



NETCONF/YANG

Shishio Tsuchiya

shtsuchi@cisco.com

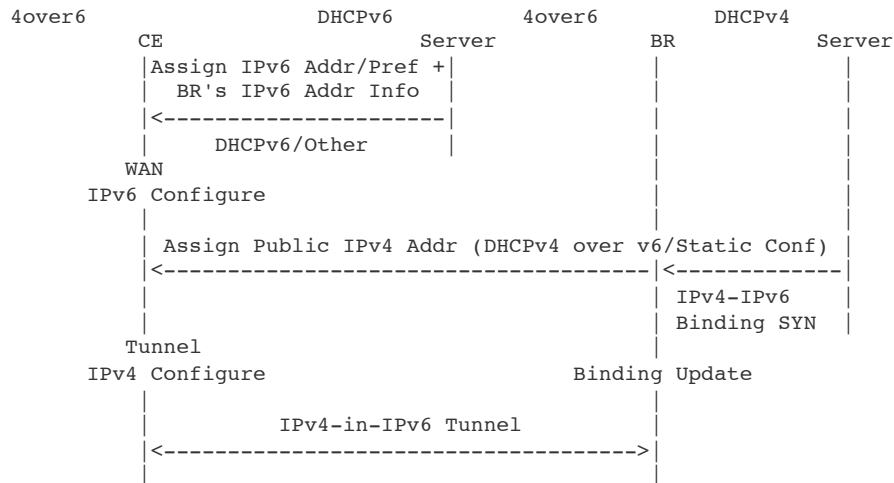
はじめに

- 社内のトレーニングで使った内容をチュートリアルで応募しました
- 本プログラムで…という事になったので、こんな感じで

Agenda

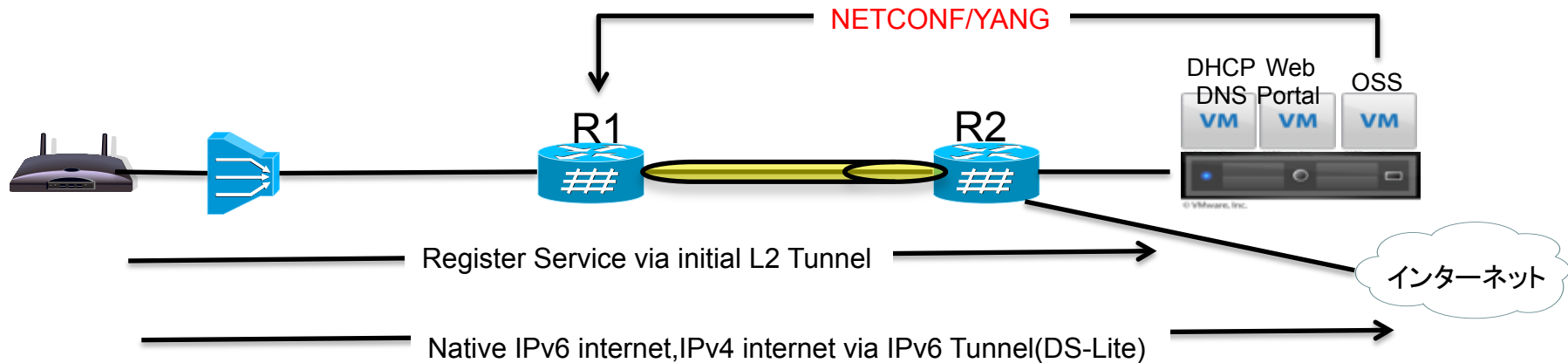
- 何故にNETCONF/YANG?
- XML
- NETCONF
- YANG
- Q&A

RFC7040 Public IPv4-over-IPv6 Access Network



- MAPなどの比較技術
- IPv4とIPv6を1:1でBind
- アドレス情報の伝播はDHCP
- Encapの設定方法は？

Real-Time OSS for SDN



- 顧客がCPEにケーブルを刺すと、L2トンネルを経由し、Cloud Service Centerに接続する
- 顧客の登録により、OSSよりR1に設定変更が行われ、インターネットに接続が出来るようになる

Writable MIB Module IESG Statement

<https://www.ietf.org/iesg/statement/writable-mib-module.html>

- The Internet Engineering Steering Group(IESG)は OPSエリアで標準化ベースの設定手法として NETCONF/YANGを強く推奨してるのにも関わらず、それぞれのWGが同一用途で特定MIBを定義しつつづけてるのをみて、下記を提言した
 - ✓ それぞれのWGは標準の設定手法として**NETCONF/YANGを推奨**
 - ✓ SNMPのWrite運用の用途がはっきりしており、OPS ADと MIB Doctorとコンセンサスが取れている時のみ、作成と修正を行う

Tail-f買収

<http://www.cisco.com/web/about/ac49/ac0/ac1/ac259/tail-f.html#~faq>

[Products & Services](#)[Support](#)[How to Buy](#)[Training & Events](#)[Partners](#)

Acquisitions

Tail-f Systems is Now Part of Cisco

Tail-f Systems is a leader in network services orchestration technologies.

[Read Press Release](#)



[Overview](#)[FAQs](#)[Sales](#)[Services & Support](#)[Social Media](#)

Q: Who is Tail-f Systems?

A: Tail-f Systems is a first mover and leader in the emerging network services configuration and orchestration space for SDN and NFV environments, and a leading vendor in the On-device Configuration Management market. Tail-f is also the thought-leader around the development and implementation of the NETCONF protocol and YANG data modeling language. NETCONF and YANG are becoming the preeminent standards to simplify network programmability and automation as well as service orchestration across both traditional physical and virtual infrastructures. These technologies promise to drastically improve on the current generation of disparate, expensive, and slow script-based automation solutions in use by most service providers today.

