
Inmanta Documentation

Release 2.6.2

Inmanta NV

Sep 11, 2024

CONTENTS:

- 1 Introduction** **1**
- 1.1 Prerequisites 1
- 1.2 Supported Features and Platforms 3
- 1.3 Inmanta Connect Lab 4
- 1.4 Inventory 15
- 1.5 Testing 19
- 1.6 Configuration and Customization 27
- 1.7 Extension 74
- 1.8 Frequently Asked Questions (FAQ) 76

- 2 Additional resources** **77**

- 3 PDF version** **79**

INTRODUCTION

Inmanta connect provides a means to seamlessly employ MEF standards covering Carrier Ethernet services. Carrier Ethernet is a service provided by ISPs to customers who want to extend their local networks over long (MAN and WAN) geographical locations.

The Carrier Ethernet service comes in three flavors:

- **Ethernet Virtual Private Line** or **E-Line**: a service connecting two customer Ethernet ports over a WAN.
- **Ethernet Virtual Private LAN** or **E-LAN**: a multi-point service connecting a set of customer endpoints, giving the appearance to the customer of a bridged Ethernet network connecting the sites.
- **Ethernet Virtual Private Tree** or **E-Tree**: a multi-point service connecting one or more roots and a set of leaves, but preventing inter-leaf communication.

More information regarding the Carrier Ethernet standards defined by MEF can be found [here](#).

1.1 Prerequisites

This section provides the required information about the connect module, ultimately enabling you to interact with customers more efficiently and to propose a solution respectively.

Who this guide is intended for:

- Pre-sales Engineers
- Developers/Deployment Engineers
- Development Managers

1.1.1 Planning

There certain factors that you should consider when approaching a customer:

1. Number and location of their sites (campus, data center)
2. Number of networking gears (routers, multi-layer switches, firewalls, etc.) that they manage
3. Vendor and OS version of their network equipment
4. Features and technologies that they utilize for instance:
 - Epipe
 - L3VPN
 - LDP

- etc...

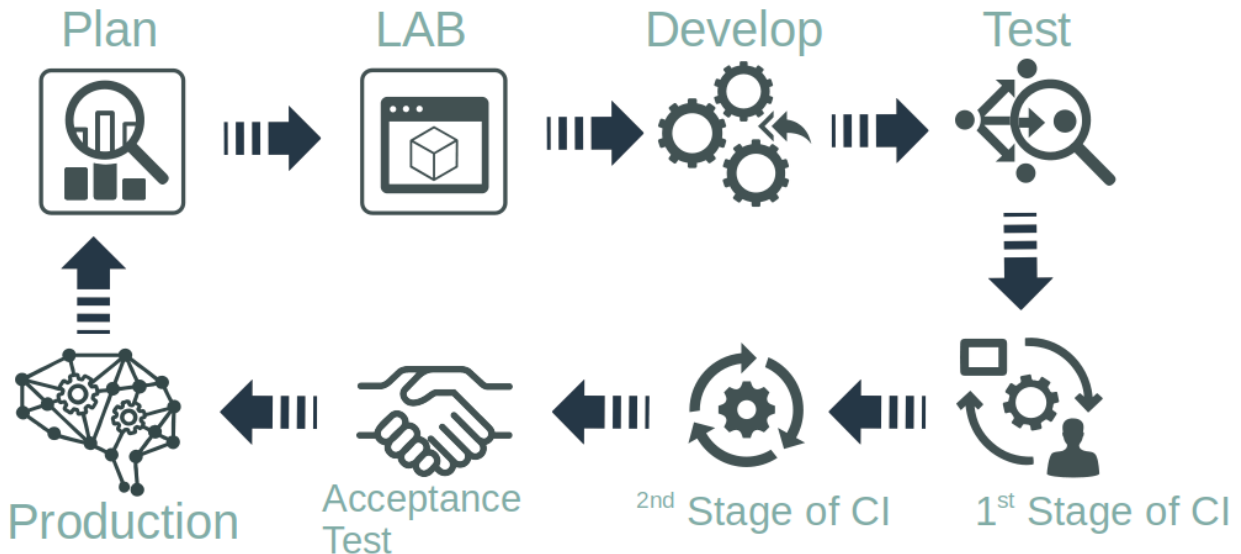
5. inventory system in use:

- Kind/brand
- Quality of data in the inventory system
- Does it support API calls? If so:
 - How is the quality of the API?
 - How is the response time of the API?
 - The slower the response, the more time it takes to develop/automate
- Could Inmanta run it in a LAB environment?
- Size of the data model
- Interaction mode with the inventory:
 - transaction support
 - planning mode
 - etc...

1.1.2 Development

In general each developer will receive a virtual lab to develop and test on. Upon committing code, tests on that commit will run on our CI/CD solution. The CI server has one or more virtual lab instances of its own, for on-commit and nightly builds. If the LABs have components that can not be virtualized or are too expensive to have many, we share LABs, but that increases the cost of making the tests.

The figure below depicts our development workflow:



Under certain circumstances, we use physical LABs for our testing and development.

In order to set up a development environment, you need to follow these steps:

1. Check out our [developer getting started guide](#)
2. Set up a base project using [this guide](#)

3. Check out our supported *platforms and features*
4. Set up a test LAB using *this guide*
5. Develop or test the current module
6. Run tests using *this guide*

In case you could not find the answer to your question, please check out our [FAQ page](#).

1.2 Supported Features and Platforms

Inmanta Connect supports the following platforms:

- Nokia SR-OS 20.10 or later through netconf/yang. This means that a recent version of SR-OS is required and MD-CLI has to be enabled.
- Juniper MX R18 or later through netconf/yang in rfc-compliant mode.
- Cisco XR 7.3 or later on ASR 9k through netconf/yang

We test our products against multiple vendors and versions. For specific versions and devices please contact sales or support for more details.

We are constantly developing new features and are adding more vendors to our arsenal. The table below depicts our current supported features and platforms:

| Features | Cisco | Juniper | Nokia |
|--------------------------------|---------|---------|---------|
| Customization | | | |
| Configuration based | * | * | * |
| Template based | * | * | * |
| Network backend | | | |
| LDP | * | * | * |
| EVPN | * | * | * |
| EVPN - Multihoming | Planned | Planned | * |
| Services | | | |
| Point-to-point EVC | * | * | * |
| Multipoint EVC | Planned | * | * |
| L3VPN | Planned | Planned | Planned |
| Layer 1 connections | Planned | Planned | Planned |
| Testing and development | | | |
| Full test suite | * | * | * |
| Containerized lab | * | * | * |

1.3 Inmanta Connect Lab

Table of Contents

1.3.1 Virtual Lab Setup

Summary

This guide describes the required steps to set up a lab using Containerlab. Containerlab enables us to quickly create a variety of topologies with support for different vendors using docker containers and experiment with Inmanta Connect module.

In this guide, we use a simple topology containing:

- 2 routers (or 4 if testing multihoming) from [our supported vendors](#)
- 4 clients (NFV-Test-API)
- Inmanta service orchestrator to test services which we deploy to the routers.

Set Up A Virtual Machine

1. Using the hypervisor of your choice, spin up a CentOS machine with either of the below specs:

| System Requirements | CPU Cores | RAM (GB) | Disk Space (GB) |
|-------------------------------------|-----------|----------|-----------------|
| Minimum (only applicable for Nokia) | 4 | 16 | 20 |
| Recommended | 4 | 32 | 35 |

2. Take note of the VM's IP address.
3. Once the VM is up, install `git` and `docker`:

- Install `git`:

```
sudo yum update
sudo yum install git
```

- Install `docker`:

```
sudo dnf config-manager --add-repo=https://download.docker.com/linux/centos/
↪docker-ce.repo
sudo dnf install docker-ce docker-ce-cli containerd.io
sudo systemctl enable --now docker
```

4. Check `docker` status:

```
sudo systemctl status docker
```

The output should look like this:

```
$ sudo systemctl status docker
docker.service - Docker Application Container Engine
Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset:┐
```

(continues on next page)

(continued from previous page)

```
→disabled)
Active: active (running) since Fri 2021-06-04 10:56:34 UTC; 3 days ago
Docs: https://docs.docker.com
```

5. Add your user to docker group:

```
sudo usermod -aG docker $USER
```

Install Containerlab

1. Before installing [Containerlab](#) we need to update and reboot the VM:

```
sudo yum update
sudo reboot
```

2. Install Containerlab:

```
sudo yum-config-manager --add-repo=https://yum.fury.io/netdevops/ && \
echo "gpgcheck=0" | sudo tee -a /etc/yum.repos.d/yum.fury.io_netdevops_.repo

sudo yum install containerlab
```

This is a short and starch-free version of Containerlab setup. For more information, you can visit its official docs:

- [Installation](#)
- [Quick Start](#)

Warning Due to some [breaking changes](#) in Containerlab 0.15, Connect LAB with version $\geq 0.3.0$ only works with clab version ≥ 0.15 . If you want to use a Containerlab older than 0.15 you will need to install one of the earlier versions of Connect LAB.

You can check which version of Containerlab you have installed using the `clab version` command.

Download And Setup Images

The specific steps to download and setup the images of each vendor could be viewed in the following pages:

Cisco LAB Setup

Cisco Image

Please follow the steps [here](#) to set up a functioning virtual Cisco XRv router.

From now on, we will expect this image to be tagged as `containerlab/vr-xrv:6.3.1`:

```
$ docker images
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
containerlab/vr-xrv 6.3.1       e006d956bb22    30 seconds ago  911MB
```

Cisco Base Configuration

At the time of this writing, Containerlab is not able to automatically place a base configuration for Cisco routers and this step has to be done manually:

- Head to `connect-lab/labs/config/cisco` directory.
- *Using this section*, connect to routers and paste the contents of for instance `east.cfg` file in `router-east`.

Based on the desired topology, you need to paste a specific configuration to its respective router.

Juniper LAB Setup

Juniper Image

Please follow the steps [here](#) to set up a functioning virtual Juniper router.

From now on, we will expect this image to be tagged as `vrnetlab/vr-vmx:18.3R1.9`:

```
$ docker images
```

| REPOSITORY | TAG | IMAGE ID | CREATED | |
|-----------------|----------|--------------|-------------|---|
| ↪ SIZE | | | | |
| vrnetlab/vr-vmx | 18.3R1.9 | 91e0035e6f3b | 8 weeks ago | ↪ |
| ↪ 4.75GB | | | | |

Juniper Base Configuration

At the time of this writing, Containerlab is not able to automatically place a base configuration for Juniper routers and this step has to be done manually:

- Head to `connect-lab/labs/config/juniper` directory.
- *Using this section*, connect to routers and paste the contents of for instance `east.cfg` file in `router-east`.

Based on the desired topology, you need to paste a specific configuration to its respective router.

Nokia LAB Setup

Nokia Image And License

Please follow the steps [here](#) to set up a functioning virtual Nokia router.

From now on, we will expect this image to be tagged `vrnetlab/vr-sros:20.10.R1`.

```
$ docker images
```

| REPOSITORY | TAG | IMAGE ID | CREATED | |
|------------------|----------|--------------|---------------|---|
| ↪ SIZE | | | | |
| vrnetlab/vr-sros | 20.10.R1 | 5835fa21ae15 | About an hour | ↪ |
| ↪ ago 992MB | | | | |

Nokia routers need to have a **license** to function. Please visit [here](#) for more information.

`clab deploy` command for routers should be run from the directory that contains the license file. In this case `connect-lab/labs`.

Get The Connect LAB Resources

The Connect LAB project, containing the configuration files and c1ab topology files, can be downloaded from Inmanta packages repository:

```

DEFAULT_VERSION="1.0.5"
read -rp "Enter the version you want to install (defaults to $DEFAULT_VERSION): " \
↪VERSION \
  && if [[ $VERSION == "" ]]; then VERSION=$DEFAULT_VERSION; echo "No version provided, \
↪defaulting to $VERSION"; fi \
  && read -rsp "Enter your private token here: " TOKEN \
  && echo "" \
  && curl -s0 --fail https://packages.inmanta.com/$TOKEN/connect/raw/versions/$VERSION/ \
↪connect-lab-$VERSION.tar.gz \
  || ( \
    echo "FAILED to get package" \
    && cat connect-lab-$VERSION.tar.gz \
    && echo "" \
    && exit 1 \
  ) \
  && tar -zxf connect-lab-$VERSION.tar.gz \
  && echo "All done! The lab project is available under connect-lab-$VERSION/ directory"

```

Currently the connect-lab version that is used for this guide is 1.0.5.

