OpenFlowを使ったネットワークテスト自動化システムの構築

沖縄オープンラボラトリ
L1patch応用ネットワークテストシステム
プロジェクト

2016/2/6

沖縄オープンラボラトリ/株式会社アドックインターナショナル
吉田 正之

OKINAWA OPEN LABORATORY

PJモチベーション

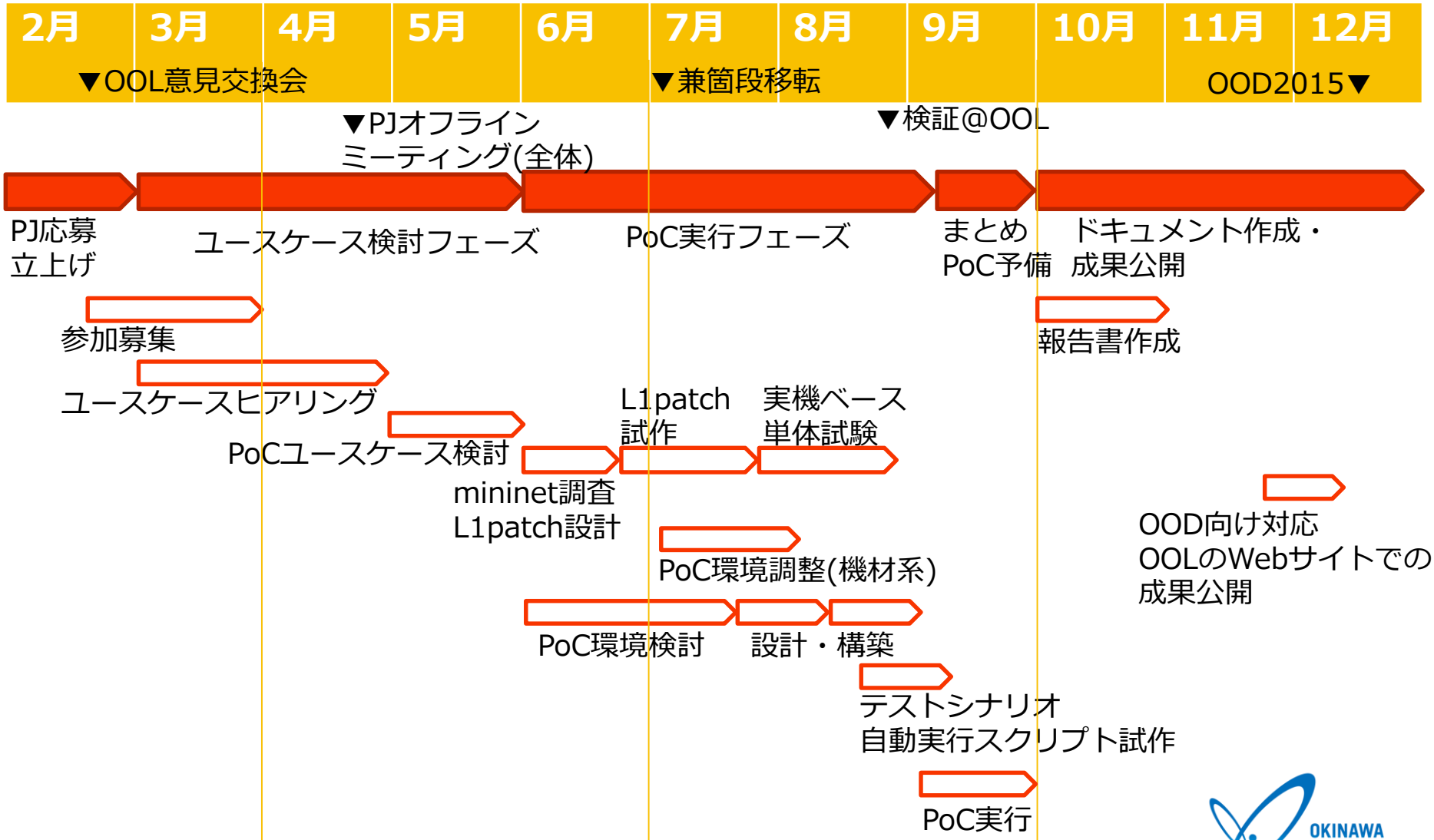
- サービスとしてはNW(を含む)テストが必要

NW構築、NW制御、運用管理システムの開発・実装
→　<u>実際、実装したとおりに動作するか?</u>

- NWのテストは手作業が多く自動化が難しい。
  - NWは「物理的な位置」を抽象化する: NW自体のテストは物理的な構成・トポロジ・位置情報に依存してしまう。
  - テストに時間がかかってしまってはAgilityが改善しない。
    → 手作業に頼っている間は大きな改善は困難。

L1patchの応用で「NWに対するテスト」を変えられないか?

