RIFT(ROUTING IN FAT TREES)

～IP FABRIC ROUTING OPEN STANDARD AND ZERO OPEX～

Juniper Networks
APAC CoE シニアコンサルティングエンジニア
塚本 広海

AGENDA

- RIFT登場の背景
- RIFTとは
  - 概要
  - 機能紹介
  - 参考 – JUNOSサンプル
- まとめ

DATA CENTER TRENDS

トラフィックの
肥大化

DCの巨大化

管理機器台数の増加

サーバ間通信
(East-West) の増加

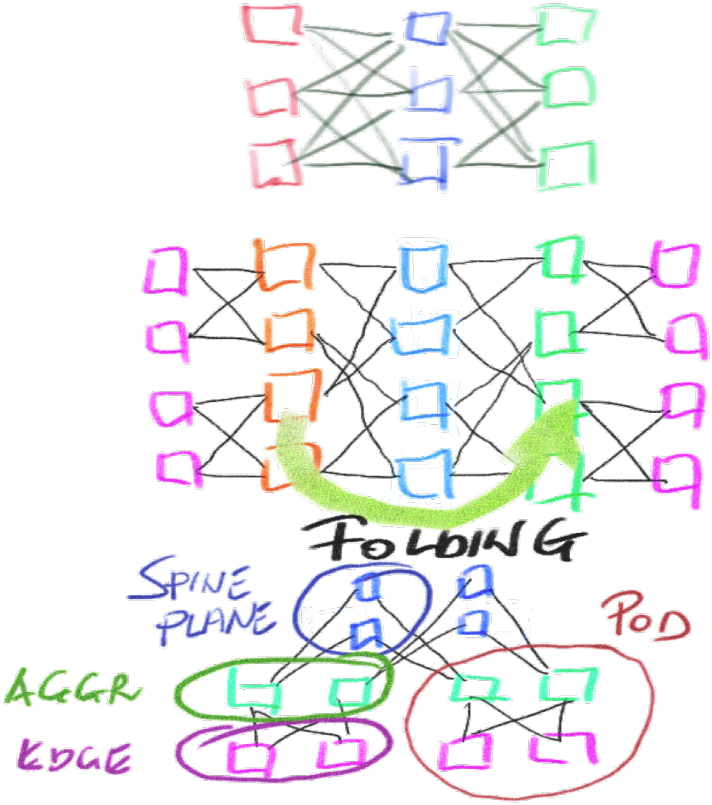
スケールアウト
ハードウェアの
コモディティ化

障害発生時における
サービス提供の維持

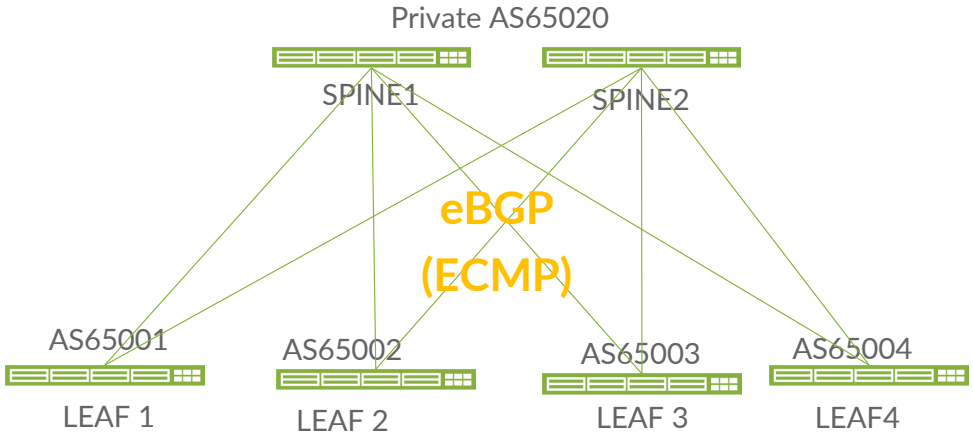
DC FABRIC TOPOLOGY – FAT-TREES TOPOLOGY

- Clos Network by Charles Clos
- Diagonal(toroidal) mesh networks
- Dragonfly Topology
- **”Folded Fat-Tree”のLeaf- SPINEトポロジ**
  - 昨今のデータセンターのデファクト
  - スケールアウトや障害時の影響も少
  - East-West Trafficにも最適
  - 完全なClosではないオーバーサブスクリプション

トポロジーは均一かつ規則性を持つ



DC FABRIC – IP FABRIC – PROTOCOL



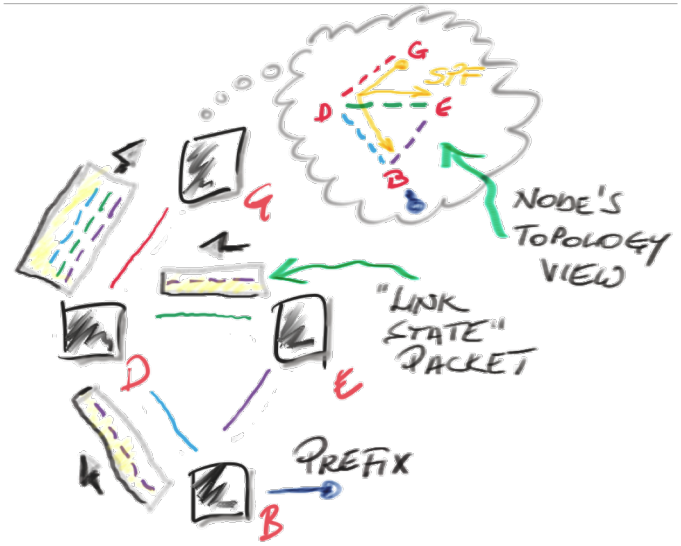
RFC7938 Informational
Use of BGP for Routing in Large-Scale Data Centers

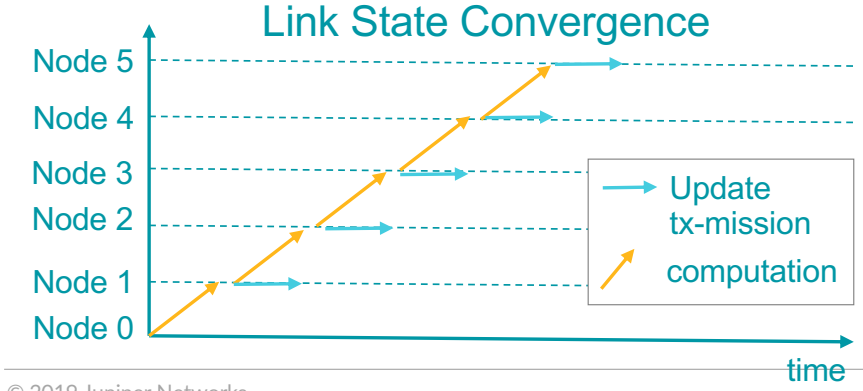
- L2の障害ドメインの波及なくeBGPでL3制御
- VMモビリティは同一ラック内
- “Path Hunting”制御のためのAS番号採番
- ECMPにのためのベストパス選択変更
- デフォルトタイマーの変更

複雑な設定、外部ツール、ピア自動検出実装必要、状態確認の困難さ、複雑なベストパス選定、影響範囲の最小化、ルーティングテーブル肥大化、IGPの見直し etc.

