JAIST Repository
https://dspace.jaist.ac.jp/

Title	運用中ネットワークの資源を利用した検証用ネットワークの構築と置き換えに関する研究
Author(s)	村上, 正樹
Citation	
Issue Date	2018-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/15201
Rights	
Description	Supervisor: 篠田 陽一, 先端科学技術研究科, 修士(情報科学)

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

運用中のネットワーク資源を利用した
検証用ネットワークの構築と置き換えに関する研究

北陸先端科学技術大学院大学
先端科学技術研究科

村上 正樹

平成 30 年 3 月

修 士 論 文

運用中のネットワーク資源を利用した検証用ネットワークの構築と置き換えに関する研究

1610185　　村上 正樹

主指導教員　篠田陽一 教授
審査委員主査　篠田陽一 教授
審査委員　丹康雄 教授
知念賢一 特任准教授
Razvan BEURAN 特任准教授

北陸先端科学技術大学院大学
先端科学技術研究科 [情報]

平成 30 年 2 月

Copyright © 2018 by Masaki Murakami

概 要

インターネットを利用した様々なサービスが提供されている。インターネットを介したサービスは日々高度化し、複雑化している。サービスの数自体も日々増加している。これらの高度化したサービスを維持、提供するためには、新サービス提供開始時の可用性が重要となる。ここでに述べる可用性とは、サービスが正常に動作し続けることである。可用性を確保する手段の１つとして、事前検証が考えられる。事前検証の精度が高ければ高いほど、新サービス提供開始後の不具合発生を抑制することができる。しかし、複雑化したサービスは、検証が必要な項目が多岐に渡り、また検証項目自体も複雑になるため検証が難しい。このため、新サービスを提供する段階で不具合が発生することがある。

本研究では新サービスの検証が難しい要因は、新サービスを検証する環境が実際にサービスを提供する環境と異なる点が多いからである。特に、実環境に近い検証環境を実現するためには、実環境と同等なパケット及び実環境と同等なネットワーク構成が必要である。一方で、ネットワークを構築する機器の高度化が進み、ネットワークに利用される資源も増加している。ここれらの資源が利用可能となることでネットワークのより効率的な運用が可能となる。

本研究では、実環境と同等なパケットを扱え、実環境と同じ構成のネットワークを持つ検証用環境を実現することを目的とする。同時に、十分な検証を行ったにも関わらずサービス提供開始後に不具合が発生した際のサービス停止時間を抑制する手段についても考える。加えて、ネットワーク資源の効率化の観点から、普段使われていないネットワーク資源の利用についても本研究の目的とする。

そこで本研究では、これらの目的を達成するために、実環境とパケット及びネットワーク構成が同等の置き換え可能な検証用環境構築と運用手法を提案する。本提案で構築する検証環境は仮想的な環境であり、運用中のネットワーク資源の一部を用いて構築する。検証環境の構築と運用のために、サーバの安定運用のために用いられる手法である Bule-Green Deployment や Immutable Infrastructure の考え方を応用した。検証環境は、運用中のサービスのネットワーク構成ごと複製することで、運用中のサービスのネットワーク構成を再現する。また、ここで構築した検証用ネットワークには、運用中ネットワークで実際に流れているパケットのミラーリングにより、パケット的に等価な環境を再現する。加えて、

マネジメントネットワークから運用中の環境と検証用の環境のリンクを切り替えることで、相互に置き換え可能なネットワークを実現する。

本研究では、これらの機能を実現する概念としてDIVINEを提案する。そして、DIVINEを実現する手法としてMayaconという選択的パケットミラーリング機構を利用して、実環境と同等のパケットが流れる検証用環境を構築する。

実験用ネットワークを対象に、DIVINE及びMayaconを用いた提案手法の実現性を検証する実験を行った。実験では、同一物理マシンの資源を用いたネットワークの複製が可能であることと、Mayaconを用いたパケットミラーリングによって検証用ネットワークでライブパケットを利用できることを確認した。また、仮想的なワイヤーリンクの切り替えを利用したネットワークの置き換えが可能であることを確認した。