The project structure is as follows:

```

connect-lab/
├── docs
│   └── repair-a-lab.md
├── labs
│   ├── config
│   │   ├── cisco-evc-2R
│   │   │   ├── east.cfg
│   │   │   └── west.cfg
│   │   ├── cisco-evc-4R
│   │   │   ├── east.cfg
│   │   │   ├── north.cfg
│   │   │   ├── south.cfg
│   │   │   └── west.cfg
│   │   ├── juniper-evc-2R
│   │   │   ├── east.cfg
│   │   │   └── west.cfg
│   │   ├── juniper-evc-4R
│   │   │   ├── east.cfg
│   │   │   ├── north.cfg
│   │   │   ├── south.cfg
│   │   │   └── west.cfg
│   │   ├── nfv-test-api
│   │   │   └── config.yaml
│   │   ├── nokia-evc-2R
│   │   │   ├── east.cfg
│   │   │   └── west.cfg
│   │   └── nokia-evc-4R

```

(continues on next page)

Starting the LAB

The topology consists of two routers; or four in case of multihoming, four clients, Inmanta service orchestrator and a DB for the orchestrator. Inmanta service orchestrator uses the management network to connect to the routers and the orchestrator DB.

Before you proceed, check that all the required images are present:

```
$ docker images
```

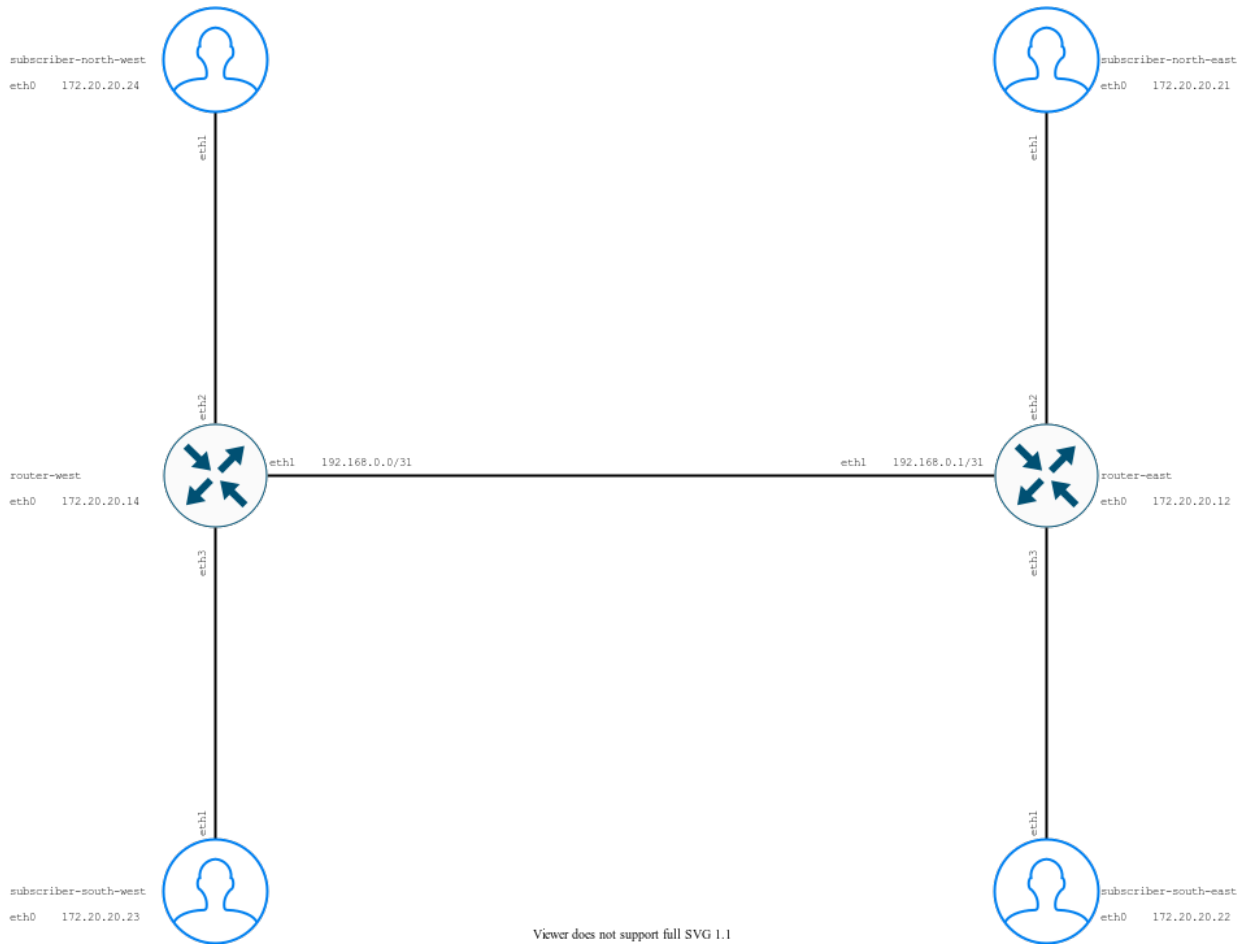
| REPOSITORY | TAG | IMAGE ID |
|--|----------|--------------|
| ↪CREATED | | |
| SIZE | | |
| docker.cloudsmith.io/inmanta/containers/service-orchestrator | 4 | 5139265d16e3 |
| ↪7 days ago | | |
| 1.28GB | | |
| vrnetlab/vr-sros | 20.10.R1 | 0ff03483e5c4 |
| ↪6 weeks ago | | |
| 994MB | | |

Inside `connect-lab/labs` directory there are a bunch of `yaml` files starting with the name of a vendor; for instance, `cisco-evc.clab.yaml`. Deploying the aforementioned `yaml` file will yield the topology with **2 routers** as depicted below.

Connection Map With 2 Routers

| Name | Interface | Name | Interface |
|-------------|-----------|-----------------------|-----------|
| router-west | eth1 | router-east | eth1 |
| router-west | eth2 | subscriber-north-west | eth1 |
| router-west | eth3 | subscriber-south-west | eth1 |
| router-east | eth2 | subscriber-north-east | eth1 |
| router-east | eth3 | subscriber-south-east | eth1 |

Topology With 2 Routers



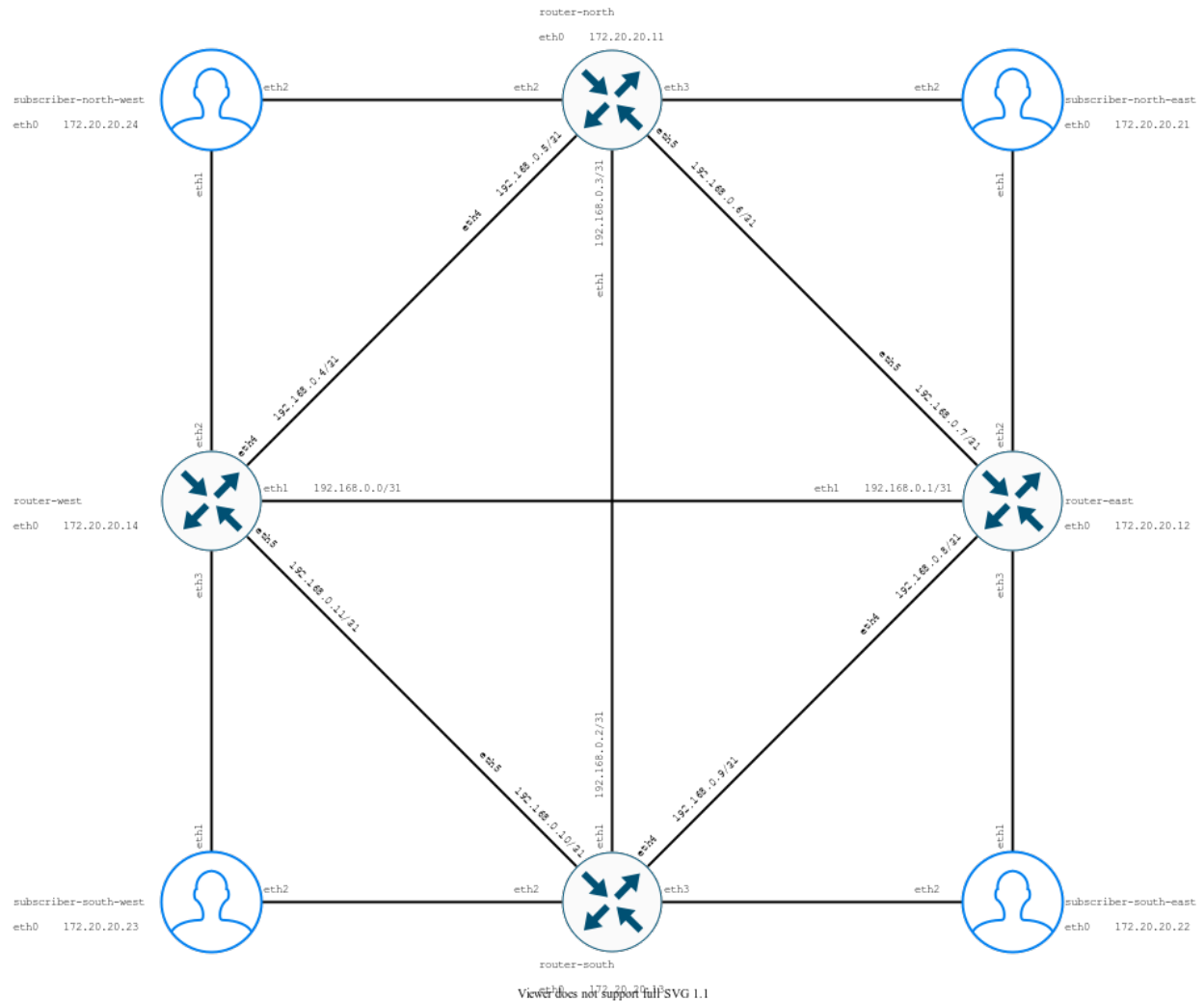
The rest of this document is based on `cisco-enc.clab.yml` deployment.

On the other hand, a **4 router** topology can be set up by deploying the files that have 4R in their name. As an example for `cisco-enc-4R.clab.yml` the result is depicted as below.

Connection Map With 4 Routers

| Name | Interface | Name | Interface |
|--------------|-----------|-----------------------|-----------|
| router-north | eth1 | router-south | eth1 |
| router-north | eth2 | subscriber-north-west | eth2 |
| router-north | eth3 | subscriber-north-east | eth2 |
| router-north | eth4 | router-west | eth4 |
| router-north | eth5 | router-east | eth5 |
| router-west | eth1 | router-east | eth1 |
| router-west | eth2 | subscriber-north-west | eth1 |
| router-west | eth3 | subscriber-south-west | eth1 |
| router-west | eth5 | router-south | eth5 |
| router-east | eth2 | subscriber-north-east | eth1 |
| router-east | eth3 | subscriber-south-east | eth1 |
| router-east | eth4 | router-south | eth4 |
| router-south | eth2 | subscriber-south-west | eth2 |
| router-south | eth3 | subscriber-south-west | eth2 |

Topology With 4 Routers

**NOTE:**

- In this configuration Inmanta service orchestrator uses port 2222 of the VM for remote access.
- Router's initial configuration files are in the `connect-lab/labs/config/` directory. Currently non-Nokia routers require a manual copy and paste of their configuration in order to function. The procedure has been elaborated for each vendor *in this section*.

To run the LAB for the first time:

```
cd connect-lab/docs/
sudo clab deploy --topo cisco-evc.clab.yml
```

To re-run the LAB after making modifications:

```
sudo clab deploy --topo cisco-evc.clab.yml --reconfigure
```

To delete the LAB:


```
sudo clab destroy --topo cisco-evc.clab.yml
```

After a few minutes, the router containers should be shown as healthy. In our example we use Nokia routers:

```
$ docker ps
CONTAINER ID   IMAGE                                COMMAND                                CREATED
↳STATUS      PORTS
↳
↳NAMES
9472bdfd566e   vrnetlab/vr-sros:20.10.R1          "/launch.py --trace ..."          4 minutes ago
↳Up 4 minutes (healthy)   80/tcp, 443/tcp, 5000/tcp, 10000-10099/tcp, 57400/tcp, 161/
↳udp, 0.0.0.0:21022->22/tcp, :::21022->22/tcp, 0.0.0.0:21830->830/tcp, :::21830->830/
↳tcp   clab-nokia-nokia-west
f59435aa0ed6   vrnetlab/vr-sros:20.10.R1          "/launch.py --trace ..."          4 minutes ago
↳Up 4 minutes (healthy)   80/tcp, 443/tcp, 5000/tcp, 10000-10099/tcp, 57400/tcp, 161/
↳udp, 0.0.0.0:20022->22/tcp, :::20022->22/tcp, 0.0.0.0:20830->830/tcp, :::20830->830/
↳tcp   clab-nokia-nokia-east
fb201bf0beae   postgres:10                         "docker-entrypoint.s..."          4 minutes ago
↳Up 4 minutes           5432/tcp
↳
↳clab-nokia-inmanta-db
ec307468311c   inmantaci/nfv-test-api:0.6.1       "sh -c '/bin/sleep 5..."          4 minutes ago
↳Up 4 minutes           0.0.0.0:2002->8080/tcp, :::2002->8080/tcp
↳
↳clab-nokia-subscriber-south-east
448344a5ab1f   inmantaci/nfv-test-api:0.6.1       "sh -c '/bin/sleep 5..."          4 minutes ago
↳Up 4 minutes           0.0.0.0:2001->8080/tcp, :::2001->8080/tcp
↳
↳clab-nokia-subscriber-north-east
f56297c91866   inmantaci/nfv-test-api:0.6.1       "sh -c '/bin/sleep 5..."          4 minutes ago
↳Up 4 minutes           0.0.0.0:2003->8080/tcp, :::2003->8080/tcp
↳
↳clab-nokia-subscriber-north-west
de139aeef0c4   inmantaci/nfv-test-api:0.6.1       "sh -c '/bin/sleep 5..."          4 minutes ago
↳Up 4 minutes           0.0.0.0:2004->8080/tcp, :::2004->8080/tcp
↳
↳clab-nokia-subscriber-south-west
9f73ca4fd97e   service-orchestrator:4             "sh -c '/usr/bin/chm..."          4 minutes ago
↳Up 4 minutes           0.0.0.0:8888->8888/tcp, :::8888->8888/tcp, 0.0.0.0:2222->22/
↳tcp, :::2222->22/tcp
↳ clab-nokia-inmanta-server
```

