



Juniper Apstra Freeform概要 & 簡易オペレーションガイド

2023/8/4 updated

CONFIDENTIALITY AND LEGAL NOTICE

This material contains information that is confidential and proprietary to Juniper Networks, Inc. Recipient may not distribute, copy, or repeat information in the document.

This statement of product direction sets forth Juniper Networks' current intention and is subject to change at any time without notice. No purchases are contingent upon Juniper Networks delivering any feature or functionality depicted in this presentation.

subject to a license agreement that describes program terms and conditions.

本資料は融資でベストエフォートで記載している資料となります。
内容に不備がある場合はご了承ください。
最新の状況などは公式のマニュアルをご確認ください。
また、内容は予告なしに変更になる場合があります。

はじめに

本資料はJuniper ApstraのFreeformのネットワーク構築手順を概要をまとめたものです。

内容は予告なく変更する可能性もあり、また都度修正させていただく可能性があります。
本資料のオペレーション部分は参考資料としてご活用ください。

不明点はJuniper Networks、またはパートナー様にご連絡いただくか、
Apstraのマニュアルを参照下さい。

Juniper Apstra 4.2 User Guide Freeform Introduction

<https://www.juniper.net/documentation/us/en/software/apstra4.2/apstra-user-guide/topics/concept/freeform.html>



- Apstra Freeform概要
- Apstra Freeformオペレーション概要
 - 初期セットアップ手順
 - Config Templates
 - Apstra Telemetry
 - オプション機能
- 参考リンク

APSTRAの2つのデザインパターン



**DC Reference
Architecture**

Best of Everything

**Freeform
Architecture**

New

容易な自動化を
ベストプラクティスリファレンスと共に

すべてのDC環境の自動化に

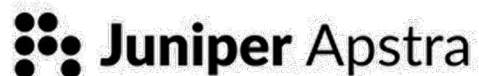


Meal kit - DC Reference



DIY - Freeform

APSTRAの2つのデザインパターン



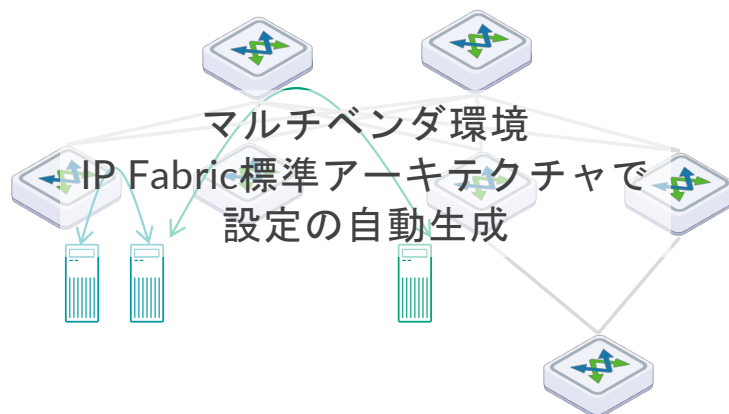
DC Reference Architecture

Best of Everything

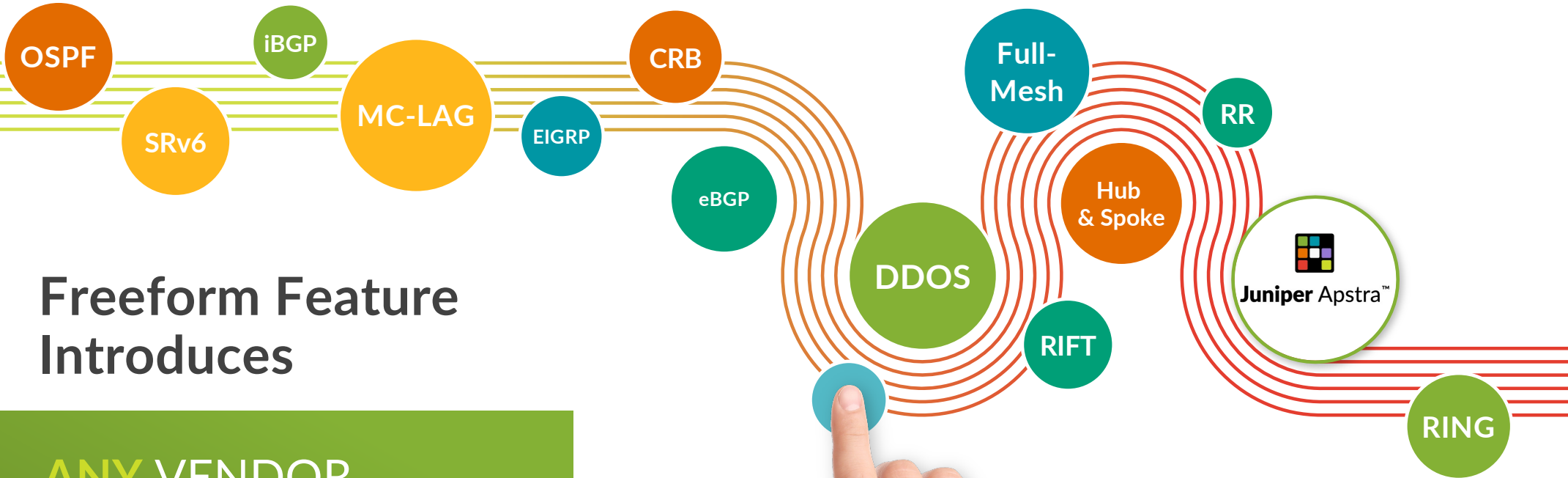
Freeform Architecture ^{New}

データセンターネットワーク向けの
ベストプラクティスアーキテクチャと
トポロジを評価済み設定で自動化・可視化

すべてのトポロジとフィーチャで
どのような設定でも
自動化・可視化基盤として利用可能



※現在はJUNOSのみ



Freeform Feature Introduces

ANY VENDOR

ANY TOPOLOGY

ANY DATA CENTER

※現在はJUNOSのみのサポート

Freeform Overview

- Network Designer & Config Templatesにより柔軟に設定を自動作成
- 可視化、ロールバック、OSアップグレード、など運用に必要な付加価値機能も利用

Network Designer

Config Templates & device context

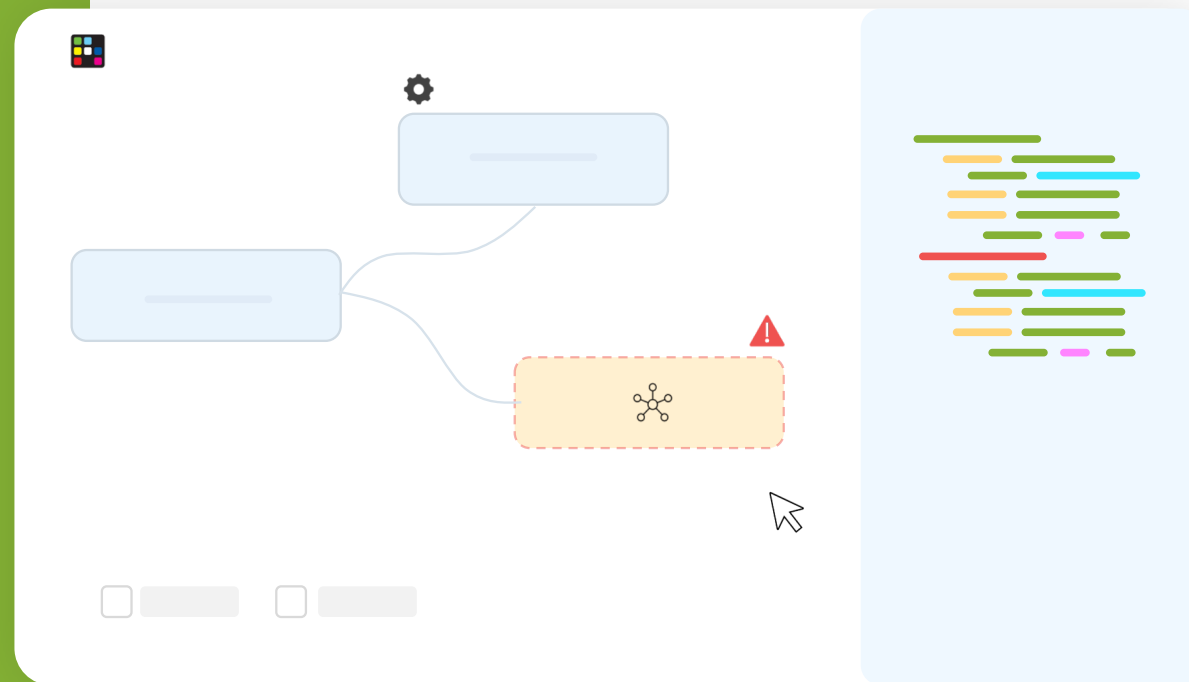
Property Sets

Closed-Loop Validation

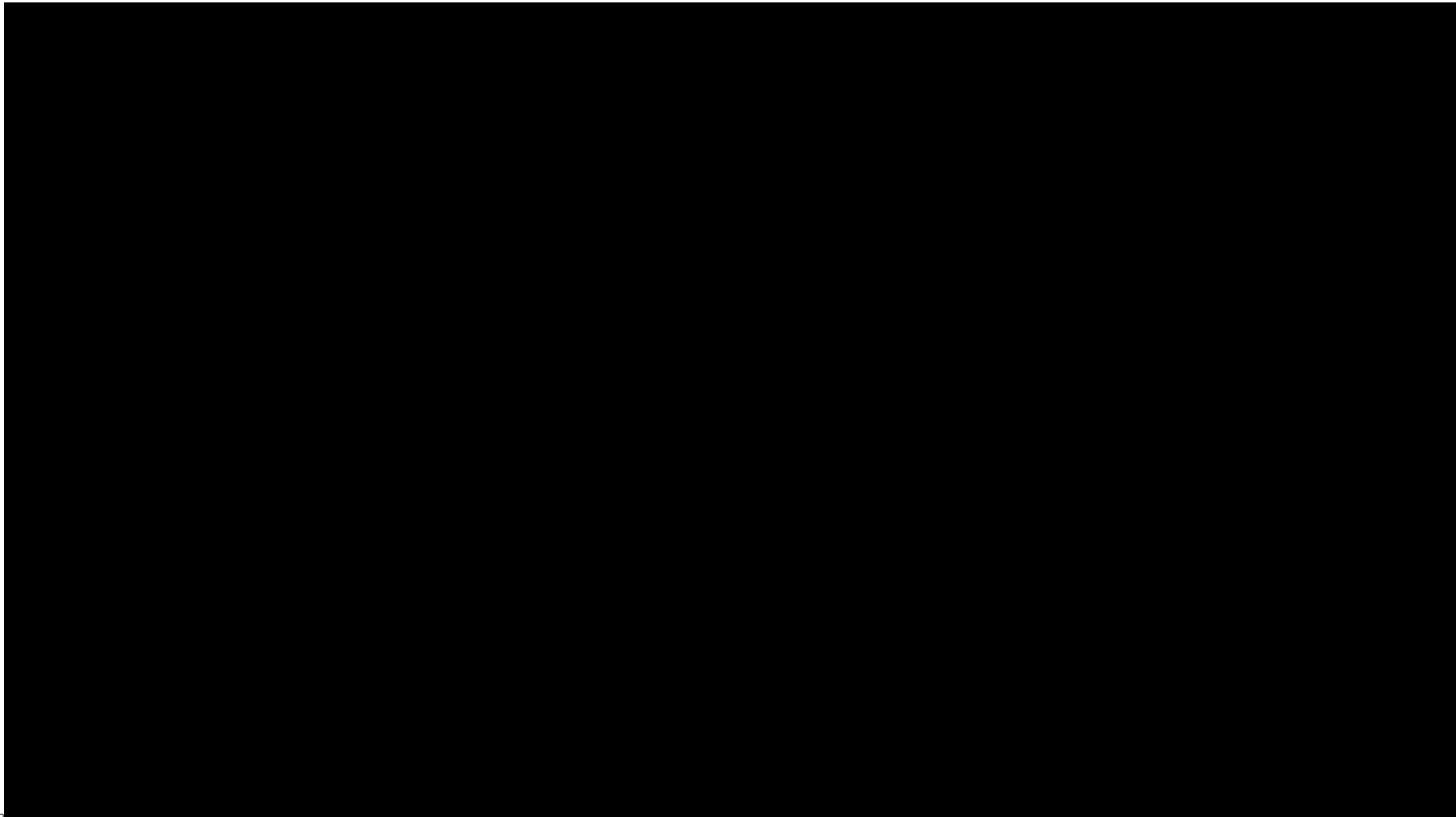
Telemetry, IBA

Time Voyager (Roll Back)