実験用ネットワークに対して提案手法が有効であることが分かった。

DIVINEは、エッジ機器のリンクの変更がマネジメントネットワークを通して可能であることなどの条件を満たすネットワークであれば、どのような構成を持つネットワークにおいても利用可能である。本研究で提案するDIVINEは、余剰なネットワーク資源を用いて置き換え可能な実環境に近い検証用環境を実現することで、効率的で可用性の高いネットワークの実現に寄与するものである。

目 次

第 1 章　はじめに **1**

1.1　背景 1

1.2　目的 2

1.3　本論文の構成 2

第 2 章　ネットワークの検証と運用の現状と課題 **3**

2.1　関連技術 3

2.1.1　Blue-Green Deployment 3

2.1.2　Immutable Infrastructure 4

2.2　ネットワーク検証の関連研究及び関連手法 5

2.2.1　Shadow Configuration 5

2.2.2　Immutable Network Infrastructure 6

2.2.3　ステージング 6

2.3　ネットワーク検証に置ける問題点 7

第 3 章　DIVINE の提案 **8**

3.1　DIVINE の概要 8

3.2　DIVINE における検証用ネットワーク 9

3.2.1　運用中ネットワークの複製 9

3.2.2　検証環境の再現性 9

3.3　運用中ネットワークと検証用ネットワークの置き換え 10

第 4 章　手法の設計と実装 **11**

4.1　手法の概念 11

4.2　Mayacon 13

4.2.1　Mayacon TypeA 14

4.2.2 Mayacon TypeB 14
4.3 検証用ネットワークの構築 15
4.3.1 VirtualBox のクローンを用いた複製 16
4.3.2 運用中ネットワーク内パケットのミラーリング 18
4.4 ハイパーバイザを介した置き換え手法 18
4.4.1 Mayacom 挿入 20
4.4.2 ネットワークの移行 22
4.4.3 Mayacon 抜去 23

第 5 章 動作実験 24
5.1 動作実験 24
5.1.1 動作実験で用いた環境 24
5.1.2 実験の目的 26
5.1.3 実験シナリオ 27
5.2 実験結果 37

第 6 章 考察 38

第 7 章 おわりに 40
7.1 まとめ 40
7.2 今後の課題と展望 40

図 目 次

2.1 Blue-Green Deployment 3
2.2 Shadow Configuration のイメージ 5

3.1 DIVINE の概要 8

4.1 提案手法の概念図 11
4.2 Mayacon の仕組み 13
4.3 ネットワーク複製のイメージ 16
4.4 Mayacon の挿入 20
4.5 ネットワークの移行 22

5.1 実験用ネットワーク 25
5.2 運用中ネットワークの複製 28
5.3 実験用ネットワーク [検証] への Mayacon の挿入 29
5.4 実験用ネットワーク [運用] への Mayacon の挿入 30
5.5 置き換えの操作 31
5.6 遅延の生成 36

第1章　はじめに

本章では、本研究の背景、目的、本論文の構成を述べる。

1.1　背景

インターネットは多種多様なサービスによって利用されている。代表的なインターネットを利用したサービスとして、情報検索、オンラインショッピング、動画配信などがある。政府や自治体などの公共組織でもｅ－Ｊａｐａｎ戦略 [1] や世界最先端 IT 国家創造宣言 [2] に基づく各種公的な申請や、公共データの民間解放 (オープンデータ) の提供などが行われている。今後もインターネットを利用したサービスが増加していくことが予想される。それに伴い、インターネットを構成するネットワークも複雑で大規模なものになっていく。また、ネットワークの仮想技術の進歩もネットワークの可用性を広げると同時に、ネットワーク構成をより複雑なものにしている要因である。このような社会のクリティカルインフラであるインターネットに障害が発生した場合、社会的、経済的にも大きな影響を及ぼすことが考えられる。そのため、インターネットを構成する一つ一つのネットワークが正常に動作することが、インターネット全体のサービス品質の向上に繋がる。ネットワークが正常に動作するかについては、事前に検証を行うなどすることで、サービスインした際のネットワーク全体への影響をある程度事前に防ぐことが必要である。しかしながら、新たなサービス導入後のネットワーク全体の挙動を事前に検証する手法は確立されていない。