Agenda

- 何故にNETCONF/YANG?
- XML
- NETCONF
- YANG
- Q&A

XMLとは？



- eXtensible Markup Language
- World Wide Web Consortium (W3C) により策定・勧告
- データ記述の為の言語
- シリアル化(直列化)とデータ識別に役立つ
- 完全なプログラム言語やデータベース言語ではない
- HTMLとの比較
 - ✓HTML: データを表示する 大文字/小文字を区別しない
 - ✓XML : データを記述する 大文字/小文字を区別する

XMLの記述サンプル

プログラムが取り出しやすい

```
<person>
  <name>
    <first>Shishio</first>
    <last>Tsuchiya</last>
  </name>
  <organization>
    Cisco Systems G.K.
  </organization>
  <title>
    Consulting Systems Engineer
  </title>
</person>
```

HTMLの記述サンプル

人間が分かりやすい様に表示

```
<p>  
  <b>First Name:</b>Shishio<br>  
  <b>Last Name:</b>Tsuchiya<br>  
  <b>Organization:</b>Cisco Systems G.K.<br>  
  <b>Title:</b>Consulting Systems Engineer<br>  
</p>
```

<p>～</p>パラグラフ

～太字

改行

XMLヘッダー

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

- version:1.0 (mandatory)
- encoding:記述されるデータの文字コードを示す (optional,UTF-8 デフォルト)
- standalone:XML文章がスタンドアロンかどうか (optional,デフォルト no)

XMLエレメント(要素)

```
<name>Shishio</name>  
<empty contents />
```

- XMLタグを *Element* と呼ぶ
- スタートとエンドの間のデータが *Contents*(内容)
- 全てのElementはスタートとエンドのタグを持たなければならない
- 空のContentsの際には エンドで”/>”で表すことが出来る

XMLアトリビュート(属性)

```
<presentation name="NETCONF/YANG">  
</presentation>
```

- elementの補足情報
- “”もしくは’ ’ で囲む

XMLエンティティ

```
<!ENTITY myName "Shishio Tsuchiya">  
<!ENTITY RFC2629 SYSTEM http://xml.resource.org/public/rfc/bibxml/reference.RFC.2629.xml>  
<name>&myName;</name>  
<read>&RFC2629;</read>
```

- 他のXMLデータで使用する為のシンボルの指定
- 外部エンティティを参照することも可能
- &はエスケープ文字にも使用

<	<
>	>
&	&
"	"
'	'

キャラクターデーター

```
<sample>  
<![CDATA[ 2 < 1]]>  
</sample>
```

- ・ <![CDATA[で始まり]]>で終わる
- ・ パーサーが適用されず、文字列と認識される

XMLコメント

```
<presentation title="NETCONF/YANG">  
  <1st>  
    XML  
    <!--現在進行中-->  
  </1st>  
</presentation>
```

- コメントアウト
- <!--で始まり -->で終わる

Namespace

- element nameの衝突を防ぐために使用

```
<table>
<tr>
  <tb>Shishio Tsuchiya</tb>
  <tb>Masataka Mawatari</tb>
  <tb>Toshiya Mabuchi</tb>
  <tb>Hiroki Nonaka</tb>
</tr>
</table>
```

```
<table>
  <color>White</color>
  <width>50</width>
  <length>100</length>
</table>
```

- HTML <table>タグとXMLでの家具<table>で衝突
- xmlns:namespaceURLで表現する

Agenda

- 何故にNETCONF/YANG?
- XML
- **NETCONF**
- YANG
- Q&A

NETCONFとは？



- **NET**work **CONF**iguration Protocol
- コンフィグデータの管理においてSNMPのSMI(Structured Managed Interface)の欠点を補う為に作成
 - ✓ コンフィグのバックアップやリストア能力の欠如
 - ✓ トランザクションの概念が無い
 - ✓ 変なまたは古いコンセプト
- IETFにて作成 RFC6241 Network Configuration Protocol (NETCONF)
- NETCONFではXMLを使用する

NETCONFへの要求事項

- 簡単に使えなければならない
- コンフィグとオペレーションの明確な区別
- 1つベンダーに絞るよりも、スケーラビリティのあるネットワークワイドなコンフィグ
- コンフィグのバックアップとリストア方法の提供
- 一貫したコンフィグを提供するためのエラーチェック
- 結果はdiffやCVSといったテキスト管理システムで処理/保管出来る事が望まれる
- コンフィグの修正およびそのアクティベーションを行う事は区別出来る事

NETCONFとSNMPの比較

	SNMP	NETCONF
データモデル	MIB	XMLスキーマー文章
データモデル言語	SMI	XMLスキーマー
オペレーション	SNMP	NETCONF
RPCエンキャプスレーション	BER	XML
トランスポート	UDP	TCP

NETCONFプロトコル

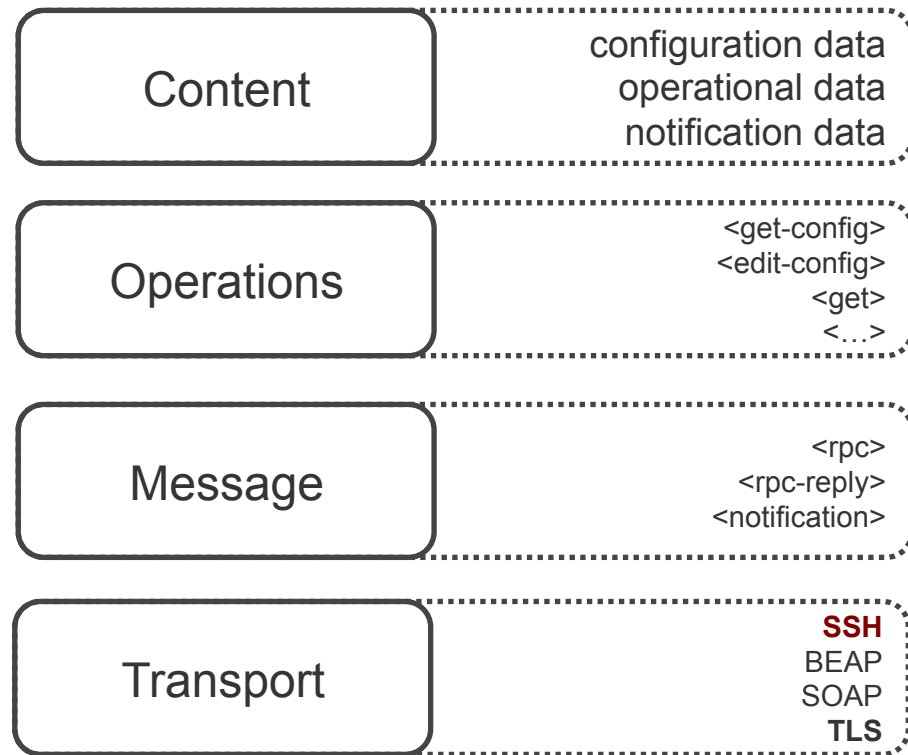
- 様々なオペレーションの提供

get-config, get, edit-config, commit, validate, copy-config, discard-changes, delete-config, lock, unlock, create-subscription

- 様々なコンフィグデータの保管

(candidate, running, startup)

プロトコルレイヤー



NETCONFフロー - hello



```
cisco@server-1:~$ ssh cisco@10.255.0.54 netconf format
```

