type="checkbox"/> 172.16.5.95	172.16.5.95
<input type="checkbox"/> 172.16.5.96	172.16.5.96

Keeping it simple was a deliberate approach, and the intent here is to mask all the complexity at the network layer from the network operator. The service-id entered in the user form forms the values for VLAN ID, Router distinguisher, and Route Target values.

Once we apply the model, ATOM first checks for resource availability on the devices involved and then generates and pushes the device's commands to complete the provisioning.

The below screenshot shows the service level view of a provisioned service.

```
Vpws-Service ↗ ✕
```

```
<vpws-services xmlns="http://anutanetworks.com/evpn-vpws-service">
  <vpws-service>
    <devices>172.16.4.99</devices>
    <interface>xe-0/1/9</interface>
    <svc-id>1144</svc-id>
  </vpws-service>
</vpws-services>
```

 Copy  Download

The network device selected is a Juniper MX device that supports native YANG. Based on the resource qualification, ATOM generates the NETCONF payload to provision the VPWS service using EVPN.

The below screenshot shows the NETCONF payload generated. The service-id entered in the user form translates into the VLAN-ID, and the RD and RT values as intended.

```
Create: vpws-service xe-0/1/9,1144 ✕
```

```
Logs  Commands
```

```
<interface>
  <name>xe-0/1/9</name>
  <unit nc:operation="create">
    <name>1144</name>
    <encapsulation>vlan-ccc</encapsulation>
    <vlan-id>1144</vlan-id>
  </unit>
</interface>
</interfaces> <!-- filler -->
<routing-instances xmlns="http://yang.juniper.net/junos/conf/routing-instances"> <!-- filler -->
  <!-- subtree:CREATE /controller/devices/device=172.16.4.99/atom-junos-mount/junipermx194R110/junos-conf-root:configuration/junos-conf-routing-instances:routing-
instances/instance=VPWS-1144 -->
  <instance nc:operation="create">
    <name>VPWS-1144</name>
    <instance-type>evpn-vpws</instance-type>
    <route-distinguisher>
      <rd-type>99:1144</rd-type>
    </route-distinguisher>
    <vrf-target>
      <community>target:99:1144</community>
    </vrf-target>
  </instance>
</routing-instances>
</interface>
  <name>xe-0/1/9,1144</name>
```

The commands provisioned on the Juniper device is also shown below.

```
admin@GRE-vMX-5-99> show configuration | display set | match 1144
set interfaces xe-0/1/9 unit 1144 encapsulation vlan-ccc
set interfaces xe-0/1/9 unit 1144 vlan-id 1144
set routing-instances VPWS-1144 protocols evpn interface xe-0/1/9.1144 vpws-service-id local 1144
set routing-instances VPWS-1144 protocols evpn interface xe-0/1/9.1144 vpws-service-id remote 1144
set routing-instances VPWS-1144 instance-type evpn-vpws
set routing-instances VPWS-1144 interface xe-0/1/9.1144
set routing-instances VPWS-1144 route-distinguisher 99:1144
set routing-instances VPWS-1144 vrf-target target:99:1144

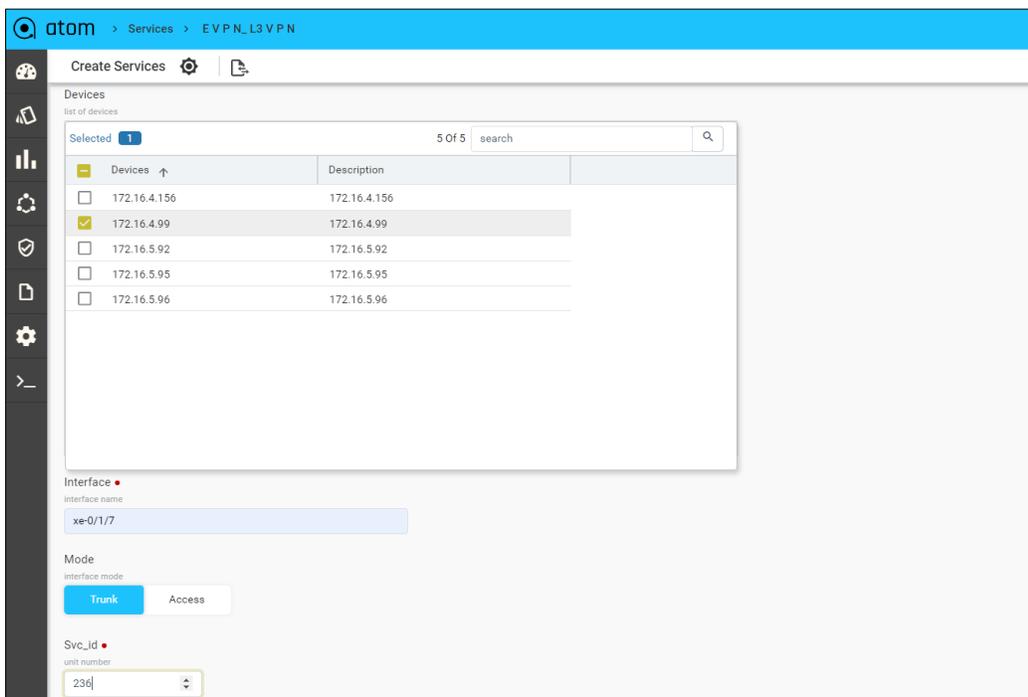
admin@GRE-vMX-5-99>
```