一方で、それらの高度化及び複雑化するネットワークを支える計算機器の性能も日々向上し続けている。例えば、CPU の高速化やメモリの高集積化、記憶媒体の大容量化などがあげられる。これらの計算機器の高性能化に伴い、普段は使われない余剰な計算機資源が発生することがある。これらの資源は、一時的な通信量の増加などに備えて準備されているものもあるが、多くのそれらの多くの資源は利用される時間が限られている。このような余剰資源を効率的に利用することで、潜在的な資源を利用したより高度なサービスが

提供できると考えられる。

1.2　目的

現在運用中のネットワークに対して、設定の変更や機器の追加などサービスの変更を行う際に、事前に実環境と近い環境で検証を行うことがサービスの安定性を確保する上で望ましい。本論文では、運用中ネットワークに対してサービスの変更を行う際に事前の検証を実環境と近い環境で行う手法を提案する。また、サービスの安定性の観点から旧サービスから新サービスへの移行時のサービスへの影響を抑えることも重要であると考える。そこで、旧サービスから新サービスへの移行時のダウンタイム等のサービスへの影響を少なくする手法についても考察を行う。運用中ネットワークの余剰な資源を用いて検証用ネットワークを構築することし、それぞれのネットワーク間でサービスネットワークとしの役割の置き換えが可能なネットワークを実現することで、新サービスへの移行時の安全な移行が可能となると考えられる。したがって本提案は、余剰なネットワーク資源を用いて、新サービスへのネットワーク移行時のサービスへの障害を抑制し、効率的で安定したサービスの提供に寄与する。

1.3　本論文の構成

本論文は、本章を含めて 7 章から構成される。2 章では、ネットワークの検証の手法や検証環境の構築手法、ネットワークの検証における課題について述べる。また、本提案に関連する技術及び、本提案との関係についても述べる。3 章では、本論文で提案する手法の概念について述べる。4 章では、3 章で説明した概念を実現するための手法の説明と、その設計及び実装について詳しく述べる。5 章では、4 章で設計及び実装した手法を実験用ネットワークを用いて行った実験及びその結果を述べる。6 章では、5 章の実験結果を受けた考察及び、3 章、4 章の概念や手法についての考察を述べる。7 章では、論文全体のまとめと、今後の展望や課題について述べる。

第2章　ネットワークの検証と運用の現状と課題

2.1　関連技術

本研究の提案概念に関連する既存の技術について述べる。

2.1.1　Blue-Green Deployment

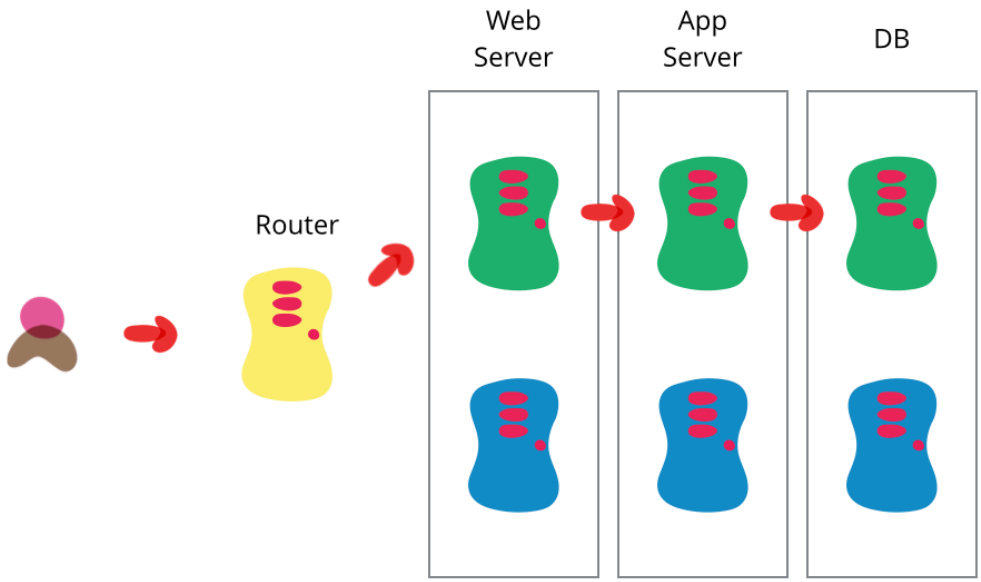

図 2.1: Blue-Green Deployment

Blue-Green Deployment [1]