BGPにパッチをあてて機能拡張させ独自に利用することになり、障害波及、スケーラビリティ、容易な運用　が達成出来たとは言えない

ROUTING PROTOCOL REVIEW – LINK STATE





各ノードが、トポロジを構成する要素(ノード、リンク、経路)を含むリンクステート情報を生成して、リンクステート情報をフラッド

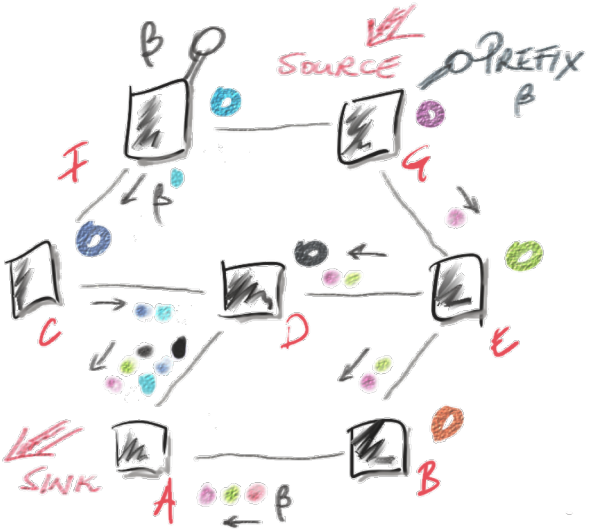
ADVANTGES

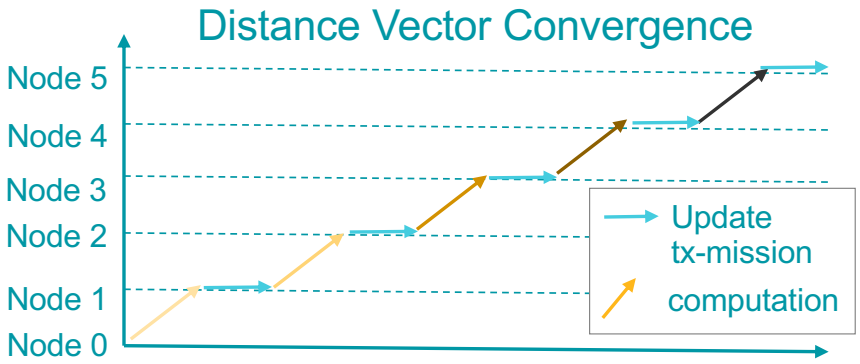
- 各ノードが全体のトポロジを見る
- 各ノードが全体への到達性を計算
- コンバージェンスは非常に高速

DISADVANTAGES

- あらゆる障害はネットワーク全体に波及
- フラッディングは到達性を提供するために過剰な負荷
- 更新のために定期的なフラッディング

ROUTING PROTOCOL REVIEW – DISTANCE VECTOR





プレフィックスは、通ってきたリンクの合計コストを保持。それぞれの受信者は、最適な結果を計算し、それを渡す

ADVANTGES

- メトリックを厳密に増加させることで、容易にループを防ぐことが可能
- 到達性よりもポリシーを重視
- 適切に実装された場合、Link Stateよりも高いスケール

DISADVANTAGES

- コンバージェンスは遅い
- ECMPのために特別な配慮必要(Add-Pathなど)
- 受信者は、全ての経路を保持する必要がある。もしそうでないならば、再送を要求

SPINE-LEAFトポロジーに最適なプロトコルとは？

DCファブリックの特性:
・トポロジの形は概ね一意 (CLOS, Fat-Tree)
・ノード間の接続は非常に多い

既存のルーティングプロトコル:
・不規則なネットワークトポロジ
・ノード間の接続はそう多くない

相反する特徴

- 不規則なトポロジに対応するため、仕組みがとても複雑
- 結果として、現在用いられるルーティングプロトコルは
  - 高速だが、スケールしない (link-state)
  - 遅いが、スケールする (distance-vector)　　のどちらか

IPファブリックはハードウェアはコモディティ化してきたが、
ファブリックの運用についてもコモディティ化する必要ためのプロトコルが必要

REQUIREMENTS BREAKDOWN (RFC7938+) FOR A “MINIMAL OPEX FABRIC”

Problem / Attempted Solution	BGP modified for DC (all kind of “mods”)	ISIS modified for DC (RFC7356 + “mods”)	RIFT Native DC
ピアの自動検出/True ZTP/誤配線の防止	⚠	⚠	✓
ToRにおけるサーバにとって最小限の情報の保持、サーバマルチホーミングへの対応	✗	✗	✓
高いECMPスケール (BGPではたくさんの追加設定やメモリ消費、ASパス計算におけるRFC違反を許容する必要がある)、LFA	⚠	✓	✓
等コストでないマルチパス、Anycast、MC-LAGの置き換え	✗	✗	✓
ネクストホップやプレフィックスの変更によるTraffic Engineering	✓	✗	✓
PCE/SRを使用するための、全てのリンクの可視性	⚠	✓	✓
任意のデータ(key-value)の効率的な伝搬	✗	⚠	✓
サービス断を伴わないノードの迅速な取り外し	✗	✓	✓
ブラックホールやパケットの逆走を防ぐ為の経路の自動分割	✗	✗	✓
障害発生時の影響範囲の最小化 (障害時にネットワークを”揺れないよう”にする)	✗	✗	✓
可能な限りの最速なコンバージェンス	✗	✓	✓
帯域の負荷分散	✗	✗	✓
プロトコルの初期実装の容易さ	✓	✗	✗