EVPN L3VPN Automation

The model addresses the provisioning of EVPN L3VPN along with its integration to the legacy L3VPN solution. As in the case of above model, the operator needs to feed in minimal inputs to activate the service. As seen in the below screenshot, the only inputs taken include:

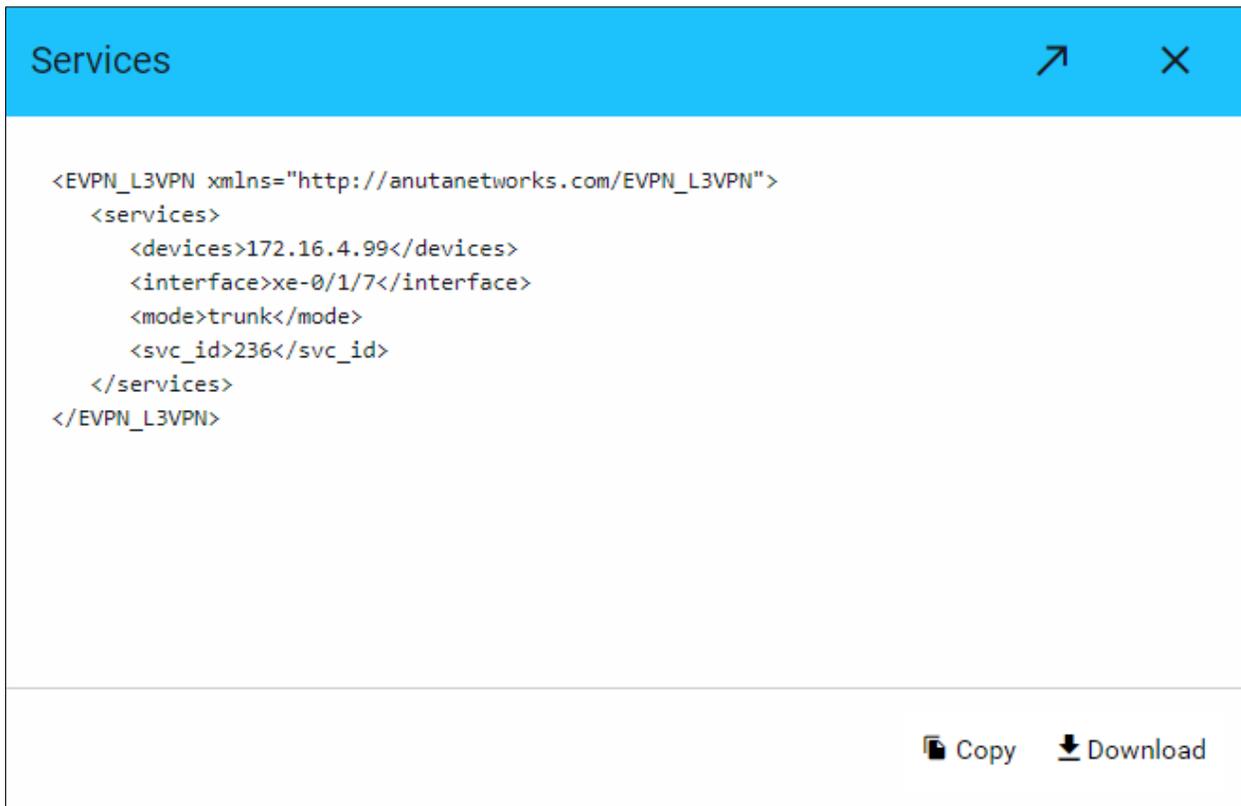
1. Devices
2. Interface
3. Port Mode [Trunk or Access]
4. Service_id

The Service_Id here is restricted to a range of 0 to 254 as it now translates into an octet on the IP address for the service, along with configurations around sub-interfaces, part of bridge domain configurations, and as Route distinguisher & Route target.