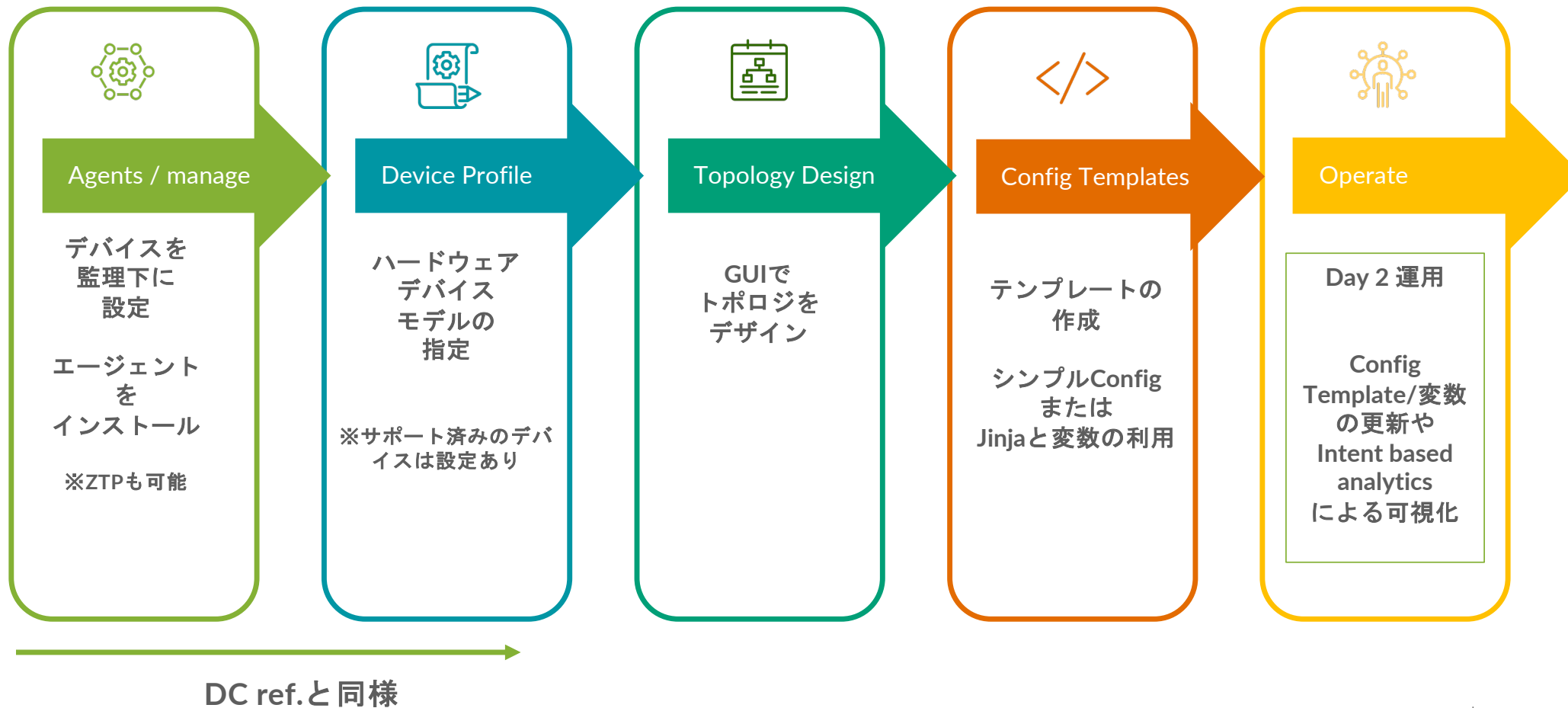
NOS Upgrades, ZTP,



Apstra Operation Overview - 4.1.1 - 4 min video

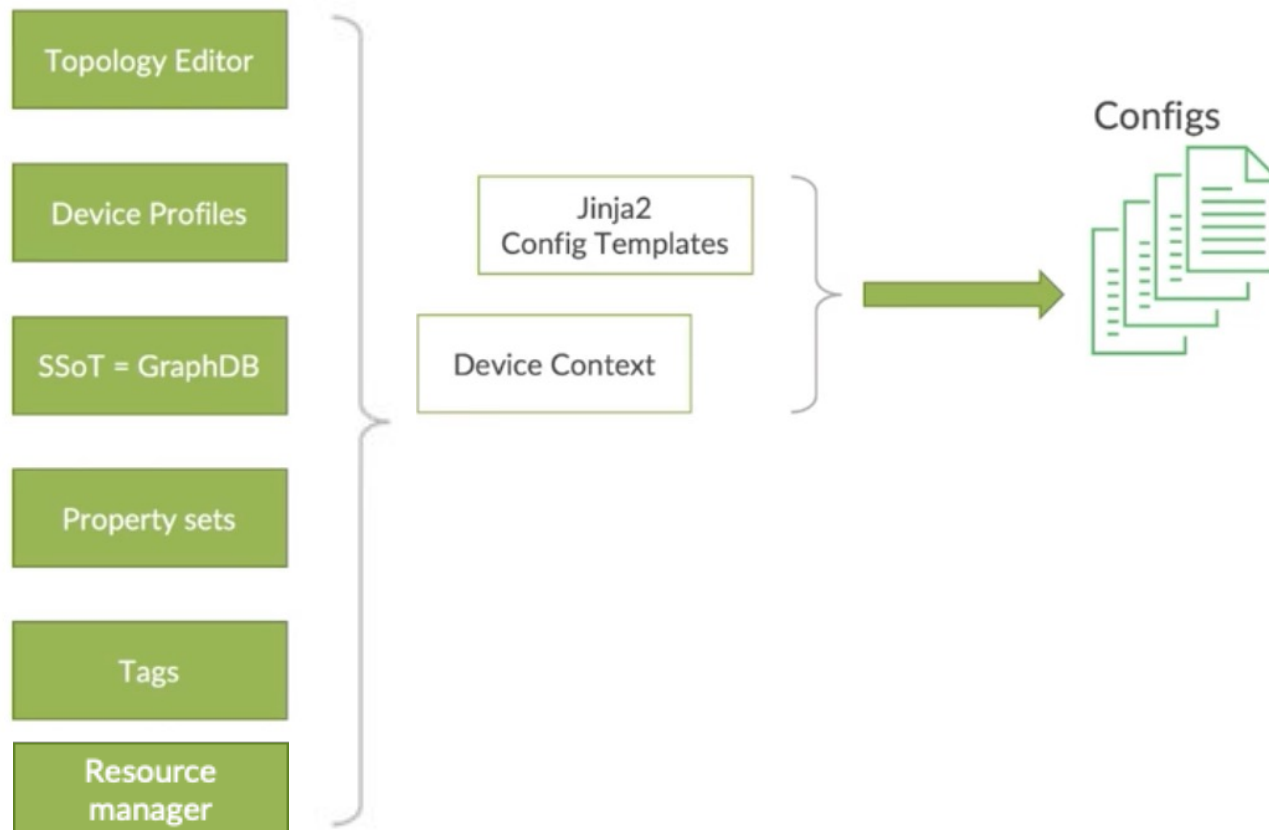


Building a Fabric with Freeform



Device Context & Config Templates

- Network Designer & Config Templatesにより柔軟に設定を自動作成
- Config TemplatesはJinja 2 ベースのテンプレートを準備し、設定の自動生成を行う



Telemetry Realtime Analytics for Freeform

“Dashboard”-ネットワーク全体の健康状態の把握

“Analytics”-BGPセッション、memory, cpuなど各種各種パラメーターを取得可能

The dashboard provides a comprehensive overview of network health. At the top, it shows 'All Probes' with 0 anomalies. Below this, there are sections for 'Fabric' and 'Deployment Status', each with four circular indicators representing different components (Cabling, Interface, Node, Hostname for Fabric; Deployment, Config Dev., Config Re...) and all showing 0 anomalies. The interface includes navigation tabs for Dashboard, Analytics, Staged, Uncommitted, Active, and Time.

This view shows the configuration and data sources for a specific device (172.27.114.92). It includes tabs for Device, Agent, and Pristine Config. Below these are filters for various data sources: Anomalies, Config, Interface, MAC, LLDP, Hostname, and Counters. The interface is clean and organized for easy navigation.

The Analytics page for BGP Session Flapping provides detailed insights into BGP session statuses across all switches. It includes a search bar, filters for FSM State and Status, and a table of session details. The table shows System ID, AF, Dest Asn, Dest Ip, Source Asn, Source Ip, Vrf Name, Flap Count, Flap Count Increment, FSM State, and Status.

System ID	AF	Dest Asn	Dest Ip	Source Asn	Source Ip	Vrf Name	Flap Count	Flap Count Increment	FSM State	Status
52540028C197 oxford-circus	ipv4	23	192.168.0.1	22		default	0	0	established	up
52540028C197 oxford-circus	ipv4	47	192.168.0.2	22		default	0	0	established	up
52540028C197 oxford-circus	ipv4	86	192.168.0.5	22		default	0	0	established	up

Filter selected by all selected only unselected only

<input type="checkbox"/>	Service Name	Service Started?	Interval (s)	Input	Run Count	Success Count	Failure Count	Max Run Count	Execution Time, ms	Waiting Time, ms	Last Run Timestamp	Last Error Timestamp	Error message
<input type="checkbox"/>	ARP	yes	120		2248	2244	0		1236.28	0.25	2022-06-20, 13:26:46		N/A
<input type="checkbox"/>	INTERFACE	yes	120		2244	2243	0		771.95	0.25	2022-06-20, 13:26:46		N/A
<input type="checkbox"/>	DISK UTIL	yes	120		2051	2050	0		304.60	658.90	2022-06-20, 13:26:47		N/A
<input type="checkbox"/>	LLDP	yes	10		26784	26777	0		369.42	0.75	2022-06-20, 13:27:47		N/A

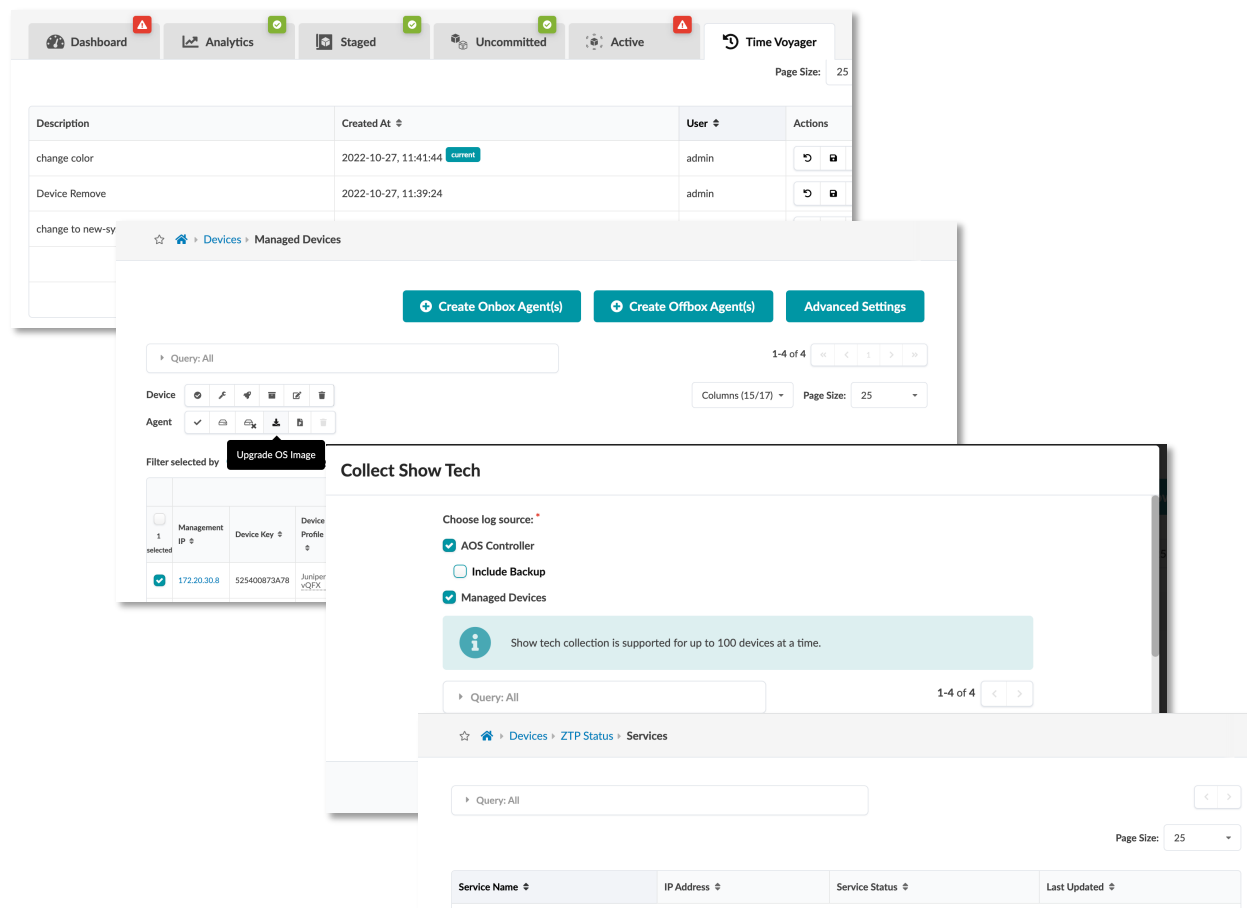
多様な運用オプション機能

Time Voyager (Roll Back)

Device OS Upgrade

Collect Show Tech

ZTP



Capabilities Across All of Apstra

1 SWITCH OR 100'S

豊富な機能

- Zero Touch Provisioning
- OS Upgrades
- RBAC
- Metadata Tags
- Cable Maps Exports

シンプル運用

- Single software solution
- Vendor Agnostic ※
- Optimized UI experience
- Full API programmability
- Turn-Key and Custom Designs

革新的なオペレーション

- Digital Network Designer
- Time Voyager
- Golden Config Validation(DC)
- Analytics and IBA
- Real-Time validations

DC Reference DesignとFreeformの比較

	DC Ref. Design	Freeform
デザイン	Logical Device/Racks/Templatesから作成	トポロジーとプロトコルの制限なし (デバイス依存)
コンフィグ	Juniper/Apstraで動作保証済みの設定を自動生成	必要な設定を検討・検証し、テンプレートを作成した後に自動生成
監視	フルセット	DC Ref Designの一部 + IBA
CLI syntaxの知識	基本不要だがConfigletsには必要	必要
Jinja2の知識	Configletsには必要なことがある	必要(またはjinjaを使わず静的な設定を流し込みも可能)
Pythonの知識	APIには必要	必要な場合がある
Time Voyager	使える	使える
NOS Upgrade	できる	できる
ベンダーサポート	Juniper(JUNOS/EVO)/SONiC/NXOS/EOS	Juniperのサポート済みデバイスのみ(他はroadmap)
Brownfieldサポート	要PS相談	要PS相談(但し既存設定も利用可能)



- Apstra Freeform概要
- Apstra Freeformオペレーション概要
 - 初期セットアップ手順
 - Config Templates
 - Apstra Telemetry
 - オプション機能
- 参考リンク

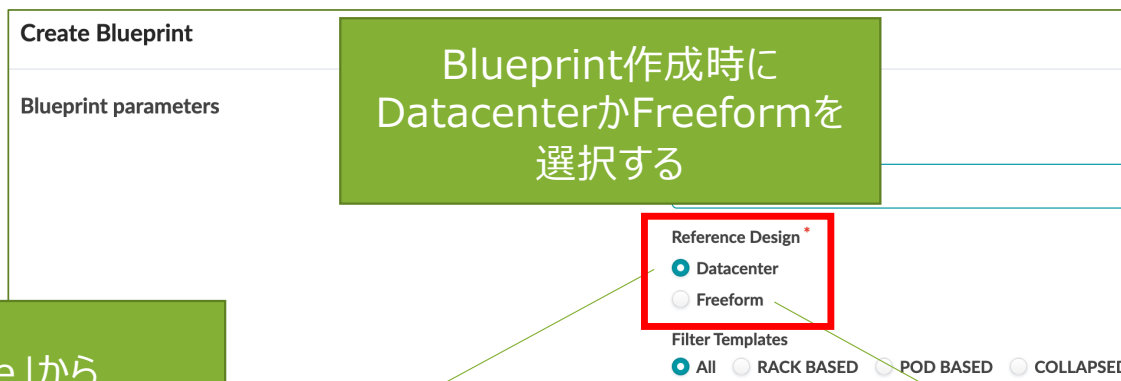
Freeform初期セットアップの流れ

■ 初期セットアップ

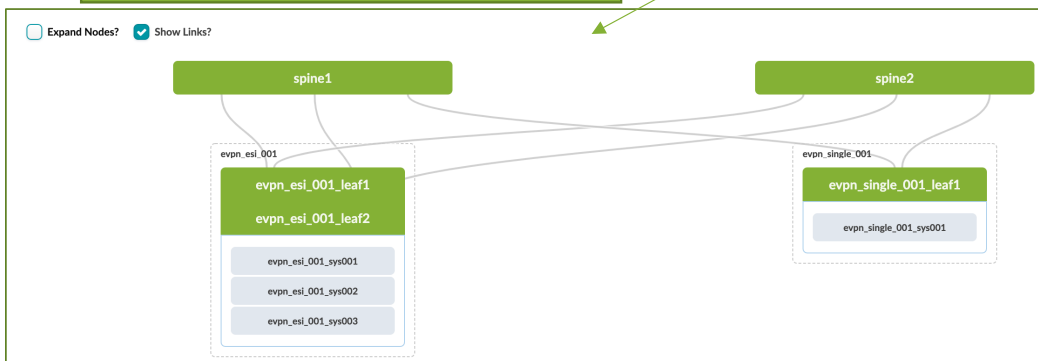
※ Apstraの起動はDC Referenceと同様のため省略

- Blueprint作成時にFreeformを選択
- Managed Device登録
- System作成
- Topology作成
- Topology Link作成
- Device Profiles & System(Agent)アサイン
- Config Templatesアサイン
- Commitでデバイスに反映

