OPEN DAYLIGHT

Netconf/YANG tutorial

ENOG15 Moscow - June 2018

Evgeny Zobnitsev (FACTOR GROUP)

Agenda

- **Orchestration problem statement**
- Netconf overview
- Restconf overview
- Yang overview
- OpenDaylight overview
- DEMO

Network configuration management today

OPEN DAYLIGHT

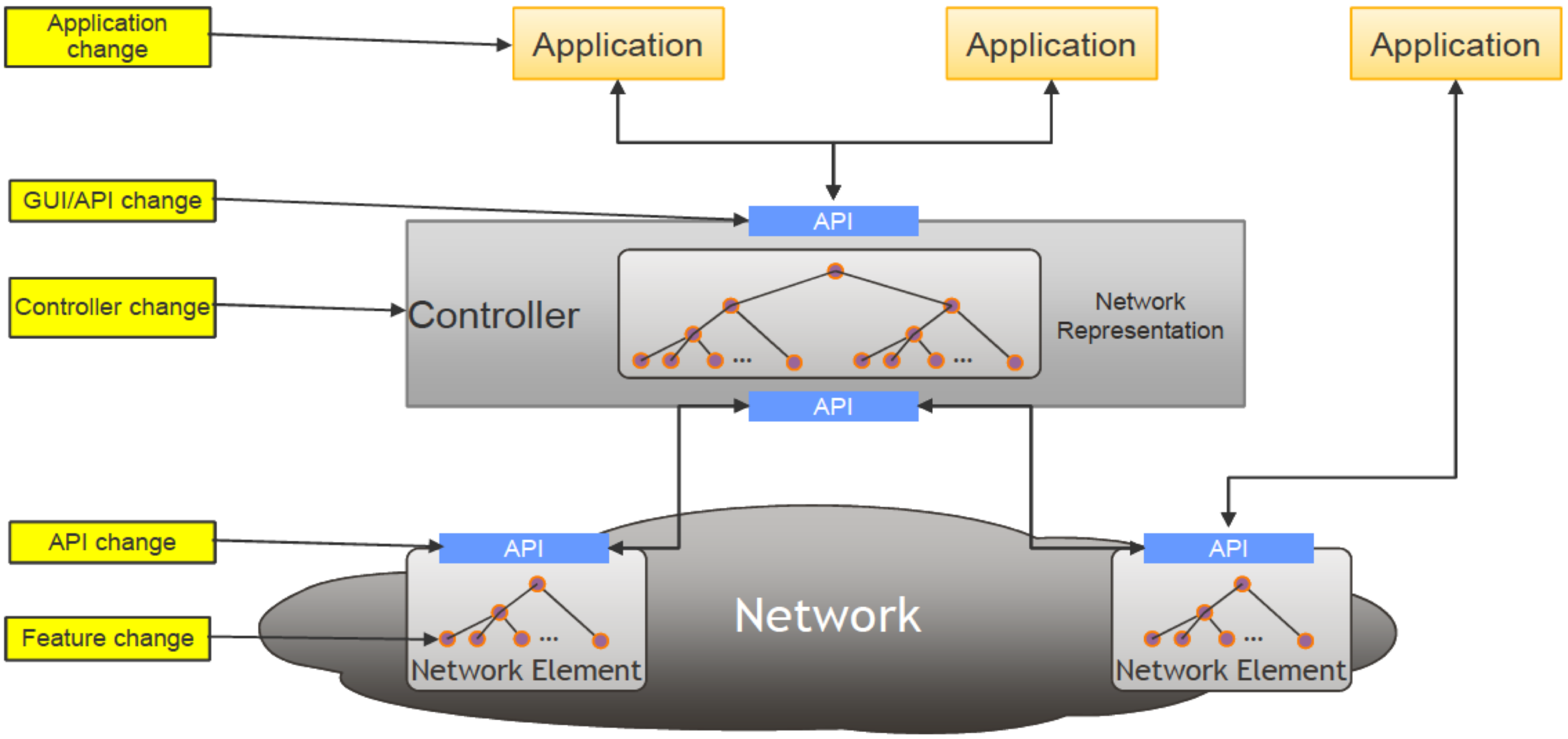
- Various proprietary CLIs of the network elements (NEs)
- Various proprietary APIs of the network elements and controller
- Manual configuration or expect scripts, as automation
- Imperative, incremental NEs configuration, lack of abstractions
- Configuration screen scraping from NEs, using parsing
  - The most bad thing, no programmatic knowledge when the CLI command finished.
  - CLI designed to work with humans, not machines.
- SNMP - has not solved this problems (see RFC3535)

Network configuration management today

OPEN DAYLIGHT

- Service lifecycle management - workflow-based
  - Workflows describes to the provisioning system, in detail, which steps to take to reach a final state.
  - Each service livecycle action has to be explicitly defined: service create, service change, service edit, service delete.

