Yang role

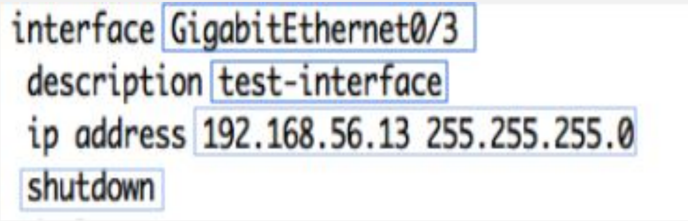
Ganesh Nalawade

Agenda

- Introduction to Yang
- Yang variants
- Introduction to Netconf
- Yang Role deepdive
- Demo
- Questions

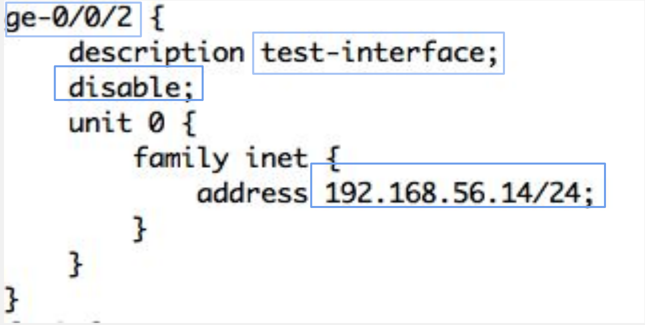
Proprietary data model

ios interface configuration

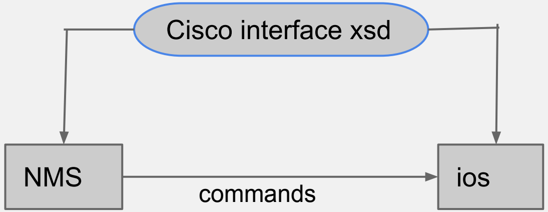


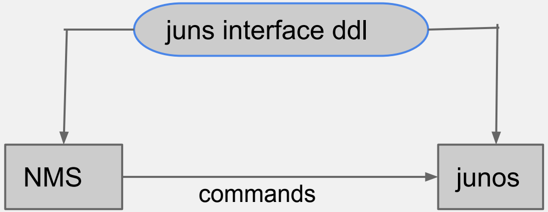
```
interface GigabitEthernet0/3
 description test-interface
 ip address 192.168.56.13 255.255.255.0
 shutdown
```

junos interface configuration



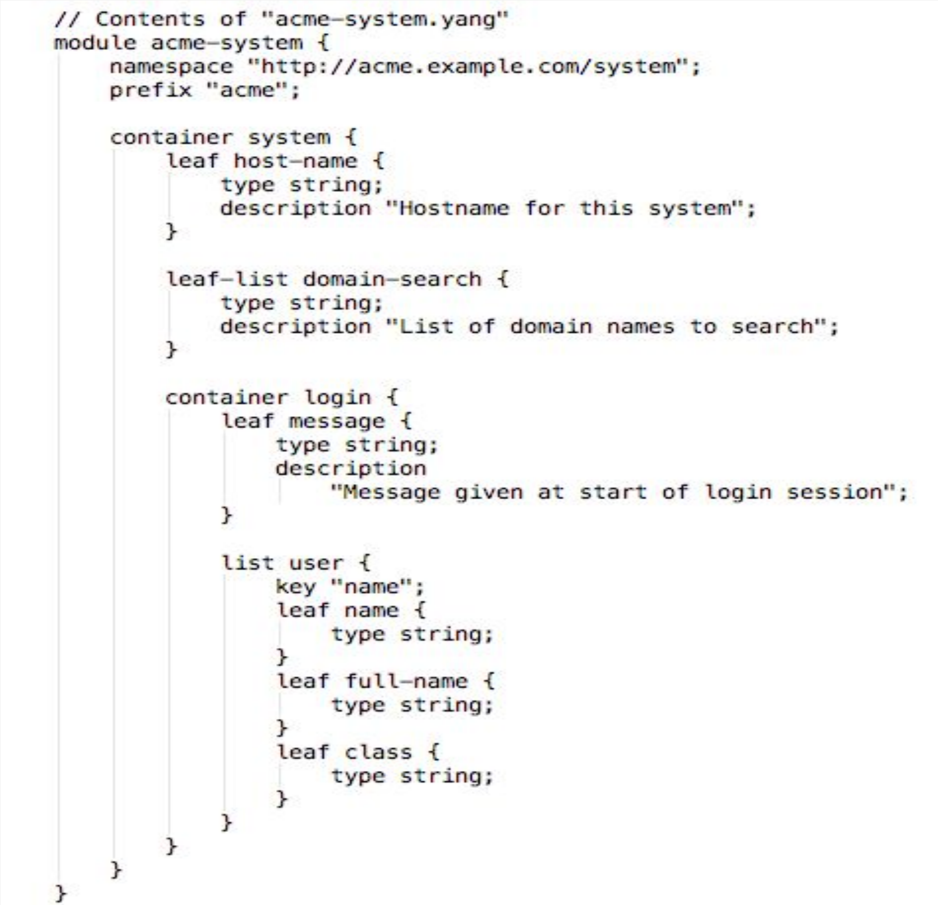
```
ge-0/0/2 {
    description test-interface;
    disable;
    unit 0 {
        family inet {
            address 192.168.56.14/24;
        }
    }
}
```