[1]引　用　元:https://martinfowler.com/bliki/images/blueGreenDeployment/blue_green_deployments.png

デプロイの種類は、大きく分けて以下の 2 に分類できる。

In place

新しいリビジョンのアプリケーションのみをその場で反映する手法である。SSH での実行が可能であるため、オンプレミス環境で有効である。一方で同一のインスタンスに変更を加えるため、一貫性が損なわれる恐れがある。そのため、インスタンスのロールバックが難しい。

Blue/Green

新しいリビジョンを反映させた新しいインスタンスを作り、入れ替える手法である。インスタンスが新しいリビジョン専用の構成であるため、常に一貫性が保たれる。そのため、ロールバックも容易である。一方で、新しいリビジョンを作るたびに新たなインスタンスを作るため、余分なインスタンスが必要になる。これについては、必要な分だけリソースをレンタルすることができるクラウドサービスを利用することで、コストを抑えることができる。

本研究では、この 2 つのデプロイ手法のうち Blue-Green Deployment の考え方を応用する。

また、Blue-Green Deployment は反映のスピードによって以下のように分類することができる。本研究でのネットワークの置き換えでは、All at once 方式を採用している。

All at once

全てのインスタンスを一斉にデプロイする手法である。

Batch

数台ずつ順次デプロイを行う手法である。

One by one

1 台ずつ順次デプロイを行う手法である。

2.1.2 Immutable Infrastructure

Immutable Infrastructure は、システム更新の都度現行のインスタンスに変更を加えるのではなく、新しくインスタンスを作成して、更新後のサービスは新インスタンスで行うとい

う手法である。また、旧インスタンスは使わずに捨ててしまう。Immutable Infrastructure は、システム更新のために新しいインスタンを作り、新しくできたインスタンスを使うという点では Blue-Green Deployment と似た考え方である。しかし、Blue-Green Deployment が旧システムと新システム切り替え時のシステム全体への影響を抑えることを重要していることに対して、Immutable Infrastructure はインスタンス毎の一貫性の保証による更新作業の効率化に重点を置いている。この一貫性とは、あるインスタンスがある時点の更新から一度も変更されていない状態が保たれていることである。反対に一貫性がない状態とは、対象のインスタンスに対してなんども異なる更新や OS 等のアップデートを加えた結果つぎはぎのような状態になっていることである。

2.2 ネットワーク検証の関連研究及び関連手法

ネットワークの検証手法及び、検証環境の構築に関する研究及び、技術について述べる。

2.2.1 Shadow Configuration

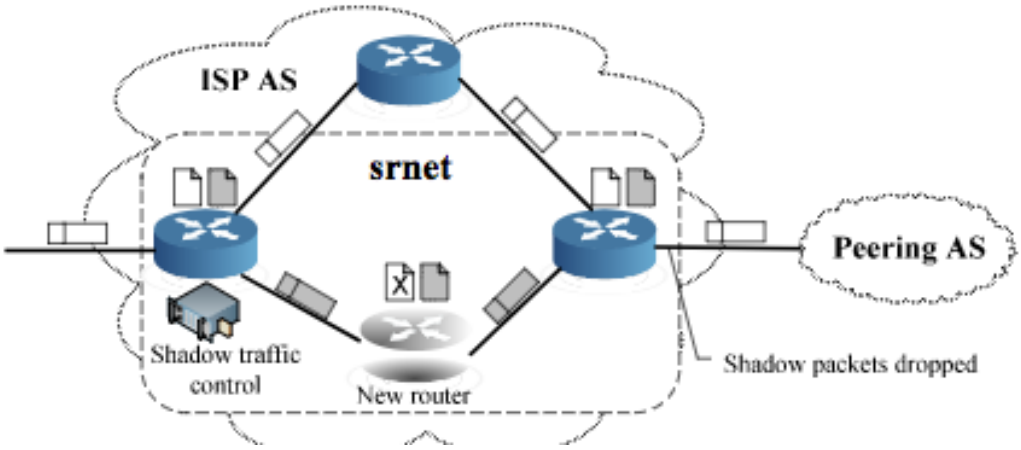

図 2.2: Shadow Configuration のイメージ

Shadow Configuration のイメージ [2]