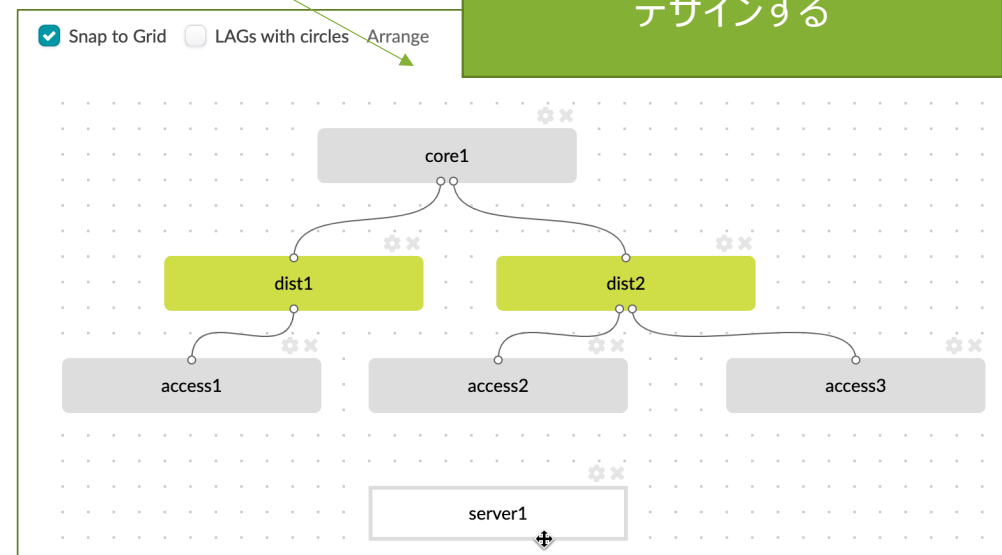
Freeform作成方法



「Template」から作成される



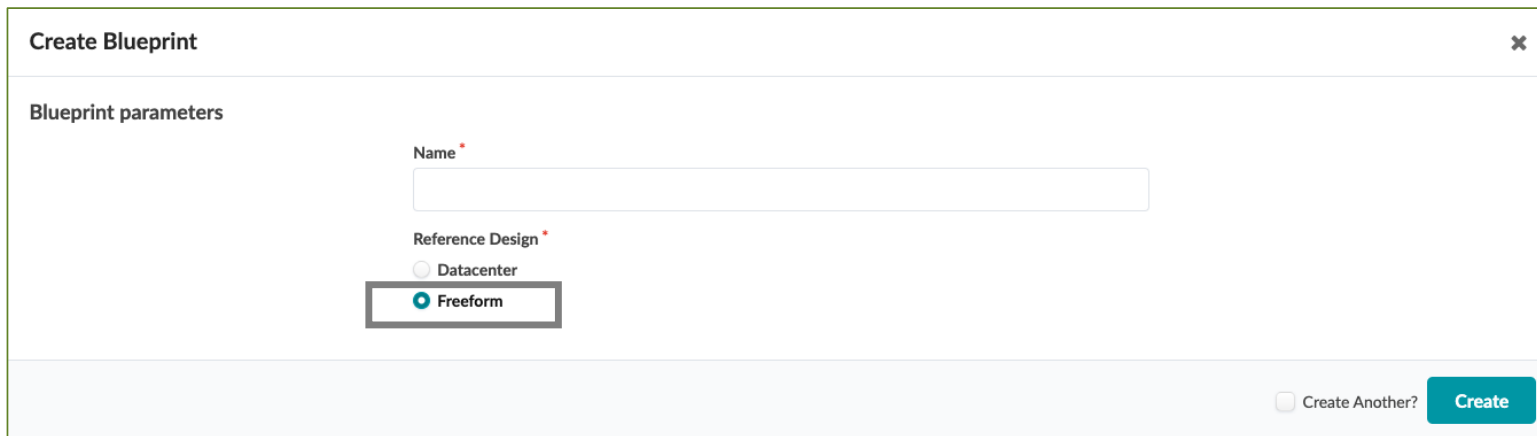
GUI上で自由にデザインする



Blueprint作成時にFreeformを選択

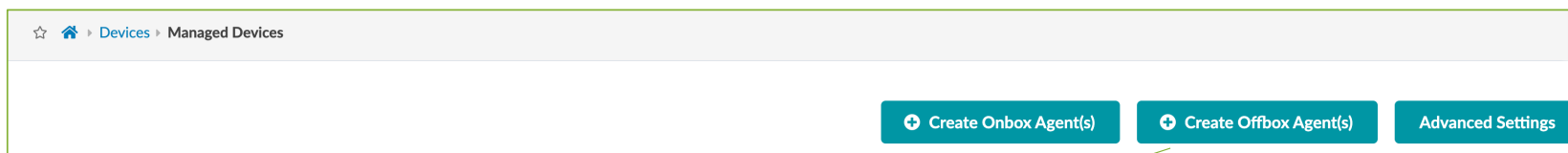
Blueprintを作成するときにDatacenter or Freeformを選択する。

- Blueprint作成時にFreeformを選択



The screenshot shows a 'Create Blueprint' dialog box with a close button (X) in the top right corner. Under the heading 'Blueprint parameters', there is a 'Name' field with a red asterisk and an empty text input box. Below that is the 'Reference Design' section, also with a red asterisk, containing two radio button options: 'Datacenter' and 'Freeform'. The 'Freeform' option is selected and highlighted with a grey border. At the bottom right of the dialog, there is a 'Create Another?' checkbox and a blue 'Create' button.

Managed Device登録 (DC ref/freeform同様手順)



Create System Agent(s)

Agent Parameters 各DeviceのManagement IPをRangeで指定

Device Addresses (25 max) *

172.20.51.12-172.20.51.16

Comma-separated list of hostnames, individual IP addresses, and IP address ranges, e.g. '192.168.1.5-192.168.1.10,mydevice.local' →

172.20.51.12
172.20.51.13
172.20.51.14
172.20.51.15
172.20.51.16

5台のIPが表示されていることを確認

Operation Mode

FULL CONTROL TELEMETRY ONLY

Platform *

Junos

Username *

root

Password *

.....

Agent Profile

Select...

Packages 0

Create

Juniper機器の場合"Junos"

Agentが各DeviceにloginするためのUser/Pass

入力完了後、Create

Device Profilesインポート

Managed Deviceとして登録していない場合、
利用するデバイスのDevice Profilesをインポート

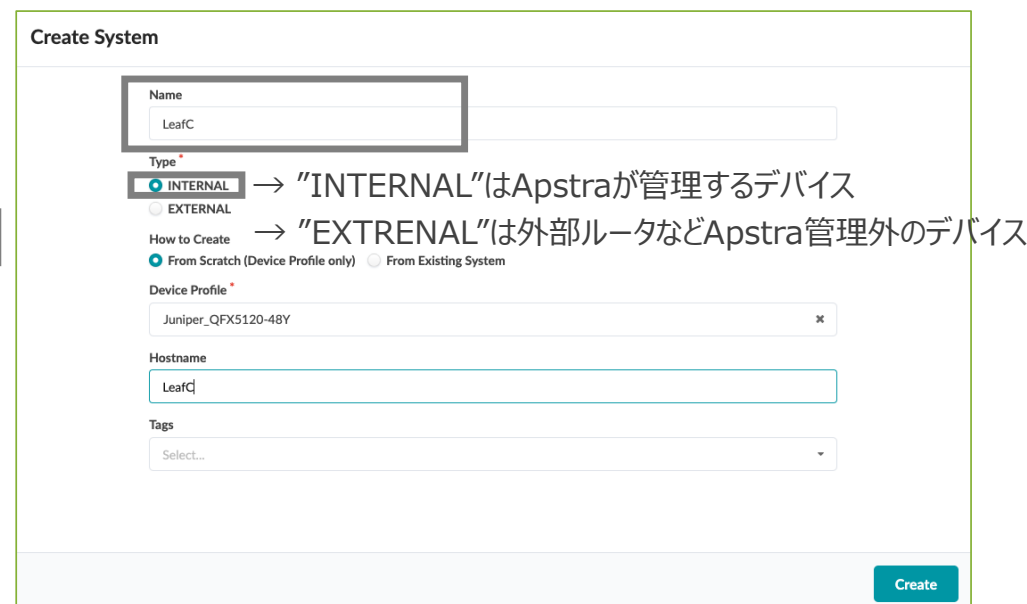
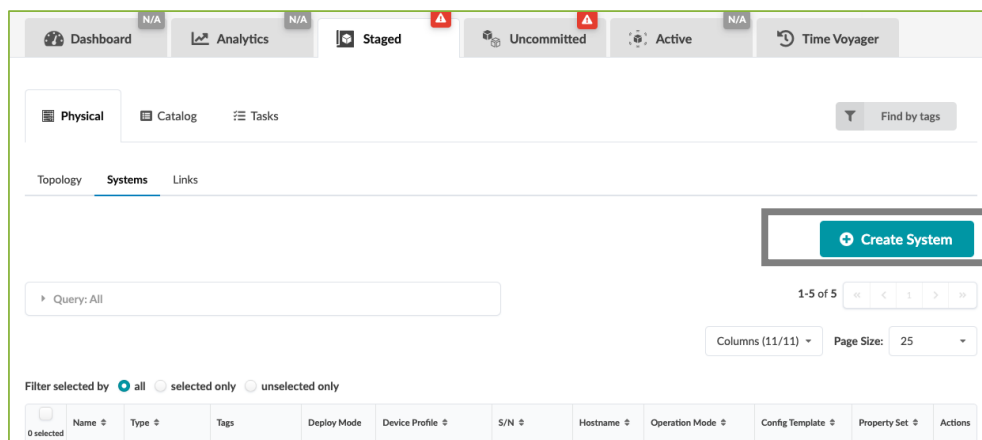
Catalog -> Device Profilesで利用するDevice Profilesをimportする

The screenshot shows the Juniper Networks management interface. The breadcrumb navigation is: Blueprints > New-freeform > Staged > Catalog > Device Profiles. The main navigation bar includes Dashboard, Analytics, Staged, Uncommitted, Active, and Time Voyager. The 'Catalog' tab is selected, and the 'Device Profiles' sub-tab is active. A red box highlights the 'Import Device Profile(s)' button. Below the button is a search query field containing 'Query: All'. The table below shows two existing device profiles:

Name	Manufacturer	Hardware Model	Device Profile Type	OS Family	OS Version	ASIC
Juniper_QFX5120-32C	Juniper	QFX5120-32C.*	monolithic	Junos	(1[89] 2[0-2])\..*	T3
Juniper_QFX5120-48Y	Juniper	QFX5120-48Y.*	monolithic	Junos	(1[89] 2[0-2])\..*	T3

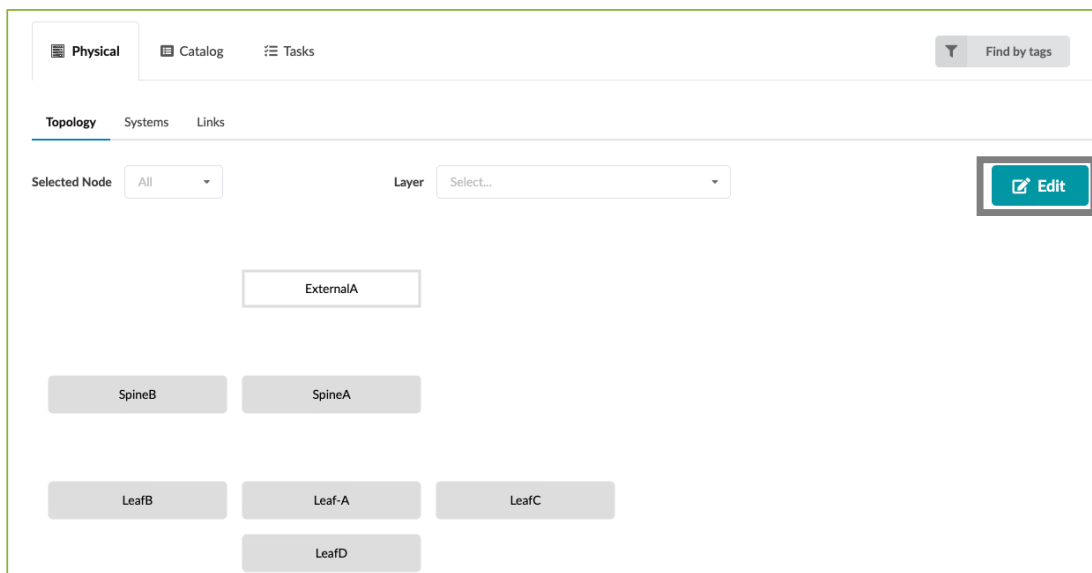
System作成

Topologyで使用する各デバイスをSystemとして作成



Topology 作成

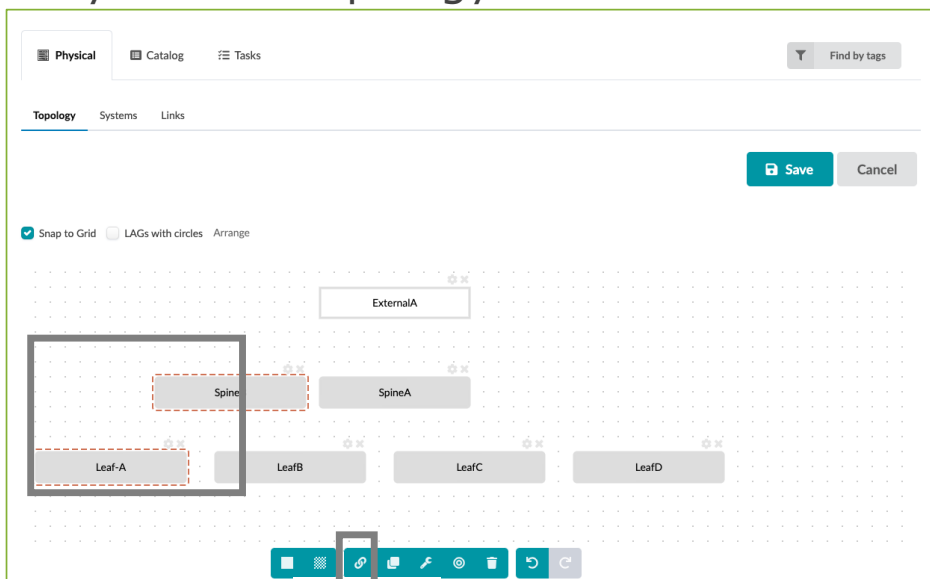
Physical -> Topology画面のEditからノードやリンクを作成可能



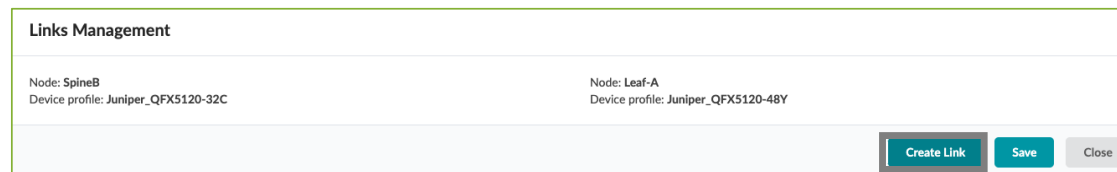
Topology Link作成

Link接続したい2つのスイッチを選択(Alt/Commandを押しながら選択するか、クリックをしながらグラブして対象を選択)、

Physical -> Topology



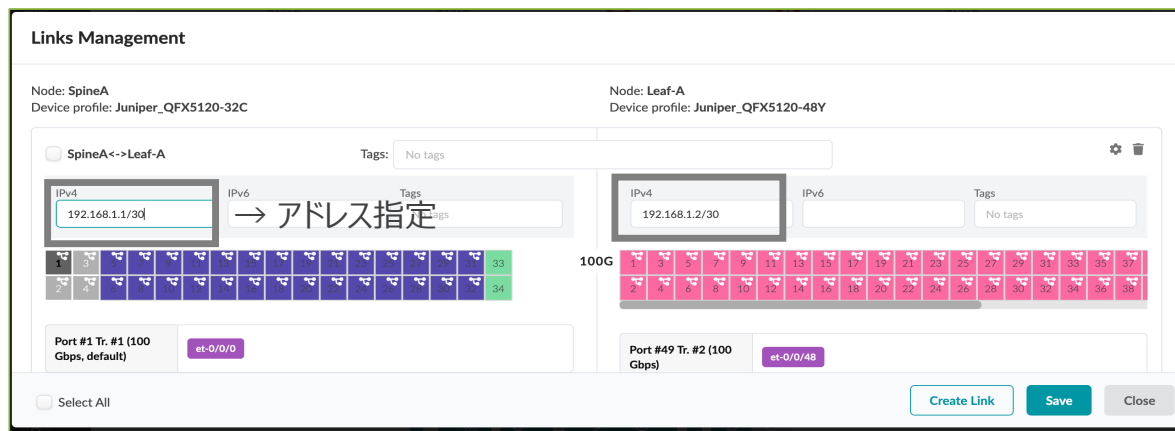
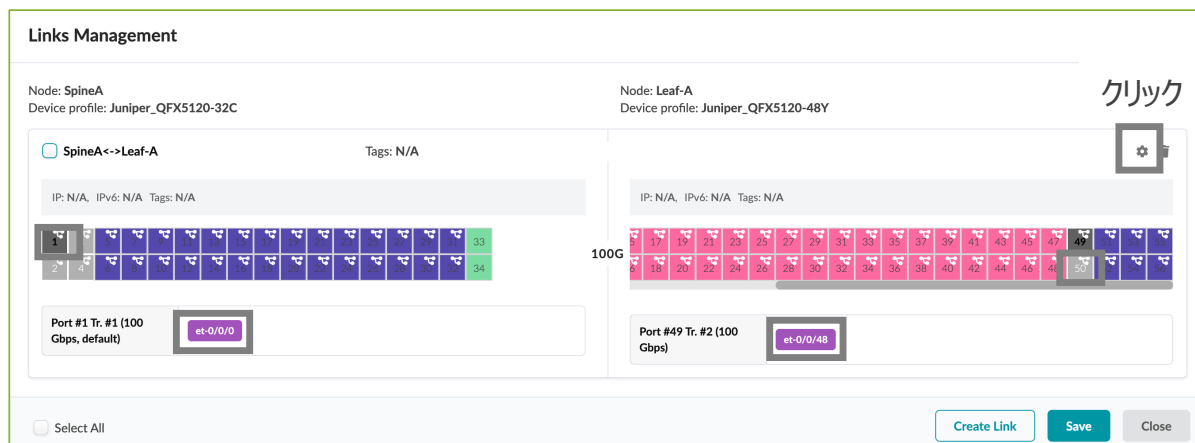
クリック