NETCONFフロー

Capability Exchange<hello>



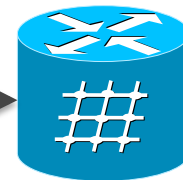
クライアント

```
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
    <capability>
      urn:ietf:params:netconf:capability:candidate:1.0
    </capability>
    <capability>
      urn:ietf:params:netconf:capability:notification:1.0
    </capability>
  </capabilities>
  <session-id>
    285212672
  </session-id>
</hello>
]]>]]>
```

NETCONFフロー

Capability Exchange<hello>

```
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
    <capability>
      urn:ietf:params:netconf:capability:candidate:1.0
    </capability>
    <capability>
      urn:ietf:params:netconf:capability:notification:1.0
    </capability>
  </capabilities>
</hello>
]]>]]>
```



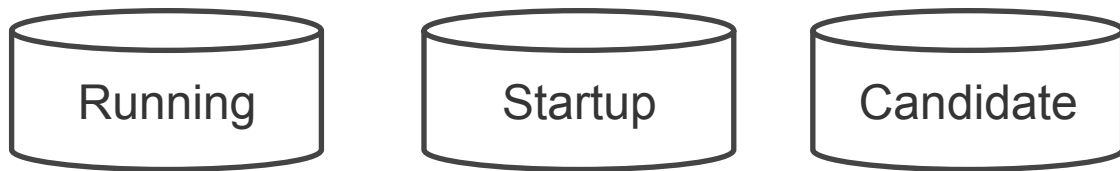
サーバー

NETCONF Capability

- helloメッセージの際にnamespaceで定義されたcapabilityを交換する

Capability	Capability Identifier	Reference
:writable-running	urn:ietf:params:netconf:capability:writable-running:1.0	[RFC6241]
:candidate	urn:ietf:params:netconf:capability:candidate:1.0	[RFC6241]
:confirmed-commit	urn:ietf:params:netconf:capability:confirmed-commit:1.0	[RFC4741]
:confirmed-commit:1.1	urn:ietf:params:netconf:capability:confirmed-commit:1.1	[RFC6241]
:rollback-on-error	urn:ietf:params:netconf:capability:rollback-on-error:1.0	[RFC6241]
:validate	urn:ietf:params:netconf:capability:validate:1.0	[RFC4741]
:validate:1.1	urn:ietf:params:netconf:capability:validate:1.1	[RFC6241]
:startup	urn:ietf:params:netconf:capability:startup:1.0	[RFC6241]
:url	urn:ietf:params:netconf:capability:url:1.0	[RFC6241]
:xpath	urn:ietf:params:netconf:capability:xpath:1.0	[RFC6241]
:notification	urn:ietf:params:netconf:capability:notification:1.0	[RFC5277]
:interleave	urn:ietf:params:netconf:capability:interleave:1.0	[RFC5277]
:partial-lock	urn:ietf:params:netconf:capability:partial-lock:1.0	[RFC5717]
:with-defaults	urn:ietf:params:netconf:capability:with-defaults:1.0	[RFC6243]
:base:1.0	urn:ietf:params:netconf:base:1.0	[RFC4741][RFC6241]
:base:1.1	urn:ietf:params:netconf:base:1.1	[RFC6241]

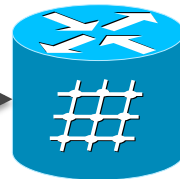
NETCONF Datastore



- NETCONFではいくつかのデータストアを定義
- 現在稼働している設定ファイル(Running)
- <running>は:baseで定義される(必須)
- <startup>は:startupで定義されるブートアップ時の設定
- candidateは操作(commit/変更/追加/削除)をする為の設定ファイル

NETCONFフロー –<get-config>

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
    <filter>
      <Configuration>
        <InterfaceConfigurationTable>
          <InterfaceConfiguration>
            <Naming>
              <Active>act</Active>
              <InterfaceName Match="Loopback.*"/>
            </Naming>
          </InterfaceConfiguration>
        </InterfaceConfigurationTable>
      </Configuration>
    </filter>
  </get-config>
</rpc>
]]>]]>
```



サーバー

NETCONFフロー –<get-config>



クライアント



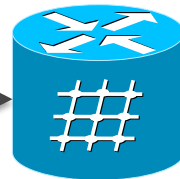
サーバー

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <Configuration>
      <InterfaceConfigurationTable MajorVersion="5" MinorVersion="4">
        <InterfaceConfiguration>
          <Naming>
            <Active>
              act
            </Active>
            <InterfaceName>
              Loopback0
            </InterfaceName>
          </Naming>
          <InterfaceVirtual>
            true
          </InterfaceVirtual>
          <Description>
            Loopback
          </Description>
          <IPv4Network MajorVersion="6" MinorVersion="4">
            <Addresses>
              <Primary>
                <Address>
                  192.168.255.1
                </Address>
                <Netmask>
                  255.255.255.255
                </Netmask>
              </Primary>
            </Addresses>
          </IPv4Network>
        </InterfaceConfiguration>
      </InterfaceConfigurationTable>
    </Configuration>
  </data>
</rpc-reply>
]]>]]>
```

NETCONFフロー

インターフェース追加<edit-config>

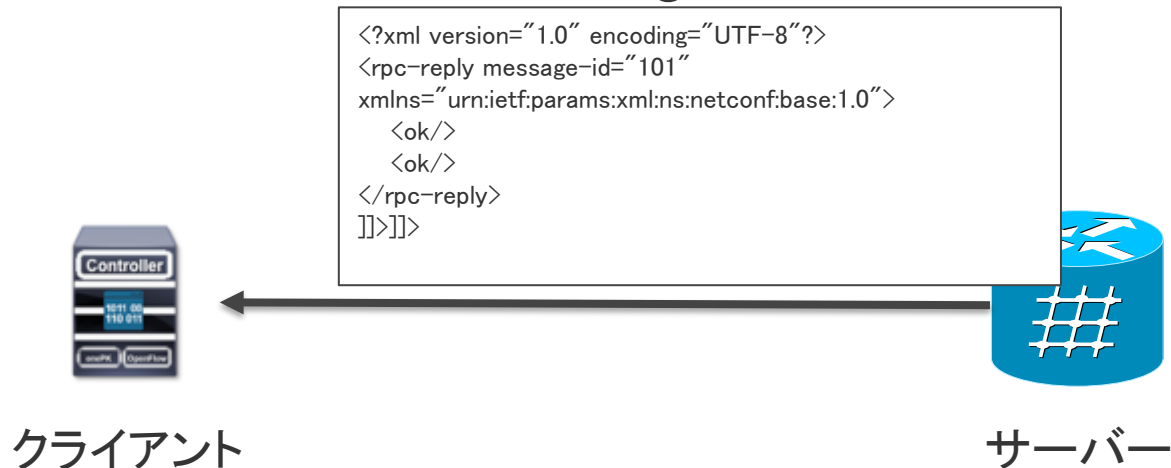
```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <config>
      <Configuration>
        <InterfaceConfigurationTable>
          <InterfaceConfiguration>
            <Naming>
              <Active>act</Active>
              <InterfaceName>Loopback100</InterfaceName>
            </Naming>
            <InterfaceVirtual>true</InterfaceVirtual>
            <IPV4Network><Addresses><Primary>
              <Address>172.16.255.1</Address>
              <Netmask>255.255.255.255</Netmask>
            </Primary></Addresses></IPV4Network>
          </InterfaceConfiguration>
        </InterfaceConfigurationTable>
      </Configuration>
    </config>
  </edit-config>
  <commit/>
</rpc>
]]>]]
```