Accessing Containers

To access containers individually and remotely:

| Container | Service | URL |
|-----------------------|---------|---|
| orchestrator | WEB | http://vm_ip:8888 |
| orchestrator | SSH | ssh://vm_ip:2222 |
| subscriber-north-east | API | http://vm_ip:2001 |
| subscriber-south-east | API | http://vm_ip:2002 |
| subscriber-north-west | API | http://vm_ip:2003 |
| subscriber-south-west | API | http://vm_ip:2004 |

To access containers individually on the VM:

| Container | Access |
|-----------------------|---|
| nokia-east | ssh admin@clab-nokia-nokia-east |
| nokia-west | ssh admin@clab-nokia-nokia-west |
| subscriber-north-east | docker exec -ti clab-lab-subscriber-north-east bash |
| subscriber-south-east | docker exec -ti clab-lab-subscriber-south-east bash |
| subscriber-north-west | docker exec -ti clab-lab-subscriber-north-west bash |
| subscriber-south-west | docker exec -ti clab-lab-subscriber-south-west bash |

Conclusion

This guide showed how to set up a small lab using `container lab` to test Inmanta connect module. If you are having any questions or suggestions regarding this guide, please get in touch.

1.3.2 External lab

This document describes how to use an external lab (physical or virtual) to evaluate, develop and test on Inmanta Connect. There are three steps in this process:

1. Ensure that the lab network supports Inmanta Connect
2. Installing the Inmanta Service Orchestrator
3. Load Inmanta Connect in the Inmanta Service Orchestrator

Support Inmanta Connect network

Inmanta Connect requires a number of things from your network:

1. Each device has a management interface that is reachable by the orchestrator. For each vendor the netconf interface needs to be reachable.
2. Inmanta Connect only manages the endpoints of the services. This means that it assumes that there is backbone connectivity between all routers on which endpoints are managed (PE routers). It also assumes that all the required peerings and protocols are enabled between all PEs. What this means depends on the selected network backends (LDP, EVPN, ...).

3. Document the endpoints in a *compatible inventory*. Inmanta Connect needs to know all the user connects to the network: ports (and ethernet segments), network elements (logical devices and management interface details). The quickest way to get started is using our built-in yaml inventory and pick one of files included in the built-in lab.

Install the service orchestrator and connect

Inmanta Connect is built on the Inmanta Service Orchestrator. The first step is to install this orchestrator and then load Inmanta Connect. The current version of Inmanta Connect expects version Inmanta Service Orchestrator 4. There are several options to install the orchestrator:

- On a (virtual) machine with a RHEL 7 or 8 compatible operating system and [install it from RPM](#).
- [Install it using containers](#) on docker (or podman).

It is important to correctly set the router credentials. The inventory supplies Inmanta Connect with the names of the *environment variables*. These variables are set in the environment file of the service orchestrator (The documentation explains this for an [RPM install](#) and [containers](#)

Load Inmanta Connect

The final step is to load Inmanta Connect on the service orchestrator. There are mainly two methods to perform this:

- Customize the test suite and run it against the lab. The test suite will deploy Inmanta Connect with all your customizations.
- Customize the inmanta connect template project and load it in an environment.

Inmanta Connect is a solution that allows you to fully automate the deployment and management of connectivity services. We recommend and encourages to create a lab environment to test and validate Inmanta Connect and the customization you have made on top of it. There are several options:

- Use the [built-in virtual lab](#). There are labs for each vendor. These labs also include the Inmanta Service Orchestrator on which Inmanta Connect was built.
- Use an [external physical](#) or [virtual lab](#) that already exists.

1.4 Inventory

Connect requires one or more inventories to function. The purpose of this document is to describe what the inventory is used for, which integrations are available and how to add additional integrations.

1.4.1 Purpose

The purpose of the inventory integration can be threefold:

1. A resource inventory to resolve a provided UNI reference to one or more ports in the network. This results in the port, the device, router ip, management ip, credentials, ... Without this information Inmanta Connect will not be able to orchestrate any services.
2. Allocate identifiers for resources such as VLANs, service identifiers, ... By default Inmanta Connect requires a range it can manage and allocate identifiers from. However, Inmanta Connect can also allocate them based on the inventory.
3. Document the orchestrated services in an external service inventory. All services are always available in the built-in service inventory of Inmanta Connect.

Resource inventory

Inmanta Connect uses the MEF reference model for requesting new services such as carrier ethernet services. This reference model uses the concept of a UNI (User Network Interface) reference. A UNI is an abstract concept to model the attachment of a customer (user) to the provider network. There can be a one-to-one mapping between a UNI and a port on a router or it could also model a LAG implemented using multi-homing in an EVPN network.

Inmanta Connect expects a reference to a UNI that it can resolve in a resource inventory to one or more ports in the network. This reference is a URI that points to the corresponding resource in the resource inventory. Example of a such a reference are:

- UNI 123-852-456 in the built-in inventory of connect: `inmanta:123-852-456`
- UNI 1234 in Netbox: `netbox:1234`
- UNI with name `customer1` in a TMF639 compliant inventory: `tmf639://192.0.2.42/api/resource/?@type=typeUNI&name=customer1`

The inventory integration will provide Inmanta Connect with parameters such as:

- The name of the port
- The name of the router
- The mgmt ip
- ...

Identifier allocation

Inmanta Connect requires unique identifiers to work correctly. For example a circuit id, which needs to be unique within the network. The default mode of operation is assigning a range of identifiers to Inmanta Connect. Connect allocates an available identifier from the provided range. See the configuration settings for more details. By default only service ids are allocated from a range. Identifiers such as VLANs have to be provided when the a service instance is created.

This behaviour can be customized by developing a custom allocator. This allocator can then use an external inventory to find an available VLAN, service id or other identifier.

Service inventory

Inmanta Connect can integrate with external service inventories to document the created services. Inmanta Connect currently does not offer out of the box integrations. The orchestration model behind Inmanta Connect has been built to easily add service documentation.

1.4.2 Inventory integration

Inmanta Connect can integrate with any inventory that has an external API. The default settings enable the built-in resource inventory. Other inventories can be enabled or integrated. Multiple inventories can be enabled at the same time.

Built-in resource inventory

The built-in resource inventory loads the inventory from a yaml file. Inmanta Connect also contains the topologies to work with the connect reference labs. The inventory contains only the data required for Connect to orchestrate new services.

The inventory models `Device`, `Network Element (NE)` and `User Network Interface (UNI)` resources in a yaml file. The default location for this file is `/files/connect-inventory.yaml`, in the inmanta project. The inventory path can be modified using the `main.inventory_path` attribute of the configuration.

The snippet below shows an example of the contents in an inventory file:

```

devices:
  dev-2:
    mgmt_ip: 172.20.20.12
    mgmt_port: 830
    vendor: Nokia
    model: 7750 SR
    os: TiMos
    version: "20.10"
    username_env: NETCONF_DEVICE_USER
    password_env: NETCONF_DEVICE_PASSWORD
  dev-4:
    mgmt_ip: 172.20.20.14
    mgmt_port: 830
    vendor: Nokia
    model: 7750 SR
    os: TiMos
    version: "20.10"
    username_env: NETCONF_DEVICE_USER
    password_env: NETCONF_DEVICE_PASSWORD

ne:
  - id: "2"
    name: router-east
    router_ip: 10.255.255.2
    device: ref#devices.dev-2
  - id: "4"
    name: router-west
    router_ip: 10.255.255.4
    device: ref#devices.dev-4

port:
  - id: "3"
    name: 1/1/c2/1
    network_element: ref#ne[id=2]
  - id: "4"
    name: 1/1/c3/1
    network_element: ref#ne[id=2]
  - id: "7"
    name: 1/1/c2/1
    network_element: ref#ne[id=4]
  - id: "8"
    name: 1/1/c3/1

```

(continues on next page)

```

network_element: ref#ne[id=4]

uni:
- id: inmanta:456-852-789
  ports:
  - ref#port[id=3]
- id: inmanta:456-985-752
  ports:
  - ref#port[id=4]
- id: inmanta:123-852-456
  ports:
  - ref#port[id=8]
- id: inmanta:456-852-798
  ports:
  - ref#port[id=7]

```

Built-in resources

Below are the built-in resources that the built-in inventory models:

- **devices**: The network devices in the inventory. Its attributes are:
 - **mgmt_ip**: the ip address Inmanta Connect should use to access the netconf/yang interface of this device
 - **mgmt_port**: the port the netconf/yang interface is running
 - **vendor**: the name of the vendor. Inmanta Connect uses this to determine what vendor a device is. Currently Cisco, Nokia and Juniper are supported.
 - **model**: the router model to which is begin deployed. For example: ASR9001 or MX480. This value is currently not used and is only there for reference.
 - **os**: the operating system running on the device. For example Junos, SR-OS, ... This value is currently not used and is only there for reference.
 - **version**: the version of the os running on the device. This is used for version specific work arounds.
 - **username_env**: The environment variable on the orchestrator to fetch the device username from.
 - **password_env**: The environment variable on the orchestrator to fetch the device password from.
- **ne**: The network elements in the inventory. This represents the logical router running on the **device**. Its attributes are:
 - **id**: A reference for the network element that is used internally in the inventory
 - **name**: The name of the network element.
 - **router_ip**: The loopback ip of the router. This ip is used for example for LDP configuration.
 - **device**: A reference to a device **device** resource.
- **port**: A port on a network element. A customer can connect to one or more ports (see **uni**):
 - **id**: An identifier used in the inventory to reference the port.
 - **name**: The name of the port on a device. The orchestrator uses this value to reference the port.
 - **network_element**: A reference to the network element this port belongs to.
- **uni**: A user network interface models how a user is connected to the network. Its attributes are:

- `id`: An identifier used to reference this UNI. The built-in resolver will use this identifier when used as a uni reference. The resolver is used when the URI scheme is `inmanta:`.
- `ports`: A list of references to the ports that make up this UNI.
- `[lag_id]`: An optional attribute (combined with `es_id`). When multi-homing is used with EVPN, it is important to configure ethernet segments lag association. When this is enabled in the configuration, this value is used.
- `[es_name]`: An optional attribute (combined with `es_id`). When multi-homing is used with EVPN, it is important to configure ethernet segments. When this is enabled in the configuration, this value is used.
- `[es_id]`: See `es_name`

The inventory supports internal references to other resources. These references start with `ref#`. After that a dictpath expression is used.

1.5 Testing

This guide includes:

- *Testing*
 - *Constructing A LAB File*
 - *LAB Topology Constraints*
 - *Vendor specific tests*
 - * *Example*
 - * *Adding New Markers*
 - *Topology File Structure*
 - *Assembling A Basic Test*
 - *Running Tests*
 - *Running Mypy Type Check*

By following the steps [here](#), you should have a `virtual` environment and a clone of `connect` module locally.

1.5.1 Constructing A LAB File

All the tests are parametrized so that multiple developers can execute them on the same LAB without interfering with each other. The user specific data is presented in the LAB file, under `tests/labs/user/` directory.

To select a LAB when running the tests, use the `--lab` option, specifying in argument the name of the file, except the `.yaml` extension. All LAB files should end with the `.yaml` extension.