これまでの活動状況


2月
3月
4月
5月
6月
7月
8月
9月
10月
11月
12月
▼OOL意見交換会
▼兼箇段移転
OOD2015▼
▼PJオフライン
ミーティング(全体)
▼検証@OOL
PJ応募
立上げ
ユースケース検討フェーズ
PoC実行フェーズ
まとめ
PoC予備
ドキュメント作成・
成果公開
参加募集
報告書作成
ユースケースヒアリング
L1patch
試作
実機ベース
単体試験
PoCユースケース検討
mininet調査
L1patch設計
OOD向け対応
OOLのWebサイトでの
成果公開
PoC環境調整(機材系)
PoC環境検討
設計・構築
テストシナリオ
自動実行スクリプト試作
PoC実行

プロジェクト成果の発表

プレスリリースでの活動報告

OODでのプロジェクト紹介

プロジェクト成果報告

OKINAWA
OPEN
LABORATORY

プロジェクトの概要

- 実機・実体としてのネットワークを効率よくテストするためのシステムとユースケースのアイディアを実証する。
  - 一般的なNWテストのプロセスの自動化
  - NWの物理・論理構成パターンを網羅するテスト用計算機リソース確保と配置問題の解決
  - テストパターンの定義によるテスト自動実行
- PoC実装
  - oolorg/ool-l1patch-dev · GitHub
    https://github.com/oolorg/ool-l1patch-dev

OKINAWA OPEN LABORATORY

目的

SDN/SDx:

NWの構築・実装のAgility

- NW構築・制御、運用管理システムの開発・実装で求められる"保証/確認"
  - 「実装したとおりに制御されている」
  - 「すべての機器が想定通り動作している」
- 開発のスピードに追従できるネットワークのテスト

SDN: Software Defined Networking
SDx: Software Defined Anything

「NWのテスト」もSoftware-Definedになる

そのためにはどのようなシステムがあればよいか?

OKINAWA OPEN LABORATORY

そのために解決すべき課題

ネットワークは物理的な配置を抽象化する

ネットワークそれ自身に対する操作は、物理配置を考慮しなければいけない

「現地・現物」の操作 → 人による操作

「人による」操作は自動化できない
→ スケールしない。

多数のデバイス・論理環境のかさねあわせ

- 複雑さの増大
  - 仮想化に伴う「実際何が起きているのか」の把握しにくさの拡大

“現地・現物・人海戦術“からの脱却

×

ソフトウェア化による拡張性と複雑さへの対応

OKINAWA OPEN LABORATORY

実現したいこと

テストにおける人手での介入をなくす

- 人が直接作業する・移動するということをやめたい
  - 作業がスケールしない
  - 属人性が高い
  - 単純に時間(コスト)がかかる

試験パターンの機械的処理

- リソースに合わせてテストを「縮退」させたくない
  - NWテストのほとんどはパターンに沿った単純なテスト
  - 本来注力すべきは網羅したいテストパターンのそのものの検討
  - 実施すべきテストは全パターン実施したい。
    1. ネットワークに対する回帰テスト(regression test)をしたい。