クリック

Topology Link作成

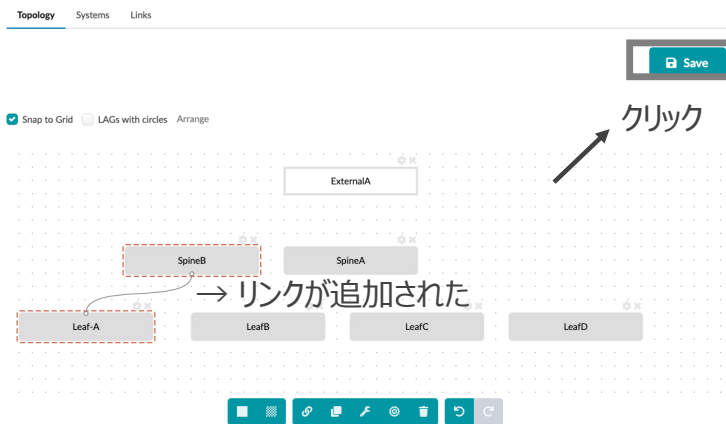
接続するポートとインタフェース名(スピード)を選択し、Create Link



(Option)リンクのアドレス設定
これらのインタフェース名や
アドレス情報は後で使用するDevice
Contextとしてconfigテンプレートの変
数値として利用することが可能となる。

Topology Link作成

Saveで保存



同様に物理接続する各デバイスのリンクとアドレスをそれぞれ設定する

Device Profiles & System(Agent)アサイン

各システムにDevice ProfileとSystem(Managed Device)をアサインする。
ただし、Managed Deviceを登録していない場合は、System(Managed Device)をアサインせずに動作確認することは可能。

Leaf-A

Label
Leaf-A

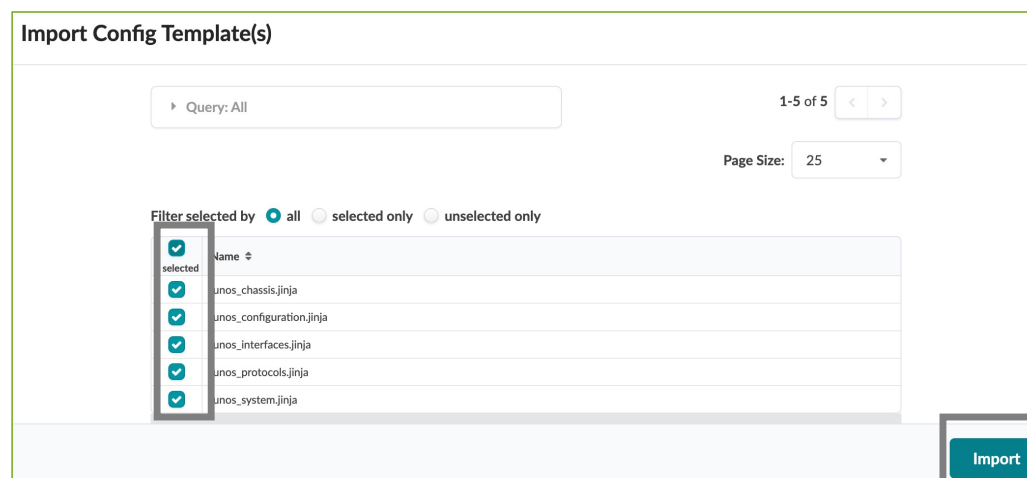
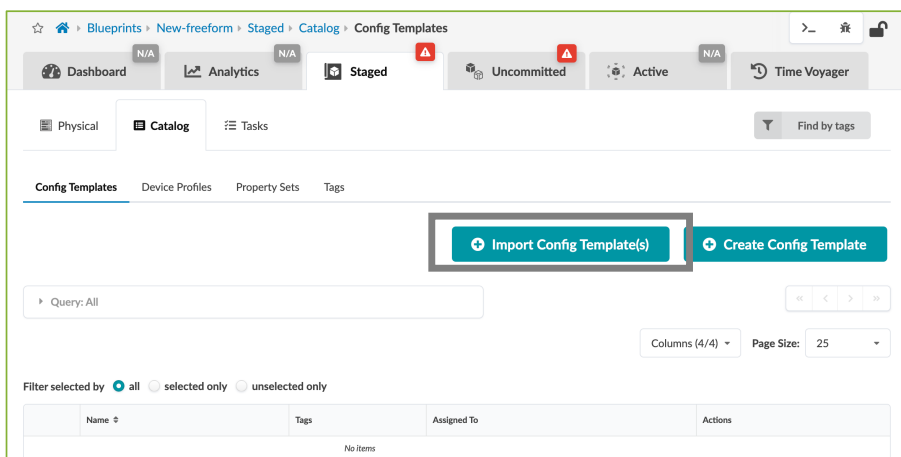
Device Profile
Juniper_QFX5110-32Q

System
WT3717370020 (172.2 7.114.92) -

tags
No tags

Config templateアサイン (import)

各デバイスに既存のconfigテンプレートをアサインする場合は、既存のテンプレートをインポートしアサインする。



※Config templateについては別の章で解説

Config templateアサイン (assignment)

各デバイスにconfigテンプレートをインポートまたは作成し、Jintaテンプレートをアサインする。

The screenshot shows the Juniper Networks configuration interface. The breadcrumb navigation is Blueprints > New-freeform > Staged > Physical > Systems. The top navigation bar includes Dashboard, Analytics, Staged, Uncommitted, Active, and Time Voyager. A 'Create System' button is visible. The main content area shows a table of systems with the following columns: Name, Type, Tags, Deploy Mode, Device Profile, S/N, Hostname, Operation Mode, Config Template, Property Set, and Actions. The table contains 7 rows, with the first row selected. A tooltip 'Update Config Template Assignments' is displayed over the 'Leaf-A' row. The table data is as follows:

Name	Type	Tags	Deploy Mode	Device Profile	S/N	Hostname	Operation Mode	Config Template	Property Set	Actions
ExternalA	EXTERNAL	external	Undeploy	N/A	N/A	ExternalA	UNMANAGED	N/A	N/A	[Trash]
Leaf-A	INTERNAL		Undeploy	Juniper_QFX5120-48Y	Not assigned	LeafA	FULL CONTROL	Not assigned	Not assigned	[Trash]
LeafB	INTERNAL		Undeploy	Juniper_QFX5120-48Y	Not assigned	LeafB	FULL CONTROL	Not assigned	Not assigned	[Trash]
LeafC	INTERNAL		Undeploy	Juniper_QFX5120-48Y	Not assigned	LeafC	FULL CONTROL	Not assigned	Not assigned	[Trash]
										[Trash]
										[Trash]

Configuration 確認

各デバイスのRendered configを確認すると、テンプレートで定義した変数(interfaceやアドレス等)が補完され適用されていることがわかる。

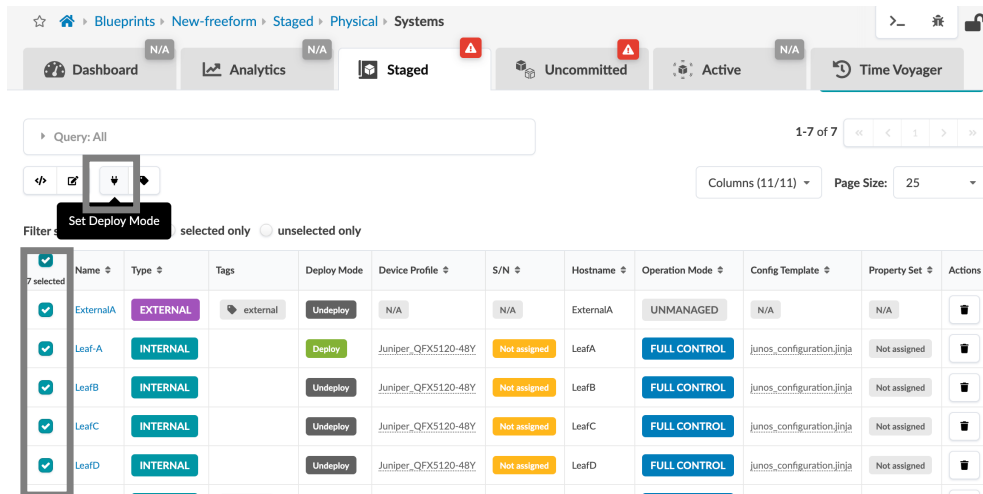
The screenshot shows the Juniper configuration management interface. At the top, there is a navigation bar with 'Blueprints > New-freeform > Staged > Physical > Systems > Details'. Below this, there are tabs for 'Dashboard', 'Analytics', 'Staged', 'Uncommitted', 'Active', and 'Time Voyager'. The main area displays a network diagram with 'Leaf-A' connected to 'SpineA' and 'SpineB'. The 'Leaf-A' device has two interfaces: 'et-0/0/48' and 'et-0/0/49'. The 'SpineA' and 'SpineB' devices also have 'et-0/0/0' interfaces. A configuration panel for 'Leaf-A' is open, showing 'Deploy Mode' (undeploy), 'S/N' (Not assigned), 'Hostname' (LeafA), and 'Config' (Rendered, Incremental, Device Context).

Leaf-A Rendered Config Preview

```
1 system {
2   host-name LeafA;
3 }
4 interfaces {
5   replace: et-0/0/48 {
6     description "facing_spinea:et-0/0/0";
7     mtu 9216;
8     unit 0 {
9       family inet {
10        address 192.168.1.2/30;
11      }
12    }
13  }
14  replace: et-0/0/49 {
15    description "facing_spineb:et-0/0/0";
16    mtu 9216;
17    unit 0 {
18      family inet {
19        address 192.168.2.1/30;
```

Deploy Mode “Deploy”

各デバイスのDeploy Modeを“Deploy”に変更。
複数選択し、Deploy Modeに変更することも可能。



Commit

Commitにより設定をデバイスに反映。

☆ Home > Blueprints > New-freeform > Uncommitted > Logical Diff

Revert Commit

Dashboard Analytics Staged Uncommitted Active Time Voyager

Logical Diff Full Nodes Diff Build Errors Warnings

Query: All 1-2 of 2

Page Size: 25

Type	Action	Name
Config Template	ADDED	device_property-set_sample.jinja
Config Template	CHANGED	junos_configuration.jinja



- Apstra Freeform
 - 初期セットアップ手順
 - **Config Templates**
 - Apstra Telemetry
 - オプション機能
- 参考リンク

Config Templatesオペレーション記載内容

Config Templates概要

- Config Templateサンプル

新規Config Template作成 Day-0

- Jinja reference document
- プリインストールのデフォルトサンプル
- 複数Config Templatesの利用方法

変数値の利用

- Device Context概要
- Device Context変数値確認方法
- Property Sets概要 – 静的変数値
- Property Sets利用方法
- Property Setsサンプル
- Resource Management – 動的変数値利用方法

Config Template 編集方法 Day-2/Advanced編

- Config Template編集
- クローン方法/デバイス毎の異なるConfig Template
- Day2オペレーションサンプル – vlan追加

Advanced編

- Day2オペレーションサンプル – Drain Modeサンプル
- タグベースの動的なコンフィグ生成

Config Templates with Jinja2 Templates概要

Direct Config

```
1 system {
2   host-name piccadilly-circus;
3 }
4 interfaces {
5   replace: xe-0/0/0 {
6     unit 0 {
7       description "facing_oxford-circus:xe-0/0/0";
8       family inet {
9         address 192.168.0.1/31;
10      }
11    }
12  }
13  replace: xe-0/0/1 {
14    unit 0 {
15      description "facing_green-park:xe-0/0/2";
16      family inet {
17        address 192.168.0.9/31;
18      }
19    }
20  }
21  replace: xe-0/0/2 {
22    unit 0 {
```



Jinja2 Template

```
1 system {
2   host-name {{hostname}};
3 }
```



Nesting - Composable

複数jinjaテンプレート利用

```
1 {% include "system.jinja" %}
2
3 {% include "interfaces.jinja" %}
4
5 {% include "protocols.jinja" %}
```

※ Jinja2 Templates形式でなく JUNOS CLIをそのまま貼り付けてデバイスに適用することも可能

Config Templates with Jinja2 プレビュー

Blueprint作成後に、Templateから各デバイスにどのようなconfigが生成されるか動的に確認可能
Blueprint > Staged > Catalog > Config Templates >

Innovative Interactive Editor

Update Config Template

Name *
interfaces.jinja

Config Preview

Device Context ? Jinja function reference System: Bond-Street Apply Mode: Complete Preview

Template Text *

```
1  {% %}
2  interfaces {
3  {% for interface_name, iface in interfaces.iteritems() %}
4      replace: {{ interface_name }} {
5          unit 0 {
6              description "{{iface['description']}}";
7              {% if iface['ipv4_address'] and iface['ipv4_prefixlen'] %}
8                  family inet {
9                      address {{iface['ipv4_address']}}/{{iface['ipv4_prefixlen']}};
10                 }
11             {% endif %}
12         }
13     }
14 {% endfor %}
15     replace: lo0 {
16         unit 0 {
17             family inet {
18                 address {{ property_sets.data[this_router]['loopback'] }}/32;
19             }
20         }
21     }
```

Preview (as rendered for device Bond-Street)

Config is empty.

Save Changes Update

新規Config template作成

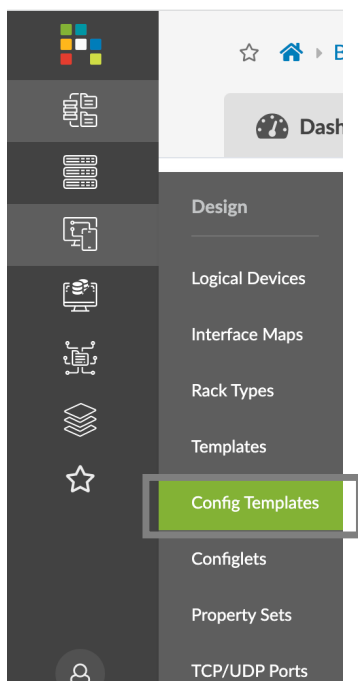
テンプレートを新規作成する場合は“Create Config Template”で作成。

The screenshot shows the Juniper Apstra interface for managing Config Templates. The breadcrumb navigation is "Design > Config Templates". A "Freeform Only" filter is active. A red box highlights the "Create Config Template" button. The left sidebar shows the "Design" menu with "Config Templates" selected. The main content area shows a table of existing templates with columns for name and actions.