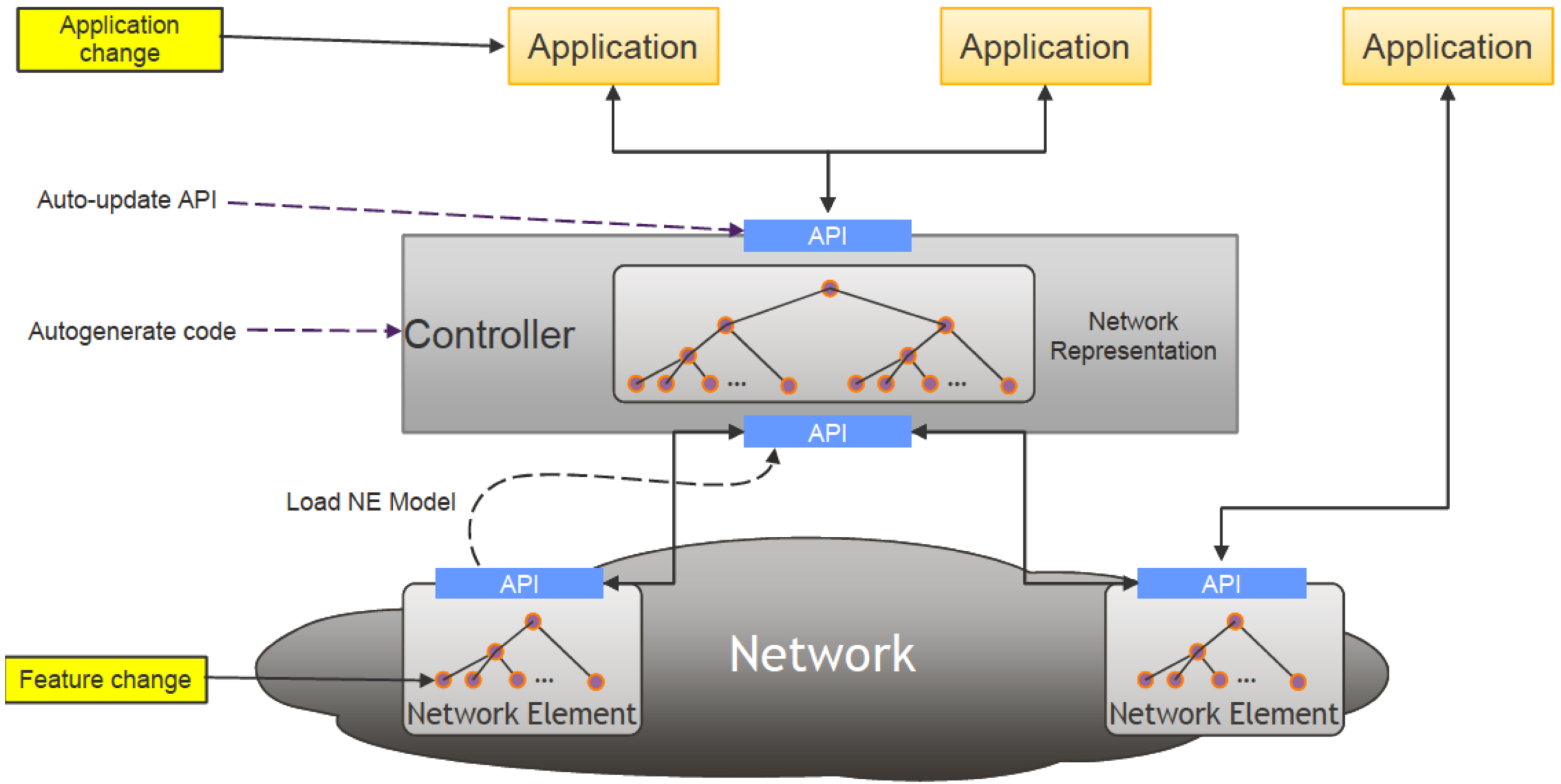
Network application life cycle today



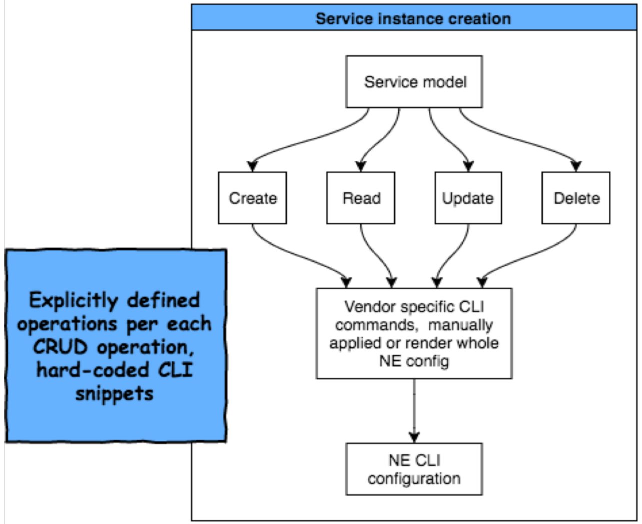
Network configuration management today

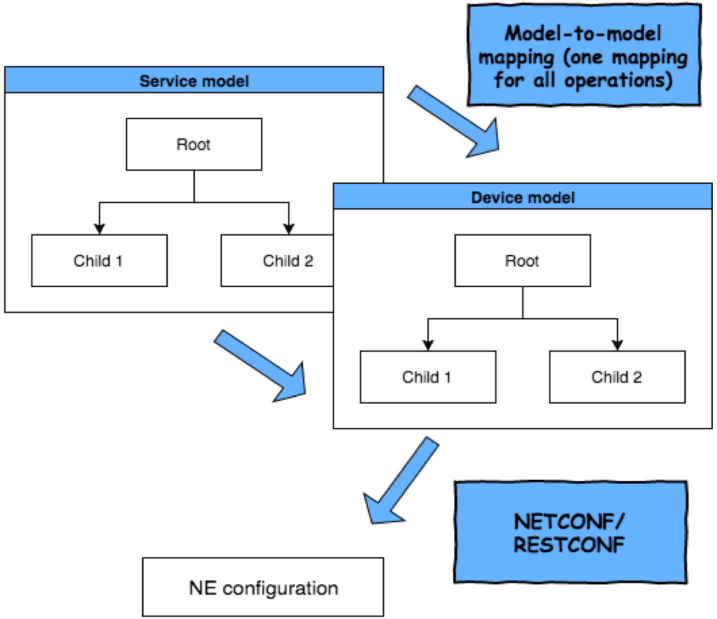
- **Model-driven topology of the network**
  - describes structure of the network.
- **Model-driven services description, auto-generating APIs**
  - describes structure of the network services, APIs.
- **Model-driven configuration of the network elements (NEs)**
  - describes structure of network element configuration, provided and supported by vendor.
- **Model-driven telemetry of the network elements (NEs)**
  - describes structure of network element supported telemetry, provided and supported by vendor.

Network application life cycle (End-to-End Model-Driven)



Comparison of Workflow-based and Model-to-model service-to-network mapping





Agenda

OPEN DAYLIGHT

- Orchestration problem statement
- **Netconf overview**
- Restconf overview
- Yang overview
- OpenDaylight overview
- DEMO

FACTOR GROUP

Key Features of NETCONF (RFC6241)

- **Domain-Specific Knowledge**
  - NETCONF was specifically developed to support network configuration
- **Support for Transactionality**
  - NETCONF support Atomicity, Consistency, Isolation, Durability (ACID) transactionality
- **Vendor Device Independence**
  - NETCONF has abilities to get capabilities just from device during <hello> exchange
- **Support of run RPCs to run actions**
  - NETCONF has abilities to run the RPC that device support
- **Device versions built-in support**
  - NETCONF has abilities to run <get-schema> RPC, to download YANG model of device configuration just from device itself

NETCONF Transport

NETCONF messages are encoded in XML

- Each message is framed by:
  - NETCONF 1.0: a character sequence]]>]]>
  - NETCONF 1.1: a line with the number of characters to read in ASCII
- NETCONF messages are encrypted by SSH,SSH provides authentication, integrity and confidentiality
- NETCONF is connected oriented using TCP
- When a NETCONF Manager connects to a NETCONF Server (NE), they send <hello> message
- The contents of the <hello> message declares which NETCONF Capabilities each party is capable of.
- By declaring support for a capability in <hello>, the manager will know which operations it can submit to the NE.
- Extensions go in separate XML namespaces, which makes it easier to build backwards and forwards compatible management applications

Agenda

- Orchestration problem statement
- Netconf overview
- **Restconf overview**
- Yang overview
- OpenDaylight overview
- DEMO

Key Features of RESTCONF (RFC 8040)

OPEN DAYLIGHT

- uniform, standardized way for Web applications to access the configuration data, state data, data-model-specific Remote Procedure Call (RPC) operations, and event notifications within a network element.
- operates on the configuration datastores defined in NETCONF and defines a set of CRUD operations that can be used to access these datastores.
- YANG language defined
- uses URI-encoded path expressions to identify the YANG data nodes unlike NETCONF, that using XPath expressions, starting from the document root to the target resource.
- operates on a hierarchy of resources are accessed via a set of URIs using syntax specified in RFC 8040

RESTCONF HTTP methods vs NETCONF

RESTCONF operation	Description	NETCONF operation
HEAD	Get without a body	<none>
OPTION	Discover which operations are supported by a data resource	<none>
GET	Retrieve data and metadata	<get>, <get-config>
POST	Create a data resource	<edit-config> (nc:operation="create")
POST	Invoke an RPC operation	Call RPC directly