OKINAWA OPEN LABORATORY

テストにおける人手の介入をなくす

(1) テスト対象の物理構成を操作したい

トポロジの組み替え・障害試験の自動化

物理構成は人手で操作 → 作業スケーラビリティのネック

特に検証環境で、
トポロジ変更やノードの追加・削除などを伴うテストの自動化。

(2) テスト用リソースの拡張性

任意の箇所(物理・論理)にテスト用のノードを置きたい

複数のテスト用ノードをまとめてコントロールしたい

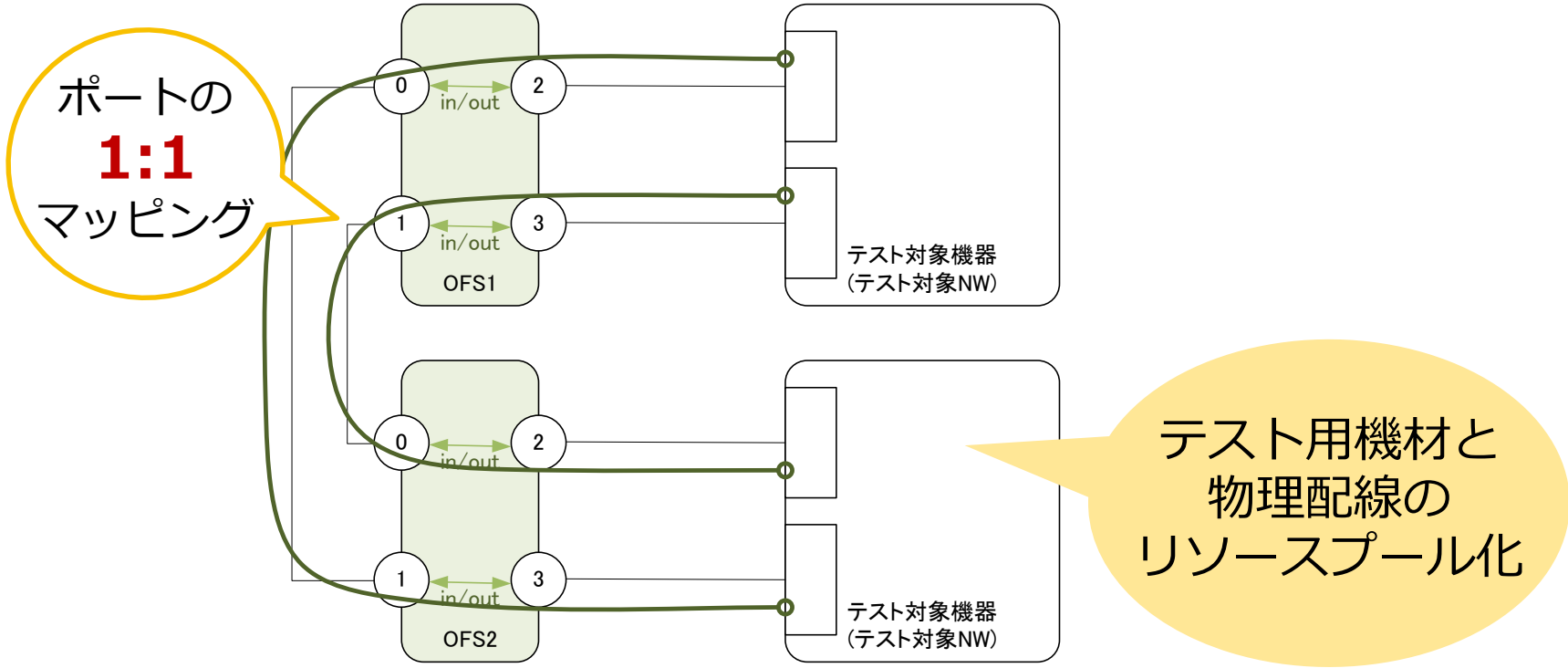
本番・検証を問わず
複数のテスト用トラフィックを効率よく流すための仕組み。

OKINAWA OPEN LABORATORY

(1) 物理構成の操作

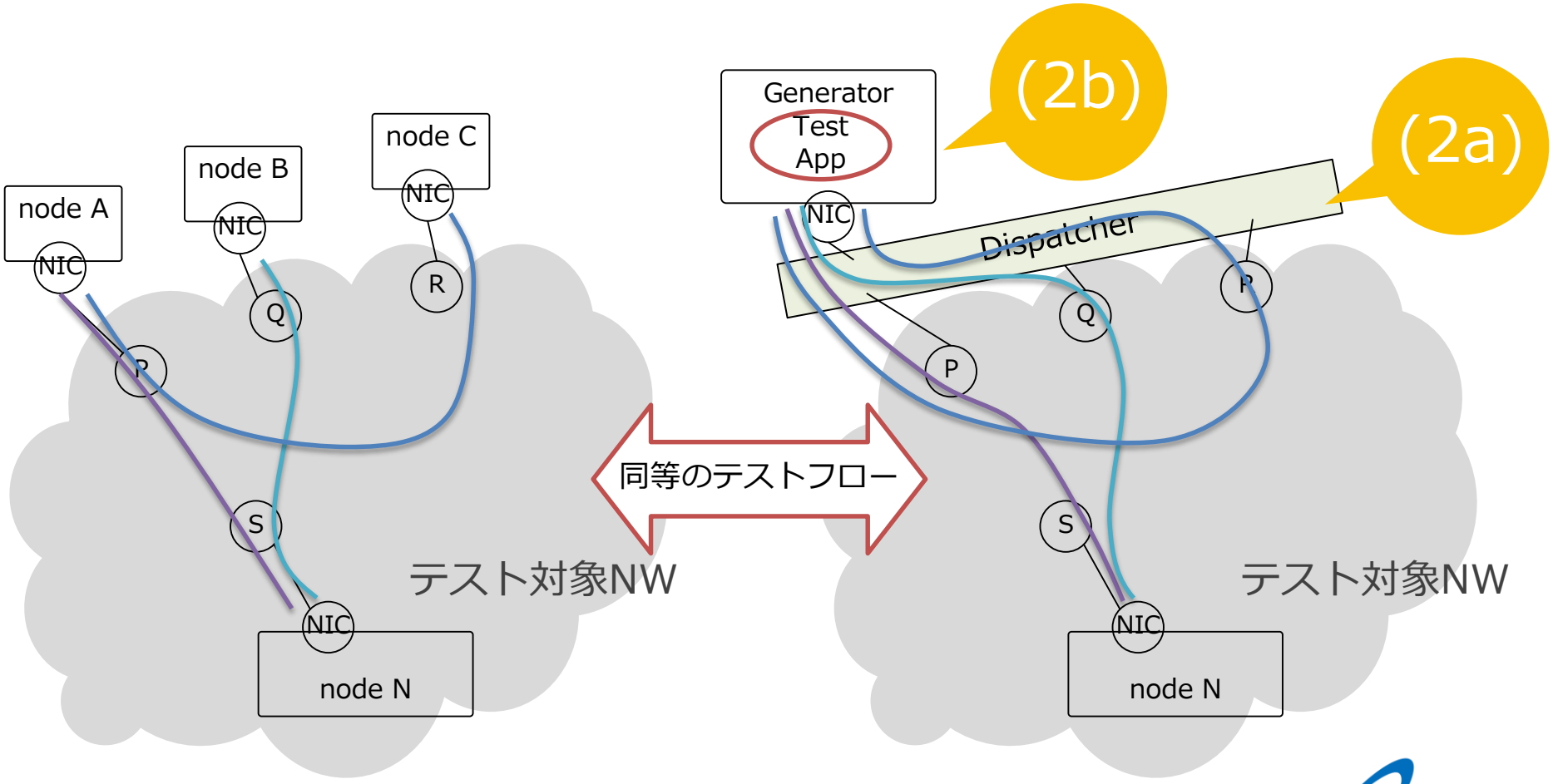