[2]引用元:http://www.sigcomm.org/sites/default/files/ccr/papers/2008/October/1402946-1402972.pdf

本研究の関連研究として、R.Alimi らにより提案された Shadow Configuration がある。この研究は、サービス用及び検証用の 2 つの設定を用意して、その設定を利用して論理的なネットワークを構築するというものである。また、検証用のネットワークには実ネットワークのコピーパケットを流している。このパケットは、ネットワークの出入口に設置された shadow traffic control によって制御されており、検証対象ネットワーク以外のネットワークにコピーパケット及び、検証用ネットワークで利用しているパケットが流れでないようにしている。

2.2.2 Immutable Network Infrastructure

本研究の関連研究として、上野幸杜氏の Immutable Network Infrastructure がある。この研究では、仮想レイヤ 2 及び 3 のネットワークを物理ネットワーク上に構築し、その仮想ネットワークの上で検証を行う手法を提案している。仮想レイヤ 2 及び 3 はそれぞれ、VLAN 及び VRF(Virtual Routing and Forwarding) を用いて実現している。また、この手法はエッジのレイヤ 2 スイッチにおいてトラフィックコピーを行うことで、検証用の仮想ネットワークに物理ネットワークに流れるパケットを流すことができる。

2.2.3 ステージング

ステージングは、サービスを実際に運用する本番環境と同等の環境を構築し、その中で新サービスをテストすることである。ステージングにおける環境とは、コンピュータや基本ソフト、通信設備などを指す。ステージングという言葉は、テスト環境を指す場合と、テストそのものを指す場合があるが、本論文ではテスト環境をステージング環境、テストそのものをステージングと表すこととする。高機能化とともに複雑化するシステムでは、開発段階では見落とされていた不具合が稼働後に分かることが少なくない。ステージングを行うことで、本番環境と近い環境でのテストが可能になるため、実際にサービスを稼働しないと分からなかった不具合を事前に検知することができる。結果的に、サービス稼働後の不具合発生の抑制に繋がる。

2.3 ネットワーク検証に置ける問題点

本研究で着目するネットワーク検証における問題点について述べる。問題点は以下の 3 つである。

1. 新サービスを実環境に投入した際に発生する不具合を事前に検証することが難しい点
2. 実環境で流れているパケットを使った新サービスの検証が難しい点
3. 検証を終え、新サービスを稼働する段階で不具合が発生した場合のサービスへの影響を抑える手段が確立されていない点

第3章　DIVINEの提案

本章では、項で述べたという問題を解決する手段として DIVINE(Device Integrated Virtualized InterNetworking Emulator) を提案する。