RESTCONF HTTP methods vs NETCONF

RESTCONF operation	Description	NETCONF operation
PUT	Create or replace a data resource	<edit-config> (nc:operation="create/ replace"), <copy-config> (PUT on datastore)
PATCH	Create or update but not delete a data resource	<edit-config> (nc:operation depends on patch content)
DELETE	Delete a data resource	<edit-config> (nc:operation="delete")

Agenda

- Orchestration problem statement
- Netconf overview
- Restconf overview
- **Yang overview**
- OpenDaylight overview
- DEMO

Key Features of Yang (RFC6020)

OPEN DAYLIGHT

- Service & NE Data Models vs. Information Models (UML)
  - Yang is the data modeling language
- Domain-Specific Language
  - Yang was specifically developed to support network configuration
- NE configuration modeling
  - Yang is rich enough to model NE configuration (often follow the CLI)
- Service configuration modeling
  - Yang is rich enough to model services in the same language as the NE
- Network topology modeling
  - Any new device can be supported, by publishing Yang
- Vendors must create and publish Yang models of their own devices via ietf-netconf-monitoring RFC feature

New Features of Yang (RFC7950)

OPEN DAYLIGHT

YANG version 1.1 is a maintenance release of the YANG language, addressing ambiguities and defects in the original specification [RFC6020].

Key changes (not all of them):

- A server advertises support for YANG 1.1 modules by using ietf-yang-library [RFC7895] instead of listing them as capabilities in the <hello> message.
- Added a set of new XML Path Language (XPath) functions
- Allow "augment" to add conditionally mandatory nodes
- Allow imports of multiple revisions of a module
- Made the "yang-version" statement mandatory in YANG version "1.1”
- Allow "if-feature" in "bit", "enum", and "identity”
- Allow notifications to be tied to data nodes
- Added a new statement "action", which is used to define operations tied to data nodes
- Allow notifications to be tied to data nodes

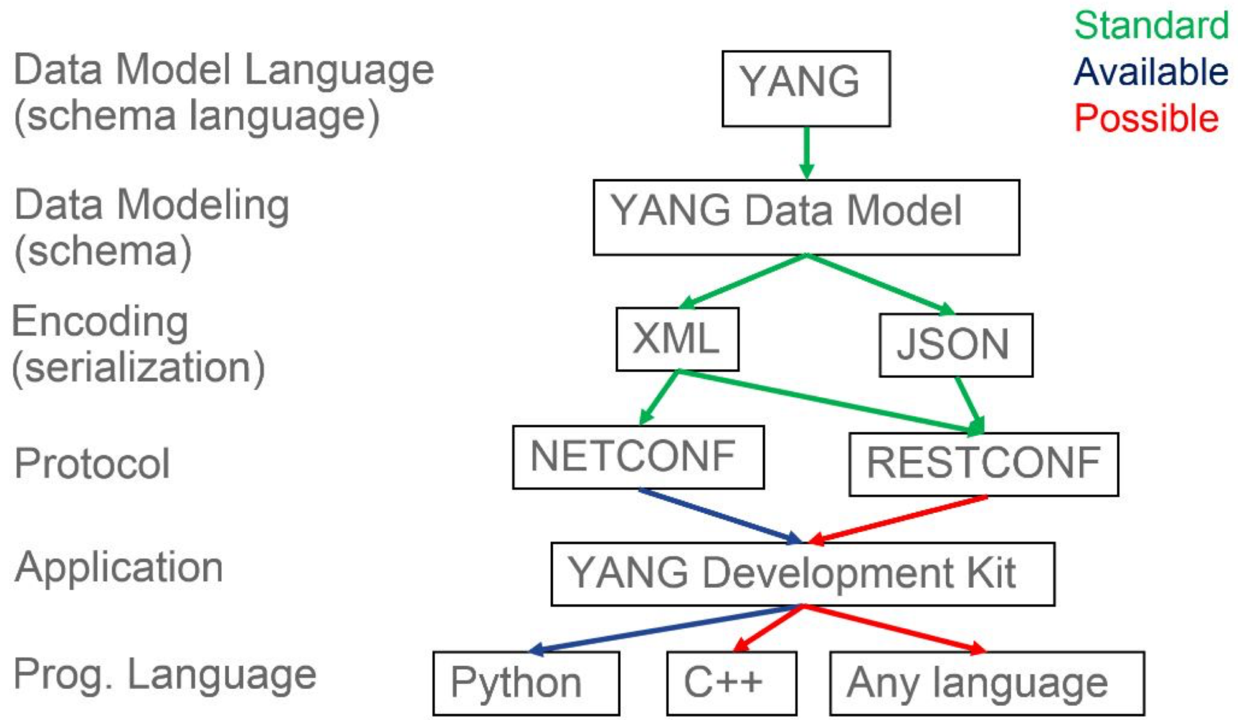
Yang module definition example

OPEN DAYLIGHT

```
module mplsl3vpn {
  namespace "http://ru/fgts/mplsl3vpn";
  prefix mplsl3vpn;
  import ietf-inet-types {
    prefix inet;
  }
    import junos {
    prefix junos;
  }
  include "ietf-bgp-l3vpn";
  organization "Factor Group";
  contact e@zobnitsev.ru;
  description "This is MPLS L3VPN
service for FGTS Lab";
  revision 2014-02-20 {
    description "Initial revision.";
  }
```

```
list endpoint {
  leaf pe-device {
    description "PE-device selection";
    mandatory true;
    type leafref {
      path "/system:devices/system:device\
          /system:name";
    }
  }
  container interfaces {
    description "Interface parameters on the PE-device";
    leaf interface {
      type leafref {
      path deref(../../pe-device)/../system:config/
junos:configuration/junos\
          :interfaces/junos:interface/junos:name;
      }
    }
}
```

Protocols interaction & standardization



* Source - Benoît Claise blog: http://www.claise.be/2016/12/yang-opensource-tools-for-data-modeling-driven-management/

Extracting Yang modules out from RFCs

OPEN DAYLIGHT

- To extract the Yang module(s) out of IETF draft for the validation, Jan Medved created the [xym.py](), the eXtracting Yang Module Python script for IETF drafts and RFCs.
- [pyang is Yang validator](), transformator and code generator, written in python, lead by Martin Bjorklund, and constantly improved by the community , especially now that the Yang 1.1 specifications [RFC 7950] have published. At the IETF, pyang is now part of the submission tool (thanks to Qin Wu and Dapeng Liu): when posting an IETF draft containing a Yang module, the Yang module language is validated.