Yang

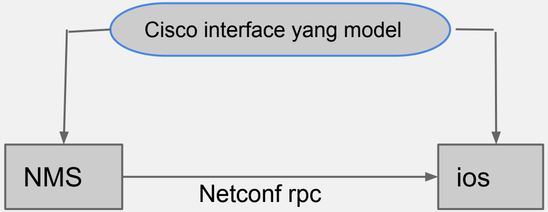
- YANG is a data modeling language used to model configuration and state data
- It is a IETF standard defined by RFC 6020
- Human readable representation of data-mode
- Hierarchy data representation
- Build in data types and constraints
- Extensible

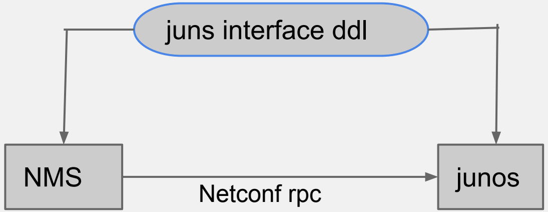


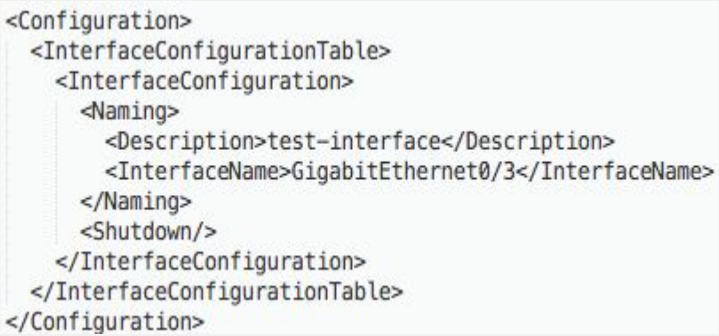
```
// Contents of "acme-system.yang"
module acme-system {
    namespace "http://acme.example.com/system";
    prefix "acme";

    container system {
        leaf host-name {
            type string;
            description "Hostname for this system";
        }

        leaf-list domain-search {
            type string;
            description "List of domain names to search";
        }

        container login {
            leaf message {
                type string;
                description
                    "Message given at start of login session";
            }

            list user {
                key "name";
                leaf name {
                    type string;
                }
                leaf full-name {
                    type string;
                }
                leaf class {
                    type string;
                }
            }
        }
    }
}
```

Yang variants (vendor defined)