AGENDA

- RIFT登場の背景
- RIFTとは
  - 概要
  - 機能紹介
  - 参考 – JUNOSサンプル
- まとめ

RIFT: NOVEL ROUTING ALGORITHM FOR CLOS UNDERLAY

Link-State and Distance Vector

両方の‘いいとこ’取り

- 高速なコンバージェンス
- 自動的なトポロジー検出
- ToRで保持する経路の最小化
- 高スケールのECMP
- ノードの高速な切り離し

良くない点の改善

- 大量のフラッディング
- 手動での隣接関係設定

RIFT独自の機能

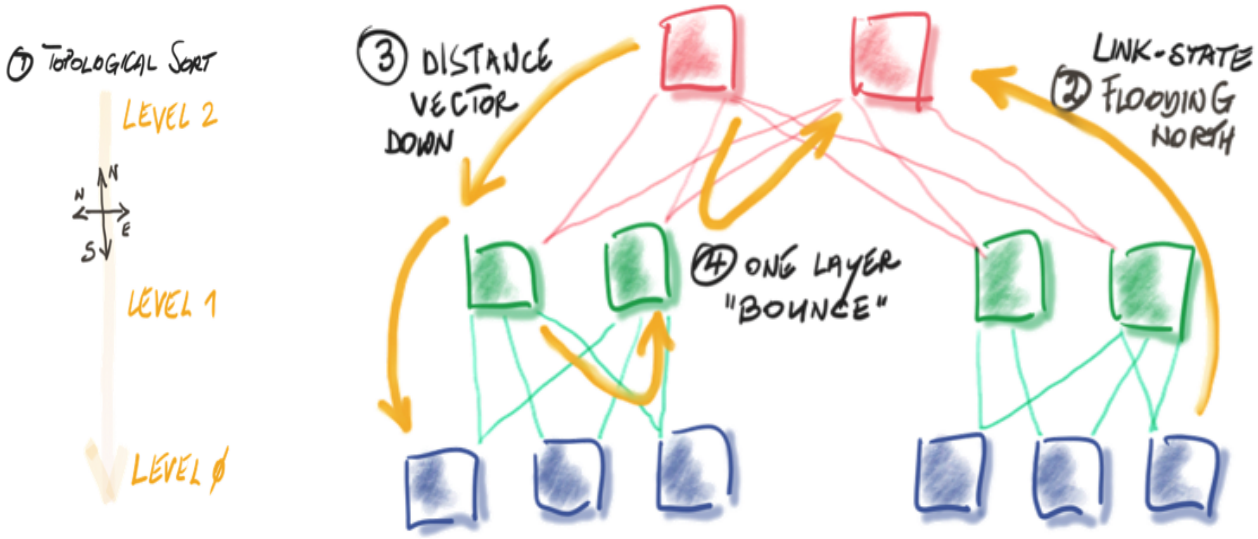
- ゼロタッチプロビジョニング
- 障害時の自動的な経路再分割
- 障害時の影響範囲の最小化
- ループフリーで全てのパス有効
- 非イコールコストのマルチパス
- Key-Value Storeによる拡張

データセンターにおける様々なユースケースをサポート

RIFT ARCHITECTURE OVERVIEW

1.Topological sort
方向性と階層の概念をルーティングに導入

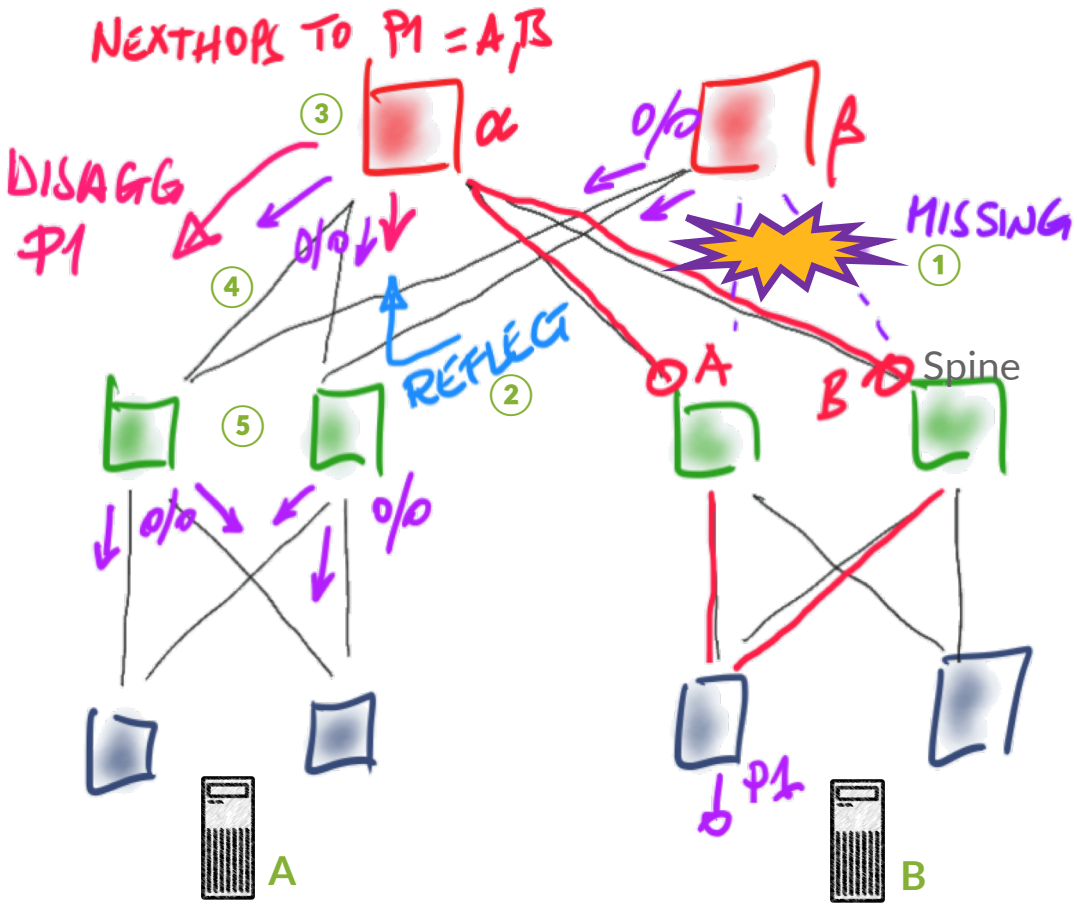
2.Link-State flood Up (North)
全体のトポロジーと経路は最上位のスパインのみが知る



4. Bounce
- フラッディングの最小
- 同一レイヤーの障害検出と経路の自動再分割

3.Distance Vector Down (South)
- 上位から下位の機器へはデフォルトルートのみ
- より細かい経路を持つ場合は、障害時における経路の分割 、Policy Guided Prefix (PGP) for TE