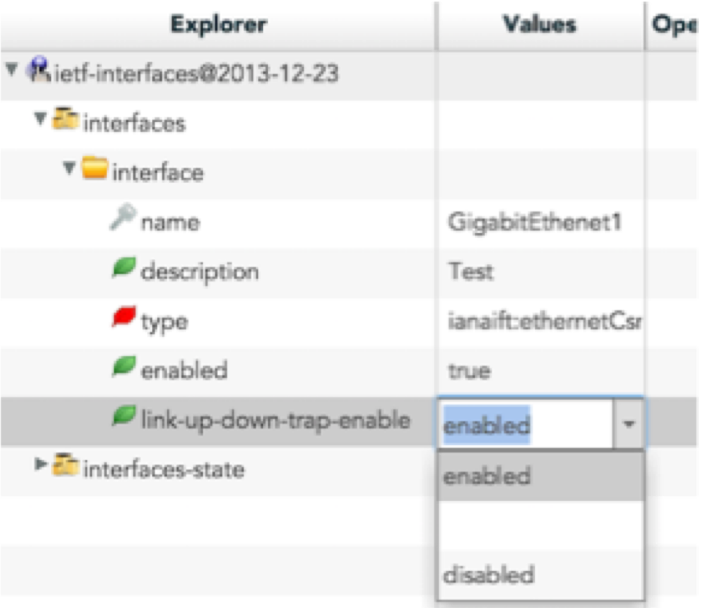
```
ez@eztp pyang -p NSO/ncs/src/ncs/yang -p NSO/run-ncs/packages/ -f
tree NSO/run-ncs/packages/mplsl3vpn/src/yang/mplsl3vpn.yang

module: mplsl3vpn
  augment /ncs:services:
    +--rw mplsl3vpn* [name]
       +--rw name                         string
       +--ro modified
       |  +--ro devices*         -> /ncs:devices/device/name
       |  +--ro services*        instance-identifier
       |  +--ro lsa-services*    instance-identifier
       +--ro directly-modified
       |  +--ro devices*         -> /ncs:devices/device/name
       |  +--ro services*        instance-identifier
 +--rw rt                              string
       +--rw endpoint* [id]
          +--rw id                 string
          +--rw rd                 string
          +--rw pe-device          -> /ncs:devices/device/name
          +--rw interfaces
          |  +--rw ip-address               inet:ipv4-address
          |  +--rw netmask                  inet:ipv4-address
 +--rw (junos)?
          |  |  +--:(junos)
          |  |     +--rw interface                -> deref(../../pe-
device)/../ncs:config/junos:configuration/interfaces/interface/name
          |  |     +--rw ip-address-with-mask     junos:ipv4addr
          |  +--rw vlan                     uint16
          |  +--rw bandwidth                string
          +--rw description
```

FACTOR GROUP

Yang tools

- **OpenDaylight YANG Tools** – Tools supporting NETCONF and YANG, code generation from YANG models
  - https://wiki.opendaylight.org/view/YANG_Tools:Main
- **ONOS YANG Tools** – Yang Runtime and Java application support
  - https://gerrit.onosproject.org/onos-yang-tools
- **YANG Explorer** – YANG Browser / RPC Builder
  - https://github.com/CiscoDevNet/yang-explorer
- **YANG Development Kit (YDK)** – APIs bindings generator
  - https://github.com/CiscoDevNet/ydk-gen
  - https://github.com/CiscoDevNet/ydk-py
  - https://github.com/CiscoDevNet/ydk-go
  - https://github.com/CiscoDevNet/ydk-cpp



Visual dependency tool – f.e. ietf-ip

Legend

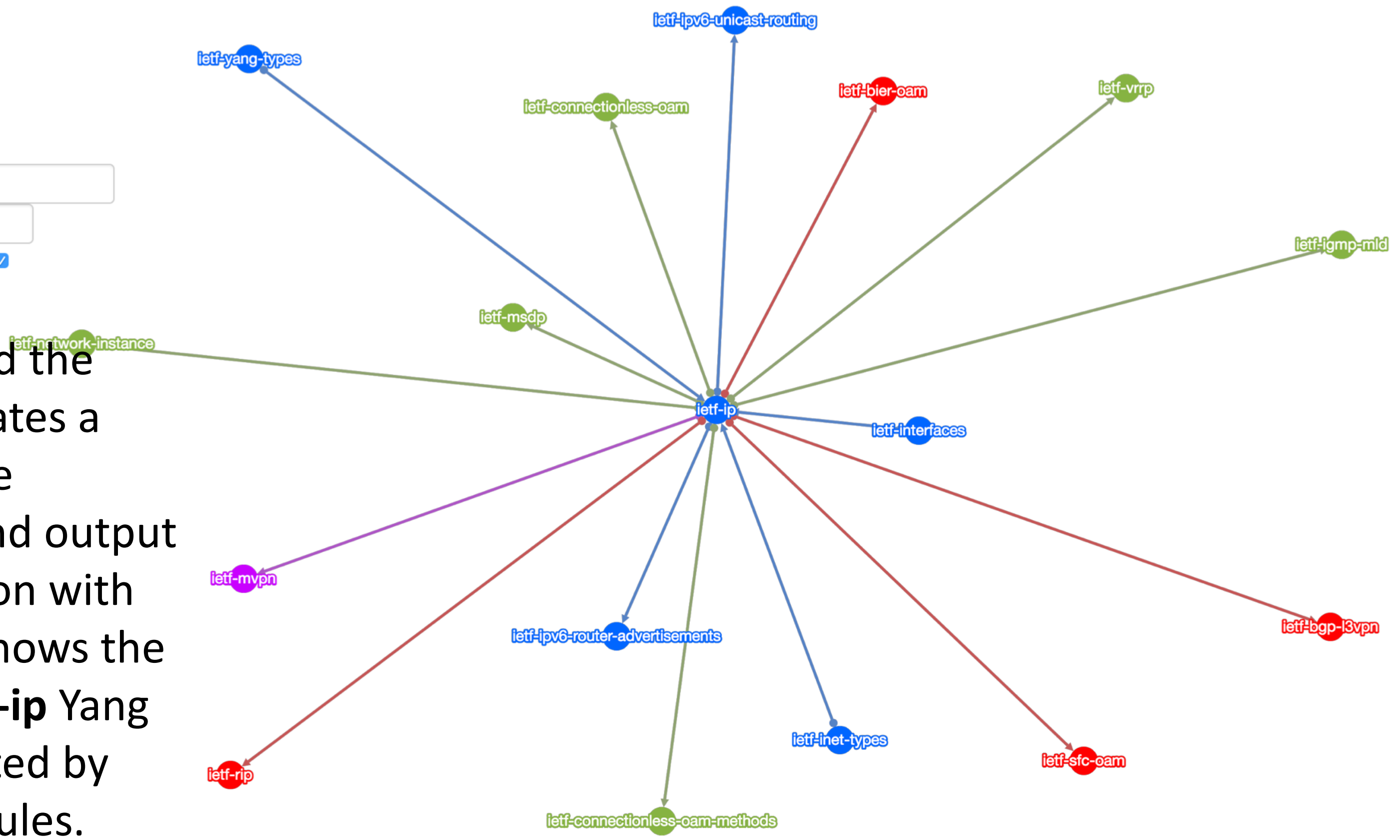
Status: Compilation Failed
Status: IETF:INDIVIDUAL DRAFT
Status: IETF:WG DRAFT
Status: IETF:RFC

Modules: ietf-ip x

Orgs:

Recursion Levels: 0 **Include Standards?**

Generate Export



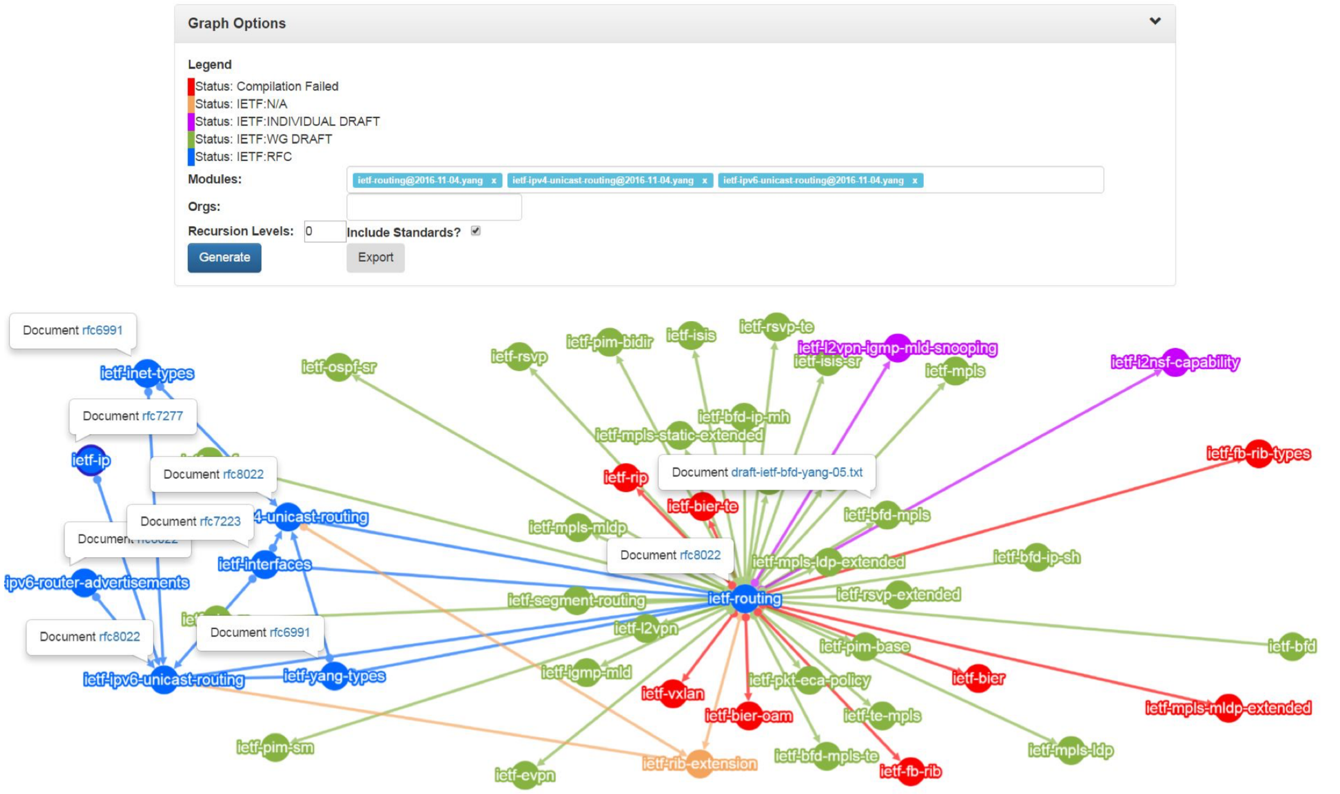
Jan Medved developed the symd.py, which generates a variety of yang module dependency graphs and output suitable for visualization with D3.js tools. Example shows the importance of the **ietf-ip** Yang module, as it is imported by many other Yang modules.

Yang Catalog - Yang DB search

The Yang DB search laid a framework for the multi-SDO impact analysis, including a color scheme for the standard maturity levels.

Currently it includes:

- IETF
- BBF
- OpenConfig



Agenda

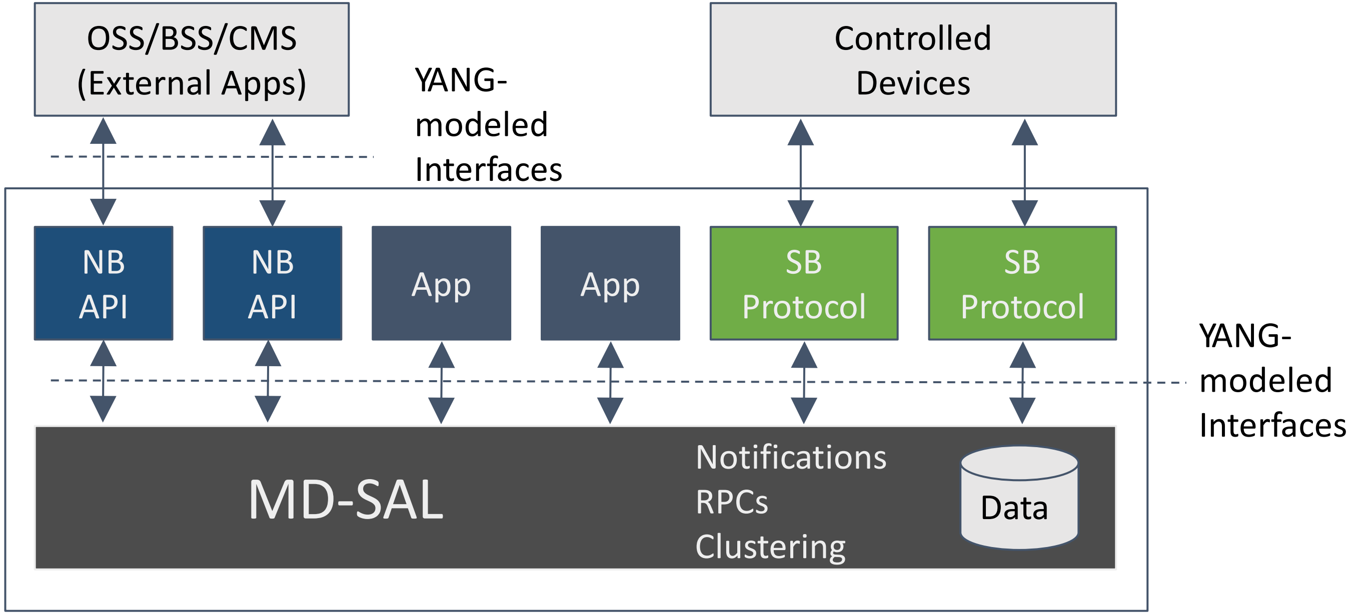
- Orchestration problem statement
- Netconf overview
- Restconf overview
- Yang overview
- **OpenDaylight overview**
- DEMO

What is OpenDaylight?