Query: All	1-5 of 5	Page Size: 25	Actions
inja			[Copy] [Edit] [Delete]
ation.jinja			[Copy] [Edit] [Delete]
es.jinja			[Copy] [Edit] [Delete]
s.jinja			[Copy] [Edit] [Delete]
inja			[Copy] [Edit] [Delete]





























Config Templates プリインストールデフォルト サンプル

予めプリロードされたConfig Templatesを活用・参照し、Templates作成の利活用可能。



既存テンプレートの一覧

デフォルトでjunos_configuration.jinjaなどいくつかのテンプレートが用意されているためそれらを複製し作成することも可能。

Name	Actions
junos_chassis.jinja	   
junos_configuration.jinja	   
junos_interfaces.jinja	   
junos_ospf-static.jinja	   
junos_protocols.jinja	   
junos_system.jinja	   
ospf.jinja	   

Config Templateの利用パターン

Jinja2テンプレートで構文を利用する方法と、シンプルにJUNOS設定をそのまま利用するケースがあり。

1. Jinja2構文活用して様々な変数や繰返処理を使ってテンプレートを作ることも、
2. 変数を使用せず静的にJUNOS設定をそのまま定義することでもテンプレートを作成することも可能。

1. Jinja2構文活用 (advanced)

Create Config Template

unnumbered.jinja

Device Context: Leaf-A System: Leaf-A Apply Mode: Complete Preview

Template Text *

```
1 {% set dev = namespace(facae=[], peer_iface=[], ae_count=0, vlans=[], rid=None, loopback=None, asn=None) %}
2 {##### Parse the devices variable #####}
3 {% if property_sets.get('devices', []).get('hostname', []).get('rid') %}
4 {% set dev.rid = property_sets.devices[hostname].rid %}
5 {% endif %}
6 {% if property_sets.get('devices', []).get('hostname', []).get('loopback') %}
7 {% set dev.loopback = property_sets.devices[hostname].loopback %}
8 {% endif %}
9 {% if property_sets.get('devices', []).get('hostname', []).get('asn') %}
10 {% set dev.asn = property_sets.devices[hostname].asn %}
11 {% endif %}
12 {##### Parse the switch_interfaces variable #####}
13 {% for system, iface in property_sets.get('switch_interfaces', []).iteritems() if hostname == system or system.startswith('{{ dev.asn }}' % hostname) or system
    .endswith('{{ dev.asn }}' % hostname) %}
```

Create

2. 静的なJUNOS設定(変数は最小限) (no-code)

Create Config Template

Name: static-ospf.jinja

Template Text *

```
1 protocols {
2   ospf {
3     area 0.0.0.0 {
4       interface xe-0/0/0.0;
5     }
6   }
7 }
```

Create Another? Create

複数のConfig Template 利用

- 複数のテンプレートを1台のデバイスに適用したい場合、ひとつ親となるテンプレートを作成し、そこに複数のテンプレートを指定することで適用。
 - デフォルトサンプルのjunos_configuration.jinjaの中身を確認すると他のテンプレートをネस्टドし、テンプレートを作成していることが確認できる。
- ※{{変数}}については次ページにて記載。

Expanded View Compact View

Common Parameters

Name	junos_configuration.jinja
Tags	
Assigned To	0 systems

親となる、デバイスに適用するテンプレート

Text

```

1 {% block system %}
2   {% include "junos_system.jinja" %}
3 {% endblock system %}
4
5 {% block chassis %}
6   {% include "junos_chassis.jinja" %}
7 {% endblock chassis %}
8
9 {% block interfaces %}
10  {% include "junos_interfaces.jinja" %}
11 {% endblock interfaces %}
12
13 {% block protocols %}
14  {% include "junos_protocols.jinja" %}
15 {% endblock protocols %}
16

```

複数のjinjaテンプレートを利用する指定をすることで、ひとつのコンフィグレーションとして生成可能になる

Name	junos_interfaces.jinja
Tags	
Assigned To	0 systems

```

1 {% for interface_name, iface in interfaces.iteritems() %}
2   {% if loop.first %}
3   interfaces {
4     {% endif %}{# loop.first #}
5     {% set is_l2 = function.is_l2_interface(iface) %}
6     {% set is_part_of_parent_intf = iface.get("part_of") %}
7     {# Ensure any Layer 3 interface, interfaces with bond members are rendered in configuration. #}
8     {% if iface.get("description") or iface.get("ipv4_address") or iface.get("ipv6_address") or iface.get("part_of") or iface.get("composed_of") %}
9     replace: {{interface_name}} {
10      {% if iface.get("operation_state") == 'down' %}
11      disable;
12      {% endif %}{# iface.get("operation_state") == 'down' #}
13      {% if iface.get("description") %}
14      description "{{iface["description"]}}";
15      {% endif %}{# iface.get("description") #}
16      {% if portSetting is defined and portSetting.get(interface_name, {}).get("interface", {}).get("speed") %}
17      speed {{ portSetting[interface_name]["interface"]["speed"] }};
18      {% endif %}{# portSetting is defined and portSetting.get(interface_name, {}).get("interface", {}).get("speed") #}
19      {% if portSetting is defined and portSetting.get(interface_name, {}).get("interface", {}).get("link_mode") %}
20      link-mode {{ portSetting[interface_name]["interface"]["link_mode"] }};
21      {% endif %}{# portSetting is defined and portSetting.get(interface_name, {}).get("interface", {}).get("link_mode") #}

```

Config Templatesオペレーション記載内容

Config Templates概要

- Config Templateサンプル

新規Config Template作成 Day-0

- Jinja reference document
- プリインストールのデフォルトサンプル
- 複数Config Templatesの利用方法

変数値の利用

- **Device Context概要**
- **Device Context変数値確認方法**
- **Property Sets概要 – 静的変数値**
- **Property Sets利用方法**
- **Property Setsサンプル**
- **Resource Management 利用方法**

Config Template 編集方法 Day-2/Advanced編

- Config Template編集
- クローン方法/デバイス毎の異なるConfig Template
- Day2オペレーションサンプル – vlan追加

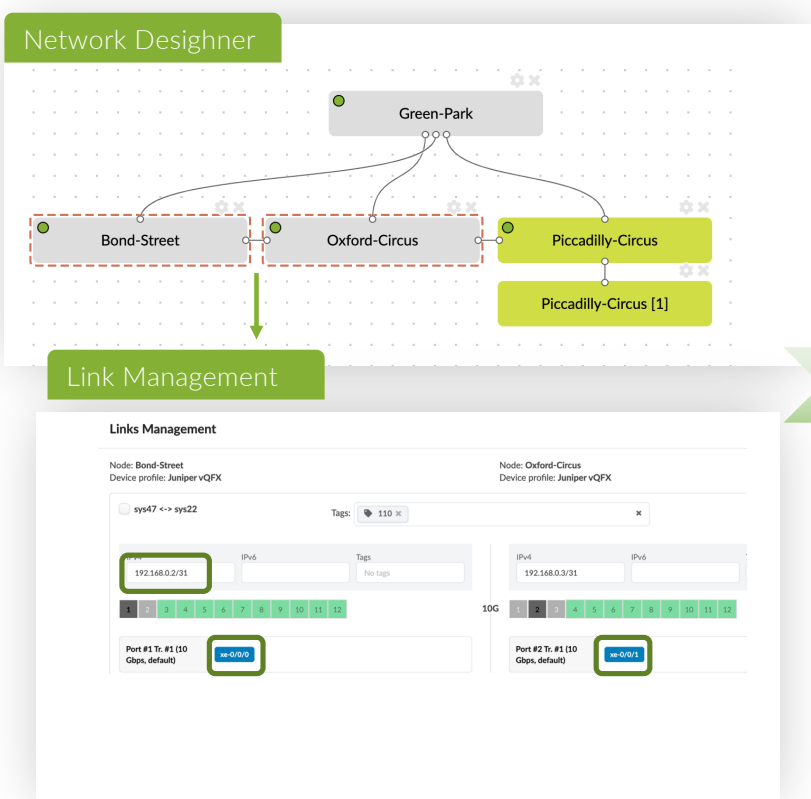
Advanced編

- Day2オペレーションサンプル – Drain Modeサンプル
- タグベースの動的なコンフィグ生成

Device Context 変数値の利用の概要

Network DesignerやDevice Profile の情報を変数値として利用しコンフィグの自動生成

例:Network Designerでアサインしたアドレスから動的にコンフィグ作成



```
Device Context
└─ Interfaces
  {
    xe-0/0/0
    {
      link_tags [ ... ]
      neighbor_interface { ... }
      port_settings { ... }
      composed_of: []
      description: "facing_oxford-circus:xe-0/0/1"
      id: "fwXE70AvM9HfzuocY04"
      if_type: "ethernet"
      index: 0
      ipv4_address: "192.168.0.2"
      ipv4_prefixlen: 31
      ipv6_address: null
      ipv6_prefixlen: null
      label: null
      lag_mode: null
      name: "xe-0/0/0"
      part_of: null
      port_channel_id: null
      port_id: 1
      role: "internal"
      tags: []
      transformation_id: 1
    }
  }
  xe-0/0/1 { ... }
  xe-0/0/10 { ... }
```

```
Jinja2 Template
1 {% set this_router=hostname %}
2 interfaces {
3   {% for interface_name, iface in interfaces.iteritems() %}
4     replace: {{ interface_name }} {
5       unit 0 {
6         description "{{iface['description']}}";
7         {% if iface['ipv4_address'] and iface['ipv4_prefixlen'] %}
8           family inet {
9             address {{iface['ipv4_address']}}/{{iface['ipv4_prefixlen']}};
10          }
11        {% endif %}
12      }
13    }
14  }
```

```
Config
Bond-Street Rendered Config Preview
7 interfaces {
8   replace: xe-0/0/0 {
9     unit 0 {
10      description "facing_oxford-circus:xe-0/0/1";
11      family inet {
12        address 192.168.0.2/31;
13      }
14    }
15  }
```

KB:[Juniper Apstra] Jinja Configlets using dynamic data from deviceModel

https://juniper.lightning.force.com/lightning/r/Knowledge__kav/ka03c00000001P6IAAC/view

Device Context 変数値の利用サンプル

例として以下のように{{hostname}}としてConfig Templateに設定した変数が、“LeafA”の値として設定に反映されていることが確認できる。

Name	junos_system.jinja
Tags	
Assigned To	0 systems

```
1 {% if hostname is defined and hostname %}
2 sys
3   host-name {{{hostname}}}
4 }
5 {% endif %}
6
```

Device Context

```
▶ all_resources { ... }
▶ interfaces { ... }
  aos_version: "4.1.1"
  configured_system_type: "internal"
  deploy_mode: "deploy"
  hostname: "LeafA"
  id: "s0TfSty6fNKtBCumWG8"
  management_ip: null
  model: "Juniper_QFX5120-48Y.*"
  name: "Leaf-A"
  os_family: "Junos"
  property_sets: {}
  reference_architecture: "freeform"
  resources: []
  routing_instance_supported: true
  system_tags: []
  system_type: "internal"
```

Leaf-A Rendered Config Preview

```
1 system {
2   host-name LeafA;
3 }
4 interfaces {
5   replace: et-0/0/48 {
6     description "facing_spinea:et-0/0/0";
7     mtu 9216;
8     unit 0 {
9       family inet {
10        address 192.168.1.2/30;
11      }
12    }
13  }
14  replace: et-0/0/49 {
15    description "facing_spineb:et-0/0/0";
16    mtu 9216;
17    unit 0 {
18      family inet {
19        address 192.168.2.1/30;
```

Device Context確認 (デバイス変数値)

各デバイスの変数/パラメーターは、Device Contextから確認可能。
このインタフェースやアドレスはTopologyで設定したものを変数値として利用可能。

The screenshot displays the Juniper Networks management interface. The top navigation bar includes 'Dashboard', 'Analytics', 'Staged', 'Uncommitted', 'Active', and 'Time Voyager'. The main area shows a network diagram with 'Leaf-A' and two 'Spine' devices (SpineA and SpineB). The 'Device Context' panel is open, showing fields for 'Deploy Mode' (undeploy), 'S/N' (Not assigned), 'Hostname' (LeafA), and 'Config' (Rendered, Incremental, Device Context).

Device Context

```
▶ all_resources { ... }
▼ interfaces
{
  ▼ et-0/0/48
  {
    ▶ neighbor_interface { ... }
    composed_of: [ ]
    description: "facing_spine:et-0/0/0"
    id: "3N7xaLPB3c2g16Tt-HY"
    if_type: "ethernet"
    index: 0
    ipv4_address: "192.168.1.2"
    ipv4_prefixlen: 30
    ipv6_address: null
    ipv6_prefixlen: null
    label: null
    lag_mode: null
    link_tags: [ ]
```

Property Sets 概要

各デバイス固有の変数値を設定し、Jinja2で動的にアサインしのコンフィグの自動生成

Property Sets

例:各デバイス毎のas番号をproperty setとして定義し、動的にコンフィグの生成

Name	data
Values	<pre>Properties └─ { 4 items └─ bond-street : { 2 items └─ asn : 47 └─ loopback : "10.0.0.2" } └─ piccadilly-circus : { 2 items └─ asn : 23 └─ loopback : "10.0.0.1" } └─ green-park : { 2 items └─ asn : 86 └─ loopback : "10.0.0.3" } └─ oxford-circus : { 2 items └─ asn : 22 └─ loopback : "10.0.0.0" } }</pre>

Jinja2 Template

```
46 bgp {
47   group external-peers {
48     type external;
49     export send-direct;
50   }
51   {% for interface_name, iface in interfaces.iteritems() if iface.get('ipv4_address') and iface.get
52     ('neighbor_interface', {}).get('ipv4_address') %}
53     {% set peer_hostname=iface.neighbor_interface.system_hostname %}
54     neighbor {{ iface.neighbor_interface.ipv4_address }} {
55       peer-as {{ property_sets.data[peer_hostname]['asn'] }};
56       export add-med {{ iface.link_tags[0] }};
57     }
58   {% endfor %}
59 }
60
61 routing-options {
62   autonomous-system {{ property_sets.data[this_router]['asn'] }};
63 }
```

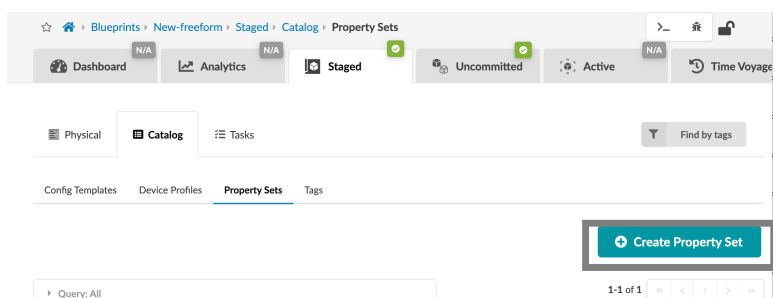
Config

Bond-Street Rendered Config Preview

```
123 bgp {
124   group external-peers {
125     type external;
126     export send-direct;
127     neighbor 192.168.0.3 {
128       peer-as 22;
129       export add-med-110;
130     }
131     neighbor 192.168.0.7 {
132       peer-as 86;
133       export add-med-177;
134     }
135   }
136 }
137 }
138 routing-options {
139   autonomous-system 47;
140 }
```


Property Sets利用方法

Config Template内の変数は、Device Contextから利用できる変数の他に Blueprint内のProperty Setsとして変数定義も可能。



Create Property Set

System [?]
Select... → 特定のデバイスを指定しない場合は全デバイスが対象

Input Type
 Builder Editor

Properties
{ 1 item + → "Builder"の場合、" + "で変数名を追加し、
SPINE1 : NULL ✎ ✖
}

Create

Property Setサンプル

例として以下のように各デバイスに対して変数と値を定義し、ridの値を各デバイスに適用することも可能。これらを活用し、junos設定全体のテンプレート定義を行うこともできる。

Update Property Set

Name *

devices

System

all systems

Input Type

Builder Editor

Properties

4 items

- SpineA : { 3 items }
 - rid : "10.1.1.1"
 - asn : 65001
 - loopback : "10.1.1.1"
- LeafB : { 3 items }
 - rid : "10.1.1.4"
 - asn : 65004

Update Config Template

Name

device_property-set_sample.jinja

Config Preview

Device Context System:

Template Text *

```
1 {% set dev = namespace(rid=None) %}
2
3 {% if property_sets.get('devices', {}).get(hostname, {}).get('rid') %}
4   {% set dev.rid = property_sets.devices[hostname].rid %}
5 {% endif %}
6
7 {
8   routing-options {
9     {% if dev.rid %}
10    router-id {{ dev.rid }};
11    {% endif %}
12  }
13 }
14
```

Preview (as rendered for device LeafB)

```
1
2 {
3   routing-options {
4     router-id 10.1.1.4;
5   }
6 }
7
```

Preview (as rendered for device SpineA)

```
1
2 {
3   routing-options {
4     router-id 10.1.1.1;
5   }
6 }
7
```

Resource Management in Freeform

4.1.2以前は変数に対する値はproperty setまたはDevice Contextから静的にアサインする必要があったが、Freeformのリソースアロケーションにより、リソースプールから動的な割当が可能となりました。

Resource Management概要 ~動的変数値割当~

リソースのレンジから動的に各デバイスシステムやリンクにアドレスを割り当てjinjaからアサイン。

例:リソースレンジから各デバイス毎のas番号を動的にアサインし、動的にコンフィグの生成

Allocation Group

Name	Type	Stats	Assigned Resource
as	ASN	6/6	Private-64512-64534

Group and Resource

Name	Type	Value	Generated By	Allocated From	Assigned To	Actions
tokyo-as	ASN	64513	tokyo-as	as	Spine1	
tokyo-as	ASN	64515	tokyo-as	as	sys-002	
tokyo-as	ASN	64512	tokyo-as	as	sys-003	

Jinja2

Update Config Template

Jinja function reference

Template Text *

```

7 }
8 }
9
10
11 {% set local_asn = function.get_resource_value(resources, 'tokyo-as', 'tokyo-group') %}
12 routing-options {
13   autonomous-system {{ local_asn }};
14 }

```

Config

Preview (as rendered for device Spine1)

```

8 }
9 routing-options {
10   autonomous-system 64513;
11 }
12

```

Resource Management in Freeform

変数の生成と割当は作成は以下の手順で行う。それぞれの手順を次頁から記載する。

1. 値のレンジ設定

[Resource]

2. 上記Resourceを該当Blueprintで有効化 - Allocation Group

[Blueprint] -> [Staged] -> [Create Allocation Group]

3. [Group]を作成し値を割り当てる対象を生成 - Group / Group Generator

[Blueprint] -> [Staged] -> [Resource Management] → [Group / Group Generator]

※各デバイス単位や、各リンク単位等で割当ることが可能。

3-1:デバイス単位の割当、3-2:リンク単位での割当の例を示す。

4. [Resource]の値を、作成した[Group]に割り当て - Resource Generator

[Blueprint] -> [Staged] -> [Resource Management] → [Resource Generator]

5. ConfigTemplateで変数の指定

1.Resource – 変数値の指定

DC. Referenceと同様に[Resource]にて利用する値のレンジを設定。
ここでは例として、1.ASNを指定するサンプルを示す。

The screenshot displays the Juniper Apstra interface for managing ASN Pools. On the left, a navigation sidebar highlights the 'Resources' section. The main content area shows a table of existing ASN Pools and a 'Create ASN Pool' form.

Pool Name	Total Usage	Range Usage	Status	Actions
Private-64512-65534	1.08%	1.08%	IN USE	[Edit] [Delete]
Private-4200000000-4294967294	0%	0%	NOT IN USE	[Edit] [Delete]

Create ASN Pool

Name: Freeform-ASN

Ranges: 65000 - 65100

Create Another?

2.Allocation Group – Blueprint内での有効化

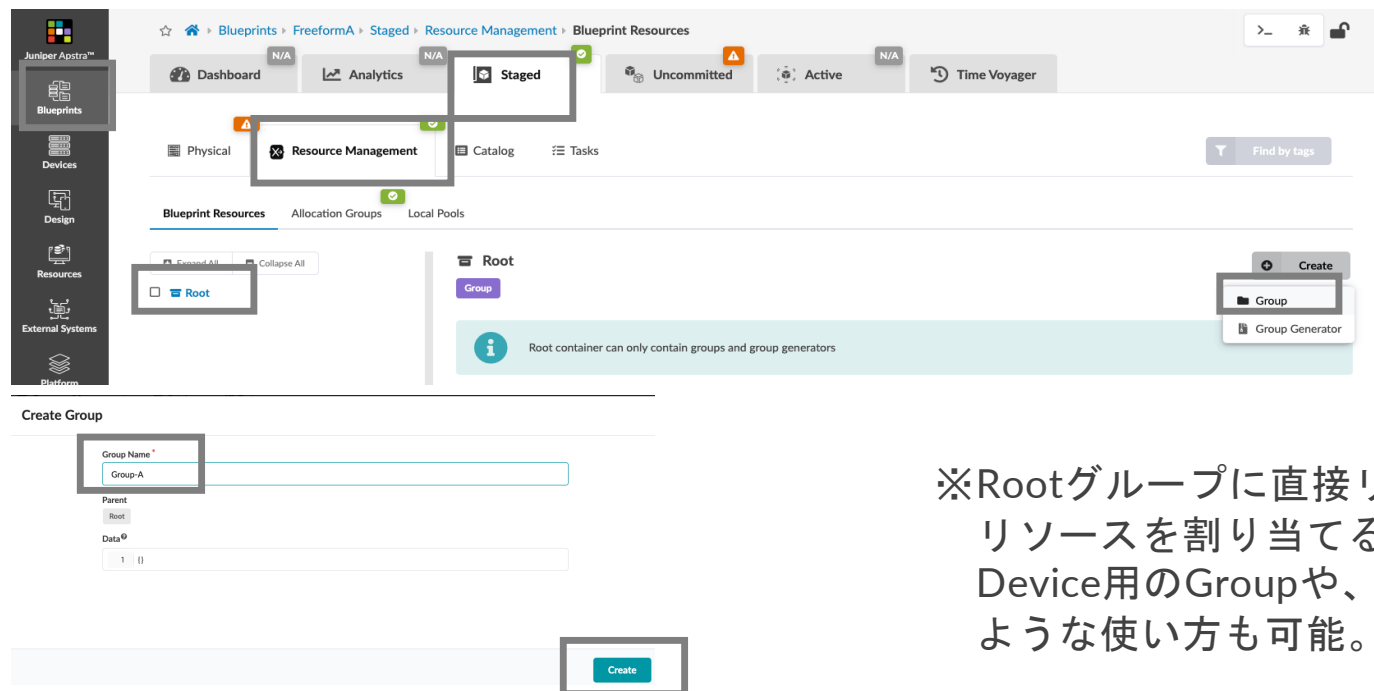
[Resource]で作成した値を、該当するBlueprintで利用できるように割り当て。

The screenshot shows the Juniper Apstra interface with the 'Create Allocation Group' dialog open. The dialog is titled 'Create Allocation Group' and has a close button (X) in the top right corner. The 'Group Name' field contains 'ASN_alloc'. The 'Type' is set to 'ASN' (selected with a radio button). The 'Resource Pools' section shows a search bar with 'Query: All' and a pagination indicator '1-3 of 3'. Below this, there are three radio buttons for filtering: 'all' (selected), 'selected only', and 'unselected only'. A table lists the resource pools with their names, total usage, range usage, and status.

Pool Name	Total Usage	Range Usage	Status
Private-420000000-4294967294	0%	0%	NOT IN USE
Freeform-ASN	0%	0%	NOT IN USE
Private-64512-65534	1.08%	1.08%	IN USE

3.Resource Management Group

それぞれの値を割り当てるためのグループ分けとして、Resource Groupを作成。
Resource Groupとは、フォルダ構造のような階層型のグループになり、リソースを保存する



※Rootグループに直接リソースを作成できないため、リソースを割り当てるためのグループを作成。Device用のGroupや、VRF毎のGroup等を分割するような使い方も可能。

3. Resource Management Group Generator

リソースから動的に割り当てる方法としては、いくつかの割当方法があるが、ここでは以下の2つの例を示します。

1. ASN, loopbackのように各デバイスに変数をアサイン

- ・ 各デバイス用のResource Groupを作成
- ・ Resource Generatorで各GroupにASNまたはアドレスを割り当てる

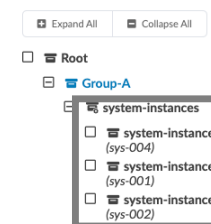
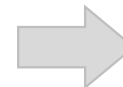
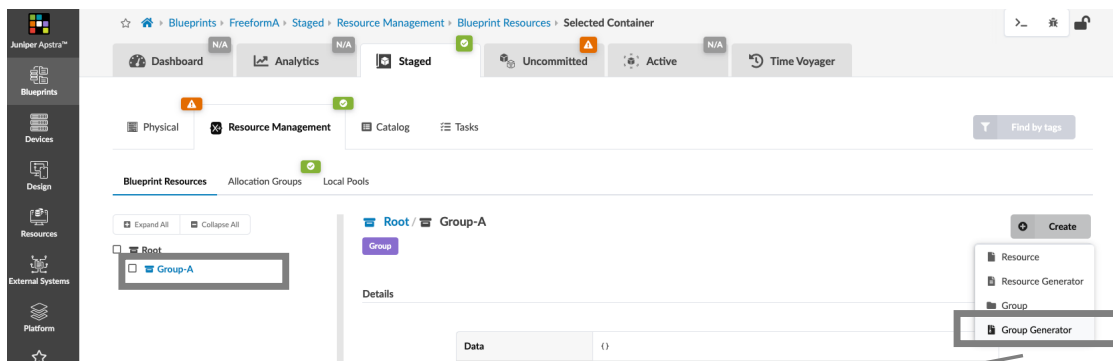
2. 各リンクにアドレスをアサインする

- ・ Resource Generatorでリンクにアドレスを割り当てる

3-1. Resource Management Group Generator

- デバイス/ノードGroup作成

例として、”ASNやloopbackのような各デバイスに固有の値を割り当てるような例”を示す。各デバイスに値を割り当てるためのGroupを作成する場合は、“Group Generator”を利用。
※Groupを作成せずにリソースを割り当てることも可能だが、管理を容易にするために作成することを推奨。



3つデバイスのそれぞれに
対してGroupが自動作成される。

Create Group Generator

Group Generator Name *

Parent
Group-A

Scope *

1 | `node('system', system_type='internal', name='target')`

Create

※以下をScopeとしてGroup Generatorを行うことで、各ノード/デバイス毎のResourceが生成される。

```
node('system', system_type='internal', name='target')
```

この指定方法はApstraが持っているGraph DBから対象とするGroupを生成しており、指定の方法によって様々な対象でGroupが作成できる。

Scopeの記述方法はここでは割愛。

4-1.Resource Generator – ASN割当

事前に設定していたAllocation Groupのリソースレンジから各デバイスに値を割り当てる。以下では各デバイスにAS番号を割り当てる例を示す。

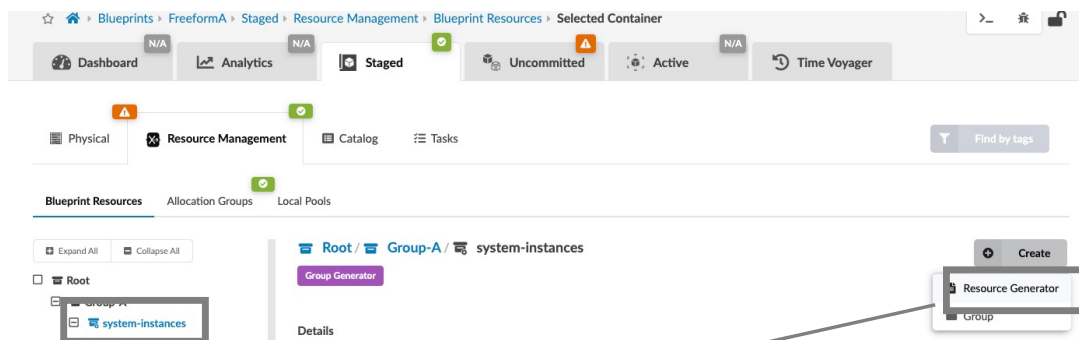
割り当てる各デバイスの上位のGroupを選択

各ノート/デバイス毎にASNが生成された

Name	Type	Value	Generated By	Allocated From	Assigned To	Actions
asn_gen	ASN	45002	asn_gen	ASN_alloc	Not Assigned	
asn_gen	ASN	45001	asn_gen	ASN_alloc	Not Assigned	
asn_gen	ASN	45000	asn_gen	ASN_alloc	Not Assigned	

4-1.Resource Generator – loopback アドレス割当

ASNと同じ用にloopback用の[Resource]と[Allocation Groups]をまずは作成する。
その後、リソースレンジから各デバイスに値を割り当てる。



Create Resource Generator

Name: loopback_alloc_group

Container Type: Group Generator

Container: system-instances

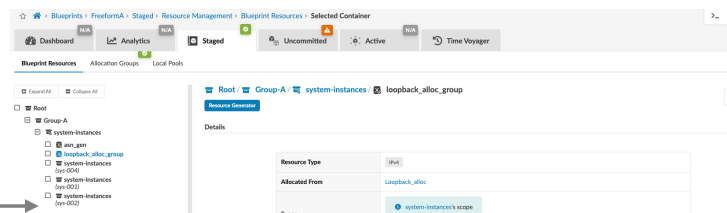
Group Generator Scope: node['system', system_type='Internal', name='target']

Type: IPv4 Host IPv4 IPv6 Host IPv6 ASN VNI VLAN Integer

Allocation Group: Loopback_alloc

Subnet Prefix Length: 32

Create

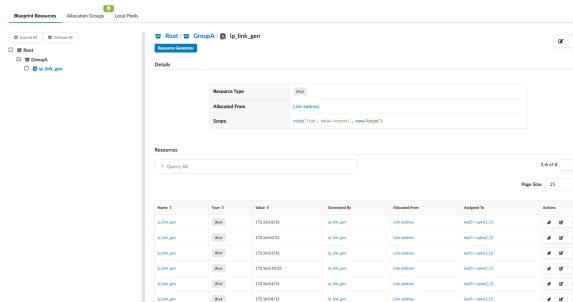
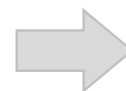
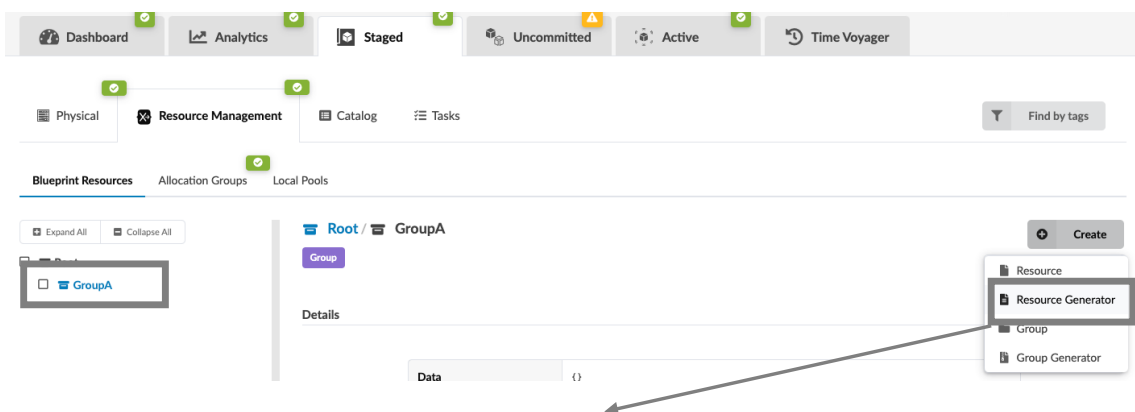


各ノード/デバイス毎にloopbackのアドレスが生成された

Name	Type	Value	Generated By	Allocated From	Assigned To	Actions
loopback_alloc_group	IPv4	10.0.0.1/32	loopback_alloc_group	Loopback_alloc	Not Assigned	# ⌵
loopback_alloc_group	IPv4	10.0.0.18/32	loopback_alloc_group	Loopback_alloc	Not Assigned	# ⌵
loopback_alloc_group	IPv4	10.0.0.19/32	loopback_alloc_group	Loopback_alloc	Not Assigned	# ⌵