Example

Here is an example of the LAB file, you can copy it in this directory, and modify the values.

```
environment:
  id: 8404deea-3621-4bca-9076-6a612115f810 # Change this
  name: ci-1 # Change this
  project:
    id: 3fa85f64-5717-4562-b3fc-2c963f66afa6
```

(continues on next page)

(continued from previous page)

```
name: connect

prefix: ci-1 # Change this

service_id_range:
  start: 10000 # Change this
  stop: 10099 # Change this

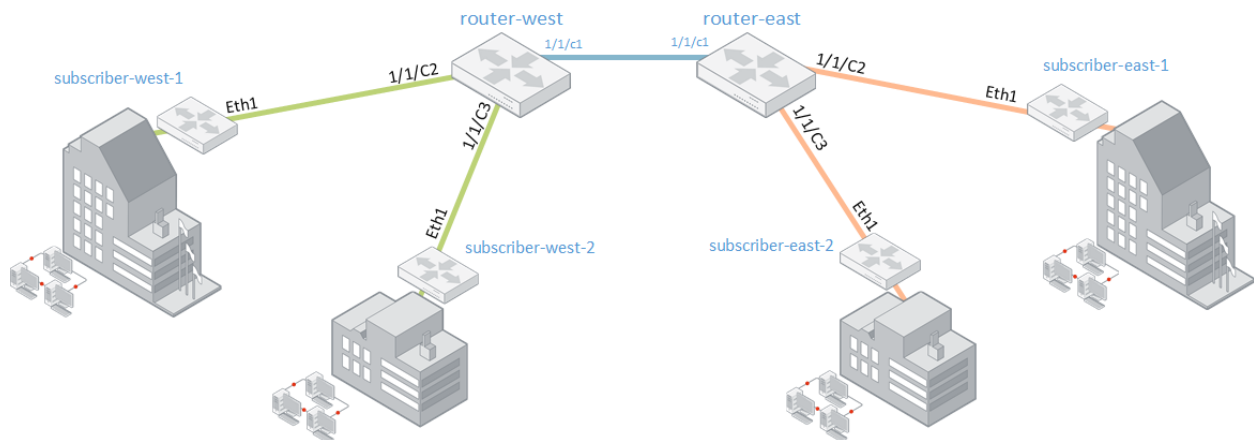
vlan_id_range:
  start: 600 # Change this
  stop: 649 # Change this
```

1.5.2 LAB Topology Constraints

In the test cases some assumptions are made on the structure of the lab. Those assumptions are checked when the tests using the LAB manager are setup. Here are the assumptions:

1. The topology file contains at least four subscribers, named: subscriber-1, subscriber-2, subscriber-3 and subscriber-4.
2. Side A (subscriber-1 and subscriber-2) are connected to the same provider router.
3. Side B (subscriber-3 and subscriber-4) are connected to the same provider router.
4. Side A and B do not have a direct connection
5. All routers in the LAB have the same vendor.

Here is a visual representation of the topology:



1.5.3 Vendor specific tests

Some tests can only be executed with a specific topology file because they test the behavior of the module against a specific vendor. To avoid running this test case in unsupported configurations, pytest's markers can be used.

Example

This is already the case for two test cases:

- tests/test_model/test_carrier_ethernet_evc_nokia/test_LDP
- tests/test_model/test_carrier_ethernet_evc_nokia/test_EVPN

These verify that the generated yang config contains the desired information. In order to mark those tests as suitable only for Nokia topology, they get marked with the marker `nokia_only`:

```
@pytest.mark.nokia_only
def test_LDP(project: Project, lab_config: LabConfig) -> None:
```

This marker can be reused for any test that requires a Nokia topology file.

By default those tests will be skipped. If you are running the tests with a topology file containing only Nokia devices, you can either set the environment variable `INMANTA_CONNECT_LAB_KIND` or the cli parameter `--connect-lab-kind` to *Nokia*. In that case, and in that case only, those two tests will be executed.

Adding New Markers

To add some additional markers, i.e. when other vendor-specific tests are added apart from Nokia, there are three things to do:

1. Extend the `pytest_configure` function in `tests/conftest.py` to support the additional marker.
2. Extend the `pytest_runtest_setup` function in `tests/conftest.py` to skip the test if the marker is detected but the vendor is not the one we want.
3. Add the marker on top of the test you wish to limit to the specific vendor.

1.5.4 Topology File Structure

The topology file should contain four main entries:

- **routers:** A dictionary containing the routers of the LAB.
- **subscribers:** A dictionary containing the subscribers of the LAB.
- **links:** A list containing the links between different element (routers and subscribers) in the LAB.
- **inventory:** A dictionary containing a valid inventory that can be given to the module for this LAB.

Here is an example of the topology file:

```
routers:
  nokia-east:
    vendor: Nokia
    mgmt_address: 192.168.2.33
    netconf_port: 20830
    ssh_port: 20022
```

(continues on next page)

(continued from previous page)

```
mgmt_username_env: NETCONF_DEVICE_USER
mgmt_password_env: NETCONF_DEVICE_PASSWORD
```

nokia-west:

```
vendor: Nokia
mgmt_address: 192.168.2.33
netconf_port: 21830
ssh_port: 21022
mgmt_username_env: NETCONF_DEVICE_USER
mgmt_password_env: NETCONF_DEVICE_PASSWORD
```

subscribers:**subscriber-1:**

```
mgmt_address: 192.168.2.33
api_port: 2001
uni: inmanta:456-852-789
```

subscriber-2:

```
mgmt_address: 192.168.2.33
api_port: 2002
uni: inmanta:456-985-752
```

subscriber-3:

```
mgmt_address: 192.168.2.33
api_port: 2003
uni: inmanta:123-852-456
```

subscriber-4:

```
mgmt_address: 192.168.2.33
api_port: 2004
uni: inmanta:652-784-963
```

links:

- endpoints:
 - type: router
 - device: nokia-east
 - interface: eth1
 - type: subscriber
 - device: subscriber-1
 - interface: eth1
 - namespace: east1
- endpoints:
 - type: router
 - device: nokia-east
 - interface: eth2
 - type: subscriber
 - device: subscriber-2
 - interface: eth1
 - namespace: east1
- endpoints:
 - type: router
 - device: nokia-west
 - interface: eth1
 - type: subscriber
 - device: subscriber-3

(continues on next page)

(continued from previous page)

```
    interface: eth1
    namespace: west1
- endpoints:
- type: router
  device: nokia-west
  interface: eth2
- type: subscriber
  device: subscriber-4
  interface: eth1
  namespace: west1

inventory:
devices:
  dev-1:
    mgmt_ip: 172.20.20.31
    mgmt_port: 22
    vendor: Nokia
    model: 7750 SR
    os: TiMos
    version: "20.10"
    username_env: NETCONF_DEVICE_USER
    password_env: NETCONF_DEVICE_PASSWORD
  dev-2:
    mgmt_ip: 172.20.20.21
    mgmt_port: 22
    vendor: Nokia
    model: 7750 SR
    os: TiMos
    version: "20.10"
    username_env: NETCONF_DEVICE_USER
    password_env: NETCONF_DEVICE_PASSWORD

ne:
- id: "1"
  name: Nokia_west
  router_ip: 10.255.255.2
  device: ref#devices.dev-1
- id: "2"
  name: Nokia_east
  router_ip: 10.255.255.1
  device: ref#devices.dev-2

uni:
- id: inmanta:456-985-752
  port: 1/1/c3/1
  network_element: ref#ne[id=1]
- id: inmanta:456-852-789
  port: 1/1/c2/1
  network_element: ref#ne[id=1]
- id: inmanta:652-784-963
  port: 1/1/c3/1
  network_element: ref#ne[id=2]
```

(continues on next page)

(continued from previous page)

```
- id: inmanta:123-852-456
  port: 1/1/c2/1
  network_element: ref#ne[id=2]
```

Here are some notes about parametrized values:

- Some values in the file can be parametrized using the prefix `opt:` prefix.
- The remaining part of the value will be extracted and resolved.
- The remaining value should match the key of one of the `TestParameter` object set in the tests `conftest.py` file.

For instance, if you set `opt:inn_con_lab_mgmt_ip` as a value for `routers.nokia-east.mgmt_address`, when the topology file is loaded, this value will be replaced by the value passed to the `--lab-mgmt-ip` argument or set in `INMANTA_CONNECT_LAB_MGMT_IP`. Please note that CLI arguments have more priority than `env` variables.

1.5.5 Assembling A Basic Test

We also need to make a temporary directory for the tests and utilize it by environment variables:

```
mkdir /tmp/env
export INMANTA_TEST_ENV=/tmp/env
```

The most basic type of testing is a compile test which can be conducted like:

```
import uuid
import pytest
from pytest_inmanta.plugin import Project

def test_basics(project: Project) -> None:
    """
    Simple example of instantiation of a CarrierEthernetEvc entity. We just try to
    ↪ compile it.
    """
    model = f"""
import connect
import connect::infra as infra
import connect::mock

connect::CarrierEthernetEvc(
  identifier="my-evc-001",
  connectionType="POINT_TO_POINT",
  ext_networkBackend=connect::Ext_networkBackend(type="EVPN"),
  evcEndpoints=[
    connect::CarrierEthernetEvcEndPoint(
      identifier="my-evc-ep-1",
      egressBandwidthProfilePerEndPoint=[
        connect::EgressBwpFlow(
          cir=1,
        ),
        connect::EgressBwpFlow(
          cir=2,
```

(continues on next page)

(continued from previous page)

```

    ),
  ],
  evcEndPointMap=connect::VlanIdListOrUntag(
    type="LIST",
    vlanIdList=[
      connect::VlanId(vlanId=200),
    ],
  ),
  ↪carrierEthernetSubscriberUni=connect::CarrierEthernetSubscriberUniRef(
    href="inmanta:456-985-752",
  ),
  _uni=infra::UserNetworkInterface(
    id="inmanta:456-985-752",
    port="1/1/c3/1",
    network_element=infra::NetworkElement(
      id="1",
      name="ne-1",
      router_ip="1.2.3.4",
      device=infra::Device(
        mgmt_ip="1.2.3.4",
        mgmt_port=22,
        vendor="Nokia",
        model="7750 SR",
        os="TiMos",
        version="20.10",
        username_env="NETCONF_DEVICE_USER",
        password_env="NETCONF_DEVICE_PASSWORD",
      )
    )
  ),
),
connect::CarrierEthernetEvcEndPoint(
  identifier="my-evc-ep-2",
  egressBandwidthProfilePerEndPoint=[
    connect::EgressBwpFlow(
      cir=1,
    ),
    connect::EgressBwpFlow(
      cir=2,
    ),
  ],
  evcEndPointMap=connect::VlanIdListOrUntag(
    type="LIST",
    vlanIdList=[
      connect::VlanId(vlanId=201),
    ],
  ),
  ↪carrierEthernetSubscriberUni=connect::CarrierEthernetSubscriberUniRef(
    href="inmanta:652-784-963",
  ),

```

(continues on next page)

(continued from previous page)

```

        _uni=infra::UserNetworkInterface(
            id="inmanta:652-784-963",
            port="1/1/c3/1",
            network_element=infra::NetworkElement(
                id="2",
                name="ne-2",
                router_ip="1.2.3.5",
                device=infra::Device(
                    mgmt_ip="1.2.3.5",
                    mgmt_port=22,
                    vendor="Nokia",
                    model="7750 SR",
                    os="TiMos",
                    version="20.10",
                    username_env="NETCONF_DEVICE_USER",
                    password_env="NETCONF_DEVICE_PASSWORD",
                )
            )
        ),
    ],
    purge_resources=false,
    instance_id="{uuid.uuid4()}",
)
"""

project.compile(model, no_dedent=False)

```

1.5.6 Running Tests

```

# There are multiple env files with the names of vendors. These match a specific
↳ topology file
# in tests/labs/topology
# Sourcing one of this files will also run the common-env.sh script. This script loads
↳ some
# environment variables common to any developer of the module.
# If you wish to load some env variables automatically, this script looks for the
↳ existence
# of personal_env.sh file and executes it, if found. You can add all private envs you
↳ require there.
source <topology>-env.sh
export INMANTA_CONNECT_LAB_USER="" # The name of the lab file you created (if not added
↳ to personal_env.sh)
export INMANTA_LSM_ENVIRONMENT="" # An environment on mentioned orchestrator (if not
↳ added to personal_env.sh)

pytest tests \
    --use-module-in-place \
    --log-cli-level=debug \
    tests/

```

1.5.7 Running Mypy Type Check

The use of Mypy is encouraged since it makes the code easier to read and debug should a problem occur. Mypy is an optional static type checker for Python that aims to combine the benefits of dynamic or "duck" typing and static typing.

```
# Check typing in the plugins
make mypy-plugins

# Check typing in the tests
make mypy-tests

# Check typing in both plugins and tests
make mypy
```

To see if you improved the typing, do make `mypy-save` before you make changes and make `mypy-diff` afterwards

1.6 Configuration and Customization

Connect can be configuration and customized using configuration files and templates:

- Topologies
- Hardware/vendors
- Allocation Policies
- UNI Resolvers
- Naming conventions
- Etc.