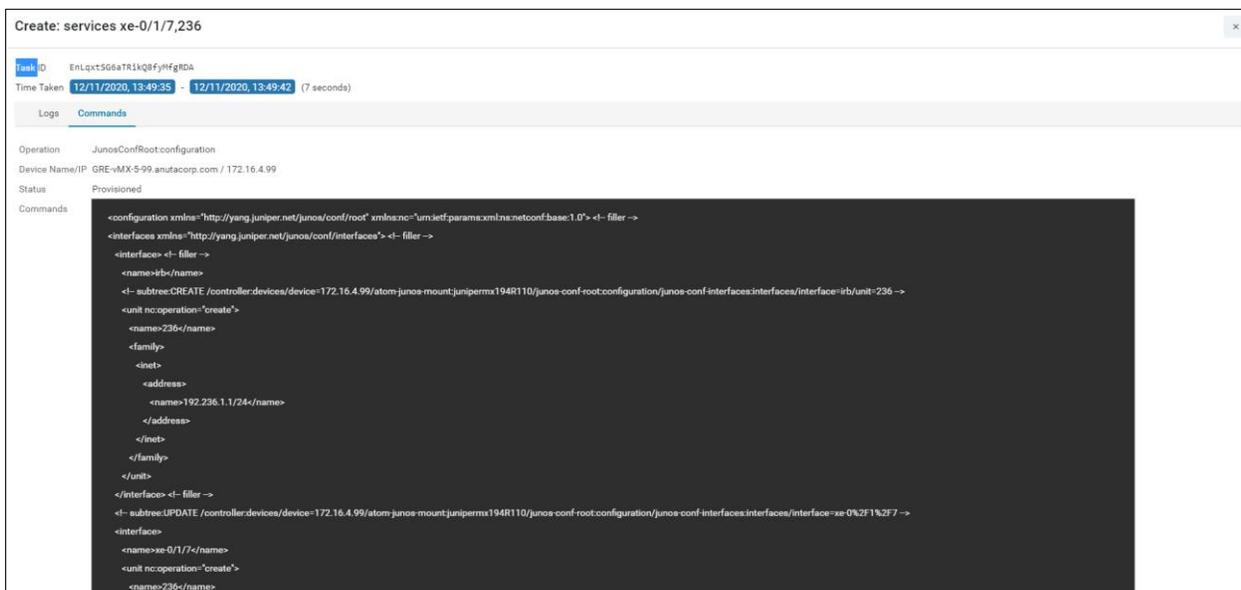
Once we apply the model, ATOM first checks for resource availability on the devices involved and then generates and pushes the device's commands to complete the provisioning.

The below screenshot shows the service level view of a provisioned service.



```
<EVPN_L3VPN xmlns="http://anutanetworks.com/EVPN_L3VPN">
  <services>
    <devices>172.16.4.99</devices>
    <interface>xe-0/1/7</interface>
    <mode>trunk</mode>
    <svc_id>236</svc_id>
  </services>
</EVPN_L3VPN>
```