L1patchによる“専有モード”ワイヤ

テスト対象NW(物理NWトポロジ)を、設定に応じて動的に作る(操作する)仕組み。



(2) テスト用リソース配置

トラフィックの生成/テスト対象へのパケット分配の分離

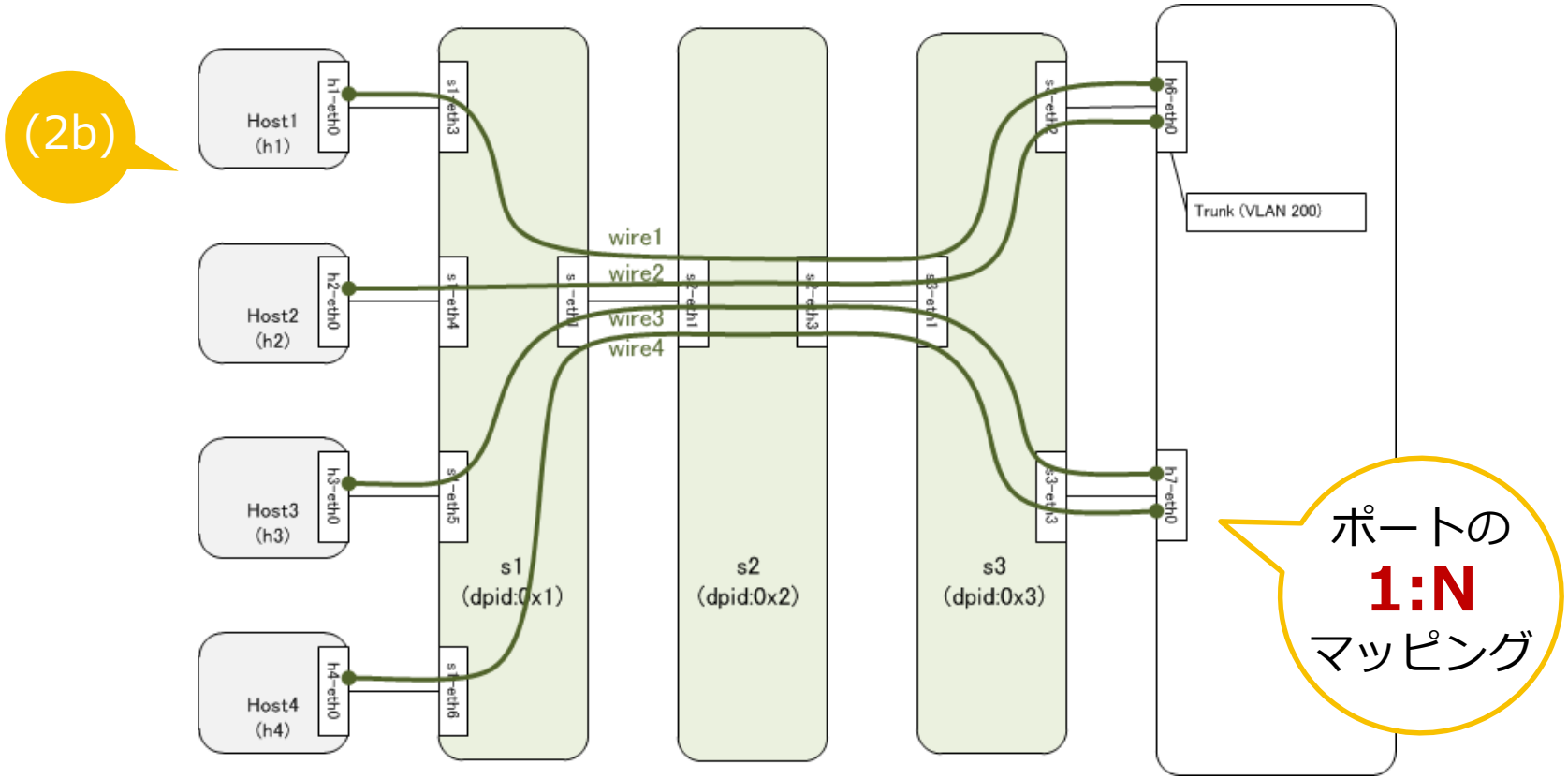


OKINAWA OPEN LABORATORY