The connect module will load its configuration from a file named `connect-config.yaml`, which should be located in the `files` folder of the project using this module. The project's structure is depicted as below:

```
connect-project/
├── files
│   └── connect-config.yaml
├── libs
│   └── connect/
├── main.cf
└── project.yml
```

1.6.1 Configuration options

Here below is the list of configuration options that the module can take. We also provide the json schema for the content of the configuration file, as well as an example of configuration file content. The example configuration file only uses all the default values, it is equivalent to providing an empty config file.

```
{
  "title": "ConnectConfig",
  "description": "This is the root of the configuration.\nEach entry in the_
```

(continues on next page)

(continued from previous page)

```

↪configuration file should correspond to one of the embedded Config objects.",
  "type": "object",
  "properties": {
    "main": {
      "$ref": "#/$defs/MainConfig"
    },
    "carrier_ethernet_evc": {
      "$ref": "#/$defs/CarrierEthernetEvcConfig"
    },
    "direct_internet_access": {
      "$ref": "#/$defs/DirectInternetAccess"
    },
    "label_distribution_protocol": {
      "$ref": "#/$defs/LabelDistributionProtocolConfig"
    },
    "nokia": {
      "$ref": "#/$defs/NokiaConfig"
    },
    "cisco": {
      "$ref": "#/$defs/CiscoConfig"
    },
    "juniper": {
      "$ref": "#/$defs/JuniperConfig"
    },
    "eth_cfm": {
      "$ref": "#/$defs/EthernetCFMConfig"
    },
    "uni": {
      "$ref": "#/$defs/UniConfig"
    }
  },
  "$defs": {
    "BackendConfig": {
      "description": "This is the CarrierEthernetEvc's backend config. It determines_
↪which backend should be\navailable and which one should be the default.",
      "properties": {
        "default": {
          "enum": [
            "LDP",
            "EVPN"
          ],
          "type": "string",
          "default": "LDP",
          "title": "Default"
        },
        "enabled": {
          "items": {
            "enum": [
              "LDP",
              "EVPN"
            ],
            "type": "string"
          }
        }
      }
    }
  }
}

```

(continues on next page)

(continued from previous page)

```

    },
    "title": "Enabled",
    "type": "array"
  }
},
"BackendConfig",
"type": "object"
},
"CarrierEthernetEvcConfig": {
  "description": "This is the CarrierEthernetEvc config.",
  "properties": {
    "service_id": {
      "$ref": "#/$defs/ServiceIdConfig"
    },
    "backend": {
      "$ref": "#/$defs/BackendConfig"
    },
    "multi_homing": {
      "$ref": "#/$defs/MultiHomingConfig"
    },
    "mtu": {
      "default": 1514,
      "maximum": 65536,
      "minimum": 576,
      "title": "Mtu",
      "type": "integer"
    }
  },
  "title": "CarrierEthernetEvcConfig",
  "type": "object"
},
"CiscoConfig": {
  "description": "This is the cisco specific config",
  "properties": {
    "netconf_retry_count": {
      "default": 5,
      "exclusiveMinimum": 0,
      "title": "Netconf Retry Count",
      "type": "integer"
    },
    "netconf_retry_interval": {
      "default": 5,
      "exclusiveMinimum": 0,
      "title": "Netconf Retry Interval",
      "type": "integer"
    },
    "xconnect": {
      "$ref": "#/$defs/CiscoXconnectConfig"
    }
  },
  "title": "CiscoConfig",
  "type": "object"
}

```

(continues on next page)

(continued from previous page)

```

    },
    "CiscoQosInputConfig": {
      "description": "Configuration related to input policy maps configuration on ↵
↵cisco routers.",
      "properties": {
        "policy_class": {
          "anyOf": [
            {
              "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
              "type": "string"
            },
            {
              "type": "string"
            }
          ],
          "default": "class-default",
          "title": "Policy Class"
        },
        "policy_name": {
          "anyOf": [
            {
              "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
              "type": "string"
            },
            {
              "type": "string"
            }
          ],
          "default": "template://connect/cisco/input_policy_name.j2",
          "title": "Policy Name"
        },
        "traffic_class": {
          "anyOf": [
            {
              "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
              "type": "string"
            },
            {
              "type": "integer"
            }
          ],
          "default": 2,
          "title": "Traffic Class"
        }
      },
      "title": "CiscoQosInputConfig",
      "type": "object"
    },
    "CiscoQosOutputConfig": {
      "description": "Configuration related to output policy maps configuration on ↵
↵cisco routers.",
      "properties": {

```

(continues on next page)

(continued from previous page)

```

    "policy_class": {
      "anyOf": [
        {
          "pattern": "^template:\\\\(\\w+)?(\\w+)+$",
          "type": "string"
        },
        {
          "type": "string"
        }
      ],
      "default": "class-default",
      "title": "Policy Class"
    },
    "policy_name": {
      "anyOf": [
        {
          "pattern": "^template:\\\\(\\w+)?(\\w+)+$",
          "type": "string"
        },
        {
          "type": "string"
        }
      ],
      "default": "template://connect/cisco/output_policy_name.j2",
      "title": "Policy Name"
    }
  },
  "title": "CiscoQosOutputConfig",
  "type": "object"
},
"CiscoXconnectConfig": {
  "description": "This is configuration specific to xconnect services on cisco\
↪n routers.",
  "properties": {
    "service_name": {
      "anyOf": [
        {
          "pattern": "^template:\\\\(\\w+)?(\\w+)+$",
          "type": "string"
        },
        {
          "type": "string"
        }
      ],
      "default": "template://connect/cisco/xconnect_service_name.j2",
      "title": "Service Name"
    },
    "group_name": {
      "default": "evpn-vpws",
      "title": "Group Name",
      "type": "string"
    }
  },

```

(continues on next page)

(continued from previous page)

```

    "l2_interface_name": {
      "default": "template://connect/cisco/l2_interface_name.j2",
      "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)$",
      "title": "L2 Interface Name",
      "type": "string"
    },
    "l2_interface_description": {
      "default": "template://connect/cisco/l2_interface_description.j2",
      "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)$",
      "title": "L2 Interface Description",
      "type": "string"
    },
    "qos_input": {
      "$ref": "#/$defs/CiscoQosInputConfig"
    },
    "qos_output": {
      "$ref": "#/$defs/CiscoQosOutputConfig"
    }
  },
  "title": "CiscoXconnectConfig",
  "type": "object"
},
"DirectInternetAccess": {
  "properties": {
    "p2p_networks": {
      "items": {
        "format": "ipv4network",
        "type": "string"
      },
      "title": "P2P Networks",
      "type": "array"
    },
    "link_local_p2p": {
      "default": false,
      "title": "Link Local P2P",
      "type": "boolean"
    },
    "vrf_name": {
      "default": "template://connect/dia/vrf_name.j2",
      "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)$",
      "title": "Vrf Name",
      "type": "string"
    }
  },
  "title": "DirectInternetAccess",
  "type": "object"
},
"EthernetCFMConfig": {
  "description": "This is the ethernet cfm quality of service config",
  "properties": {
    "enabled": {
      "default": false,

```

(continues on next page)

(continued from previous page)

```

        "title": "Enabled",
        "type": "boolean"
    },
    "level": {
        "default": 0,
        "maximum": 7,
        "minimum": 0,
        "title": "Level",
        "type": "integer"
    },
    "domain_name": {
        "default": "Network_Domain",
        "title": "Domain Name",
        "type": "string"
    },
    "mep_direction": {
        "default": "up",
        "enum": [
            "up",
            "down"
        ],
        "title": "Mep Direction",
        "type": "string"
    },
    "mep_id": {
        "default": "template://connect/eth_cfm/mep_id.j2",
        "pattern": "^template:\\\\\\\\\\(^[^\\\\]+)?(\\\\\\\\[^\\\\]+)+$",
        "title": "Mep Id",
        "type": "string"
    },
    "mep_mac_address": {
        "default": "template://connect/eth_cfm/mep_mac_address.j2",
        "pattern": "^template:\\\\\\\\\\(^[^\\\\]+)?(\\\\\\\\[^\\\\]+)+$",
        "title": "Mep Mac Address",
        "type": "string"
    },
    "check_interval": {
        "default": "1s",
        "enum": [
            "10ms",
            "100ms",
            "1s",
            "10s",
            "60s",
            "600s"
        ],
        "title": "Check Interval",
        "type": "string"
    },
    "md_admin_name": {
        "anyOf": [
            {

```

(continues on next page)

(continued from previous page)

```

        "pattern": "^template:\\\\(\\[[^\\]]+)?(\\[[^\\]]+)+$",
        "type": "string"
    },
    {
        "type": "string"
    }
],
"default": "64512",
"title": "Md Admin Name"
},
"eth_tag": {
    "default": "template://connect/eth_cfm/eth_tag.j2",
    "pattern": "^template:\\\\(\\[[^\\]]+)?(\\[[^\\]]+)+$",
    "title": "Eth Tag",
    "type": "string"
},
"eth_test": {
    "default": true,
    "title": "Eth Test",
    "type": "boolean"
}
},
"title": "EthernetCFMConfig",
"type": "object"
},
"InventorySet": {
    "description": "Container for different inventory types",
    "properties": {
        "netbox": {
            "$ref": "#/$defs/NetboxConfig"
        }
    }
},
"title": "InventorySet",
"type": "object"
},
"JuniperConfig": {
    "description": "This is the juniper specific config",
    "properties": {
        "netconf_retry_count": {
            "default": 5,
            "exclusiveMinimum": 0,
            "title": "Netconf Retry Count",
            "type": "integer"
        },
        "netconf_retry_interval": {
            "default": 5,
            "exclusiveMinimum": 0,
            "title": "Netconf Retry Interval",
            "type": "integer"
        }
    }
},
"title": "JuniperConfig",

```

(continues on next page)

(continued from previous page)

```

    "type": "object"
  },
  "LabelDistributionProtocolConfig": {
    "description": "This is the LDP config.",
    "properties": {},
    "title": "LabelDistributionProtocolConfig",
    "type": "object"
  },
  "LagIdConfig": {
    "description": "This is the Uni service's lag id config. It determines the
↪range of ids that can be used for the allocation of the lag.",
    "properties": {
      "start": {
        "default": 1,
        "title": "Start",
        "type": "integer"
      },
      "end": {
        "default": 64,
        "title": "End",
        "type": "integer"
      }
    },
    "title": "LagIdConfig",
    "type": "object"
  },
  "MainConfig": {
    "description": "This is the main config, it contains elements transient to the
↪whole module usage.",
    "properties": {
      "inventory_path": {
        "anyOf": [
          {
            "pattern": "^source:\\\\(?:[^\\"/]+)?(?:[^\\"/]+)+$",
            "type": "string"
          },
          {
            "pattern": "^file:\\\\(?:[^\\"/]+)+?$",
            "type": "string"
          }
        ],
        "default": "source:///connect-inventory.yaml",
        "title": "Inventory Path"
      },
      "inventory": {
        "$ref": "#/defs/InventorySet"
      },
      "allowed_uni_resolvers": {
        "items": {
          "type": "string"
        },
        "title": "Allowed Uni Resolvers",

```

(continues on next page)

(continued from previous page)

```

        "type": "array"
    }
},
"title": "MainConfig",
"type": "object"
},
"MultiHomingConfig": {
  "description": "This is the MultiHoming config.",
  "properties": {
    "enabled": {
      "default": false,
      "title": "Enabled",
      "type": "boolean"
    },
    "lag_id": {
      "default": "template://connect/lag_id.j2",
      "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)$",
      "title": "Lag Id",
      "type": "string"
    },
    "es_id": {
      "default": "template://connect/es_id.j2",
      "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)$",
      "title": "Es Id",
      "type": "string"
    },
    "es_name": {
      "default": "template://connect/es_name.j2",
      "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)$",
      "title": "Es Name",
      "type": "string"
    },
    "create_esi": {
      "default": true,
      "title": "Create Esi",
      "type": "boolean"
    }
  }
},
"title": "MultiHomingConfig",
"type": "object"
},
"NetboxConfig": {
  "properties": {
    "url": {
      "anyOf": [
        {
          "pattern": "^https?://[^\/?]+?(:\\d+)?/?$",
          "type": "string"
        },
        {
          "type": "null"
        }
      ]
    }
  }
}

```

(continues on next page)

(continued from previous page)

```
    ],
    "default": null,
    "title": "Url"
  },
  "token": {
    "anyOf": [
      {
        "type": "string"
      },
      {
        "type": "null"
      }
    ],
    "default": null,
    "title": "Token"
  },
  "token_env": {
    "anyOf": [
      {
        "type": "string"
      },
      {
        "type": "null"
      }
    ],
    "default": null,
    "title": "Token Env"
  },
  "agent_autostart": {
    "default": false,
    "title": "Agent Autostart",
    "type": "boolean"
  }
},
"title": "NetboxConfig",
"type": "object"
},
"NokiaBGPCConfig": {
  "description": "This is Nokia bgp config",
  "properties": {
    "resolution": {
      "default": "any",
      "enum": [
        "none",
        "filter",
        "any"
      ],
    },
    "title": "Resolution",
    "type": "string"
  },
  "tagging": {
    "default": true,
```

(continues on next page)