The below screenshots shows the NETCONF payload generated.



```
<configuration xmlns="http://yang.juniper.net/junos/conf/root" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <!-- filler -->
  <interfaces xmlns="http://yang.juniper.net/junos/conf/interfaces">
    <!-- filler -->
    <interface <!-- filler -->
      <name>xe-0/1/7</name>
      <!-- subtree:CREATE /controller:devices/device=172.16.4.99/atom:junos-mount:junipermx194R110/junos-conf-root:configuration/junos-conf:interfaces/interfaces/interface=xe-0/1/7/unit=236 -->
      <!-- unit:nc:operation="create" -->
      <name>236</name>
      <family>
        <inet>
          <address>
            <name>192.236.1.1/24</name>
          </address>
        </inet>
      </family>
    </unit>
  </interface>
  <!-- filler -->
  <!-- subtree:UPDATE /controller:devices/device=172.16.4.99/atom:junos-mount:junipermx194R110/junos-conf-root:configuration/junos-conf:interfaces/interfaces/interface=xe-0/1/7 -->
  <interface>
    <name>xe-0/1/7</name>
    <!-- unit:nc:operation="create" -->
    <name>236</name>
```

```

Create: services xe-0/1/7,236
Logs Commands
<encapsulation>vlan bridge</encapsulation>
<family>
  <bridge>
    <interface-mode>trunk</interface-mode>
    <vlan-id-list>236</vlan-id-list>
  </bridge>
</family>
</unit>
</interface>
</interfaces> <!-- filler -->
<routing-instances xmlns="http://yang.juniper.net/junos/conf/routing-instances"> <!-- filler -->
<!-- subtree:UPDATE /controller.devices/device=172.16.4.99/atom-junos-mount:junipermx194R110/junos-conf-root:configuration/junos-conf-routing-instances:routing-instances/instance=EVPN-CORE -->
<instance>
  <name>EVPN-CORE</name>
  <interface-operation>"create">
    <name>xe-0/1/7,236</name>
  </interface>
  <bridge-domains> <!-- filler -->
  <!-- subtree:CREATE /controller.devices/device=172.16.4.99/atom-junos-mount:junipermx194R110/junos-conf-root:configuration/junos-conf-routing-instances:routing-instances/instance=EVPN-CORE/bridge-domains/domain=bridge.236 -->
  <domain-operation>"create">
    <name>bridge.236</name>
    <domain-type>bridge</domain-type>
    <vlan-id>236</vlan-id>
    <routing-interface>ib.236</routing-interface>
    <bridge-options>
      <interface>
        <name>xe-0/1/7,236</name>
      </interface>
    </bridge-options>
  </domain>
</bridge-domains> <!-- filler -->
</protocols> <!-- filler -->
<!-- subtree:UPDATE /controller.devices/device=172.16.4.99/atom-junos-mount:junipermx194R110/junos-conf-root:configuration/junos-conf-routing-instances:routing-instances/instance=EVPN-CORE/protocols/evpn -->
  <evpn>
    <extended-vlan-list>236</extended-vlan-list>
  </evpn>
</protocols> <!-- filler -->
</instance>
<!-- subtree:CREATE /controller.devices/device=172.16.4.99/atom-junos-mount:junipermx194R110/junos-conf-root:configuration/junos-conf-routing-instances:routing-instances/instance=L3VPN-236 -->
<instance-operation>"create">
  <name>L3VPN-236</name>
  <instance-type>vrf</instance-type>
  <interface>
    <name>ib.236</name>
  </interface>
  <vrf-target>
    <community>target.99.236</community>
  </vrf-target>
  <protocols>
    <bgp>
      <group>
        <name>CRPD</name>
        <type>external</type>
        <import>monitor</import>
        <peer-as>1</peer-as>
        <neighbors>
          <name>192.236.1.2</name>
        </neighbors>
      </group>
    </bgp>
  </protocols>
  <route-distinguisher>
    <rd-type>99.236</rd-type>
  </route-distinguisher>
</instance>
</routing-instances> <!-- filler -->
</configuration> <!-- filler -->

```

```

Create: services xe-0/1/7,236
Logs Commands
</domain>
</bridge-domains> <!-- filler -->
</protocols> <!-- filler -->
<!-- subtree:UPDATE /controller.devices/device=172.16.4.99/atom-junos-mount:junipermx194R110/junos-conf-root:configuration/junos-conf-routing-instances:routing-instances/instance=EVPN-CORE/protocols/evpn -->
  <evpn>
    <extended-vlan-list>236</extended-vlan-list>
  </evpn>
</protocols> <!-- filler -->
</instance>
<!-- subtree:CREATE /controller.devices/device=172.16.4.99/atom-junos-mount:junipermx194R110/junos-conf-root:configuration/junos-conf-routing-instances:routing-instances/instance=L3VPN-236 -->
<instance-operation>"create">
  <name>L3VPN-236</name>
  <instance-type>vrf</instance-type>
  <interface>
    <name>ib.236</name>
  </interface>
  <vrf-target>
    <community>target.99.236</community>
  </vrf-target>
  <protocols>
    <bgp>
      <group>
        <name>CRPD</name>
        <type>external</type>
        <import>monitor</import>
        <peer-as>1</peer-as>
        <neighbors>
          <name>192.236.1.2</name>
        </neighbors>
      </group>
    </bgp>
  </protocols>
  <route-distinguisher>
    <rd-type>99.236</rd-type>
  </route-distinguisher>
</instance>
</routing-instances> <!-- filler -->
</configuration> <!-- filler -->

```

```

</group>
</protocols>
<route-distinguisher>
  <rd-type>99.236</rd-type>
</route-distinguisher>
</instance>
</routing-instances> <!-- filler -->
</configuration> <!-- filler -->

```

The commands provisioned on the Juniper device is also shown below.