- Open Source programmable platform hosted by the Linux Foundation
- ~5 Years Old
- Mature, Open Governance
- Mature code base
- ~1000 Individual Contributors from ~140 organizations
- Dozens of OpenDaylight-based solutions
- Many deployments

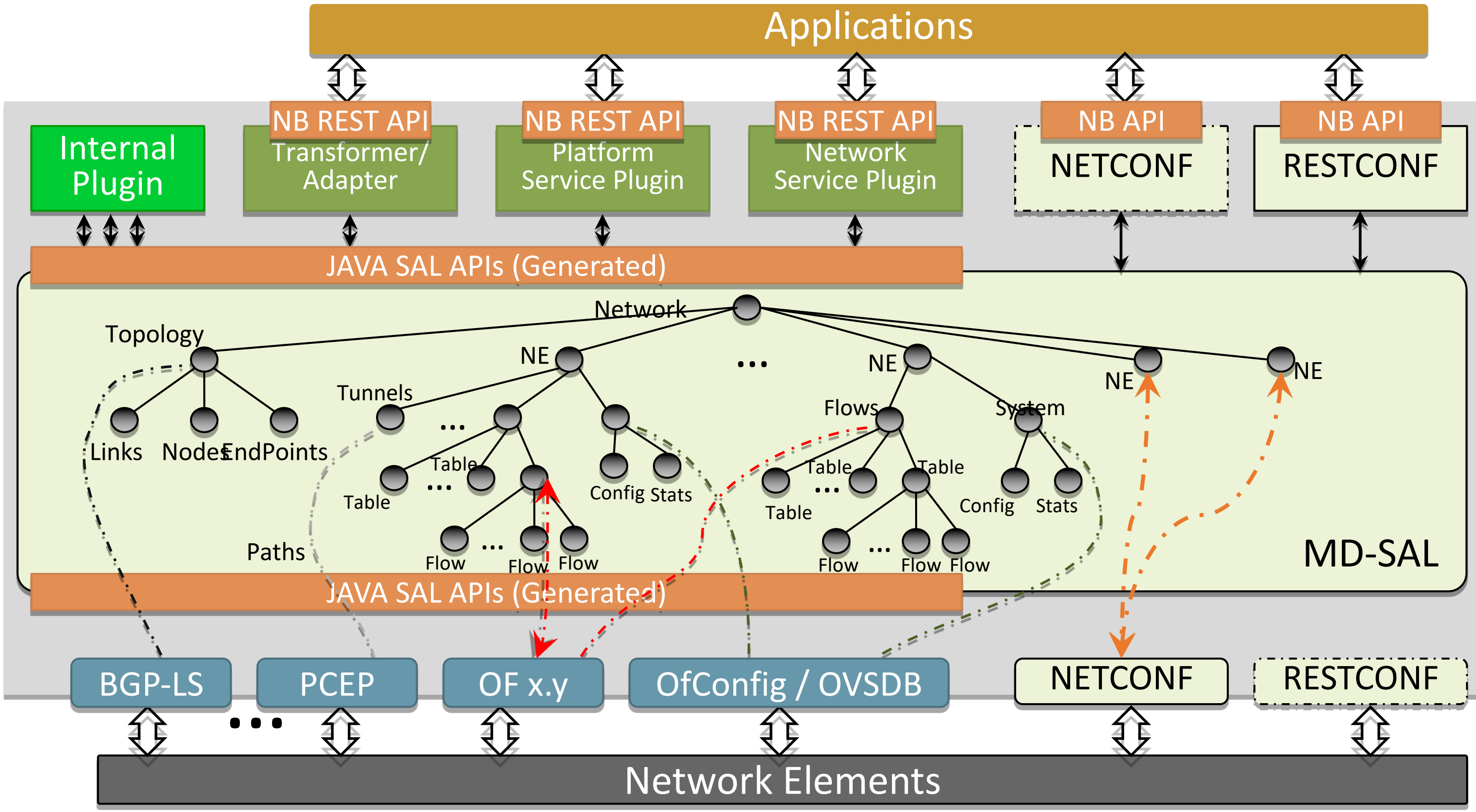
OpenDaylight: a Yang-based microservices Platform

- Based on Model-Driven Service Abstraction Layer (MD-SAL)
- Yang
- Data Modeling Language for NETCONF
- Creates well-defined APIs
- Java and RESTCONF APIs auto-generated from Yang