(continued from previous page)

```

        "title": "Tagging",
        "type": "boolean"
    },
    "filters": {
        "$ref": "#/$defs/NokiaBGPFiltersConfig"
    },
    "local_attachment_name": {
        "anyOf": [
            {
                "pattern": "^template:\\\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
                "type": "string"
            },
            {
                "type": "string"
            }
        ],
        "default": "template://connect/nokia/local_attachment_name.j2",
        "title": "Local Attachment Name"
    },
    "remote_attachment_name": {
        "anyOf": [
            {
                "pattern": "^template:\\\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
                "type": "string"
            },
            {
                "type": "string"
            }
        ],
        "default": "template://connect/nokia/remote_attachment_name.j2",
        "title": "Remote Attachment Name"
    }
},
"title": "NokiaBGPCConfig",
"type": "object"
},
"NokiaBGPFiltersConfig": {
    "properties": {
        "ldp": {
            "default": false,
            "title": "Ldp",
            "type": "boolean"
        },
        "rsvp": {
            "default": false,
            "title": "Rsvp",
            "type": "boolean"
        },
        "sr_isis": {
            "default": false,
            "title": "Sr Isis",
            "type": "boolean"
        }
    }
}

```

(continues on next page)

(continued from previous page)

```

    },
    "sr_te": {
      "default": false,
      "title": "Sr Te",
      "type": "boolean"
    }
  },
  "title": "NokiaBGPFiltersConfig",
  "type": "object"
},
"NokiaConfig": {
  "description": "This is the Nokia config.",
  "properties": {
    "netconf_retry_count": {
      "default": 5,
      "exclusiveMinimum": 0,
      "title": "Netconf Retry Count",
      "type": "integer"
    },
    "netconf_retry_interval": {
      "default": 5,
      "exclusiveMinimum": 0,
      "title": "Netconf Retry Interval",
      "type": "integer"
    },
    "customer_id": {
      "anyOf": [
        {
          "pattern": "^template:\\\\\\\\\\([^\\\\\\\\]+)?(\\\\\\\\[^\\\\\\\\]+)+$",
          "type": "string"
        },
        {
          "type": "string"
        }
      ],
      "default": "1",
      "title": "Customer Id"
    },
    "epipe": {
      "$ref": "#/$defs/NokiaEpipeConfig"
    },
    "vpls": {
      "$ref": "#/$defs/NokiaVplsConfig"
    },
    "uni": {
      "$ref": "#/$defs/NokiaUniConfig"
    }
  },
  "title": "NokiaConfig",
  "type": "object"
},
"NokiaEVPNConfig": {

```

(continues on next page)

(continued from previous page)

```

    "description": "This is Nokia EVPN config",
    "properties": {
      "bgp": {
        "$ref": "#/$defs/NokiaBGPCConfig"
      }
    },
    "title": "NokiaEVPNConfig",
    "type": "object"
  },
  "NokiaEpipeConfig": {
    "description": "This is the Nokia epipe config",
    "properties": {
      "enable_sdp": {
        "default": true,
        "title": "Enable Sdp",
        "type": "boolean"
      },
      "service_name": {
        "anyOf": [
          {
            "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
            "type": "string"
          },
          {
            "type": "string"
          }
        ],
        "default": "template://connect/nokia/epipe_service_name.j2",
        "title": "Service Name"
      },
      "qos_ingress": {
        "$ref": "#/$defs/NokiaEpipeQosIngressConfig"
      },
      "qos_egress": {
        "$ref": "#/$defs/NokiaEpipeQosEgressConfig"
      },
      "evpn": {
        "$ref": "#/$defs/NokiaEVPNConfig"
      },
      "sap_id": {
        "default": "template://connect/nokia/sap_id.j2",
        "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
        "title": "Sap Id",
        "type": "string"
      },
      "sap_description": {
        "default": "template://connect/nokia/sap_description.j2",
        "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
        "title": "Sap Description",
        "type": "string"
      },
      "esi_port_id": {

```

(continues on next page)

(continued from previous page)

```

        "default": "template://connect/nokia/port_esi.j2",
        "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
        "title": "Esi Port Id",
        "type": "string"
    },
},
"title": "NokiaEpipeConfig",
"type": "object"
},
"NokiaEpipeQosEgressConfig": {
  "properties": {
    "cbs": {
      "anyOf": [
        {
          "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
          "type": "string"
        },
        {
          "type": "string"
        },
        {
          "type": "integer"
        },
        {
          "type": "null"
        }
      ],
      "default": "auto",
      "title": "Cbs"
    },
    "mbs": {
      "anyOf": [
        {
          "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
          "type": "string"
        },
        {
          "type": "string"
        },
        {
          "type": "integer"
        },
        {
          "type": "null"
        }
      ],
      "default": "auto",
      "title": "Mbs"
    },
    "pir": {
      "anyOf": [
        {

```

(continues on next page)

(continued from previous page)

```

        "pattern": "^template:\\\\(\\w+)?(\\w+)$",
        "type": "string"
    },
    {
        "type": "string"
    },
    {
        "type": "integer"
    },
    {
        "type": "null"
    }
],
"default": "max",
"title": "Pir"
},
"policy_name": {
    "anyOf": [
        {
            "pattern": "^template:\\\\(\\w+)?(\\w+)$",
            "type": "string"
        },
        {
            "type": "string"
        }
    ],
    "default": "default-policy",
    "title": "Policy Name"
},
"queue_id": {
    "anyOf": [
        {
            "pattern": "^template:\\\\(\\w+)?(\\w+)$",
            "type": "string"
        },
        {
            "type": "integer"
        }
    ],
    "default": 3,
    "title": "Queue Id"
}
},
"title": "NokiaEpipeQosEgressConfig",
"type": "object"
},
"NokiaEpipeQosIngressConfig": {
    "properties": {
        "cbs": {
            "anyOf": [
                {
                    "pattern": "^template:\\\\(\\w+)?(\\w+)$",

```

(continues on next page)

(continued from previous page)

```

        "type": "string"
    },
    {
        "type": "string"
    },
    {
        "type": "integer"
    },
    {
        "type": "null"
    }
],
"default": "auto",
"title": "Cbs"
},
"mbs": {
    "anyOf": [
        {
            "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\\\\\[^\\\\]+)+$",
            "type": "string"
        },
        {
            "type": "string"
        },
        {
            "type": "integer"
        },
        {
            "type": "null"
        }
    ],
    "default": "auto",
    "title": "Mbs"
},
"pir": {
    "anyOf": [
        {
            "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\\\\\[^\\\\]+)+$",
            "type": "string"
        },
        {
            "type": "string"
        },
        {
            "type": "integer"
        },
        {
            "type": "null"
        }
    ],
    "default": "max",
    "title": "Pir"

```

(continues on next page)

(continued from previous page)

```

    },
    "policy_name": {
      "anyOf": [
        {
          "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
          "type": "string"
        },
        {
          "type": "string"
        }
      ],
      "default": "default-policy",
      "title": "Policy Name"
    },
    "policer_id": {
      "anyOf": [
        {
          "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
          "type": "string"
        },
        {
          "type": "integer"
        }
      ],
      "default": 3,
      "title": "Policer Id"
    }
  },
  "title": "NokiaEpipeQosIngressConfig",
  "type": "object"
},
"NokiaUniConfig": {
  "description": "This is the Nokia user network interface config.",
  "properties": {
    "netconf_retry_count": {
      "default": 5,
      "exclusiveMinimum": 0,
      "title": "Netconf Retry Count",
      "type": "integer"
    },
    "netconf_retry_interval": {
      "default": 5,
      "exclusiveMinimum": 0,
      "title": "Netconf Retry Interval",
      "type": "integer"
    },
    "port_mode": {
      "anyOf": [
        {
          "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
          "type": "string"
        },
        {

```

(continues on next page)

(continued from previous page)

```

        {
            "enum": [
                "access",
                "network",
                "hybrid"
            ],
            "type": "string"
        }
    ],
    "default": "template://connect/nokia/port_mode.j2",
    "title": "Port Mode"
}
},
"title": "NokiaUniConfig",
"type": "object"
},
"NokiaVplsConfig": {
    "description": "This is the Nokia vpls config",
    "properties": {
        "service_name": {
            "anyOf": [
                {
                    "pattern": "^template:\\\\\\\\\\([^\\\\]+)?(\\\\\\\\[^\\\\]+)+$",
                    "type": "string"
                },
                {
                    "type": "string"
                }
            ],
            "default": "template://connect/nokia/vpls_service_name.j2",
            "title": "Service Name"
        },
        "qos_ingress": {
            "$ref": "#/$defs/NokiaVplsQosIngressConfig"
        },
        "qos_egress": {
            "$ref": "#/$defs/NokiaVplsQosEgressConfig"
        },
        "evpn": {
            "$ref": "#/$defs/NokiaEVPNConfig"
        },
        "fdb": {
            "$ref": "#/$defs/NokiaVplsFdbConfig"
        },
        "proxy_arp": {
            "$ref": "#/$defs/NokiaVplsProxyArpConfig"
        },
        "sap_id": {
            "default": "template://connect/nokia/sap_id.j2",
            "pattern": "^template:\\\\\\\\\\([^\\\\]+)?(\\\\\\\\[^\\\\]+)+$",
            "title": "Sap Id",
            "type": "string"
        }
    }
}

```

(continues on next page)

(continued from previous page)

```

    },
    "sap_description": {
      "default": "template://connect/nokia/sap_description.j2",
      "pattern": "^template:\\\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
      "title": "Sap Description",
      "type": "string"
    },
    },
    "sap_dist_cpu_protection": {
      "anyOf": [
        {
          "pattern": "^template:\\\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
          "type": "string"
        },
        {
          "type": "string"
        },
        {
          "type": "null"
        }
      ],
      "default": null,
      "title": "Sap Dist Cpu Protection"
    },
    },
    "title": "NokiaVplsConfig",
    "type": "object"
  },
  "NokiaVplsFdbConfig": {
    "description": "Forwarding database (FDB) config",
    "properties": {
      "table_high_wmark": {
        "anyOf": [
          {
            "pattern": "^template:\\\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
            "type": "string"
          },
          {
            "type": "integer"
          }
        ],
        "default": 80,
        "title": "Table High Wmark"
      },
      "table_low_wmark": {
        "anyOf": [
          {
            "pattern": "^template:\\\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
            "type": "string"
          },
          {
            "type": "integer"
          }
        ]
      }
    }
  }
}

```

(continues on next page)

(continued from previous page)

```

    ],
    "default": 60,
    "title": "Table Low Wmark"
  },
  "table_size": {
    "anyOf": [
      {
        "pattern": "^template:\\\\(\\w+)?(\\w+)+$",
        "type": "string"
      },
      {
        "type": "integer"
      }
    ],
    "default": 2500,
    "title": "Table Size"
  }
},
"title": "NokiaVplsFdbConfig",
"type": "object"
},
"NokiaVplsProxyArpConfig": {
  "description": "ARP Proxy config",
  "properties": {
    "age_time": {
      "anyOf": [
        {
          "pattern": "^template:\\\\(\\w+)?(\\w+)+$",
          "type": "string"
        },
        {
          "type": "string"
        }
      ],
      "default": "900",
      "title": "Age Time"
    },
    "send_refresh": {
      "anyOf": [
        {
          "pattern": "^template:\\\\(\\w+)?(\\w+)+$",
          "type": "string"
        },
        {
          "type": "string"
        }
      ],
      "default": "300",
      "title": "Send Refresh"
    },
    "table_size": {
      "anyOf": [

```

(continues on next page)

(continued from previous page)

```

        {
          "pattern": "^template:\\\\(\\w+)?(\\w+)+$",
          "type": "string"
        },
        {
          "type": "integer"
        }
      ],
      "default": 2500,
      "title": "Table Size"
    }
  },
  "title": "NokiaVplsProxyArpConfig",
  "type": "object"
},
"NokiaVplsQosEgressConfig": {
  "description": "VPLS QoS Egress config",
  "properties": {
    "cbs": {
      "anyOf": [
        {
          "pattern": "^template:\\\\(\\w+)?(\\w+)+$",
          "type": "string"
        },
        {
          "type": "string"
        },
        {
          "type": "integer"
        },
        {
          "type": "null"
        }
      ],
      "default": "auto",
      "title": "Cbs"
    },
    "mbs": {
      "anyOf": [
        {
          "pattern": "^template:\\\\(\\w+)?(\\w+)+$",
          "type": "string"
        },
        {
          "type": "string"
        },
        {
          "type": "integer"
        },
        {
          "type": "null"
        }
      ]
    }
  }
}

```

(continues on next page)

(continued from previous page)

```

    ],
    "default": "auto",
    "title": "Mbs"
  },
  "pir": {
    "anyOf": [
      {
        "pattern": "^template:\\\\(\\w+)?(\\w+)$",
        "type": "string"
      },
      {
        "type": "string"
      },
      {
        "type": "integer"
      },
      {
        "type": "null"
      }
    ],
    "default": "max",
    "title": "Pir"
  },
  "policy_name": {
    "anyOf": [
      {
        "pattern": "^template:\\\\(\\w+)?(\\w+)$",
        "type": "string"
      },
      {
        "type": "string"
      }
    ],
    "default": "default-policy",
    "title": "Policy Name"
  },
  "queue_id": {
    "anyOf": [
      {
        "pattern": "^template:\\\\(\\w+)?(\\w+)$",
        "type": "string"
      },
      {
        "type": "integer"
      }
    ],
    "default": 1,
    "title": "Queue Id"
  }
},
"title": "NokiaVplsQosEgressConfig",
"type": "object"