サーバー

NETCONFフロー

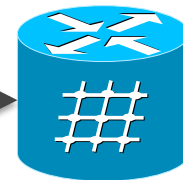
インターフェース追加<edit-config>



NETCONFフロー

インターフェース削除<edit-config>

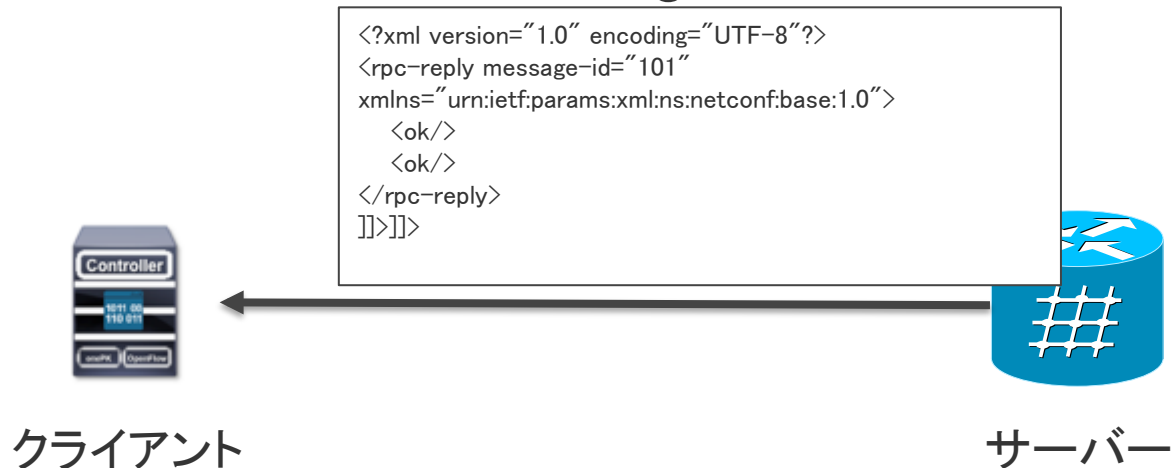
```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <config>
      <Configuration>
        <InterfaceConfigurationTable operation="delete">
          <InterfaceConfiguration>
            <Naming>
              <Active>act</Active>
              <InterfaceName>Loopback100</InterfaceName>
            </Naming>
          </InterfaceConfiguration>
        </InterfaceConfigurationTable>
      </Configuration>
    </config>
  </edit-config>
  <commit/>
</rpc>
]]>]]>
```



サーバー

NETCONFフロー

インターフェース削除<edit-config>



NETCONFメッセージ詳細

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```
<get-config>
```

```
<source>
```

```
<running/>
```

```
</source>
```

```
<filter>
```

```
<Configuration>
```

```
<InterfaceConfigurationTable>
```

```
<InterfaceConfiguration>
```

```
<Naming>
```

```
<Active>act</Active>
```

```
<InterfaceName Match="Loopback.*"/>
```

```
</Naming>
```

```
</InterfaceConfiguration>
```

```
</InterfaceConfigurationTable>
```

```
</Configuration>
```

```
</filter>
```

```
</get-config>
```

```
</rpc>
```

```
]]>]]>
```

NETCONF RPCメッセージレイヤー

Operationレイヤー

Contentsレイヤー

各ベンダー独自の
XMLスキーマー

Framing Marker

Agenda

- 何故にNETCONF/YANG?
- XML
- NETCONF
- **YANG**
- Q&A

YANGとは？



- Yet Another Next Generationモデリング言語
- NETCONFのRequest/Replyの中のObjectおよびデータを定義
- XMLスキーマーとSMIに類似（より強力）
- コンフィグレーション/オペレーション/RPCコールのモデル化
- RFC 6020 – YANG – A data modeling language for NETCONF

何故YANGが必要だったか？

- NETCONFをネットワークワイドのプロトコルにする為には共通のデータモデルを持つ必要があった
- それぞれのベンダーはCLIを持っていて、それをXMLで包むのみでは十分ではない
- YANGは全てのネットワーク機器からNETCONFを使用出来る様、共通のデータモデルを提供する
- 各々のベンダーは共通のYANGモジュールを実装しなければならない
- IETF NETMOD WGで定義中
- RFC6244 An Architecture for Network Management Using NETCONF and YANG

	SNMP	NETCONF
データモデル	MIB	YANG
データモデル言語	SMI	YANG
オペレーション	SNMP	NETCONF
RPCエンキャプスレーション	BER	XML
トランスポート	UDP	TCP

YANG module

```

container interfaces {
    list interface {key "name";
        leaf name {type string;}
        leaf description {type string;}
        leaf type {type identityref {
            base interface-type;
        }
            mandatory true;
        }
    leaf enabled {
        type boolean;
        default "true";
    }
    leaf link-up-down-trap-enable {
        if-feature if-mib;
        type enumeration {
            enum enabled {
                value 1;
            }
            enum disabled {
                value 2;
            }
        }
    }
}

}

container interfaces-state {
    config false;
    list interface {key "name";
        leaf name {type string;

```