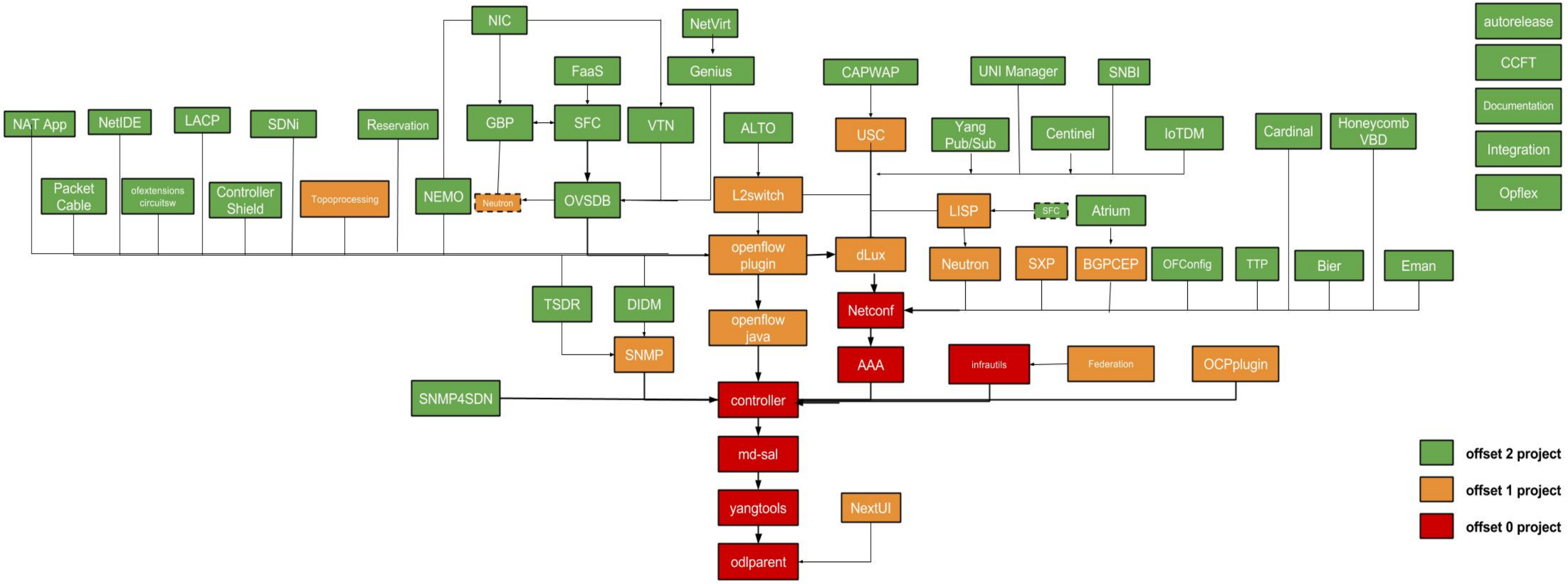


MD-SAL = Model-Driven Service Abstraction Layer

OPEN DAYLIGHT



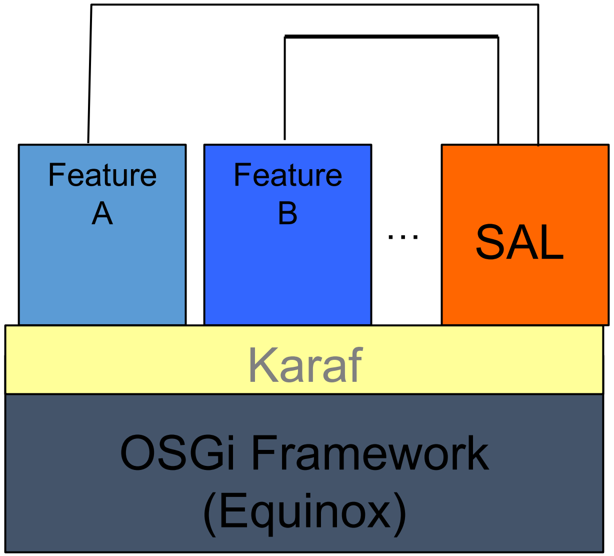
ODL project dependencies



Java, Interface, Maven, OSGi, Karaf

OPEN DAYLIGHT

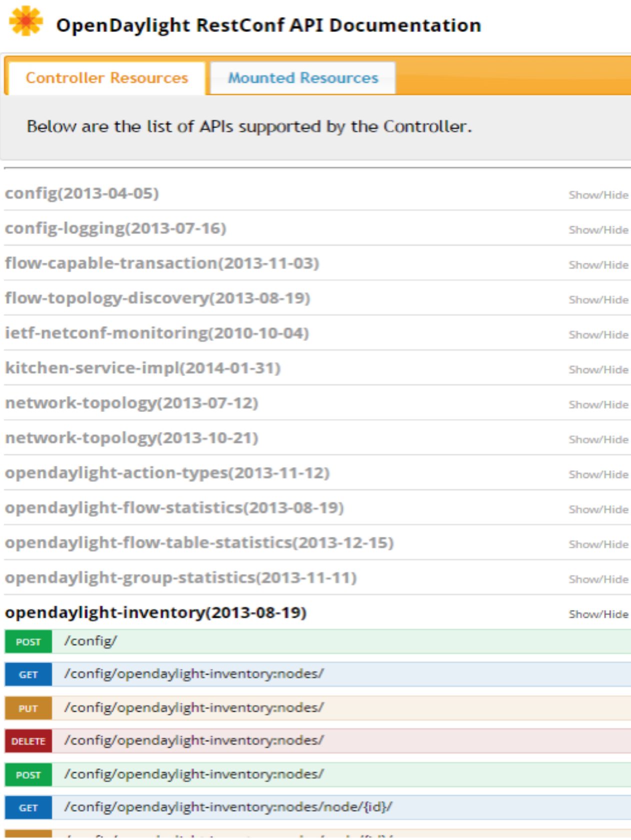
- Java chosen as an enterprise-grade, cross-platform compatible language
- Java Interfaces are used for event listening, specifications and forming patterns
- Maven – build system for Java
- OSGi:
  - Allows dynamically loading bundles
  - Allows registering dependencies and services exported
  - For exchanging information across bundles
- Karaf: Light-weight Runtime for loading modules/bundles
  - OSGi based. Primary distribution mechanism for Helium



FACTOR GROUP

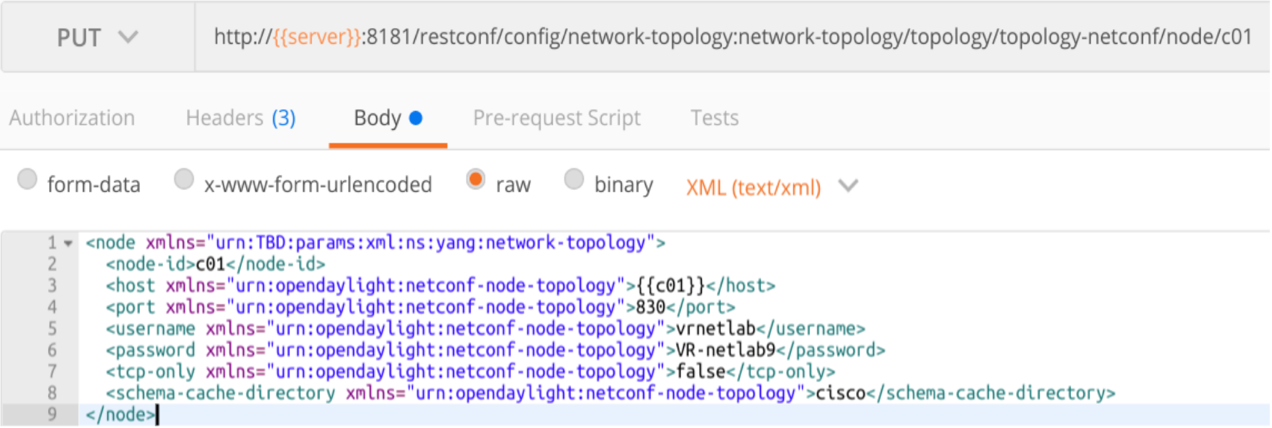
Java, Interface, Maven, OSGi, Karaf

- OpenDaylight has significant support for REST APIs
- Restconf allows for checking config and operational state
  - feature:install odl-restconf
  - http://localhost:8181/restconf/....
- List of exposed Northbound APIs are auto-generated using swagger.
  - feature:install odl-mdsal-apidocs
  - http://localhost:8181/apidoc/explorer



NETCONF connector

- To connect the NETCONF/YANG NE to ODL - use network-topology feature:
  - feature:install odl-netconf-topology odl-restconf
- Fully capable (full support for Netconf/Yang) device to mount, we need only:
  - Name
  - Host
  - Port
  - Username/Password
    (for Netconf session)



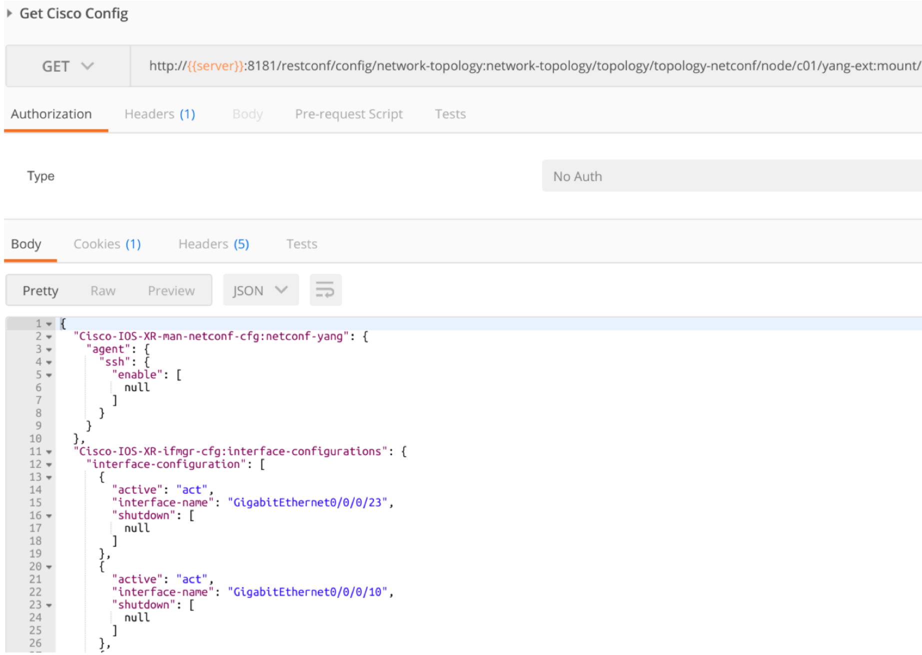
NETCONF connector (advanced config)