```

(continues on next page)

(continued from previous page)

```
},
"NokiaVplsQosIngressConfig": {
  "description": "VPLS QoS Ingress config",
  "properties": {
    "cbs": {
      "anyOf": [
        {
          "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
          "type": "string"
        },
        {
          "type": "string"
        },
        {
          "type": "integer"
        },
        {
          "type": "null"
        }
      ],
      "default": "auto",
      "title": "Cbs"
    },
    "mbs": {
      "anyOf": [
        {
          "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
          "type": "string"
        },
        {
          "type": "string"
        },
        {
          "type": "integer"
        },
        {
          "type": "null"
        }
      ],
      "default": "auto",
      "title": "Mbs"
    },
    "pir": {
      "anyOf": [
        {
          "pattern": "^template:\\\\\\\\([^\\\\]+)?(\\\\[^\\\\]+)+$",
          "type": "string"
        },
        {
          "type": "string"
        }
      ]
    }
  }
}
```

(continues on next page)

(continued from previous page)

```

        "type": "integer"
    },
    {
        "type": "null"
    }
],
"default": "max",
"title": "Pir"
},
"policy_name": {
    "anyOf": [
        {
            "pattern": "^template:\\\\(\\w+)?(\\w+)$",
            "type": "string"
        },
        {
            "type": "string"
        }
    ],
    "default": "default-policy",
    "title": "Policy Name"
},
"queue_id": {
    "anyOf": [
        {
            "pattern": "^template:\\\\(\\w+)?(\\w+)$",
            "type": "string"
        },
        {
            "type": "integer"
        }
    ],
    "default": 1,
    "title": "Queue Id"
}
},
"title": "NokiaVplsQosIngressConfig",
"type": "object"
},
"ServiceIdConfig": {
    "description": "This is the CarrierEthernetEvc's service id config. It
↪ determines from which range and using which strategy, service ids allocated to
↪ CarrierEthernetEvc's should be picked.",
    "properties": {
        "start": {
            "default": 1,
            "exclusiveMinimum": 0,
            "title": "Start",
            "type": "integer"
        },
        "end": {
            "default": 100,

```

(continues on next page)

(continued from previous page)

```

        "exclusiveMinimum": 0,
        "title": "End",
        "type": "integer"
    },
    "strategy": {
        "default": "next",
        "enum": [
            "next",
            "any"
        ],
        "title": "Strategy",
        "type": "string"
    }
},
"title": "ServiceIdConfig",
"type": "object"
},
"UniConfig": {
    "properties": {
        "lag_id_range": {
            "$ref": "#/$defs/LagIdConfig"
        }
    },
    "title": "UniConfig",
    "type": "object"
}
}
}
}

```

carrier_ethernet_evc:

```

backend:
  default: LDP
  enabled:
    - LDP
    - EVPN
mtu: 1514
multi_homing:
  create_esi: true
  enabled: false
  es_id: template://connect/es_id.j2
  es_name: template://connect/es_name.j2
  lag_id: template://connect/lag_id.j2
service_id:
  end: 100
  start: 1
  strategy: next
cisco:
  netconf_retry_count: 5
  netconf_retry_interval: 5
xconnect:
  group_name: evpn-vpws
  l2_interface_description: template://connect/cisco/l2_interface_description.j2

```

(continues on next page)

(continued from previous page)

```

l2_interface_name: template://connect/cisco/l2_interface_name.j2
qos_input:
  policy_class: class-default
  policy_name: template://connect/cisco/input_policy_name.j2
  traffic_class: 2
qos_output:
  policy_class: class-default
  policy_name: template://connect/cisco/output_policy_name.j2
  service_name: template://connect/cisco/xconnect_service_name.j2
direct_internet_access:
  link_local_p2p: false
  p2p_networks: []
  vrf_name: template://connect/dia/vrf_name.j2
eth_cfm:
  check_interval: 1s
  domain_name: Network_Domain
  enabled: false
  eth_tag: template://connect/eth_cfm/eth_tag.j2
  eth_test: true
  level: 0
  md_admin_name: '64512'
  mep_direction: up
  mep_id: template://connect/eth_cfm/mep_id.j2
  mep_mac_address: template://connect/eth_cfm/mep_mac_address.j2
juniper:
  netconf_retry_count: 5
  netconf_retry_interval: 5
label_distribution_protocol: {}
main:
  allowed_uni_resolvers:
  - inmanta
  inventory:
    netbox:
      agent_autostart: false
      token: null
      token_env: null
      url: null
    inventory_path: source:///connect-inventory.yaml
nokia:
  customer_id: '1'
  epipe:
    enable_sdp: true
    esi_port_id: template://connect/nokia/port_esi.j2
  evpn:
    bgp:
      filters:
        ldp: false
        rsvp: false
        sr_isis: false
        sr_te: false
    local_attachment_name: template://connect/nokia/local_attachment_name.j2
    remote_attachment_name: template://connect/nokia/remote_attachment_name.j2

```

(continues on next page)

(continued from previous page)

```
    resolution: any
    tagging: true
qos_egress:
  cbs: auto
  mbs: auto
  pir: max
  policy_name: default-policy
  queue_id: 3
qos_ingress:
  cbs: auto
  mbs: auto
  pir: max
  policer_id: 3
  policy_name: default-policy
sap_description: template://connect/nokia/sap_description.j2
sap_id: template://connect/nokia/sap_id.j2
service_name: template://connect/nokia/epipe_service_name.j2
netconf_retry_count: 5
netconf_retry_interval: 5
uni:
  netconf_retry_count: 5
  netconf_retry_interval: 5
  port_mode: template://connect/nokia/port_mode.j2
vpls:
  evpn:
    bgp:
      filters:
        ldp: false
        rsvp: false
        sr_isis: false
        sr_te: false
      local_attachment_name: template://connect/nokia/local_attachment_name.j2
      remote_attachment_name: template://connect/nokia/remote_attachment_name.j2
      resolution: any
      tagging: true
    fdb:
      table_high_wmark: 80
      table_low_wmark: 60
      table_size: 2500
    proxy_arp:
      age_time: '900'
      send_refresh: '300'
      table_size: 2500
  qos_egress:
    cbs: auto
    mbs: auto
    pir: max
    policy_name: default-policy
    queue_id: 1
  qos_ingress:
    cbs: auto
    mbs: auto
```

(continues on next page)

(continued from previous page)

```

    pir: max
    policy_name: default-policy
    queue_id: 1
    sap_description: template://connect/nokia/sap_description.j2
    sap_dist_cpu_protection: null
    sap_id: template://connect/nokia/sap_id.j2
    service_name: template://connect/nokia/vpls_service_name.j2
uni:
  lag_id_range:
    end: 64
    start: 1

```

uni**Type**

object

Default**uni.lag_id_range**

This is the Uni service's lag id config. It determines the range of ids that can be used for the allocation of the lag.

Type

object

Default**uni.lag_id_range.end****Type**

integer

Default

64

uni.lag_id_range.start**Type**

integer

Default

1

eth_cfm

This is the ethernet cfm quality of service config

Type

object

Default**eth_cfm.eth_test****Type**

boolean

Default

True

eth_cfm.eth_tag

Type

string

Default

template://connect/eth_cfm/eth_tag.j2

eth_cfm.md_admin_name

Type

any

Default

64512

eth_cfm.check_interval

Type

string

Default

1s

eth_cfm.mep_mac_address

Type

string

Default

template://connect/eth_cfm/mep_mac_address.j2

eth_cfm.mep_id

Type

string

Default

template://connect/eth_cfm/mep_id.j2

eth_cfm.mep_direction

Type

string

Default

up

eth_cfm.domain_name

Type

string

Default

Network_Domain

eth_cfm.level

Type

integer

Default

0

eth_cfm.enabled

Type
boolean

Default
False

juniper

This is the juniper specific config

Type
object

Default

juniper.netconf_retry_interval

Type
integer

Default
5

juniper.netconf_retry_count

Type
integer

Default
5

cisco

This is the cisco specific config

Type
object

Default

cisco.xconnect

This is configuration specific to xconnect services on cisco routers.

Type
object

Default

cisco.xconnect.qos_output

Configuration related to output policy maps configuration on cisco routers.

Type
object

Default

cisco.xconnect.qos_output.policy_name

Type
any

Default
template://connect/cisco/output_policy_name.j2

cisco.xconnect.qos_output.policy_class

Type
any

Default
class-default

cisco.xconnect.qos_input

Configuration related to input policy maps configuration on cisco routers.

Type
object

Default

cisco.xconnect.qos_input.traffic_class

Type
any

Default
2

cisco.xconnect.qos_input.policy_name

Type
any

Default
template://connect/cisco/input_policy_name.j2

cisco.xconnect.qos_input.policy_class

Type
any

Default
class-default

cisco.xconnect.l2_interface_description

Type
string

Default
template://connect/cisco/l2_interface_description.j2

cisco.xconnect.l2_interface_name

Type
string

Default
template://connect/cisco/l2_interface_name.j2

cisco.xconnect.group_name

Type
string

Default
evpn-vpws

cisco.xconnect.service_name**Type**

any

Default

template://connect/cisco/xconnect_service_name.j2

cisco.netconf_retry_interval**Type**

integer

Default

5

cisco.netconf_retry_count**Type**

integer

Default

5

nokia

This is the Nokia config.

Type

object

Default**nokia.uni**

This is the Nokia user network interface config.