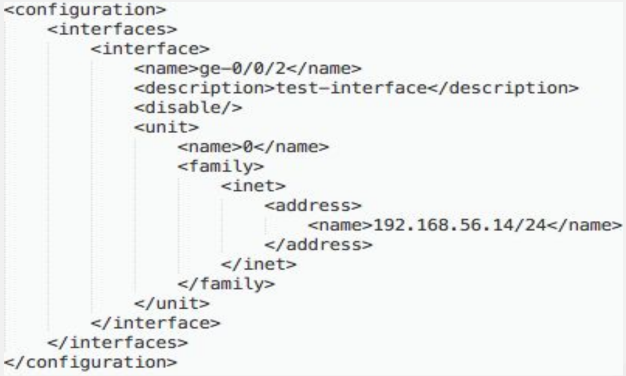
- The data represented in yang model varies based on vendor implementation and the released yang models are published and maintained by vendors themselves.







```
<Configuration>
  <InterfaceConfigurationTable>
    <InterfaceConfiguration>
      <Naming>
        <Description>test-interface</Description>
        <InterfaceName>GigabitEthernet0/3</InterfaceName>
      </Naming>
      <Shutdown/>
    </InterfaceConfiguration>
  </InterfaceConfigurationTable>
</Configuration>
```

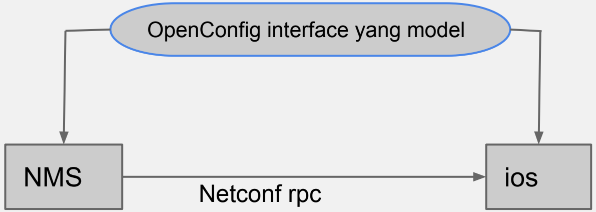


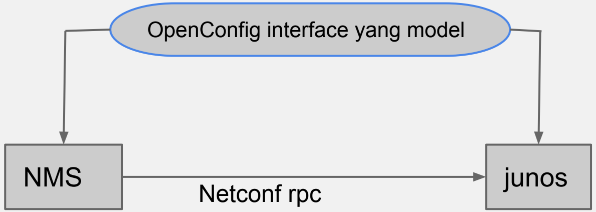
```
<configuration>
    <interfaces>
        <interface>
            <name>ge-0/0/2</name>
            <description>test-interface</description>
            <disable/>
            <unit>
                <name>0</name>
                <family>
                    <inet>
                        <address>
                            <name>192.168.56.14/24</name>
                        </address>
                    </inet>
                </family>
            </unit>
        </interface>
    </interfaces>
</configuration>
```

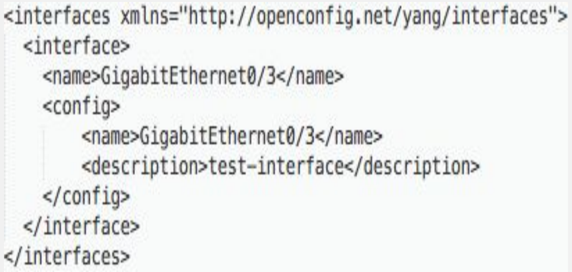
Yang variants (standard)

- Standard yang model defined by ietf, ieee
  https://github.com/YangModels/yang/tree/master/standard/ietf/RFC

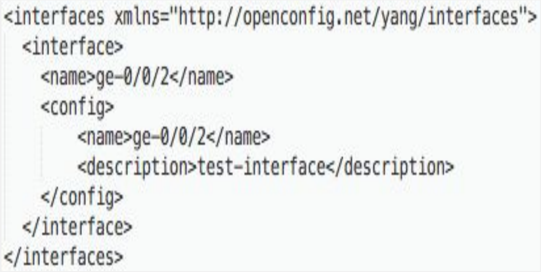
- OpenConfig which is an informal working group of network operators that promotes vendor-neutral model
  https://github.com/openconfig/public/tree/master/release/models







```
<interfaces xmlns="http://openconfig.net/yang/interfaces">
  <interface>
    <name>GigabitEthernet0/3</name>
    <config>
        <name>GigabitEthernet0/3</name>
        <description>test-interface</description>
    </config>
  </interface>
</interfaces>
```



```
<interfaces xmlns="http://openconfig.net/yang/interfaces">
  <interface>
    <name>ge-0/0/2</name>
    <config>
        <name>ge-0/0/2</name>
        <description>test-interface</description>
    </config>
  </interface>
</interfaces>
```