```

+--rw interfaces
|   +--rw interface* [name]
|   |   +--rw name                string
|   |   +--rw description?        string
|   |   +--rw type                identityref
|   |   +--rw enabled?            boolean
|   |   +--rw link-up-down-trap-enable? enumeration
+--ro interfaces-state
|   +--ro interface* [name]
|   |   +--ro name                string
|   |   +--ro type                identityref
|   |   +--ro admin-status        enumeration
|   |   +--ro oper-status         enumeration
|   |   +--ro last-change?        yang:date-and-time
|   |   +--ro if-index            int32
|   |   +--ro phys-address?       yang:phys-address
|   |   +--ro higher-layer-if*    interface-state-ref
|   |   +--ro lower-layer-if*     interface-state-ref
|   |   +--ro speed?              yang:gauge64
|   |   +--ro statistics
|   |   |   +--ro discontinuity-time yang:date-and-time
|   |   |   +--ro in-octets?         yang:counter64
|   |   |   +--ro in-unicast-pkts?   yang:counter64
|   |   |   +--ro in-broadcast-pkts? yang:counter64
|   |   |   +--ro in-multicast-pkts? yang:counter64

```

••

YANGの構成

RFC7223 ietf-interfaces

YANG module

tree structure

```

+--rw interfaces
|
|   +--rw interface* [name]
|   |   +--rw name                string
|   |   +--rw description?        string
|   |   +--rw type                 identityref
|   |   +--rw enabled?             boolean
|   |   +--rw link-up-down-trap-enable? enumeration
|   |
|   |   +--ro interfaces-state
|   |   |   +--ro interface* [name]
|   |   |   |   +--ro name                string
|   |   |   |   +--ro type                 identityref
|   |   |   |
|   |   |   |   +--ro oper-status          enumeration
|   |   |   |   +--ro last-change?        yang:date-and-time
|   |   |   |   +--ro if-index            int32
|   |   |   |   +--ro phys-address?       yang:phys-address
|   |   |   |   +--ro higher-layer-if*    interface-state-ref
|   |   |   |   +--ro lower-layer-if*    interface-state-ref
|   |   |   |   +--ro speed?              yang:gauge64
|   |   |   |
|   |   |   |   +--ro statistics
|   |   |   |   |   +--ro discontinuity-time yang:date-and-time
|   |   |   |   |   +--ro in-octets?        yang:counter64
|   |   |   |   |   +--ro in-unicast-pkts?  yang:counter64
|   |   |   |   |   +--ro in-broadcast-pkts? yang:counter64
|   |   |   |   |   +--ro in-multicast-pkts? yang:counter64
|   |   |   |   |   ...
|   |   |   |   ...
|   |   |   ...
|   |   ...
|   ...

```

RW:設定(config)とRO:運用(state/stat)の明確な区別

```

container interfaces {
  list interface {key "name";
    leaf name {type string;}
    leaf description {type string;}
    leaf type {type identityref {
      base interface-type;
    }
    mandatory true;
  }
  leaf enabled {
    type boolean;
    default "true";
  }
}

```

階層化構造

container:それ自体に値は無く、ツリーの構造を定義する為に使用
 list:コンフィグの際には必須のkeyを含む、ツリーの構造を定義する為に使用
 leaf:子ノードは持たず、値を持つ

```

}
enum disabled {
  value 2;
}
}
}
}
}
}
container interfaces-state {
  config false;
  list interface {key "name";
    leaf name {type string;

```

YANGの構成

RFC7277 ietf-ip

YANG module

tree structure

```
+--rw if:interfaces
  +--rw if:interface* [name]
    ...
    +--rw ipv4!
      +--rw enabled?          boolean
      +--rw forwarding?       boolean
      +--rw mtu?              uint16
      +--rw address* [ip]
        +--rw ip              inet:ipv4-address-no-zone
        +--rw (subnet)
          +--:(prefix-length)
            +--rw ip:prefix-length?  uint8
          +--:(netmask)
            +--rw ip:netmask?        yang:dotted-quad
      +--rw neighbor* [ip]
        +--rw ip              inet:ipv4-address-no-zone
        +--rw link-layer-address  yang:phys-address
    +--rw ipv6!
      +--rw enabled?          boolean
      +--rw forwarding?       boolean
      +--rw mtu?              uint32
      +--rw address* [ip]
        +--rw ip              inet:ipv6-address-no-zone
        +--rw prefix-length    uint8
      +--rw neighbor* [ip]
```

```
augment "/if:interfaces/if:interface" {
  leaf enabled {type boolean;default true;}
  leaf forwarding {type boolean;default false;}
  leaf mtu {
    type uint16 {range "68..max";}
    units octets;}
  list address {key "ip";
    leaf ip {
      type inet:ipv4-address-no-zone;
    }
    choice subnet {mandatory true;
      leaf prefix-length {type uint8 {
        range "0..32";}}
      leaf netmask {
        if-feature ipv4-non-contiguous-
netmasks;
        type yang:dotted-quad;
      }
    }
  }
  list neighbor {
    key "ip";
```

YANGの構成

RFC7277 ietf-ip

YANG module

tree structure

```
+--rw if:interfaces
  +--rw if:interface* [name]
    ...
  +--rw ipv4!
    +--rw enabled?          boolean
    +--rw forwarding?       boolean
    +--rw mtu?              uint16
    +--rw address* [ip]
      +--rw ip              inet:ipv4-address-no-zone
      +--rw (subnet)
        +--:(prefix-length)
          +--rw ip:prefix-length?  uint8
        +--:(netmask)
          +--rw ip:netmask?        yang:dotted-quad
    +--rw neighbor* [ip]
      +--rw ip              inet:ipv4-address-no-zone
      +--rw link-layer-address yang:phys-address
  +--rw ipv6!
    +--rw enabled?          boolean
    +--rw forwarding?       boolean
    +--rw mtu?              uint32
    +--rw address* [ip]
      +--rw ip              inet:ipv6-address-no-zone
      +--rw prefix-length    uint8
    +--rw neighbor* [ip]
```

```
augment "/if:interfaces/if:interface" {
  leaf enabled {type boolean;default true;
}
  leaf forwarding {type boolean;default false;
}
  leaf mtu {
    type uint16 {range "68..max";}
  }
  leaf ip {
    type inet:ipv4-address-no-zone;
  }
  choice subnet {mandatory true;
    leaf prefix-length {type uint8 {
      range "0..32";}}
    leaf netmask {
      if-feature ipv4-non-contiguous-
```

range:範囲を定義し、設定値を制限する事が出来る

柔軟性

choice:互いの排他的な値をリスト化出来る(radioボタン)

拡張性

augment:外部モジュールで定義されたスキーマを組み込む事が出来る

YANGの特徴まとめ

- テキストで分かりやすい
- 設定データ(RW)と運用データ(RO)の区別が明確
- 階層化構造(例:container/list/leaf-list)
- namespaceの活用による拡張性(augment)
- 制約を入れる事で設定を論理的にする事が出来る(例:range/length)
- 柔軟な選択が出来る(例:union/choice/feature)

NETCONF/YANGフロー - hello