Type

object

Default**nokia.uni.port_mode****Type**

any

Default

template://connect/nokia/port_mode.j2

nokia.uni.netconf_retry_interval**Type**

integer

Default

5

nokia.uni.netconf_retry_count**Type**

integer

Default

5

nokia.vpls

This is the Nokia vpls config

Type
object

Default

nokia.vpls.sap_dist_cpu_protection

Type
any

Default
None

nokia.vpls.sap_description

Type
string

Default
template://connect/nokia/sap_description.j2

nokia.vpls.sap_id

Type
string

Default
template://connect/nokia/sap_id.j2

nokia.vpls.proxy_arp

ARP Proxy config

Type
object

Default

nokia.vpls.proxy_arp.table_size

Type
any

Default
2500

nokia.vpls.proxy_arp.send_refresh

Type
any

Default
300

nokia.vpls.proxy_arp.age_time

Type
any

Default
900

nokia.vpls.fdb

Forwarding database (FDB) config

Type
object

Default

nokia.vpls.fdb.table_size

Type
any

Default
2500

nokia.vpls.fdb.table_low_wmark

Type
any

Default
60

nokia.vpls.fdb.table_high_wmark

Type
any

Default
80

nokia.vpls.evpn

This is Nokia EVPN config

Type
object

Default

nokia.vpls.evpn.bgp

This is Nokia bgp config

Type
object

Default

nokia.vpls.evpn.bgp.remote_attachment_name

Type
any

Default
template://connect/nokia/remote_attachment_name.j2

nokia.vpls.evpn.bgp.local_attachment_name

Type
any

Default
template://connect/nokia/local_attachment_name.j2

nokia.vpls.evpn.bgp.filters

Type
object

Default

nokia.vpls.evpn.bgp.filters.sr_te

Type
boolean

Default
False

nokia.vpls.evpn.bgp.filters.sr_isis

Type
boolean

Default
False

nokia.vpls.evpn.bgp.filters.rsvp

Type
boolean

Default
False

nokia.vpls.evpn.bgp.filters.ldap

Type
boolean

Default
False

nokia.vpls.evpn.bgp.tagging

Type
boolean

Default
True

nokia.vpls.evpn.bgp.resolution

Type
string

Default
any

nokia.vpls.qos_egress

VPLS QoS Egress config

Type
object

Default

nokia.vpls.qos_egress.queue_id

Type
any

Default
1

nokia.vpls.qos_egress.policy_name

Type
any

Default
default-policy

nokia.vpls.qos_egress.pir

Type
any

Default
max

nokia.vpls.qos_egress.mbs

Type
any

Default
auto

nokia.vpls.qos_egress.cbs

Type
any

Default
auto

nokia.vpls.qos_ingress

VPLS QoS Ingress config

Type
object

Default

nokia.vpls.qos_ingress.queue_id

Type
any

Default
1

nokia.vpls.qos_ingress.policy_name

Type
any

Default
default-policy

nokia.vpls.qos_ingress.pir

Type
any
Default
max

nokia.vpls.qos_ingress.mbs

Type
any
Default
auto

nokia.vpls.qos_ingress.cbs

Type
any
Default
auto

nokia.vpls.service_name

Type
any
Default
template://connect/nokia/vpls_service_name.j2

nokia.epipe

This is the Nokia epipe config

Type
object
Default

nokia.epipe.esi_port_id

Type
string
Default
template://connect/nokia/port_esi.j2

nokia.epipe.sap_description

Type
string
Default
template://connect/nokia/sap_description.j2

nokia.epipe.sap_id

Type
string
Default
template://connect/nokia/sap_id.j2

nokia.epipe.evpn

This is Nokia EVPN config

Type

object

Default**nokia.epipe.evpn.bgp**

This is Nokia bgp config

Type

object

Default**nokia.epipe.evpn.bgp.remote_attachment_name****Type**

any

Default

template://connect/nokia/remote_attachment_name.j2

nokia.epipe.evpn.bgp.local_attachment_name**Type**

any

Default

template://connect/nokia/local_attachment_name.j2

nokia.epipe.evpn.bgp.filters**Type**

object

Default**nokia.epipe.evpn.bgp.filters.sr_te****Type**

boolean

Default

False

nokia.epipe.evpn.bgp.filters.sr_isis**Type**

boolean

Default

False

nokia.epipe.evpn.bgp.filters.rsvp**Type**

boolean

Default

False

nokia.epipe.evpn.bgp.filters.ldap

Type
boolean

Default
False

nokia.epipe.evpn.bgp.tagging

Type
boolean

Default
True

nokia.epipe.evpn.bgp.resolution

Type
string

Default
any

nokia.epipe.qos_egress

Type
object

Default

nokia.epipe.qos_egress.queue_id

Type
any

Default
3

nokia.epipe.qos_egress.policy_name

Type
any

Default
default-policy

nokia.epipe.qos_egress.pir

Type
any

Default
max

nokia.epipe.qos_egress.mbs

Type
any

Default
auto

nokia.epipe.qos_egress.cbs

Type
any

Default
auto

nokia.epipe.qos_ingress

Type
object

Default

nokia.epipe.qos_ingress.policer_id

Type
any

Default
3

nokia.epipe.qos_ingress.policy_name

Type
any

Default
default-policy

nokia.epipe.qos_ingress.pir

Type
any

Default
max

nokia.epipe.qos_ingress.mbs

Type
any

Default
auto

nokia.epipe.qos_ingress.cbs

Type
any

Default
auto

nokia.epipe.service_name

Type
any

Default
template://connect/nokia/epipe_service_name.j2

nokia.epipe.enable_sdp

Type
boolean

Default
True

nokia.customer_id

Type
any

Default
1

nokia.netconf_retry_interval

Type
integer

Default
5

nokia.netconf_retry_count

Type
integer

Default
5

label_distribution_protocol

This is the LDP config.

Type
object

Default

direct_internet_access

Type
object

Default

direct_internet_access.vrf_name

Type
string

Default
template://connect/dia/vrf_name.j2

direct_internet_access.link_local_p2p

Type
boolean

Default
False

direct_internet_access.p2p_networks

Type
array

Default

carrier_ethernet_evc

This is the CarrierEthernetEvc config.

Type
object

Default

carrier_ethernet_evc.mtu

Type
integer

Default
1514

carrier_ethernet_evc.multi_homing

This is the MultiHoming config.

Type
object

Default

carrier_ethernet_evc.multi_homing.create_es

Type
boolean

Default
True

carrier_ethernet_evc.multi_homing.es_name

Type
string

Default
template://connect/es_name.j2

carrier_ethernet_evc.multi_homing.es_id

Type
string

Default
template://connect/es_id.j2

carrier_ethernet_evc.multi_homing.lag_id

Type
string

Default
template://connect/lag_id.j2

carrier_ethernet_evc.multi_homing.enabled

Type
boolean

Default
False

carrier_ethernet_evc.backend

This is the CarrierEthernetEvc's backend config. It determines which backend should be available and which one should be the default.

Type
object

Default

carrier_ethernet_evc.backend.enabled

Type
array

Default

carrier_ethernet_evc.backend.default

Type
string

Default
LDP

carrier_ethernet_evc.service_id

This is the CarrierEthernetEvc's service id config. It determines from which range and using which strategy, service ids allocated to CarrierEthernetEves should be picked.

Type
object

Default

carrier_ethernet_evc.service_id.strategy

Type
string

Default
next

carrier_ethernet_evc.service_id.end

Type
integer

Default
100

carrier_ethernet_evc.service_id.start

Type
integer

Default
1

main

This is the main config, it contains elements transient to the whole module usage.

Type

object

Default**main.allowed_uni_resolvers****Type**

array

Default**main.inventory**

Container for different inventory types

Type

object

Default**main.inventory.netbox****Type**

object

Default**main.inventory.netbox.agent_autostart****Type**

boolean

Default

False

main.inventory.netbox.token_env**Type**

any

Default

None

main.inventory.netbox.token**Type**

any

Default

None

main.inventory.netbox.url**Type**

any

Default

None

main.inventory_path**Type**

any

Default

source:///connect-inventory.yaml

Controlling the virtual circuit identifier (vcid) with service_id

Each deployment of a CarrierEthernetEvc using LDP backend will need to receive an unused vcid. This id is then used for configurations such as the Epipe and sdp id. This value is not passed on via the API, the model will receive it through allocation. The allocator is given a range of values that can be used, it is aware of the values it has already allocated and based on this and the allocation strategy (next, for sequential allocation, or any, for random allocation) will pick the allocated vcid.

The user of the module can influence the behavior of the allocator with three parameters in the configuration file:

- `carrier_ethernet_evc.service_id.start`: The lowest value that can be allocated.
- `carrier_ethernet_evc.service_id.end`: The highest value that can be allocated.
- `carrier_ethernet_evc.service_id.strategy`: The strategy to use when selecting the next allocated value. Possible values are:
 - `next`: Select the first free value in the range.
 - `any`: Select any free value in the range.

Controlling the ethernet virtual interconnect (evi) with service_id

Each deployment of a CarrierEthernetEvc using EVPN backend will need to receive an unused evi. This id is then used for configs such as the Epipe and evi id. This value is not passed on via the API, the model will receive it through allocation. The allocator is given a range of values that can be used, it is aware of the values it has already allocated and based on this and the allocation strategy (next, for sequential allocation, or any, for random allocation) will pick the allocated evi.

The behavior of the allocator can be influenced similar to the *vcid*, by editing the `carrier_ethernet_evc.service_id` part of the configuration.

Generating A Default Configuration File

Running the following command will generate a default configuration file using the configuration plugin:

```
python connect/docs/configuration/new_conf.py <path_to_store_config_file>
```

For example:

```
python connect/docs/configuration/new_conf.py /home/user/connect-project/files/connect-  
↪config.yaml
```

Special values

Most of the configuration fields accept a specific primitive type while others have stricter requirements. Moreover, two additional types which are of primitive constraints are introduced:

- **InmantaTemplatePath**: A path, prefixed by `template://`. The rest of the string is a path to a file in an inmanta project or module.
 - If the path starts with `/`, the path is in the project's folder. For example, `template:///example/inventory.yaml` points to a file located in `templates/example/inventory.yaml` at the root of the project using this module.
 - If the path doesn't start with `/`, the part of the string before the first `/` is the name of the module in the files of which the inventory is located. For example, `template://example/inventory.yaml` points to a file located in `templates/inventory.yaml` at the root of the module named `example`.
- **SystemPath**: A string prefixed by `file://`. The rest of the string should be an absolute path to a file on the system where the compiler is running.

Using templates

Some configuration fields accept a path to a template located in an inmanta module or project. Those fields accept a value of type `InmantaTemplatePath`, and can also accept a simpler value of a primitive type. In the former case, the path should point to a Jinja2 template (the extension of the file doesn't matter, but the content does). This template will be rendered at compile time when calling one of the `get_config_template_value` plugins (there is one by primitive type of the inmanta language). The values used for the rendering are provided in the plugin parameters.

Example: A template has already been added to this module: `nokia/epipe_service_name.j2`.

- The template path to reach it is: `inmanta://connect/templates/nokia/epipe_service_name.j2`. This is the default value of `nokia.epipe_service_name` in the configuration.
- The template content is `epipe-{{ vcid }}`
- This template is used to generate the name of the Epipe service, from the `vcid`.
- In the model, the template is rendered and accessed like this:

```
config::get_config_template_value_as_string(
    connect::config,
    "nokia.epipe_service_name",
    instance=self,
    kwargs={
        "vcid": vc_id,
    },
),
```

The plugin takes four parameters:

- `config`: The full config, from which the element at `:param dict_path:` should be queried.
- `dict_path`: The path to the configuration value in which the template path should be found.
- `instance`: (Optional) An inmanta instance, in this case the one in which implementation we call the plugin from.
- `kwargs`: (Optional) A dict, containing key-value pairs, values can only be inmanta primitive types (no entities).

1.7 Extension

Extending the Connect module requires the following steps:

- Reading the [MEF standard](#) specification for your desired feature
- Translating the MEF specification to Inmanta model
- Extending the LSM
- Developing plugins
- Writing unit tests

There are circumstances where you need to introduce additional features to the Connect API which are not part of the MEF standard. For instance, deploying E-Line with either LDP or EVPN. In such cases, we distinctively use the `Ext_` prefix for those entities and if required, define custom data types. In other words, by following these steps we update and extend the existing LSM API, and diverge from the MEF standard.

Let's take L2 MPLS VPN as an example to extend the existing API:

- Define the required types:

```
typedef connectivity_type as string matching self in ["VPWS", "VPLS"]
"""
L2 connectivity type. VPWS or VPLS.
"""
```

- Define a new entity, representing the service/technology:

```
entity Ext_L2MPLSType extends lsm::EmbeddedEntity:
    """
    The entity represents L2 MPLS VPN technology.
    :attr type: Can be VPWS or VPLS.
    """
    connectivity_type type
end
```

Please note that the entity is prefixed with `Ext_` and is **extending** the existing `lsm::EmbeddedEntity` object. All extensions to the API have to follow the same structure.

`EmbeddedEntity` contains attributes that should be embedded into a `ServiceEntity` or another `EmbeddedEntity`.

- Define the implementations (refinement) for the entities:

```
implementation with_VPWS for Ext_L2MPLSType:
    """The specific VPWS implementation goes here"""
    std::print("VPWS selected")
end

implementation with_VPLS for Ext_L2MPLSType:
    """The specific VPLS implementation goes here"""
    std::print("VPLS selected")
end
```

In `with_VPLS` implementation, the attribute types declaration is omitted for brevity; however, the steps are the same as for the `with_VPWS` example.

- Implement the refinements:

```
implement Ext_L2MPLSType using with_VPWS when self.connectivity_type=="VPWS"

implement Ext_L2MPLSType using with_VPLS when self.connectivity_type=="VPLS"
```

The `implement` is used to construct and connect the entities, refinements and their attributes together. `when` is a compile time `if`, which gives us control over how the model should be constructed.

1.7.1 Developing Plugins

There are times when you need to perform some additional validation, modification or conversions on the input data. A very simple example would be converting the `connectivity_type` value provided by the user in lowercase to uppercase. Plugins can be used to provide the aforementioned functionalities.

Head to `__init__.py` file under the `plugins` directory and implement the solution:

```
from inmanta.plugins import plugin

@plugin
def transform_to_uppercase(word: "string?") -> "string":
    return word.upper()
```

There are a few point to note here:

- It is strongly advised to use type hinting; however, the types here are accommodated with **double quotes** since they differ from Python's built-in data types.
- The `@plugin` decorator makes the function available to Inmanta model and it can then be referenced inside your model. This decorated function (`transform_to_uppercase`) can utilize any other undecorated function defined inside the `__init__.py` file for further processing.

1.7.2 Writing Unit Tests

The most basic type of testing is a `compile` test in which an Inmanta model is fed to `project.compile()` and then compiler will run the initial assessments on our provided data/model.

```
import pytest
from pytest_inmanta.plugin import Project

def test_connectivity_type(project: Project) -> None:
    project.compile(
        """
        import connect

        backend = connect::Ext_L2MPLSType(
            type="VPWS"
        )
        """
    )
```

1.8 Frequently Asked Questions (FAQ)

This section contains the frequently asked questions to gain a better understanding of our solution and customers needs.

1. I don't see my platform or vendor in the supported *platforms* and *features* what should I do?
 - We are always developing and also offer custom-tailored developments. If your platform is not in the list, please get in touch with us.
2. Where can I read about Inmanta language?
 - In [this](#) page you will find Inmanta's language reference.
3. Where can I learn more about Inmanta Service Orchestrator?
 - Inmanta Service Orchestrator is documented [here](#).
4. As a developer, how can I get started with Inmanta and how can I set up a basic project?
 - Check out our [developer getting started guide](#)
 - Set up a base project using [this guide](#).
5. What are the available testing frameworks?
 - `pytest`
 - `pytest-inmanta`
 - `pytest-inmanta-lsm`
6. When should I escalate back to Inmanta core team upon facing an error?
 - If you are certain that your model is valid, passes the tests and the error is raised by internal Inmanta modules.

For further information, please get in touch with us.

ADDITIONAL RESOURCES

- [Inmanta User Mailinglist](#)
- [Inmanta Developer Mailinglist](#)
- [Inmanta Twitter](#)

PDF VERSION

Download: [inmanta.pdf](#)