Netconf

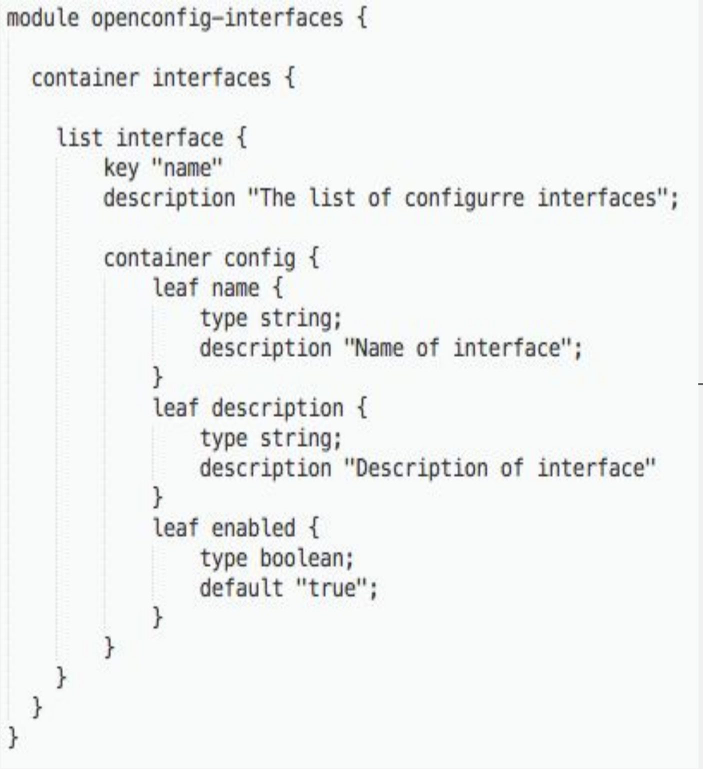
- The NETCONF protocol defines a simple mechanism through which a network device can be managed, configuration data information can be retrieved, and new configuration data can be uploaded and manipulated
- It is a IETF standard defined by RFC 6241
- It is xml based encoding mainly build on top of ssh transport as a subsystem
- Configuration rpc's
  - edit-config, get-config, copy-config, delete-config, lock, unlock
- Operational state rpc's
  - get (maps to show commands)

Ansible Netconf modules:

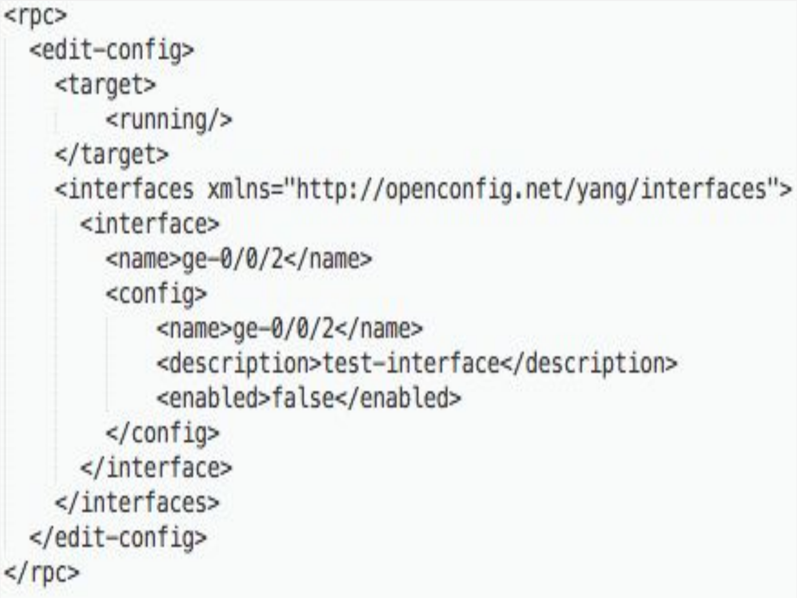
- netconf_config: Configuration management (create, update, delete)
- netconf_get: Retrieve state and configuration data (read)
- netconf_rpc: Execute generic Netconf rpc (mainly vendor specific rpc's)