(2) テスト用リソース配置

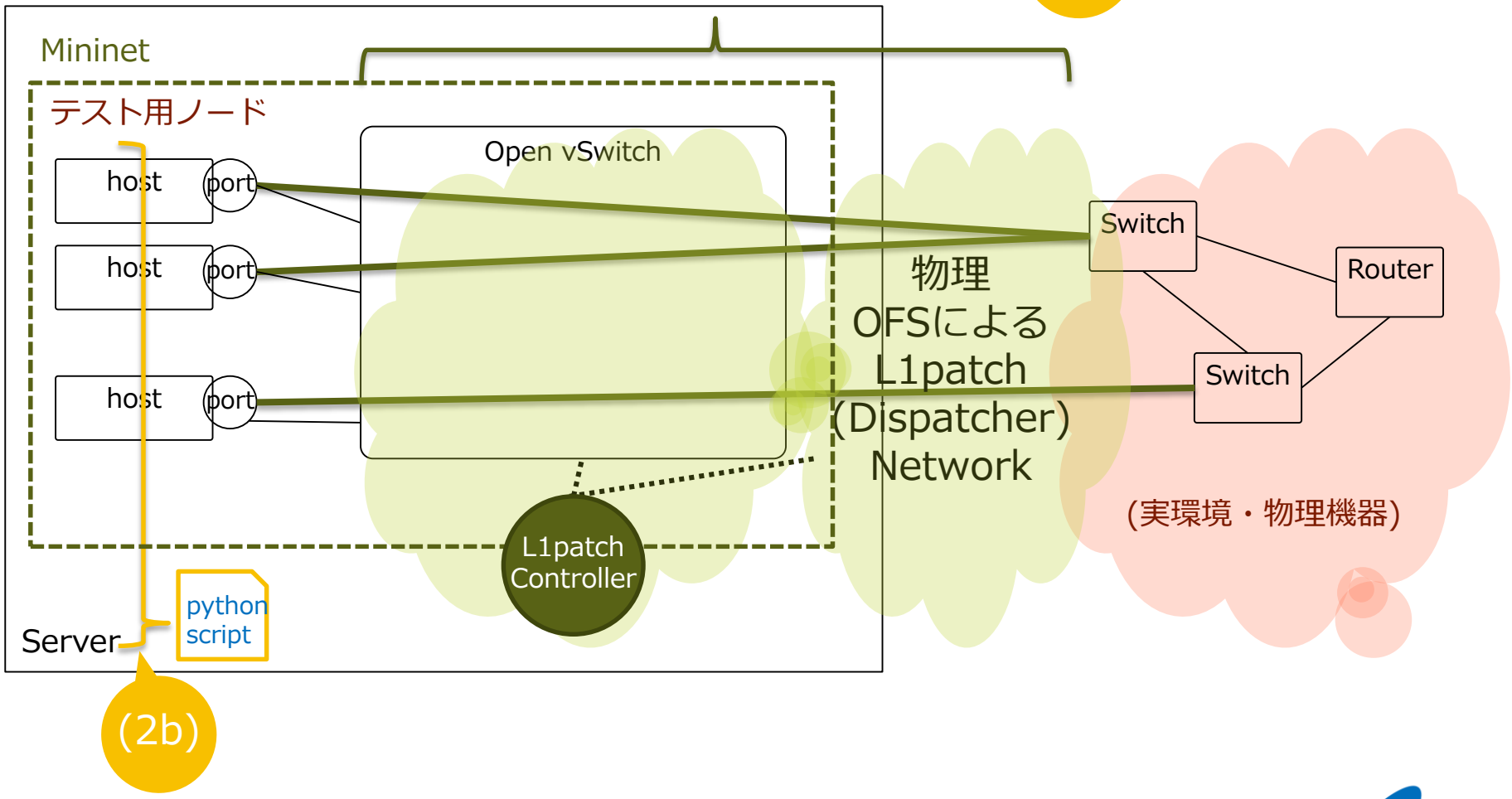
L1patchによる“共有モード”ワイヤ

物理ポート制約を回避したテスト用トラフィック分配



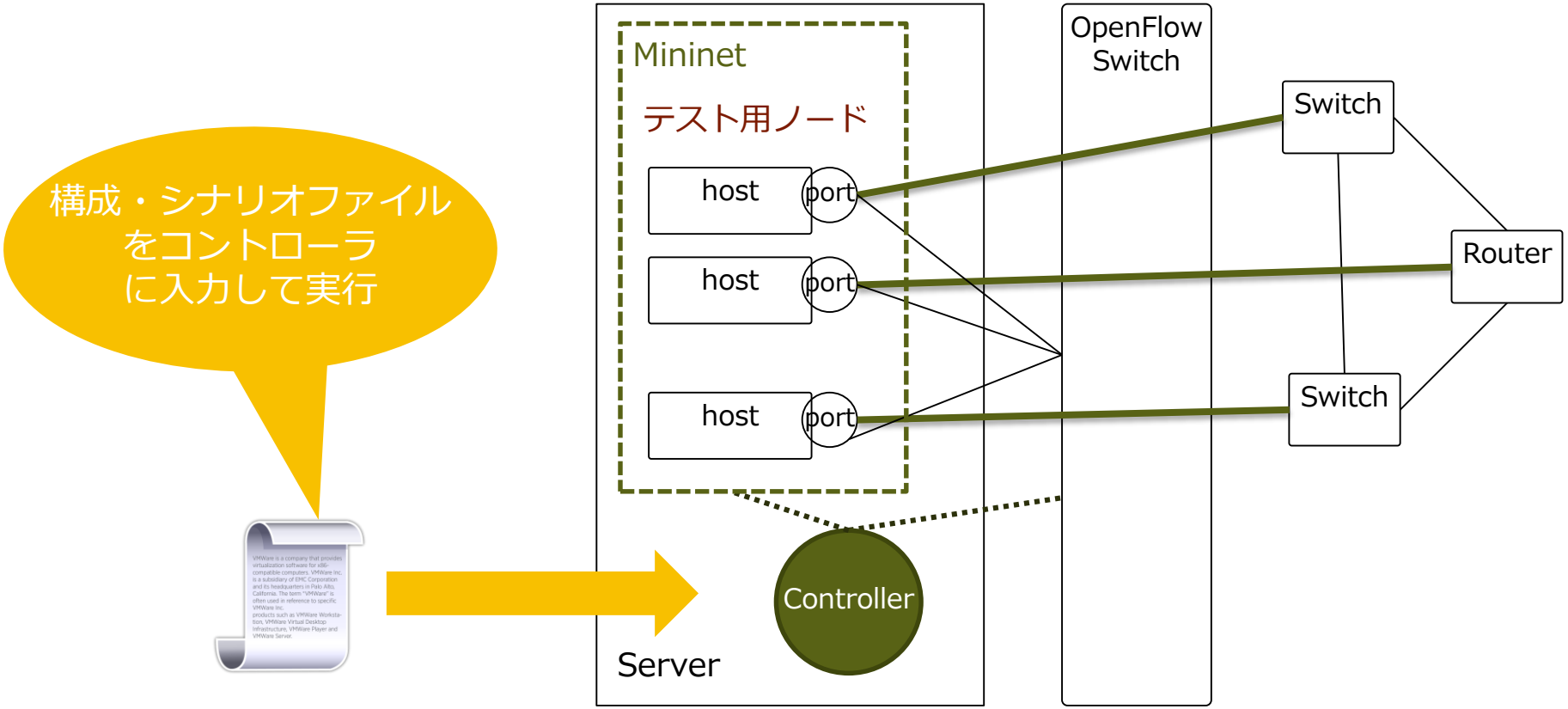
(2) 物理環境への適用

OFS全体を
L1patchとして動作させる (2a)