AUTOMATIC DE-AGGREGATION – ROUTING IN FAILURE

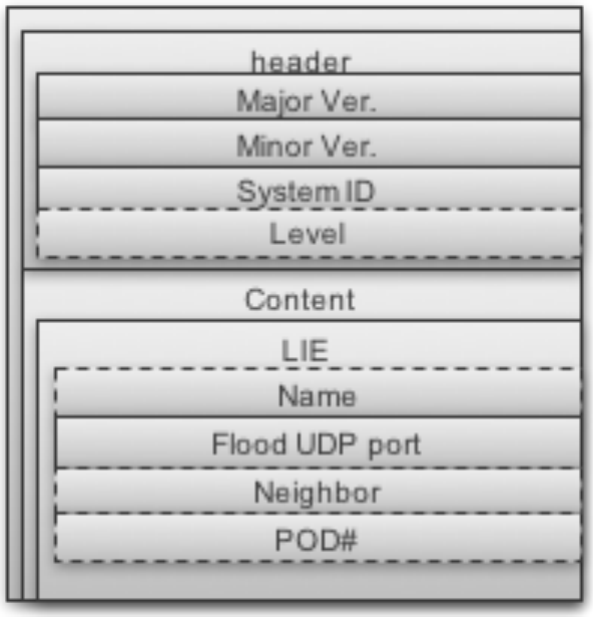


障害発生時のブラックホールを自動排除するために集約しない経路を自動伝搬

1. Spine βのダウンリンクで障害発生
2. Spine βと隣接関係を持つ下位デバイスがBounceし、アップデートをSpine αに伝達
3. Spine αは、Spine βがP1への到達性を失っていることを検出し、Spine α はP1への到達性を計算
4. Spine αはP1の経路がSpine β に行かないようにP1の経路(細かい経路)を自動生成
5. P1の経路(細かい経路) を受け取った**一段下のデバイスは、P1経路はECMPでなく、Spine α のみを利用する**ことに。 **P1経路(細かい経路)は更に下のLevel(ToR)には伝搬しない**
6. Spine βがP1への到達性を取り戻した場合、Spine αはP1経路(細かい経路)の生成を自動停止

ADJACENCY FORMATION (AKA HELLO)

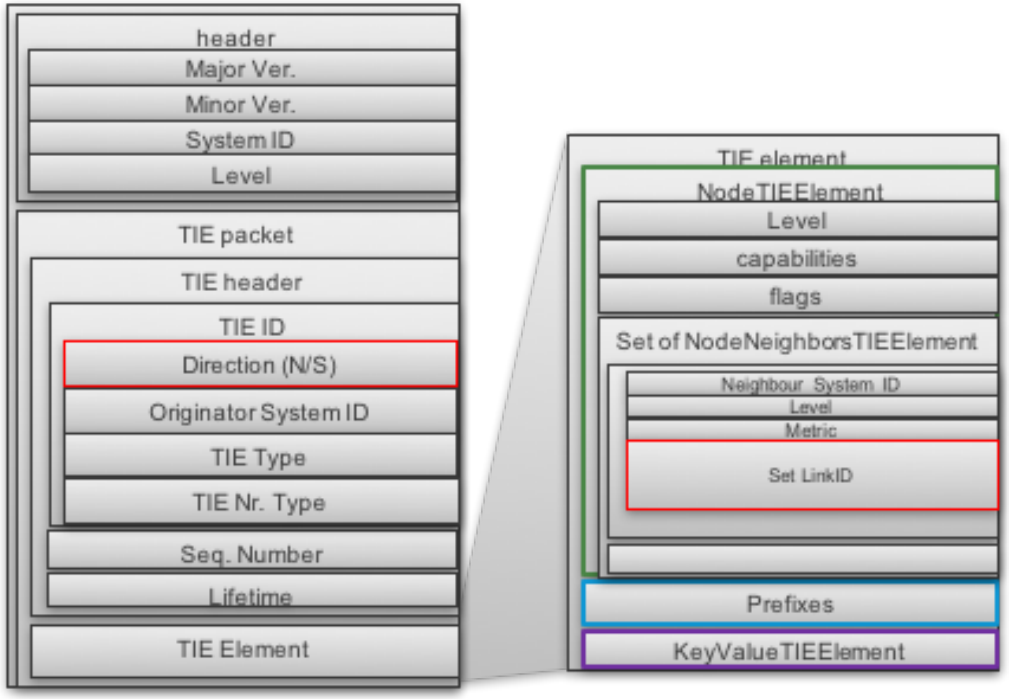
- LIE(Link Information Element)
  - Level #
    - Level # == 0 “Leaf only”
    - Level # == 24 “top_of_fabric_level”
  - POD #
    - POD # == 0 “Any POD”
- Major/Minor Version
  - Protocol Versionがcompatibleかどうかの確認
  - Major Versionが異なるとCLEANUP
- Well–Knownマルチキャストアドレスで転送
  - デフォルト 224.0.0.120
- 3-way adjacenciesを確立
  - ローカル設定をZTPに必要な情報を交換
  - BFDセッションの自動確立(May Interactions)



Thrift file for packet encodings sample

TOPOLOGY EXCHANGE (AKA LSA)

- Topology Information Element (TIE)
- North-TIE := South-TIE + carries link-state info
  - Attributes
  - Detailed topology
- BGPのようにsmall incrementalの広報
  - Not all fields always required
  - Contextual
- TIE Types
  - Node (N/S) TIE
  - Prefix (N/S)TIE
  - Positive/Negative Disaggre South TIE
  - External prefix
  - KeyValue (N/S) TIE