*Physical/Virtual device in this context refers to traditional router/switch/fw/lb including OVS

Yang mapping to Neconf



```
module openconfig-interfaces {

  container interfaces {

    list interface {
        key "name"
        description "The list of configurre interfaces";

        container config {
            leaf name {
                type string;
                description "Name of interface";
            }
            leaf description {
                type string;
                description "Description of interface"
            }
            leaf enabled {
                type boolean;
                default "true";
            }
        }
    }
  }
}
```



```
<rpc>
  <edit-config>
    <target>
        <running/>
    </target>
    <interfaces xmlns="http://openconfig.net/yang/interfaces">
      <interface>
        <name>ge-0/0/2</name>
        <config>
            <name>ge-0/0/2</name>
            <description>test-interface</description>
            <enabled>false</enabled>
        </config>
      </interface>
    </interfaces>
  </edit-config>
</rpc>
```

Yang Role deep-dive

- Functions supported:
  - **fetch:** This function fetches yang models from device (if supported) at runtime and stores it on Ansible controller
  - **configure:** Reads input json/yaml configuration, validate input configuration against yang model and pushes the configuration on device.
  - **spec:** Parses yang model and generates skeleton configuration in json and xml format and yang tree representation in easy to understand hierarchical format.

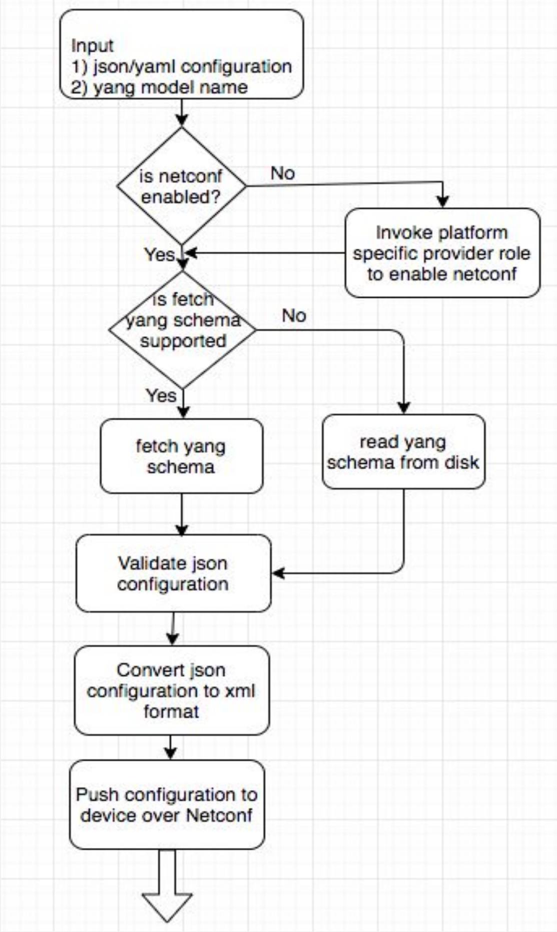
- Uses Ansible netconf modules to configure and retrieve data from network device

- Uses pyang python library for parsing of yang files and configuration data validation.

- Input to roles is json/yaml configuration which can be read using role variables.

Configure workflow

1) Enable netconf if not already enabled
2) Fetch yang schema from network device if supported else read from disk
3) Validate json configuration to check if it conforms with yang model and data type and restriction check
4) Convert json configuration to xml format which acts as a netconf rpc payload
5) Push xml config to device using netconf_config module


Input
1) json/yaml configuration
2) yang model name
is netconf enabled?
No
Invoke platform specific provider role to enable netconf
Yes
is fetch yang schema supported
No
Yes
fetch yang schema
read yang schema from disk
Validate json configuration
Convert json configuration to xml format
Push configuration to device over Netconf

Spec workflow

1) Enable netconf if not already enabled
2) Fetch yang schema from network device if supported else read from disk.
3) Generate yang tree representation as per RFC 8340 and copy to file on disk.
4) Generate json configuration skeleton and copy to file on disk.
5) Generate xml configuration skeleton and copy to file on disk.

Demo

Questions?