4-2.Resource Generator -物理リンクアドレス割当

次にデバイス毎でなく、デバイス間のリンクにアドレスを割り当ててる方法を示す
作成していた全体のGroup を指定して [Resource Generator] でリソースを割り当てる。



リンクにアドレスが割り振りされた

Create Resource Generator

Name: ip_link_gen

Container Type: Group

Container: GroupA

Scope: 1 node('link', role='internal', name='target')

Type: IPv4

Allocation Group: ip_link_alloc

Subnet Prefix Length: 31

Create

※以下をScopeとしてScope Generatorを行うことで、各Linkに毎にIPv4アドレスが生成される。
`node('link', system_type='internal', name='target')`
この指定方法はApstraが持っているGraph DBから対象とするGroupを生成しており、指定の方法によって様々な対象でGroupが作成できる。
Scopeの記述方法はここでは割愛。

補足 : Graph model Query

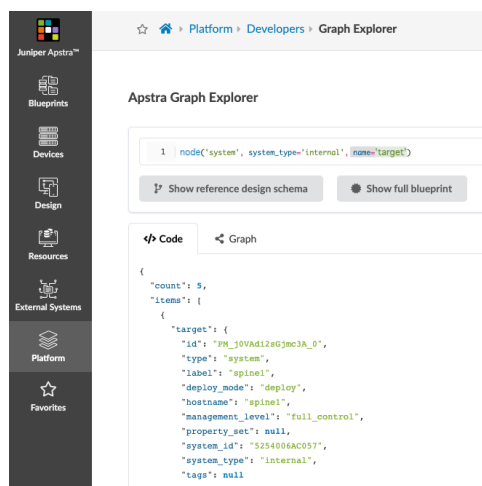
Scopeで利用したGraph model部分の詳細はドキュメントを参照のこと。

<https://www.juniper.net/documentation/us/en/software/apstra4.1/apstra-user-guide/topics/topic-map/graph.html>

例として、

`node('link', role='internal', name='target')`

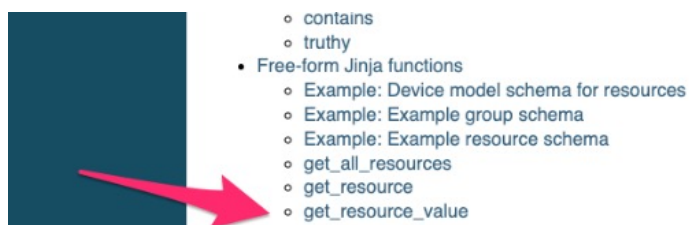
これにより、すべての内部(ファブリックに面する) 'internal' リンクのみが選択される。



Graph Explorerにてどのような値が取得可能かを確認することも可能。

5. Config TemplateのResource割当

Resource Managerにて割り当てた変数値を割り当てる場合は、“get_resource_value”を利用することで変数値を指定可能



The screenshot shows the 'Update Config Template' interface. On the left is a tree view with 'systemgen (leaf1)' selected. The main area has a 'Name' field with 'loopback_set.jinja'. Below it, 'Config Preview' shows 'Device Context' and 'System: leaf1'. A 'Jinja function reference' button is present. The main editor shows Jinja code:

```
1 {% set loopback_ipv4 = function.get_resource_value(resources, 'loopback_gen', 'groupa', 'systemgen') %}
2 replace: lo0 {
3   unit 0 {
4     family inet {
5       address {{ loopback_ipv4 }};
6     }
7   }
8 }
9
10
```

 A 'Preview' button is to the right. The 'Preview (as rendered for device leaf1)' section shows the rendered output:

```
1
2 [replace: lo0 unit 0 family inet]
3 + address 172.16.0.2/32;
4
```

 A red box highlights the rendered output. Below the preview, the text 'Leaf1のloopbackアドレスの設定が可能となる' is displayed. At the bottom are 'Save Changes' and 'Update' buttons.

※Config Templateの編集方法は次頁以降に記載

Config Templatesオペレーション記載内容

Config Templates概要

- Config Templateサンプル

新規Config Template作成 Day-0

- Jinja reference document
- プリインストールのデフォルトサンプル
- 複数Config Templatesの利用方法

変数値の利用

- Device Context概要
- Device Context変数値確認方法
- Property Sets概要 – 静的変数値
- Property Sets利用方法
- Property Setsサンプル
- Resource Management – レンジ変数値

Config Template 編集方法 Day-2/Advanced編

- **Config Template編集**
- **クローン方法/デバイス毎の異なるConfig Template**
- **Day2オペレーションサンプル – vlan追加**

Advanced編

- Day2オペレーションサンプル – Drain Modeサンプル
- タグベースの動的なコンフィグ生成

Config template Jinja 参考ドキュメント

Blueprint作成後に、以下のリンクからFreeformのReferenceドキュメントが確認可能

Blueprint > Staged > Catalog > Config Templates >

[+ Create Config Template](#)

>

?

Jinja function reference

Jinja function reference Document

Apstra Jinja Functions documentation »

Table of Contents

Apstra Jinja helper functions reference
Indices and tables

Next topic

Jinja Common Filters

Quick search

Go

Apstra Jinja helper functions reference

This document serves as a general reference against custom Jinja filters, tests, and functions as related to configuration templates and configlets.

Consult <https://jinja.palleteprojects.com/en/2.11.x/templates/> for general Jinja template designer details.

- Jinja Common Filters
 - bool
 - cidr_merge
 - intersect
 - json_query
 - to_cidr
 - to_ip
 - to_mac_format
 - to_netmask
 - to_network
 - to_prefixlen
- Jinja Common Functions
 - gen_acl_port_token
 - max_os_version
 - merge_vlans_to_list
 - min_os_version
 - raise_error
 - re_findall

Example: Extract two resource values from group underlay

Jinja for example *Extract two resource values from group underlay*

```
{% set loopback_ipv4 = function.get_resource_value(resources, 'ipv4_loopback', 'underlay') %}  
{% set bgp_asn = function.get_resource_value(resources, 'bgp_asn', 'underlay') %}  
Loopback ipv4: {{loopback_ipv4}} BGP ASN: {{bgp_asn}}
```

Device model context for example *Extract two resource values from group underlay*

```
{  
  "resources": [  
    {  
      "tags": [],  
      "label": "underlay",  
      "groups": [],  
      "data": {},  
      "id": "test_group_id-underlay",  
      "resources": [  
        {  
          "id": "test_res_id-ipv4_loopback-10.0.0.1/32",  
          "assigned_to": [],  
          "resource_type": "ip",  
          "value": "10.0.0.1/32",  
          "label": "ipv4_loopback"  
        },  
        {  
          "id": "test_res_id-bgp_asn-64512",  
          "assigned_to": [],  
          "resource_type": "asn",  
          "value": "64512",  
          "label": "bgp_asn"  
        }  
      ]  
    }  
  ]  
}
```

Expected text output for example *Extract two resource values from group underlay*

```
Loopback ipv4: 10.0.0.1/32 BGP ASN: 64512
```

Config Templateの編集

特定のシステムのテンプレートのみ変更したい場合は、クローンでコピーした後に、編集して特定のシステムに適用することも可能。

Query: All

Columns (4/4) Page Size: 25

Filter selected by all selected only unselected only

<input type="checkbox"/>	Name	Tags	Assigned To	Actions
<input type="checkbox"/>	crb_bgp.jinja		0 systems	
<input checked="" type="checkbox"/>	crb_esl.jinja		0 systems	
<input type="checkbox"/>	crb_policy_options.jinja		0 systems	
<input type="checkbox"/>	crb_root.jinja		4 systems	Clone
<input type="checkbox"/>	loopback_set.jinja		1 system	Edit

Config Templateの編集

Editで編集を行う際に、Systemを指定し、Device ContextやPreviewの確認をすることも可能。

The screenshot shows the Juniper Config Template management interface. At the top, there is a breadcrumb navigation: Blueprints > New-freeform > Staged > Catalog > Config Templates. Below this is a navigation bar with tabs: Dashboard, Analytics, Staged, Uncommitted, Active, and Time Voyager. A filter section indicates 'Filter selected by all selected only unselected only'. A table lists three templates: junos_chassis.jinja (0 systems), junos_configuration.jinja (5 systems), and junos_interfaces.jinja (0 systems). An 'Update Config Template' dialog is open, showing the 'junos_chassis.jinja' template. The dialog includes a 'Name' field, a 'Config Preview' section with 'Device Context' selected, a 'System' dropdown set to 'SpineA', and an 'Apply Mode' dropdown set to 'Incremental'. The 'Preview' button is highlighted. The dialog also shows the 'Template Text', 'Preview (as rendered for device SpineA)', and 'Device Context' sections.

Name	Tags	Assigned To	Actions
junos_chassis.jinja		0 systems	[Edit] [Delete]
junos_configuration.jinja		5 systems	[Edit] [Delete]
junos_interfaces.jinja		0 systems	[Edit] [Delete]

Update Config Template

Name: junos_chassis.jinja

Config Preview: Device Context

System: SpineA

Apply Mode: Incremental

Preview

Template Text

```
1  {# number of portchannel(ae) intfs being used need to be configured #}
2  {# qosfp port breakout channel speed need to be configured #}
3  {% set ae_ns = namespace(ae_count=0) %}
4  {% for interface_name in interfaces.iterkeys() if interface_name[:2] == 'ae' %}
5  |   {% set ae_ns.ae_count = ae_ns.ae_count + 1 %}
6  |   {% endfor %}{# interface_name in interfaces.iterkeys() if interface_name[:2] == 'ae' #}
7  |   {% set port_speed_map = None %}
8  |   {% if portSetting is defined and portSetting %}
9  |     {% set port_speed_map = function.get_junos_fpc_pic_port_speed_map(portSetting) %}
10 |   {% endif %}
11 |   {% if ae_ns.ae_count or port_speed_map %}
12 |     chassis {
```

Preview (as rendered for device SpineA)

```
1  - [system]
2  - host-name SpineA;
3
4  - [interfaces replace: et-0/0/0]
5  - description "facing_leaf-a2:et-0/0/0";
6  - mtu 9216;
7
8
9  - [interfaces replace: et-0/0/0 unit 0 fami
10 - address 172.16.1.2/30;
```

Device Context

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

Save Changes Update

Day2オペレーションサンプル : property set

例:vlan設定のJinja2テンプレートを用意し、Property Setで必要なvlan情報等を追加することで追加

vlangs.jinja

テンプレート

```
抜粋
{
  "vlans": {
    "<VLAN_ID>": {
      "vrf_name": <VRF_NAME>,
      "description": <DESCRIPTION, optional string>,
      "vxlan_id": <VXLAN_ID, optional integer>,
      "vlan_id": <VLAN_ID, required integer>,
      "ipv4_virtual_gateway_address": <IP Address without prefixlen, optional>
      "systems": {
        <SYSTEM HOSTNAME>: {
          "ipv4_address": <UNIQUE IRB IPV4 ADDRESS WITH PREFIXLEN>,
          "tagged_interfaces": [<LIST OF INTERFACE NAMES WITH VLAN TAGGING, optional strings>],
          "native_interfaces": [<LIST OF INTERFACE NAMES THIS VLAN IS NATIVE VLAN FOR, optional strings>]
        }
      }
    }
  }
}
```

Vlangs.json(property-set)

変数

```
{
  "10": {
    "vxlan_id": null,
    "description": "default_vlan10_leaf3_singleton",
    "vrf_name": "default",
    "systems": {
      "leaf3": {
        "ipv4_address": "172.16.10.1/24",
        "tagged_interfaces": [],
        "untagged_interfaces": [
          "ge-0/0/2"
        ]
      }
    },
    "vlan_id": 10,
    "ipv4_virtual_gateway_address": null
  },
  "11": {
    "vxlan_id": 10011,
    "description": "blue_vxlan10011_leaf3_singleton",
    "vrf_name": "blue",
    "systems": {
      "spine1": {
        "ipv4_address": "172.16.11.2/24"
      }
    }
  }
}
```

抜粋,

Config Templatesオペレーション記載内容

Config Templates概要

- Config Templateサンプル

新規Config Template作成 Day-0

- Jinja reference document
- プリインストールのデフォルトサンプル
- 複数Config Templatesの利用方法

変数値の利用

- Device Context概要
- Device Context変数値確認方法
- Property Sets概要 – 静的変数値
- Property Sets利用方法
- Property Setsサンプル
- Resource Management 利用方法

Config Template 編集方法 Day-2/Advanced編

- Config Template編集
- クローン方法/デバイス毎の異なるConfig Template
- Day2オペレーションサンプル – vlan追加

Advanced編

- Day2オペレーションサンプル – Drain Modeサンプル
- タグベースの動的なコンフィグ生成

Day2 オペレーションサンプル - トラフィック片寄せ

事前にトラフィック片寄せのポリシーを作成し、Deploy ModeがDrainを選択することで有効化

The image illustrates the configuration process for traffic sharding in a Juniper environment. It is divided into three main sections:

- Drain Mode:** A screenshot of the Juniper configuration interface. The 'Deploy Mode' section shows 'Drain' selected with a radio button. Below it, the 'S/N' (Serial Number) is set to '525400EE9966'. The 'Device Info' section shows the Management IP as '172.20.81.11'.
- Config Template - Policy:** A code block showing a Jinja2 template for a policy. The policy statement 'AllPodNetworks' is configured to accept traffic from the 'inet' family and reject traffic from the 'AllPodNetworks-100' family. The text '抜粋' (Excerpt) is present at the bottom right.
- Config Template - BGP:** A code block showing a Jinja2 template for BGP configuration. It includes a conditional statement: `% if deploy_mode == 'drain' %` followed by `export drain;`. The text '抜粋' (Excerpt) is present at the bottom right.

A green arrow points from the left two sections to the right section:

- Generated Config:** A screenshot of the rendered configuration for device 'spine1'. The configuration shows BGP settings for 'spine1', including 'local-address 192.168.1.1', 'peer-as 65510', and 'bfd-liveness-detect' settings. The line `export (drain);` is highlighted with a red box, indicating that the 'Drain' mode is active.

CRBサンプルポリシー : https://github.com/Juniper-SE/apstra-freeform/blob/main/CRB_stage3/crb_policy_options.jinja

タグベースの動的コンフィグ生成

タグはラベリングとして、整理、検索、参照などに活用でき、様々な構成要素(デバイス・リンク・)に設定。 Jinja でタグを使用して、コンフィグ生成時に動的変数を設定することができる。

例:リンクにタグ情報を付与し、タグ情報を元にBGPのMEDを設定

Tag

Dashboard Analytics Staged Unco

Topology Systems **Links**

Query: All

Filter selected by all selected only unselected only

	Name	Type	Tags	Speed	Role	Name	Interface
<input type="checkbox"/>							
<input type="checkbox"/>	sys47 <> sys22	Physical	110	10G	Internal	Bond-Street	xe-0/0/0
<input type="checkbox"/>	sys47 <> sys86	Physical	177	10G	Internal	Bond-Street	xe-0/0/1

Config Template

```
1 {% set this_router = hostname %}
2 {% set med_tags = namespace(meds=[]) %}
3 policy-options {
4   policy-statement send-direct {
5     term 1 {
6       from protocol direct;
7       then accept;
8     }
9   }
10 }
11
12 {% for interface_name, iface in interfaces.iteritems() %}
13 {#
14   First, building a list of link_meds so we can later deduplicate the
15   add-med-<value> policy statements
16   #}
17   {% do med_tags.meds.extend((iface.link_tags) %) %}
18   {% endfor %}
19
20   {% for link_med in med_tags.meds|sort|unique %}
21     policy-statement add-med-{{ link_med }} {
22       from {
23         route-filter 0.0.0.0/0 longer;
24       }
25       then {
26         bgp {
27           group external-peers {
28             type external;
29             export send-direct;
30           }
31           {% for interface_name, iface in interfaces.iteritems() if iface.get('ipv4_address') and iface.get('neighbor_interface', {}).get('ipv4_address') %}
32             {% set peer_hostname=iface.neighbor_interface.system_hostname %}
33             neighbor {{ iface.neighbor_interface.ipv4_address }} {
34               peer-as {{ props['peer-as-{{peer_hostname}}']|asn }};
35               export add-med-{{ (iface.link_tags|0) }};
36             }
37           {% endfor %}
38         }
39       }
40     }
41   {% endfor %}
42 }
```

Config

Bond-Street Rendered Config Preview

```
88 }
89 policy-statement add-med-110 {
90   from {
91     route-filter 0.0.0.0/0 longer;
92   }
93   then {
94     metric add-110;
95   }
96   then {
97     accept
98   }
99 }
100 policy-statement add-med-177 {
101   from {
102     route-filter 0.0.0.0/0 longer;
103   }
104   then {
105     metric add-177;
106   }
107 }
108
109 bgp {
110   group external-peers {
111     type external;
112     export send-direct;
113     neighbor 192.168.0.3 {
114       peer-as 22;
115       export add-med-110;
116     }
117     neighbor 192.168.0.7 {
118       peer-as 86;
119       export add-med-177;
120     }
121   }
122 }
```



- Apstra Freeform
 - 初期セットアップ手順
 - Config Templates
 - **Apstra Telemetry**
 - オプション機能
- 参考リンク

Telemetry Realtime Analytics for Freeform概要

“Dashboard”-ネットワーク全体の健康状態の把握

“Analytics”-BGPセッション、memory、cpuなど各種各種パラメーターを取得可能

The dashboard provides a high-level overview of network health. It features a top navigation bar with tabs for Dashboard, Analytics, Staged, Uncommitted, Active, and Time. A central status indicator shows 'All Probes' with '0 anomalies'. Below this, the 'Fabric' section displays four categories: Cabling, Interface, Node, and Hostname, each with '0 anomalies'. The 'Deployment Status' section shows three categories: Deployment, Config Dev., and Config Re., also with '0 anomalies'. An 'Anomaly History' section is visible at the bottom.