OKINAWA OPEN LABORATORY

試験パターンの機械的処理

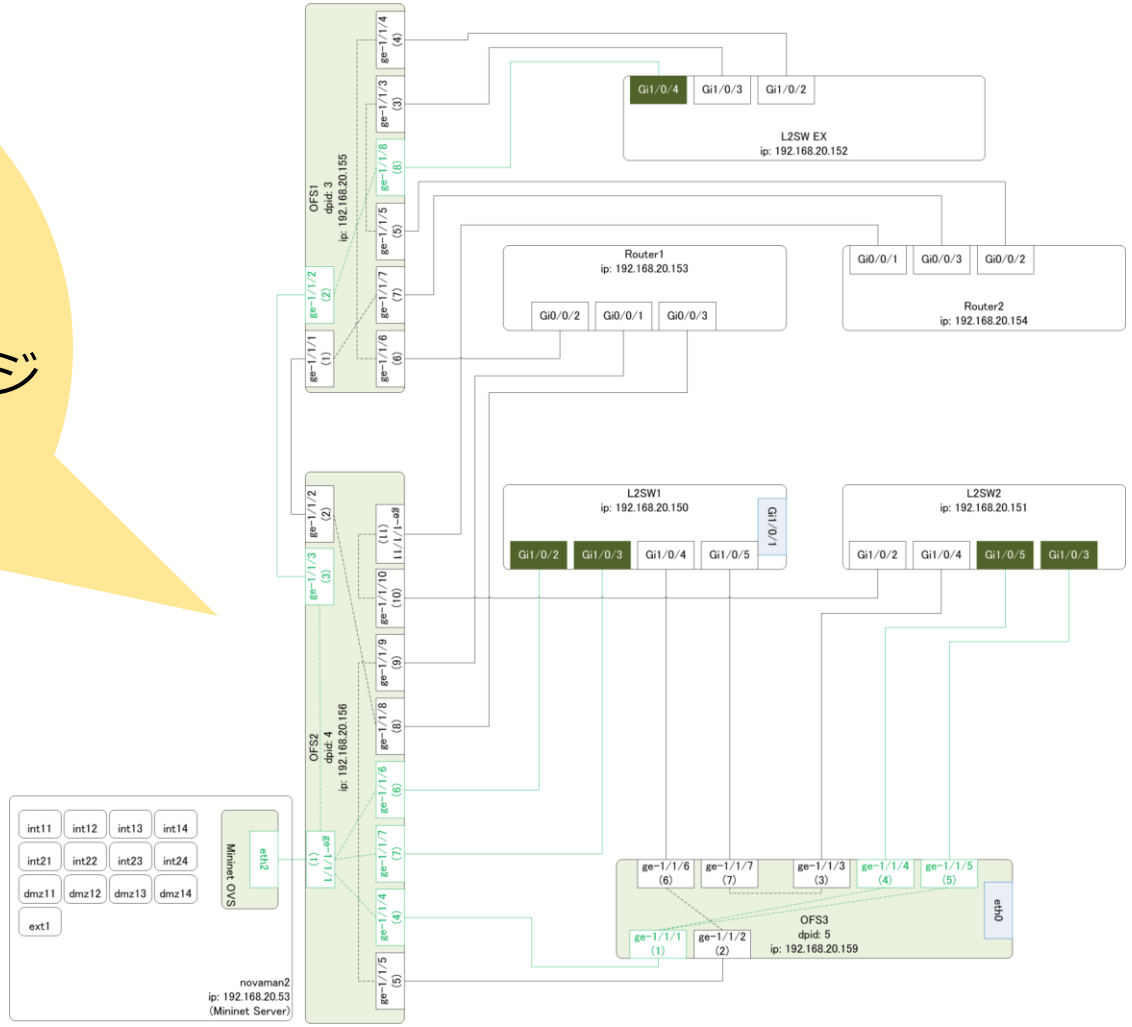


OKINAWA OPEN LABORATORY

PoC環境の物理構成について

「本当の」物理構成

L1patchによる
(1)テスト対象NWトポロジ
(2)テスト用ノード配置
のマッピング

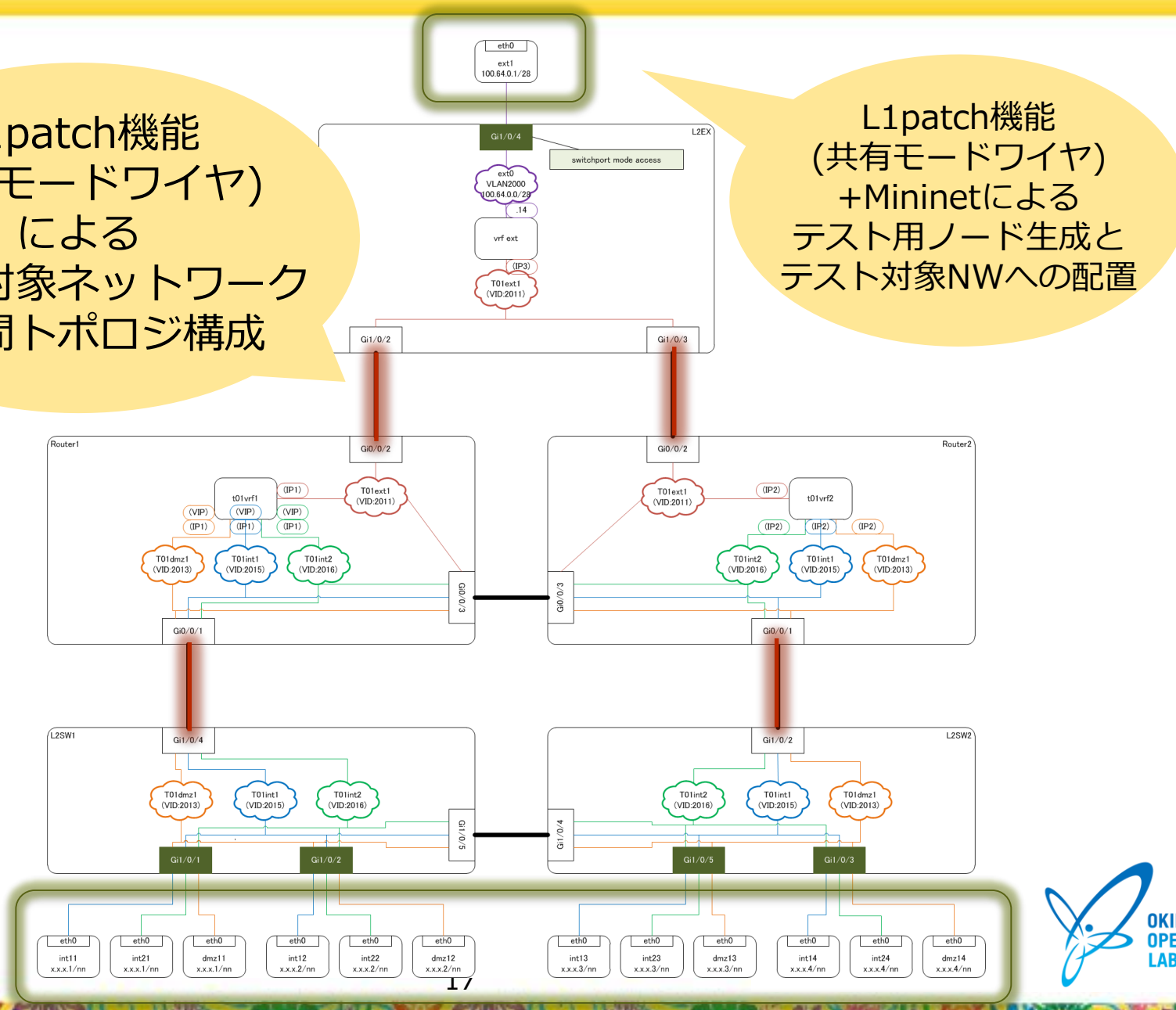


OKINAWA OPEN LABORATORY

テスト用ノードの配置と操作

L1patch機能
(専有モードワイヤ)
による
テスト対象ネットワーク
機器間トポロジ構成

L1patch機能
(共有モードワイヤ)
+Mininetによる
テスト用ノード生成と
テスト対象NWへの配置



OKINAWA OPEN LABORATORY

テストパターン

```
"dmz1(SUCCESS)": [
    [
      "dmz11",
      "dmz12"
    ],
    [
      "dmz13",
      "dmz14"
    ],
    [
      "@router1-dmz1-ip@",
      "@router2-dmz1-ip@",
      "@dmz1-vip@"
    ]
  ],
```