- Advanced configuration
  - Schema-cache-directory
  - Reconnect-on-changed-schema
  - Connection-timeout-millis
  - Default-request-timeout-millis
  - Max-connection-attempts
  - Between-attempts-timeout-millis
  - Sleep-factor
  - Keepalive-delay
  - Yang-module-capabilities
  - Yang library
  - Concurrent rpc limit

```xml
<node xmlns="urn:TBD:params:xml:ns:yang:network-topology">
  <node-id>j01</node-id>
  <host xmlns="urn:opendaylight:netconf-node-topology">{{j01}}</host>
  <port xmlns="urn:opendaylight:netconf-node-topology">830</port>
  <username xmlns="urn:opendaylight:netconf-node-topology">vrnetlab</username>
  <password xmlns="urn:opendaylight:netconf-node-topology">VR-netlab9</password>
  <tcp-only xmlns="urn:opendaylight:netconf-node-topology">false</tcp-only>
  <schema-cache-directory xmlns="urn:opendaylight:netconf-node-topology">juniper</schema-cache-directory>
    <yang-module-capabilities xmlns="urn:opendaylight:netconf-node-topology">
    <capability>http://xml.juniper.net/junos/16.2R1/junos?module=configuration&amp;revision=2016-11-22</capability>
    <capability>http://yang.juniper.net/yang/1.1/je?module=junos-extension&amp;revision=2016-11-22</capability>
<!--    <capability>http://yang.juniper.net/yang/1.1/je?module=junos-extension-odl&amp;revision=2016-11-22</capability> -->
   </yang-module-capabilities>
</node>
```

NETCONF connector - what available

- Get datastore
  - yang-ext:mount
- Invoke RPC (POST)
  - yang-ext:/mount/<module>:<operation>
- Update/Delete a netconf-connector
- Get/Set/Modify mounted NEs configuration
- Get the operational datastore
- Receive notifications from NETCONF NE



Agenda

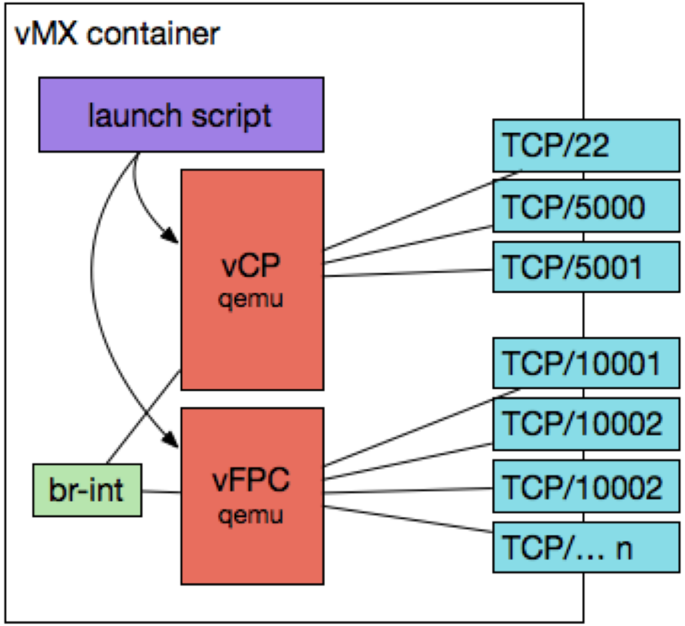
- Orchestration problem statement
- Netconf overview
- Restconf overview
- Yang overview
- OpenDaylight overview
- **DEMO**

vrnetlab – tool for service models CI/CT/CD

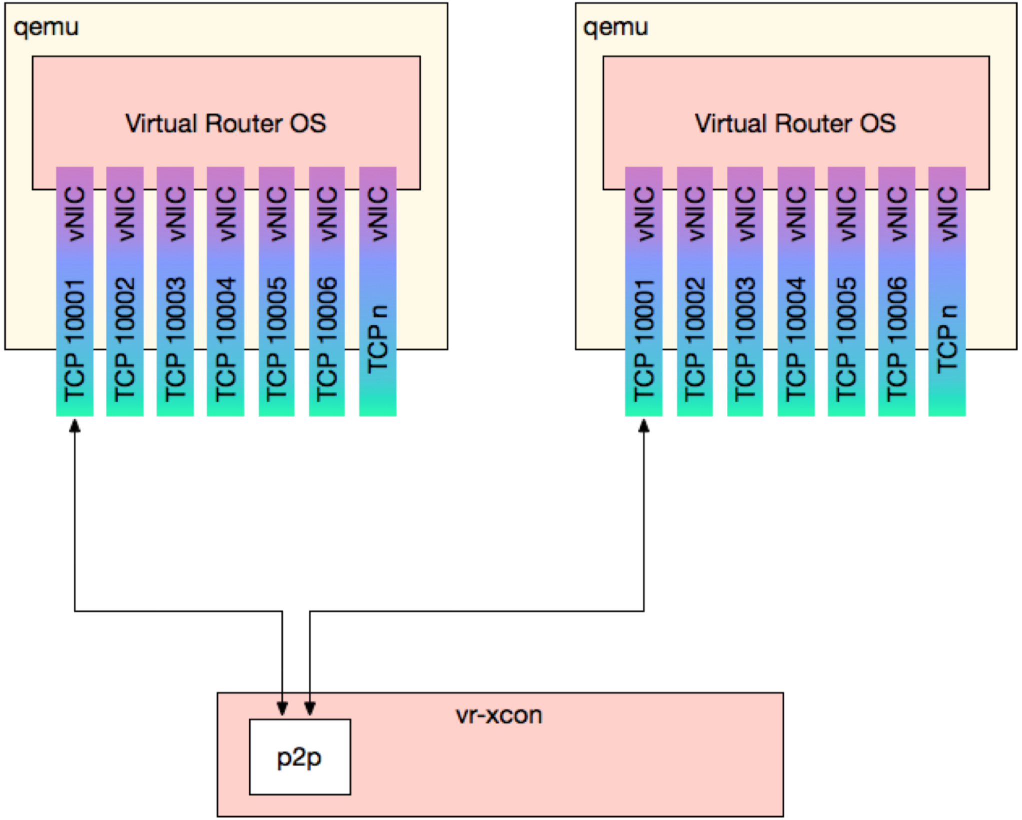
- Run your favourite virtual routers in docker for convenient labbing, development and testing.
- vrnetlab is being developed for the TeraStream project at Deutsche Telekom as part of an automated CI/CT/CD environment for testing our network provisioning system.
- It supports: Arista vEOS, Cisco CSR1000v,Cisco Nexus NX-OS, Cisco XRv, Juniper vMX, Nokia VSR
- Features:
  - Use docker and KVM
  - Ship as single unit
  - Bootstrap
  - Flexible networking
  - Simple: docker run —privileged -d vr-xrv:5.3.3

vrnetlab – tool for service models CI/CT/CD

- vMX runs two VMS:
- control plane
- forwarding plane
- vMX startup scripts is a mess
- vrnetlab makes it simple!



vrnetlab – tool for service models CI/CT/CD



Join LFN meetup!

Assign to group, we will meet soon!

https://www.meetup.com/Open-Source-Networking-Moscow/

OPEN DAYLIGHT

Thank you !

Evgeny Zobnitsev
e-mail: e@zobnitsev.ru
twitter: @ezobn

FACTOR GROUP