```
cisco@server-1:~$ ssh -p 884 cisco@10.255.0.57 -s netconf
```

NETCONF/YANGフロー

Capability Exchange<hello>

```
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.1</capability>
  </capabilities>
</hello>
]]>]]>
```



クライアント



サーバー

NETCONF/YANGフロー

Capability Exchange<hello>



クライアント

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.1</capability>
    <capability>urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring</capability>
    <capability>urn:ietf:params:netconf:capability:candidate:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:rollback-on-error:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:validate:1.1</capability>
    <capability>urn:ietf:params:netconf:capability:confirmed-commit:1.1</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-cdp-cfg?module=Cisco-IOS-XR-cdp-cfg&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-cdp-oper?module=Cisco-IOS-XR-cdp-oper&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-config-mibs-cfg?module=Cisco-IOS-XR-config-mibs-cfg&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-crypto-sam-cfg?module=Cisco-IOS-XR-crypto-sam-cfg&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-crypto-sam-oper?module=Cisco-IOS-XR-crypto-sam-oper&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-crypto-ssh-cfg?module=Cisco-IOS-XR-crypto-ssh-cfg&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-crypto-ssh-oper?module=Cisco-IOS-XR-crypto-ssh-oper&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-drivers-media-eth-cfg?module=Cisco-IOS-XR-drivers-media-eth-cfg&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-drivers-media-eth-oper?module=Cisco-IOS-XR-drivers-media-eth-oper&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-ha-eem-cfg?module=Cisco-IOS-XR-ha-eem-cfg&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-ha-eem-oper?module=Cisco-IOS-XR-ha-eem-oper&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg?module=Cisco-IOS-XR-ifmgr-cfg&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-oper?module=Cisco-IOS-XR-ifmgr-oper&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-infra-alarm-logger-cfg?module=Cisco-IOS-XR-infra-alarm-logger-cfg&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-infra-alarm-logger-datatypes?module=Cisco-IOS-XR-infra-alarm-logger-datatypes&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-infra-alarm-logger-oper?module=Cisco-IOS-XR-infra-alarm-logger-oper&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-infra-ceredundancymib-cfg?module=Cisco-IOS-XR-infra-ceredundancymib-cfg&revision=2015-01-07</capability>
    .....
    <capability>http://cisco.com/ns/yang/Cisco-IOS-XR-tty-vty-cfg?module=Cisco-IOS-XR-tty-vty-cfg&revision=2015-01-07</capability>
    <capability>http://cisco.com/ns/yang/cisco-xr-types?module=Cisco-IOS-XR-types&revision=2015-01-19</capability>
    <capability>urn:ietf:params:xml:ns:yang:ietf-inet-types?module=ietf-inet-types&revision=2013-07-15</capability>
    <capability>urn:ietf:params:xml:ns:yang:ietf-yang-types?module=ietf-yang-types&revision=2013-07-15</capability>
  </capabilities>
  <session-id>3009804360</session-id>
</hello>
]]]]]
```

RFC6022 YANG Module for NETCONF Monitoring
<get-schema>でサーバーからスキーマを受信可能

NETCONF/YANG フロー – <get-config>

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
    <filter>
      <interface-configurations xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">
        <interface-configuration>
          <interface-name>Loopback0</interface-name>
        </interface-configuration>
      </interface-configurations>
    </filter>
  </get-config>
</rpc>
]]>]]>
```



バー



NETCONF/YANGフロー –<get-config>



クライアント

```
<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <interface-configurations xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">
    <interface-configuration>
      <active>act</active>
      <interface-name>Loopback0</interface-name>
      <interface-virtual/>
      <ipv4-network xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ipv4-io-cfg">
        <addresses>
          <primary>
            <address>192.168.255.1</address>
            <netmask>255.255.255.255</netmask>
          </primary>
        </addresses>
      </ipv4-network>
    </interface-configuration>
  </interface-configurations>
</data>
```

スキーマの違い-<get-config>-

```
<Configuration>
  <InterfaceConfigurationTable>
    <InterfaceConfiguration>
      <Naming>
        <Active>act</Active>
        <InterfaceName Match="Loopback0"/>
      </Naming>
    </InterfaceConfiguration>
  </InterfaceConfigurationTable>
</Configuration>
```



NETCONFは使えるが、スキーマおよびエレメントはベンダーに依存

```
<interface-configurations xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">
  <interface-configuration>
    <interface-name>Loopback0</interface-name>
  </interface-configuration>
</interface-configurations>
```



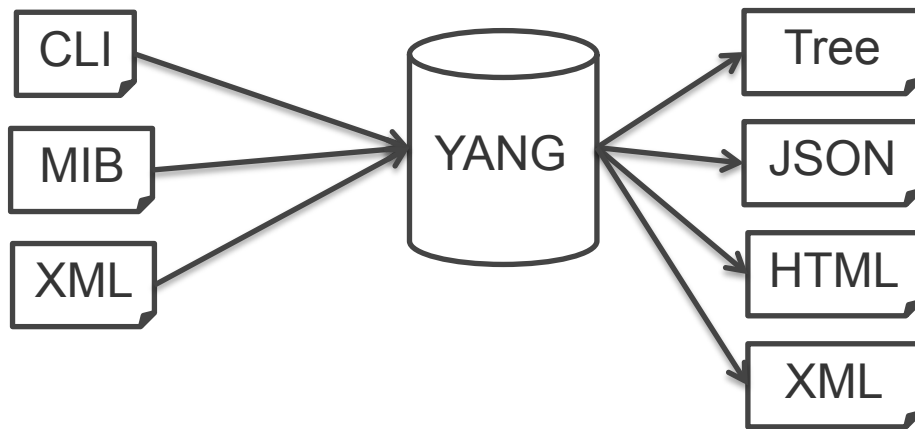
標準スキーマを使用
エレメントはベンダーに依存

```
<interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
  <interface>
    <name>Loopback0</name>
  </interface>
</interfaces>
```



標準スキーマを使用
エレメントも共通な為
全てのベンダーで使用可能

YANGの活用性



- CLI/MIB/独自XMLをYANGで記述
- YANG→XMLに変更するYIN:Yang-Independent Notation
- RESTConfの為のJSON/HTML変換も可能
- pyang <https://github.com/mbj4668/pyang>

pyang treeフォーマット