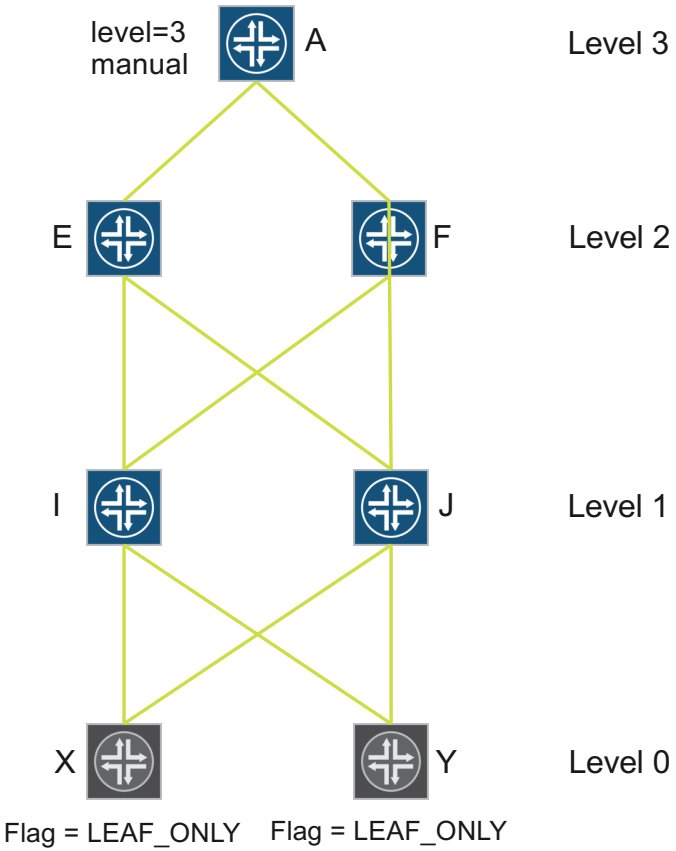


AGENDA

- RIFT登場の背景
- RIFTとは
  - 概要
  - 機能紹介
  - 参考 – JUNOSサンプル
- まとめ

ゼロタッチプロビジョニング

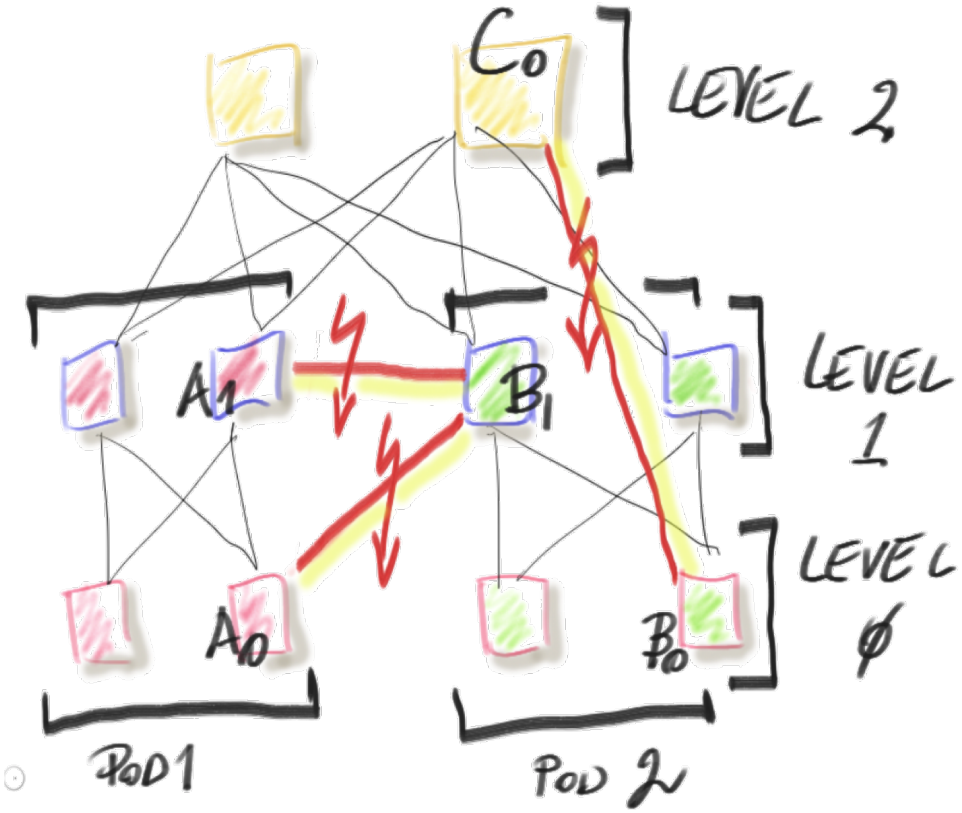
- 最上位のスイッチのみ、手動で設定
  - “level top-of-fabric”とすると最上位となる。
  - 任意のレベルを設定することも可能
    (図ではLevel 3)
- Levelが設定されていない機器が接続されると、上位のレベルから1減算したレベルが自動的に適用
  - E, F -> Level 2
  - I, J -> Level 1
- “leaf” Levelを手動設定する場合、自動的にレベル0
- 自動的にSystem IDを割り当て
  - 64ビットのIDが、ノードのEUI-64を元に生成
- 実装により、IPv4 over IPv6も可能となり、リンク間のIPv4アドレス不要



自動的なトポロジー分析：ケーブルミスマッチ

- 自動的に誤った接続を検出、隣接関係を形成しない
- “miscabled_links”として伝搬

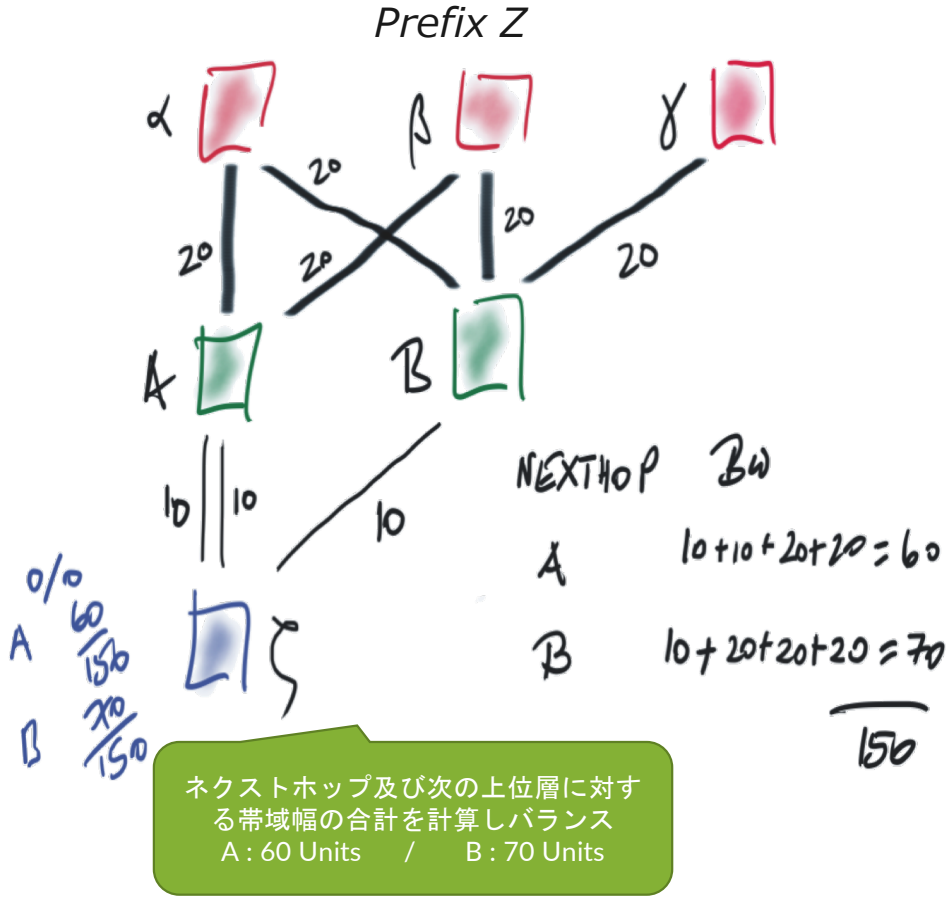
- 例:
  - A1とB1はPODが異なるため、隣接関係を形成しない
  - A0は既にA1と接続されていることから、POD1のリーフであることを学習する。これにより、異なるPODのB1との隣接関係を形成しない
  - B0とC0は階層を飛び越えているため、隣接関係を形成しない