This view shows the configuration and monitoring options for a specific device (172.27.114.92). It includes tabs for Device, Agent, and Pristine Config. Below these are several monitoring and configuration options: Anomalies, Config, Interface, MAC, LLDP, Hostname, and Counters. A filter section at the bottom allows users to select 'all', 'selected only', or 'unselected only' items.

This view displays the 'BGP Session Flapping' analytics. It shows the stage as 'BGP Session' and the data source as 'Real Time'. The interface includes a search bar, a query filter, and a table of BGP sessions. The table columns include System ID, AF, Dest Asn, Dest Ip, Source Asn, Source Ip, Vrf Name, Flap Count, Flap Count Increment, FSM State, and Status.

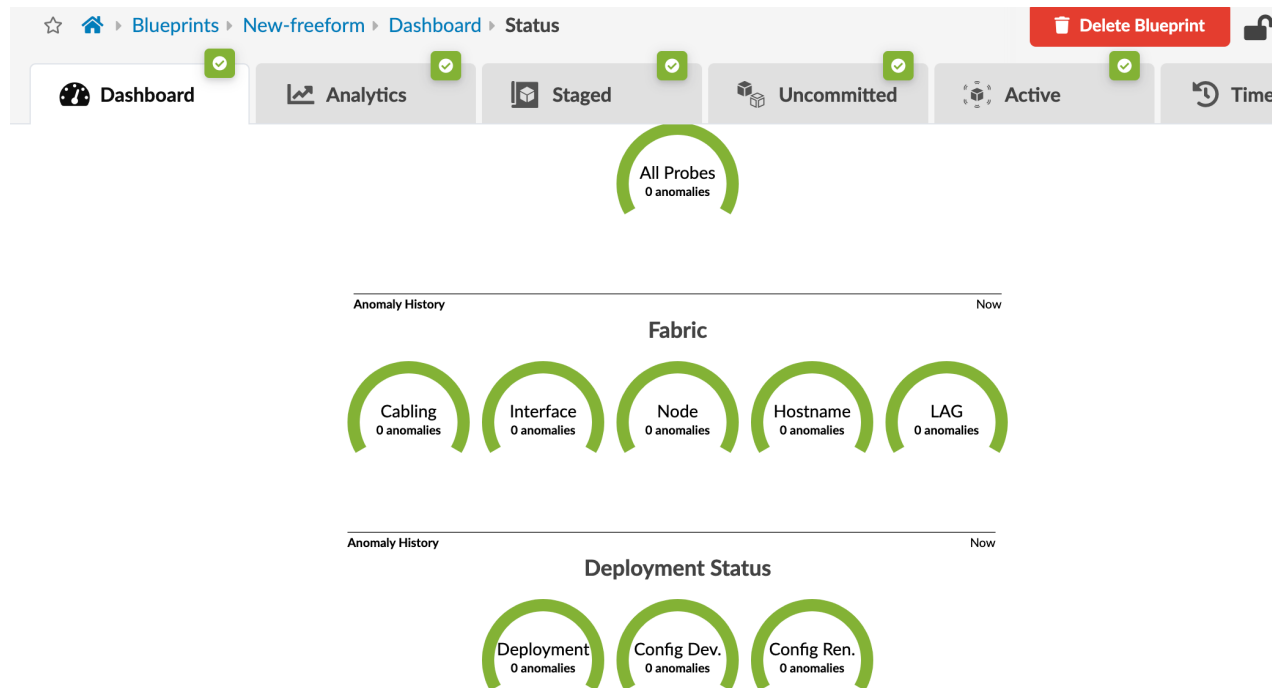
System ID	AF	Dest Asn	Dest Ip	Source Asn	Source Ip	Vrf Name	Flap Count	Flap Count Increment	FSM State	Status
52540028C197 oxford-circus	ipv4	23	192.168.0.1	22		default	0	0	established	up
52540028C197 oxford-circus	ipv4	47	192.168.0.2	22		default	0	0	established	up
52540028C197 oxford-circus	ipv4	86	192.168.0.5	22		default	0	0	established	up

Filter selected by all selected only unselected only

	Service Name	Service Started?	Interval (s)	Input	Run Count	Success Count	Failure Count	Max Run Count	Execution Time, ms	Waiting Time, ms	Last Run Timestamp	Last Error Timestamp	Error message
<input type="checkbox"/>	ARP	yes	120		2248	2244	0		1236.28	0.25	2022-06-20, 13:26:46		N/A
<input type="checkbox"/>	INTERFACE	yes	120		2244	2243	0		771.95	0.25	2022-06-20, 13:26:46		N/A
<input type="checkbox"/>	DISK UTIL	yes	120		2051	2050	0		304.60	658.90	2022-06-20, 13:26:47		N/A
<input type="checkbox"/>	LLDP	yes	10		26784	26777	0		369.42	0.75	2022-06-20, 13:27:47		N/A

Default Telemetry

DC reference architectureと同じく“Dashboard”でネットワーク全体の健康状態を確認。但し、デフォルトで用意されている監視項目(Generic System Connectivity, Liveness, Route Verification)についてはFree formではDefault Telemetryの対象外となる(4.1.1現在)



IBA Probes

DC reference architectureと同じく、Analytics > Probesで取得必要な各ステータスの取得をすることも可能。但し、利用トポロジーなどによって利用できないProbesもあり。

☆ [Home](#) > [Blueprints](#) > [New-freeform](#) > [Analytics](#) > [Dashboards](#)

Dashboard Analytics Staged Uncommitted Active Time Voyager

Dashboards Anomalies Widgets Probes

[+ Create Probe](#)

Query: All 1-5 of 5 Page Size: 25

<input type="checkbox"/>	Name ▲	Anomalies ⇅	State ⇅	Updated By ⇅	Tags ⇅	Enabled ⇅	Actions
<input type="checkbox"/>	Device System Health	✓ No anomalies	✓ Operational	System 3 days ago		<input checked="" type="checkbox"/>	Edit Copy Delete
<input type="checkbox"/>	Device Traffic	✓ No anomalies	✓ Operational	System 3 days ago		<input checked="" type="checkbox"/>	Edit Copy Delete

Device Telemetry確認

Blueprint > Active > Physical、Devices > Managed Devices > 該当デバイス > Telemetry
 で該当デバイスを選択することでConfig, Interface Counters等の状態確認可能。

Counters

Interface	Received					Transmitted					Errors				
	Bytes	Unicast packets	Broadcast packets	Multicast packets	Error packets	Discarded packets	Bytes	Unicast packets	Broadcast packets	Multicast packets	Error packets	Discarded packets	Giants	Runs	Alignment errors
et-0/0/1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
et-0/0/6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Device Info

- Management IP: 172.27.114.92
- OS: Junos 21.2R3.8
- Operation Mode: FULL CONTROL

Collection Statistics

Service Name	Service Started?	Interval (s)	Input	Run Count	Success Count	Failure Count	Max Run Count	Execution Time, ms	Waiting Time, ms	Last Run Timestamp	Last Error Timestamp	Error message
ARP	yes	120		2248	2244	0		1236.28	0.25	2022-06-20, 13:26:46		N/A
INTERFACE	yes	120		2244	2243	0		771.95	0.25	2022-06-20, 13:26:46		N/A
DISK UTIL	yes	120		2051	2050	0		304.60	658.90	2022-06-20, 13:26:47		N/A
LLDP	yes	10		26784	26777	0		369.42	0.75	2022-06-20, 13:27:47		N/A



- Apstra Freeform
 - 初期セットアップ手順
 - Config Templates
 - Apstra Telemetry
 - **オプション機能**
- 参考リンク

多様な運用オプション機能

Time Voyager (Roll Back)

Device OS Upgrade

Collect Show Tech

ZTP

The screenshot displays the Juniper Apstra management interface with several key features highlighted:

- Time Voyager:** A table showing configuration changes with columns for Description, Created At, User, and Actions. Entries include 'change color', 'Device Remove', and 'change to new-sy'.
- Device OS Upgrade:** A 'Managed Devices' page with buttons for 'Create Onbox Agent(s)', 'Create Offbox Agent(s)', and 'Advanced Settings'. It includes a table with columns for Management IP, Device Key, and Device Profile.
- Collect Show Tech:** A configuration page for 'Collect Show Tech' with options for 'AOS Controller', 'Include Backup', and 'Managed Devices'. It includes a table with columns for Service Name, IP Address, Service Status, and Last Updated.


※これらのオペレーション方法はDC Referenceと同様のため、
オペレーション方法はApstra簡易オペレーションガイドを参照ください




- オペレーション概要
 - 初期セットアップ
 - Config templates
 - 可視化/Apstra Telemetry
 - オプション
- 参考リンク

Freeform サンプル on Github

以下にJuniperのSE作成のConfig TemplatesとProperty Setsのサンプルの公開リンク：<https://github.com/Juniper-SE/apstra-freeform>



README.md



Juniper Apstra™

Freeform Collections

Config Templates and Property Sets for Juniper Apstra Freeform

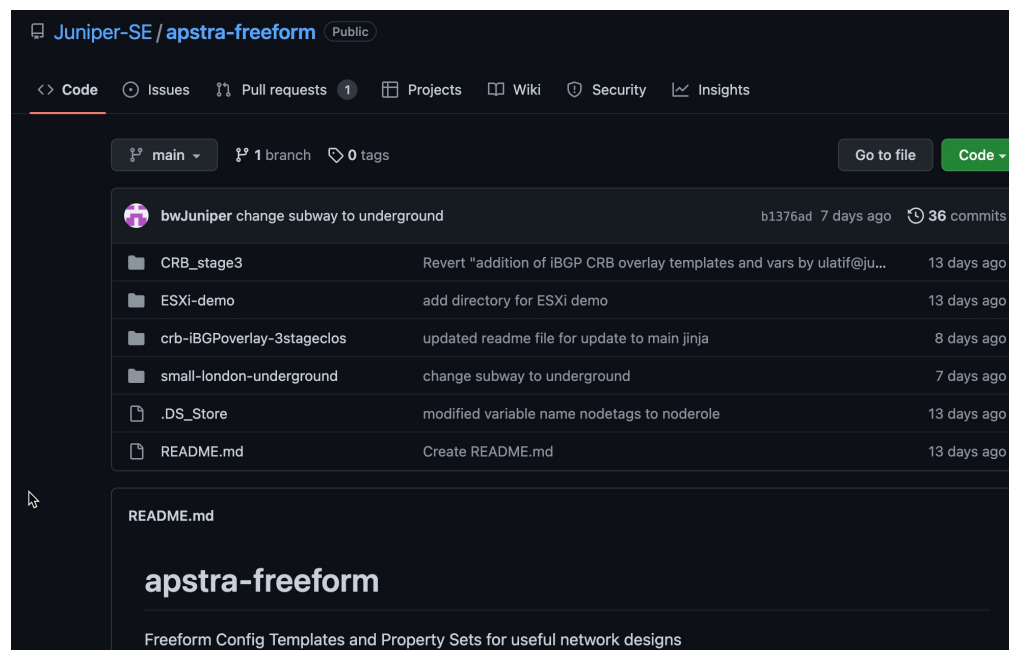
[Explore the Juniper Apstra Docs »](#)

[Start your own Demo](#) · [Report Bug](#) · [Request Feature](#)

About The Project

[Juniper Apstra](#)

Welcome to the github repo for Apstra Freeform! We hope to capture a wide variety of config templates with Jinja2 templating, property_sets and other info to build out different Freeform Topologies with Juniper Apstra. We welcome your contributions and hope this can be a public repo we can all share.



Juniper-SE / apstra-freeform Public

<> Code Issues Pull requests 1 Projects Wiki Security Insights

main 1 branch 0 tags

Go to file Code

Commit	Message	Time
bwJuniper change subway to underground	b1376ad 7 days ago 36 commits	
CRB_stage3	Revert "addition of iBGP CRB overlay templates and vars by ulatif@ju..."	13 days ago
ESXi-demo	add directory for ESXi demo	13 days ago
crb-iBGPoverlay-3stageclos	updated readme file for update to main jinja	8 days ago
small-london-underground	change subway to underground	7 days ago
.DS_Store	modified variable name nodetags to noderole	13 days ago
README.md	Create README.md	13 days ago

README.md

apstra-freeform

Freeform Config Templates and Property Sets for useful network designs

Let's try Apstra Freeform on CloudLabs



Table of Contents

- 1. Log into CloudLabs
- 2. Introduction to Apstra Freeform
 - 2.1. Apstra component parts
- 3. London Underground Network Model
- 4. Freeform Blueprints
 - 4.1. Inspecting the pre-prepared Freeform Blueprint
 - 4.2. Device Profiles in Freeform
 - 4.3. Topology Editor
 - 4.4. Tags
 - 4.5. View links between stations & their parameters
 - 4.6. Internal and External System Objects
- 5. Property Sets
 - 5.1. Update and review Property Set information.
- 6. Config Templates
 - 6.1. Nesting of Config Templates

Lab Guide - Apstra Freeform

Product SE Team - Version 0.3



Juniper Apstra™

Estimated Time: 60 minutes.

Objective

※CloudLabsの利用希望がある場合は、Juniper社員にご連絡をお願いします。
現在の利用期限は2週間です。

参考リンク

- Apstra Documentation
 - <https://www.juniper.net/documentation/product/us/en/apstra>
- Freeform Jinja テンプレート サンプル
 - <https://github.com/Juniper-SE/apstra-freeform>
- Apstra Freeform Technical Documentation In Focus 4.1.1
 - <https://www.juniper.net/documentation/us/en/software/apstra4.1/infocus-apstra-freeform-4.1.1>
- Cloud Lab Guide – Apstra Freeform
 - <https://cloudlabs.apstra.com/labguides/en/labff.html>
- 公式トレーニング - APSTRAFF
 - https://learningportal.juniper.net/juniper/user_activity_info.aspx?id=13119
- Jinja Template Designer Documentation
 - <https://jinja.palletsprojects.com/en/3.1.x/templates/#base-template>
 - <https://jinja.palletsprojects.com/en/3.1.x/changes/#version-3-1-2>
 - <https://ttl255.com/jinja2-tutorial-part-1-introduction-and-variable-substitution/>



Thank you

JUNIPER
NETWORKS

Driven by
Experience™