```
SHTSUCHI-M-V1EK:pyang-master shtsuchi$ pyang -f tree modules/ietf-interfaces.yang
```

```
module: ietf-interfaces
  +--rw interfaces
  |   +--rw interface* [name]
  |   |   +--rw name                string
  |   |   +--rw description?       string
  |   |   +--rw type                identityref
  |   |   +--rw enabled?            boolean
  |   |   +--rw link-up-down-trap-enable? enumeration {if-mib}?
  +--ro interfaces-state
  |   +--ro interface* [name]
  |   |   +--ro name                string
  |   |   +--ro type                identityref
  |   |   +--ro admin-status        enumeration {if-mib}?
  |   |   +--ro oper-status         enumeration
  |   |   +--ro last-change?        yang:date-and-time
  |   |   +--ro if-index            int32 {if-mib}?
  |   |   +--ro phys-address?       yang:phys-address
  |   |   +--ro higher-layer-if*    interface-state-ref
  |   |   +--ro lower-layer-if*    interface-state-ref
  |   |   +--ro speed?              yang:gauge64
  |   +--ro statistics
  |       +--ro discontinuity-time  yang:date-and-time
```

pyang yin(XML)フォーマット

```
SHTSUCHI-M-V1EK:pyang-master shtsuchi$ pyang -f yin modules/ietf-interfaces.yang
```

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<module name="ietf-interfaces"
```

```
  xmlns="urn:ietf:params:xml:ns:yang:yin:1"
```

```
  xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces"
```

```
  xmlns:yang="urn:ietf:params:xml:ns:yang:ietf-yang-types">
```

```
<namespace uri="urn:ietf:params:xml:ns:yang:ietf-interfaces"/>
```

```
<prefix value="if"/>
```

```
<import module="ietf-yang-types">
```

```
  <prefix value="yang"/>
```

```
</import>
```

```
----
```

```
<container name="interfaces">
```

```
  <description>
```

```
    <text>Interface configuration parameters.</text>
```

```
</description>
```

```
<list name="interface">
```

```
  <key value="name"/>
```

```
  <description>
```

```
    <text>The list of configured interfaces on the device.
```

pyang jstreeフォーマット

```
SHTSUCHI-M-V1EK:pyang-master shtsuchi$ pyang -f jstree modules/ietf-interfaces.yang
```

```
<head><title> ietf-interfaces
```

```
</title>
```

```
<style type="text/css" media="all">
```

```
body, h1, h2, h3, h4, h5, h6, p, td, table td, input, select {  
    font-family: Verdana, Helvetica, Arial, sans-serif;  
    font-size: 10pt;  
}
```

```
body, ol, li, h2 {padding:0; margin: 0;}
```

```
---
```

```
ol#root {padding-left: 5px; margin-top: 2px; margin-bottom: 1px;  
    list-style: none;}
```

```
<abbr title="Interface configuration parameters.">interfaces</abbr>
```

```
</div>
```

```
</td>
```

```
<td nowrap>container</td>
```

```
<td nowrap></td>
```

```
<td nowrap>config</td>
```

```
<td></td>
```

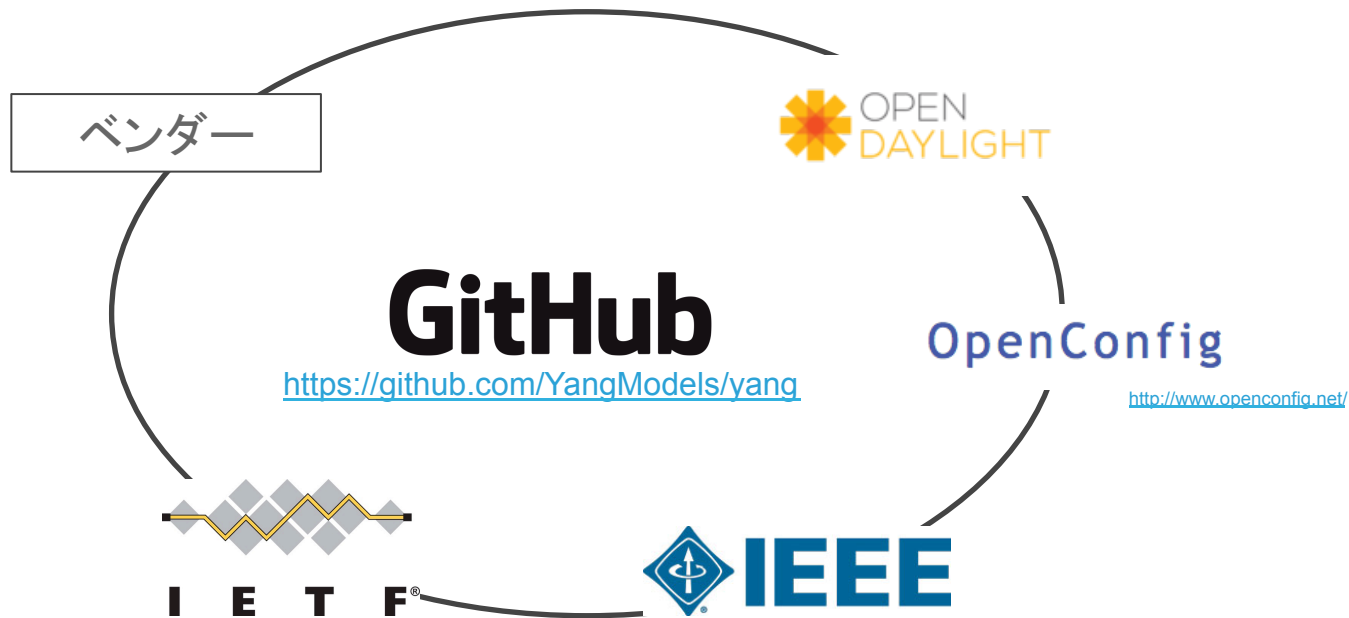
```
<td>current</td>
```

```
<td nowrap>/if:interfaces</td>
```

```
</tr>
```

```
...
```