WEIGHTED BANDWIDTH LOAD-BALANCING： ECMP最適化

アップリンクの帯域をベースに非コールコストマルチパスの実現

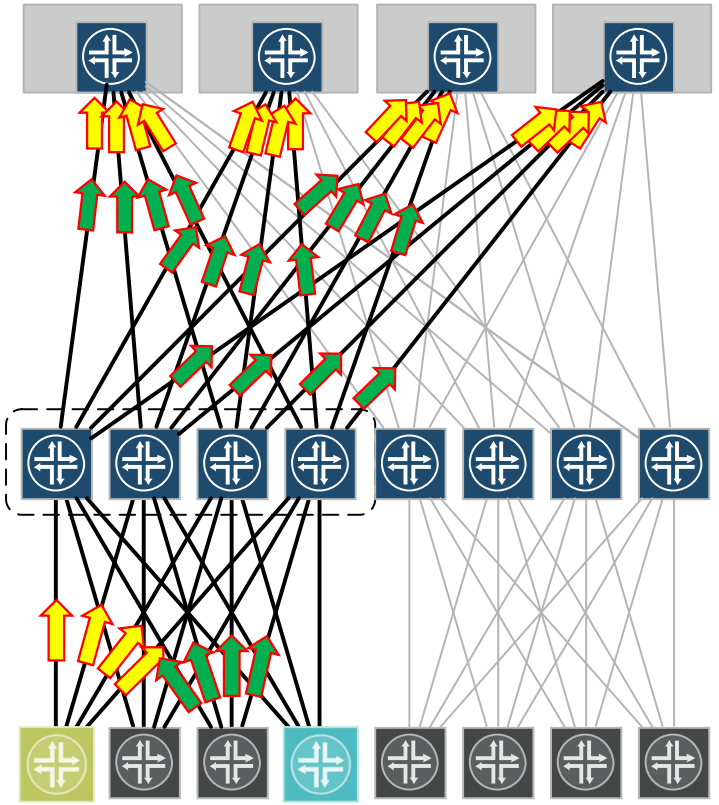
- 上流ノードとの接続帯域と、更にその上流ノードの接続している帯域を元に計算
- 経路のディスタンスとこの値を元に、ロードバランス比率を計算する。(BAD: Bandwidth Adjusted Distance)
- リンク障害時も再計算し比率を変更
- RIFTはループフリーのため、安全



FLOOD-REDUCTAION(1)：通常のフラッディングは非効率

通常のリンクステートフラッディング
無駄な重複したフラッディングが多い

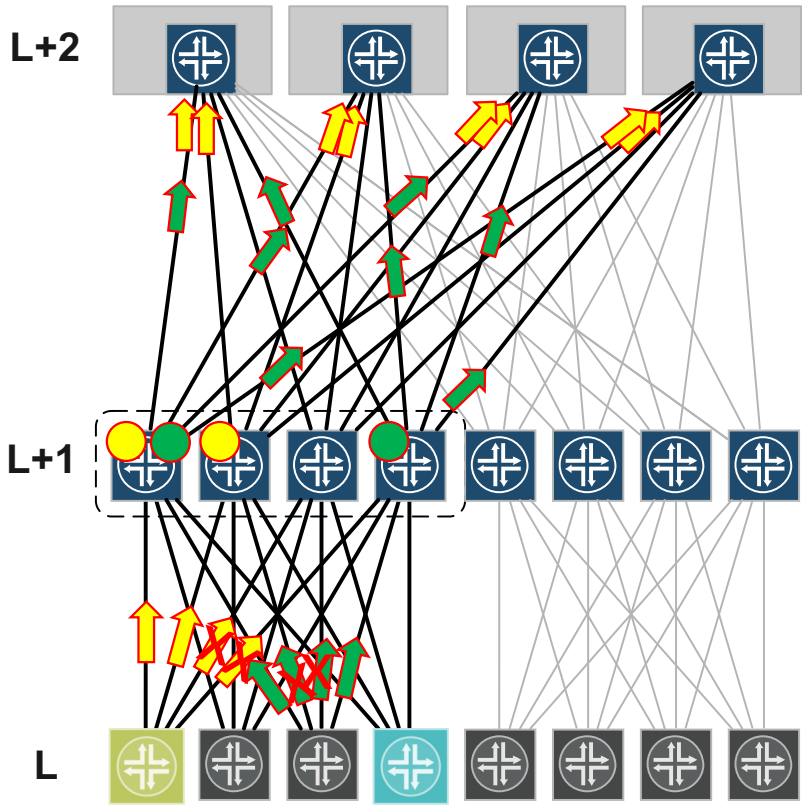
- DCにおいては、非常に高密度なノード間の配線が存在
- スーパースパイン
  - 全ポートが下位スイッチに接続
- スパイン・リーフ
  - 半数のポートが下位スイッチに接続
  - もう半数は上位スイッチに接続
- リンクステートプロトコルの課題：
  - リンクステート情報のフラッディングが非常に多く、効率が悪い
  - 冗長な情報を複数の機器から受信



FLOOD-REDUCTAION(2)：フラッディングの効率化

RIFTは不必要なフラッディングを削減

- 各ノードは、上位およびその上位がどのノードと接続している把握
- 上位ノードに対してフラッディングされるリンクステート情報は、いずれかの1台のみが,フラッドすれば十分
  ->**Flood Repeater(FR)ノードのみがフラッド**
- 選出アルゴリズムはRIFT draft (Section 4.2.3.8)
- 冗長性のため少なくとも2つのFRノード選出
- ノードLは、FRではないノードに対しては、'**do-not-reflect**' フラグを付けてリンクステート情報を送信
- 同じようなアルゴリズムが各階層で実行され、フラッディングを大幅に削減でき、障害の影響範囲も大幅に削減