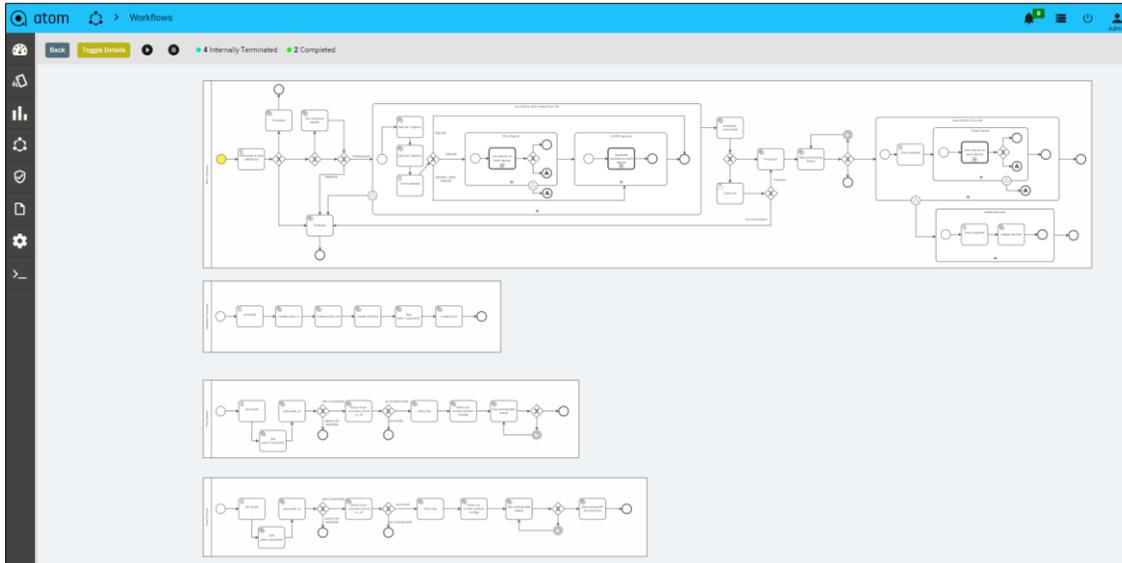
```
admin@GRE-vMX-5-99> show configuration | display set | match 236
set interfaces xe-0/1/7 unit 236 encapsulation vlan-bridge
set interfaces xe-0/1/7 unit 236 family bridge interface-mode trunk
set interfaces xe-0/1/7 unit 236 family bridge vlan-id-list 236
set interfaces irb unit 236 family inet address 192.236.1.1/24
set routing-instances EVPN-CORE protocols evpn extended-vlan-list 236
set routing-instances EVPN-CORE bridge-domains bridge.236 domain-type bridge
set routing-instances EVPN-CORE bridge-domains bridge.236 vlan-id 236
set routing-instances EVPN-CORE bridge-domains bridge.236 routing-interface irb.236
set routing-instances EVPN-CORE bridge-domains bridge.236 bridge-options interface xe-0/1/7.236
set routing-instances EVPN-CORE interface xe-0/1/7.236
set routing-instances L3VPN-236 protocols bgp group CRPD type external
set routing-instances L3VPN-236 protocols bgp group CRPD import monitor
set routing-instances L3VPN-236 protocols bgp group CRPD peer-as 1
set routing-instances L3VPN-236 protocols bgp group CRPD neighbor 192.236.1.2
set routing-instances L3VPN-236 instance-type vrf
set routing-instances L3VPN-236 interface irb.236
set routing-instances L3VPN-236 route-distinguisher 99:236
set routing-instances L3VPN-236 vrf-target target:99:236
```

Service Compliance

Anuta ATOM's service modeling capabilities not only allows a smooth vendor agnostic provisioning, but also maintains the state and sanity of the services through its service compliance feature. The service compliance module detects the configuration drift to device configurations, performs service inventory to understand the nature of the changes and notifies upon deviation. ATOM also generates the difference in configuration along with the commands to be reconciled to ensure service compliance. The user is given the authority to overwrite the device or server (ATOM) service model configurations.

Workflow Integration

Every service provisioning is governed by a method-of-procedure in organizations. Integrating ATOM's service models into ATOM workflow engine, helps to execute a set of Pre-Checks, followed by the choice of services to be provisioned and Post-Checks to ensure a smooth and successful service provisioning. Below workflow executes an instance of VPN service with pre & post validations.



Additional Resources

[Video-on-demand](#) on ATOM EVPN Service Automation

To learn how Anuta Network's ATOM Multi-Vendor Service Orchestration can accelerate your network automation journey, contact us at <https://www.anutanetworks.com>