YANG Governance Model

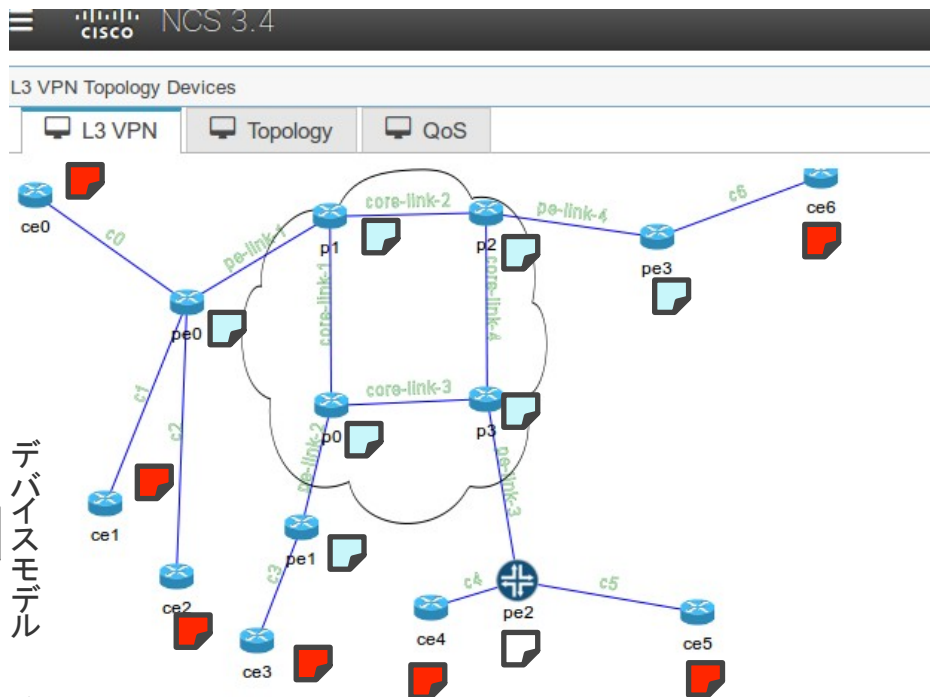
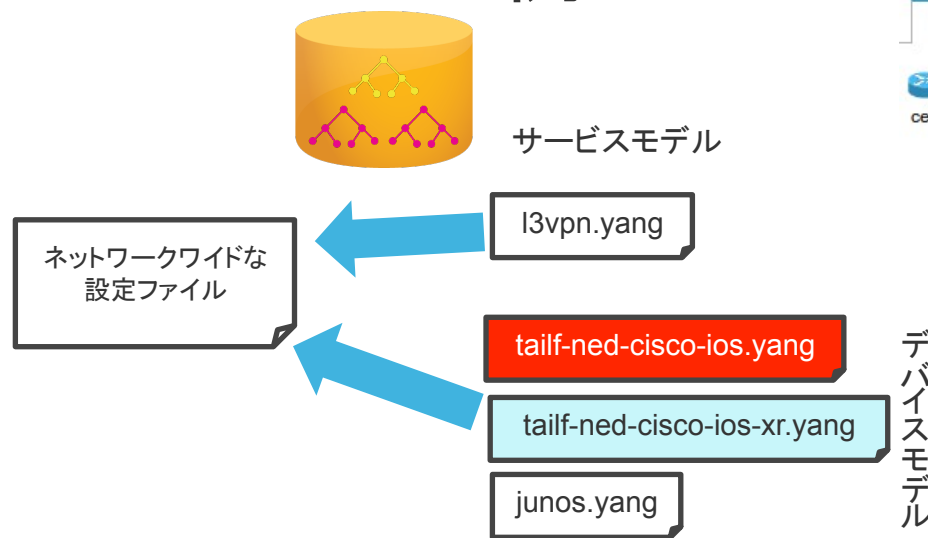


- 様々の標準化団体/オープンソース/ベンダーにより
YANGモデルを定義してる為、githubで管理

相互接続試験

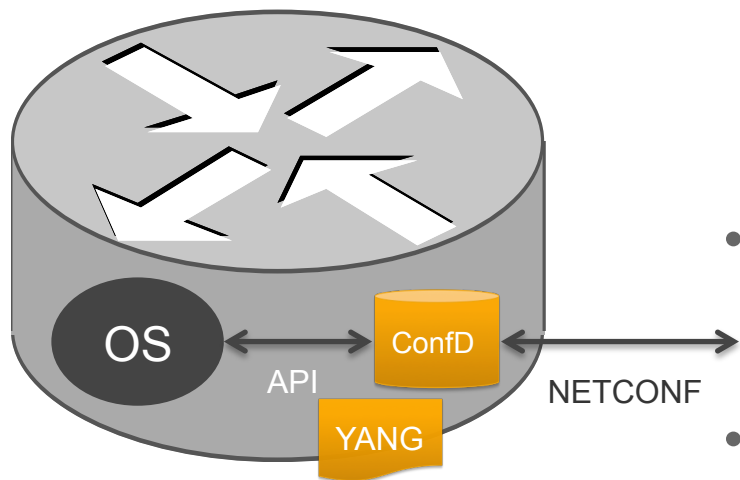
- IETF NETCONF Interop test
 - IETF85での相互接続試験 NETCONF1.1の基本機能を確認
<http://www.ietf.org/proceedings/85/slides/slides-85-netconf-3.pdf>
<http://www.internetsociety.org/publications/ietf-journal-march-2013>
- NETCONF and YANG Interoperability Test
 - EANTCでのNETCONF1.1およびYANGの相互接続試験
 - get-schemaをほとんどがサポートしていた為、デバイスから直接YANGモデルをダウンロードした
<http://www.tail-f.com/netconf-yang-interoperability-test/>

Tail-f NCSの例



- ・ プロトコル全体ではなく、必要部分のみをサービスモデルとして定義
- ・ 設定などメーカー独自の部分はそれぞれのメーカー毎にデバイスモデルを定義
- ・ サービスモデルではデバイス非依存の情報(AS番号/IPアドレスなど/接続状態/デバイスへのnamespace)
- ・ デバイスモデルは各種機器のコンフィグを作成
- ・ サービスモデル/デバイスモデルでネットワーク全体のコンフィグとして保持

ConfDの例



- ConfDはネットワーク機器と共存させる事で、外部とはNETCONFで管理が可能
- 内部ではYANGモデルで管理をし、OSとはAPIでやり取りをする
- ベーシック版はフリーで提供

<https://developer.cisco.com/site/confD/>

まとめ

- NETCONF1.1の実装が進んできた
- 現時点でRFC6022 <get-schema>は必須機能といえる
- YANGデータモデルは既存のCLI/MIB/XMLスキーマーからコンバートはしやすい
- YANGデータモデルよりRESTConfに必要なJSONなどのフォーマットへの加工もしやすい
- 現時点でYANGモデルでの運用はマルチベンダー環境についての統一された運用が可能になるかもしれない



CISCO

TOMORROW starts here.