OTHER RIFT FEATURES

- Overload bit (like Overload in IS-IS)による容易なトラフィック迂回
- Leafやサーバー上の Lightweight RIFT North RIFTのみ) として安価Leaf対応
- モビリティ(高速なホストの移動をサポート (他プロトコルより少なくとも2倍高速)
- Key-Value Store (e.g. KVSによるポリシー制御のため任意の情報伝達)
- マルチプレーン・ファブリック
- セキュリティ(authentication key, Outer authentication etc)
- トランスポート層への非依存(QUIC, TCP, UDP, UDT等でも動作させることも可能)
- オーバーレイ非依存(VxLAN, NVO3 etc - draft 4.3.3.4)
- ポリシーベースのトラフィック・エンジニアリング(Policy Guided Prefix, draft-atlas-rift-pgp)
- SRをサポート可能(SRIFT: Segment Routing In Fat Trees / draft-zzhang-rift-sr)

AGENDA

- RIFT登場の背景
- RIFTとは
  - 概要
  - 機能紹介
  - 参考 – JUNOSサンプル
- まとめ

JUNOS RIFT設定 サンプル

TOP OF FABRIC

```
protocols {
    rift {
        node-id auto;
        level top-of-fabric;
        lie-receive-address {
            family {
                inet 224.0.0.120;
                inet6 ff02::a1f7;


interfaces {
    interface-range rift-interfaces {
        member ge-0/0/*;
        unit 0 {
            family inet;
            family inet6;
    }
```

Level設定のみ

LEAF

```
protocols {
    rift {
        node-id auto;
        level auto;
        lie-receive-address {
            family {
                inet 224.0.0.120;
                inet6 ff02::a1f7;
                    }
            }
        interface ge-0/0/40.0 {
                    mode passive;
```

```
interfaces {
    interface-range rift-interfaces {
        member ge-0/0/*;
        unit 0 {
            family inet;
            family inet6;
    }

ge-0/0/40 {
  unit 0 {
     family inet {
        address 192.168.113.1/24
```

サーバーフェーシングインタフェースの設定(アドレス、passive)のみ
RIFT on Serverを使う場合は、これも設定必要なし

※passive interfaceは近々実装
現在はpolicy-optionで広報必要

JUNOS RIFT経路情報 サンプル

TOP OF FABRIC

```
root@spine> show route protocol rift table inet.0

inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.113.0/24   *[RIFT/200/100] 00:00:13, metric2 0
                    >  to fe80::5200:ff:fe20:3 via ge-0/0/2.0
192.168.114.0/24   *[RIFT/200/100] 00:00:23, metric2 0
                    >  to fe80::5200:ff:fe21:3 via ge-0/0/3.0
224.0.0.120/32     *[RIFT/20/100] 00:00:25
                      MultiRecv
```

v4の経路がv6のlink local宛の経路となっている

LEAF

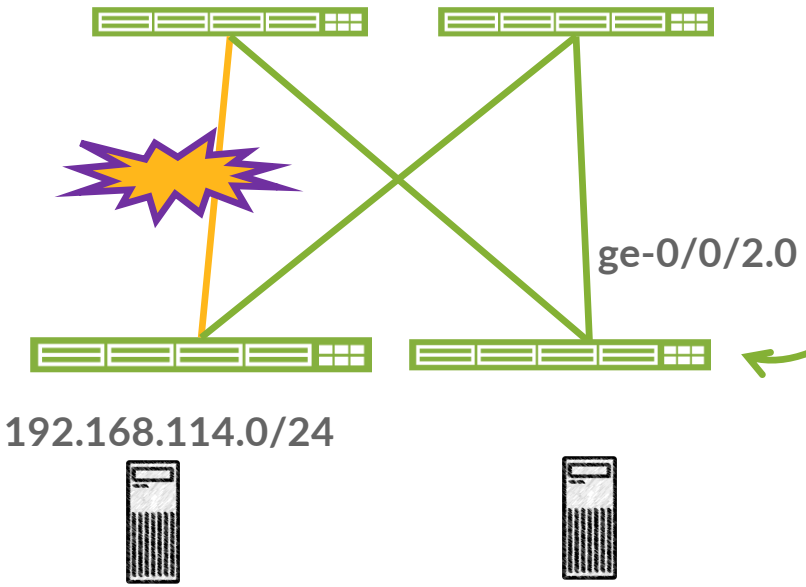
```
root@leaf> show route protocol rift table inet.0

inet.0: 7 destinations, 8 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[RIFT/20/100] 00:00:02, metric2 0
                    >  to fe80::cee1:94ff:fe2d:1ac3 via ge-0/0/1.0
                       to fe80::cee1:94ff:fe2d:1fc7  via ge-0/0/2.0
224.0.0.120/32     *[RIFT/20/100] 00:00:44
                      MultiRecv
```

LEAFではドフォルトルートのみを受信

JUNOS RIFT経路情報 サンプル – AUTOMATIC DE-AGGREGATION



LEAF

```
root@leaf> show route protocol rift table inet.0

inet.0: 7 destinations, 8 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[RIFT/20/100] 00:00:02, metric2 0
                    > to fe80::cee1:94ff:fe2d:1ac3 via ge-0/0/1.0
                      to fe80::cee1:94ff:fe2d:1fc7  via ge-0/0/2.0
192.168.114.0/24   *[RIFT/200/100] 00:00:17, metric2 0
                    > to fe80::cee1:94ff:fe2d:1fc7 via ge-0/0/2.0
224.0.0.120/32     *[RIFT/20/100] 00:00:44
                      MultiRecv
```

ECMPでブラックホールにならないように
特定のルートのみを学習しECMPで送らないように

JUNOS その他 SHOW

トポロジー & ミスケーブル

```
root@SPINE> show rift topology nodes
.                                                  +-------- Links-----+--- TIEs ----+- Prefixs -+
Lvl Name         Originator       Ovrld     Dir|  3-way|  Miscbl|Secure| Auth | Non  | V4  | V6  |Latest TIE Origination
---+----------+-----------------+-----+---+-----+------+------+------+------+-----+-----+-----------------------
 24 unknown    00cce1942d1aa000            2                                  1                 2019/11/21 21:53:28
 24 unknown    00cce1942d1fa000       N    2                                  4     1     1     2019/11/19 14:31:23
 23 unknown    004c9614f289e000       S    2                                  5     3           2019/11/21 21:53:18
 23 unknown    004c9614f2e02000       S    2          2                                4     2           2019/11/21 21:31:06

root@LEAF> show rift interface status |
Link ID: 262, Interface: xe-0/0/1.0
Status Admin True, Platform True, State: OneWay
LIE TX V4: 224.0.0.120, LIE TX V6: ff02::a1f7, LIE TX Port: 914, TIE RX Port: 915
PoD 0, Nonce 26962
Miscabled: SubnetsMismatched, Neighbor: None, Level: None, Detected: 2019/11/21 22:08:23.538
```