① ② ③

①→②
①→③
②→②
②→③

```
    {
      "source": "dmz11",
      "destination": "dmz13",
      "task": "[dmz1] ping dmz11
to dmz13",
      "command": "ping",
      "expect": "SUCCESS"
    },
    {
      "source": "dmz11",
      "destination": "dmz14",
      "task": "[dmz1] ping dmz11
to dmz14",
      "command": "ping",
      "expect": "SUCCESS"
    },
```

OKINAWA OPEN LABORATORY

効果

OKINAWA
OPEN
LABORATORY

作業リソースや工数の比較

従来(手作業)

- 作業者2名、最低2-3ノードの設定と配置の繰り返し
- 90ケースのテスト
  - テストケース作成で2h程度
  - テスト実行に0.5-1h程度
- 現地移動・現物操作でリンク障害テスト
  - リンク障害起こした後にもう一度再実行

L1patch(全自動)

- 作業者1名・サーバ1台・OpenFlowスイッチ
- 194ケースのテスト
  - パターン定義は1h弱
  - テスト実行に数分
- コマンド操作で障害テスト模擬
  - 何回でも同じテスト繰り返し

OKINAWA OPEN LABORATORY

L1patchのよいところ

テストの再現性

- テストに必要な操作・テストプロセスの記述
  - テスト対象の物理トポロジも設定で定義できる
    1. 「前やったあの構成へ戻したい」などがコンフィグ操作で可能になる
    2. 障害試験におけるトポロジの変更などもプログラム

パターン網羅

- 物理構成網羅
  - OFSポート数次第…”ネットワーク”レベルの拡張性
  - 必要なノードの生成・配置の自由度
- テストオペレーション(テスト用ノード)の集中制御
  - ログ取得なども1カ所でまとめて実行

OKINAWA OPEN LABORATORY

L1patchの課題

テスト自動化実装にかかるコスト

「やりたいテスト」を実装・テストするコストはかかる。

ある程度くりかえせる・汎用性のあるテストでないと、コスト対効果が悪い。

ベストプラクティスの蓄積、共通機能のモジュール化でハードルは下げられるのでは。

ダイナミックなテスト

複数のノードを同時に操作するというのが今のところやれていない。

レイヤの高い通信のテスト

ICMP → tcp/udp, アプリケーションレベルのテスト (FW, LB等のテスト)への応用

OKINAWA OPEN LABORATORY

将来的なターゲット

インフラのCI, TDDな基盤システム構築あるいは運用

基盤を含むオーケストレーション・自動化、そのためのシステム開発において、「ネットワークのテスト」を効率よく実行する仕組み。

検証環境でのリハーサルとテスト。あらかじめ十分なテストで確認済みのオペレーションを行えるような運用スタイルの確立。

既存の環境などへのアドオン。環境に対するオペレーションに対して都度テストを実行し、安全性を確保しながら作業できるような運用の実現。

見つけにくいネットワークの障害の早期発見

L2機器はパケット通してみないと障害がわからない。

大規模すぎて環境全部に目が届かない。

通信キャパシティや品質劣化の兆候を、デバイス単位など、なるべく狭い範囲で早めに特定したい。

OKINAWA OPEN LABORATORY

来年度の活動について

新機能の追加

■Ping以外のトラフィックを流せるようにアプリの開発。
・ファイアウォールのポリシーの網羅試験などに対応したい
■各ルータ・スイッチなどへのコンフィグ投入機能の追加

各コンポーネントのモジュール化と強化

■現在、実装済みの機能をモジュール化、機能を追加して再度公開
・トポロジー作成機能
・リソース作成・配置機能

興味のある方はぜひお声がけください！

OKINAWA OPEN LABORATORY

まとめ

■「ネットワークのテスト」において、人手の介入をなくすためのシステムの検討・PoCを行いました。

テストでの物理構成・論理構成パターン網羅

テスト用ノード生成と配置のソフトウェア化

テスト用ノードの集中制御、テストの自動化

パターンに基づいたテストケースの自動生成

■従来人手による作業が必要だった「現地・現物」作業のソフトウェアによる実行が可能であることを実証しました。

テスト作業のスケーラビリティ

再現性の向上

属人性の排除

■来年度の活動でL1Patchを拡張予定

興味のある方はぜひご参加ください！

OKINAWA OPEN LABORATORY


OKINAWA
OPEN
LABORATORY