BFD

```
root@SPINE> show bfd session
                                          Detect   Transmit
Address                  State     Interface     Time     Interval  Multiplier
fe80::4e96:14ff:fef2:8a27 Up       et-0/0/1.0    3.000    1.000        3
fe80::4e96:14ff:fef2:e067 Up       et-0/0/9.0    3.000    1.000        3
2 sessions, 2 clients
```

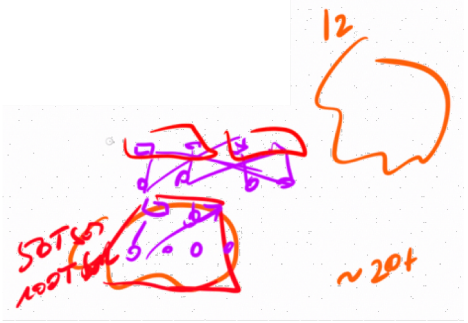
※βではrift配下でBFDタイマー設定必要

AGENDA

- 背景
- RIFTとは
- RIFT機能紹介
- サンプル動作 – JUNOSの場合
- まとめ

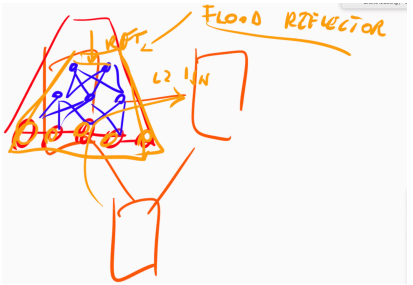
様々なDCユースケースに対応

- IP Fabric
  - シンプルなIPファブリック
  - サーバ上でRIFTデーモンを動作させる、マルチホームや、アドレスモビリティにも対応
  - SDN製品(Contrail, VMware NSXなど)のアンダーレイとしても使用可能



- Underlay for EVPN-VXLAN/NVO3 Overlay
  - アンダーレイをRIFTで構成することで、BGPはオーバーレイに特化してシンプルに

- Metro IP Fabric
  - 階層構造を持つメトロ・アクセスネットワークにおいても利用可能
  - SRと組み合わせることで様々なサービスを提供
  - シャーシデバイスのファブリック化



INDUSTRY STATUS

Standardization

- JuniperのAntoni Przygienda(Tony) が考案
- draft-ietf-rift-rift-09 (November 4th, 2019) Juniper, Cisco, Comcast, Yandex
- スタンダードトラックRFCの発行に向けてIETF
- IETF106(11/21)でも Working Groupあり
- パケットデータ構造はオープン (GPB)
- RIFT YANG model: draft-zhang-rift-yang
- **SRRIFT: SR RIFT:draft-zzhang-rift-sr**
- PGP with RIFT: draft-atlas-rift-pgp
- Multicast RIST: draft-zzhang-rift-multicast
- BIER with RIFT : draft-zzhang-bier-rift
- IETF102-104 RIFT Hackathon:
  - オープンソース実装との相互接続検証や、Chaos Monkey Testingやベンチマークなど
    参加者: Juniper, Cisco, Likedin, Yandex , Mellanox, HPE, Nuage & others

Implementation

- プロトタイプ実装が存在
- PoCテスト、パフォーマンス検証、インターオペラビリティ既に実施済

製品実装状況

- Opensource RIFT- Python
- Juniper RIFT(現在β、近々正式リリース)
- ZTE - (開発中/済?)
- FRR - (実装検討中?)
- Others –

Interop Tokyo 2019

- ShowNetにてPython RIFTとJuniperで相互接続

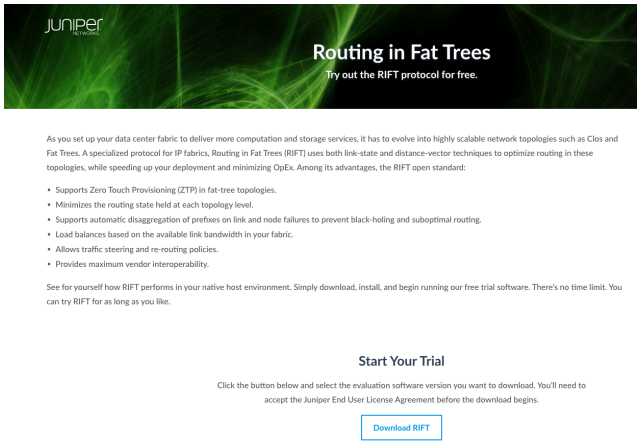
参考:JUNIPER IMPLEMENTATION AVAILABILITY

- LinuxやMac OS Xで動作する検証用バイナリを配布中
  https://www.juniper.net/us/en/dm/free-rift-trial/

- Junosのβイメージあり
  - For QFX/MX/vMX
  - RIFT packageの追加
  - REDISからリンクステートDBや統計情報の取得可能
  - マルチスレッド対応
  - Memory and Thread-Safe
  - 設定とステートは全てThriftベースでモデル化
    ※ご要望次第でイメージ提供可能

- 近々正式リリース予定

- サーバ上でRIFTが動作するcRPDにも実装予定あり



MORE MATERIAL

- Specifications in IETF Working Group
  - https://datatracker.ietf.org/wg/rift/about/
- Open Source Implementation RIFT- Python
  - https://github.com/brunorijsman/rift-python
    - **How to install * on AWS, on Vagrant**
      https://github.com/brunorijsman/rift-python/blob/master/doc/installation.md
    - **Positive Disaggregation Feature Guide**
      https://github.com/brunorijsman/rift-python/blob/master/doc/positive-disaggregation-feature-guide.md
    - **Flooding Reduction Feature Guide**
      https://github.com/brunorijsman/rift-python/blob/master/doc/flooding-reduction-feature-guide.md
- Walk Through Major Concepts & Package Explanation (RIFT Interim Recording rift-01)
  - May 2, 2018: https://www.youtube.com/watch?v=BZtFPTgcsbs
  - May 3, 2018: https://www.youtube.com/watch?v=dtxNoCkC7MA

RIFT(Routing in Fat Trees)
～IP Fabric Routing Open Standard and Zero OPEX～

DCファブリックの特性に最適化されたルーティングプロトコル

Link-State and Distance Vector		RIFT独自の機能
両方の ‘いいとこ’取り	• 高速なコンバージェンス • 自動的なトポロジー検出 • ToRで保持する経路の最小化 • 高スケールのECMP • ノードの高速な切り離し	• ゼロタッチプロビジョニング • 障害時の自動的な経路再分割 • 障害時の影響範囲の最小化 • ループフリーで全てのパス有効 • 非イコールコストのマルチパス • Key-Value Storeによる拡張
良くない点の 改善	大量のフラッディング 手動での隣接関係設定	

標準化前進中(改善要望などご意見募集)


THANK YOU
JUNIPER NETWORKS
Engineering
Simplicity