3.1　DIVINEの概要

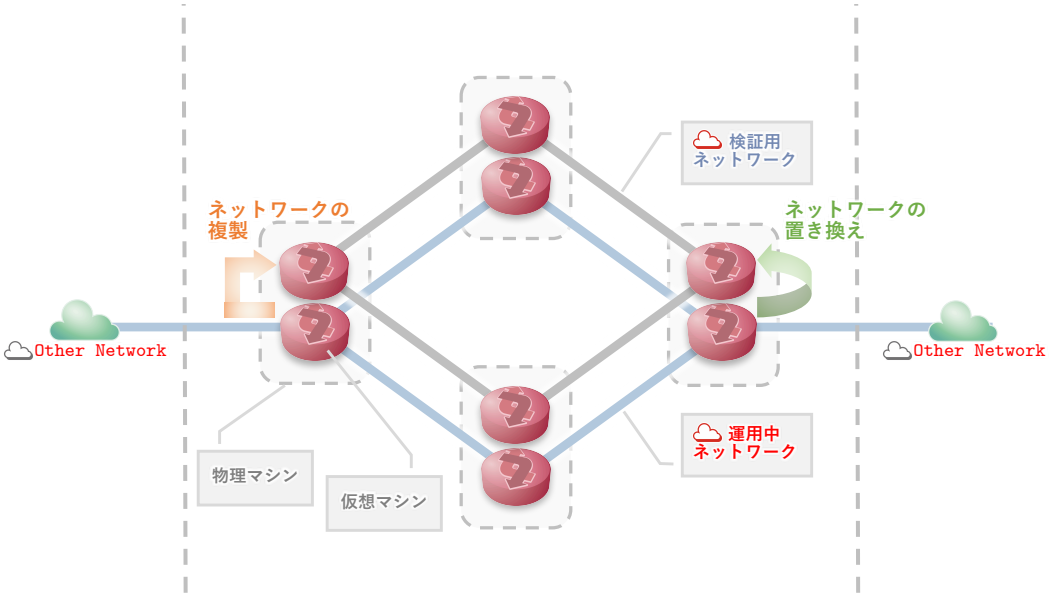

図 3.1: DIVINE の概要

DIVINE は、置き換え可能な検証環境の構築・運用手法である。DIVINE は、サービスを提供している運用中のネットワーク (以下、運用中ネットワーク) に変更を加える際に事前に検証を行えるネットワーク (以下、検証用ネットワーク) を提供する。DIVINE が提供する検証環境は、運用中ネットワークの普段使われていない資源を用いて構築する。

DIVINE の適用を想定するネットワークは、仮想化によってネットワーク機器のマシンイメージを複製可能な機器で構成されたネットワークである。例として、汎用サーバ上の仮想化環境で動作するネットワーク OS が動作する仮想マシンなどが挙げられる。

DIVINE の機能は、主に以下の 3 つである。

- 運用中ネットワークの資源を利用した検証用ネットワークの構築
- 運用中ネットワークに流れるパケットと同等のパケットの検証用ネットワークへの提供
- 運用中ネットワークと検証用ネットワークの置き換え

3.2 DIVINE における検証用ネットワーク

DIVINE は、運用中ネットワークに対する変更を検証するための検証用ネットワークを構築する。検証用ネットワークは運用中ネットワークの置き換え対象ネットワーク内の機器を全て設定もそのままに複製する。

3.2.1 運用中ネットワークの複製

DIVINE ではまず、運用中ネットワークと同じ設定、同じ構成を持つマシンイメージ複製することでもう一つのネットワークを構築する。ここで構築されたネットワークが検証用ネットワークである。検証用ネットワークは、運用中ネットワークに対する実環境では難しい検証や、新サービス提供のためのネットワークの検証を行うためのネットワークである。そのため、検証用ネットワークは運用中ネットワークで転送されているパケットと同等のパケットの流れがあることが望ましい。そこで、DIVINE では、検証用ネットワーク側に運用中ネットワークで流れているパケットと同じパケットをミラーする形で提供する。また、複製されたネットワークはそのネットワーク内で完結しており、運用中ネットワーク側からパケットが流入してくるが検証用ネットワーク側で発生したパケットは運用中ネットワークには流出しない。

3.2.2 検証環境の再現性

検証環境として DIVINE が最低限提供するものとして、運用中ネットワークの複製時の状態と同じネットワークの設定及び構成を持つネットワーク及び、運用中ネットワーク

を流れるパケットを複製したパケットである。そのため、ネットワークを構築するマシンの状況及び、検証用環境を流れるパケットはほぼ再現可能である。

3.3 運用中ネットワークと検証用ネットワークの置き換え

検証が完了した環境は多くの場合、その環境で設定したものと同じ設定を運用中のネットワークに反映させる。そのため、検証した環境をそのまま新運用ネットワークとして使うことができれば、設定反映の手間や反映作業時の人為的なミスを抑えることができる。また、実環境に近い環境で検証したネットワークをそのまま使うため、検証後の予期せぬ障害の発生を少なくすることができる。そこで、DIVINE では運用中ネットワークと並行して検証用ネットワークを構築し、それぞれのネットワークを置き換え可能なネットワークを提案する。それぞれのネットワークが置き換え可能である利点は、万が一新運用ネットワークに置き換えた際に障害などの問題が起きた時に、旧運用ネットワークに置き戻すことにより、旧ネットワークでのサービスを継続することができる。旧サービスの状態でサービス継続が可能であるため、MTTR(Meen Time To Repare) の値をあげ、サービスの信頼性を保つことができる。

第4章　手法の設計と実装

本章では、本研究の提案手法でもある DIVINE に基づく手法の設計と実装について述べる。

4.1　手法の概念

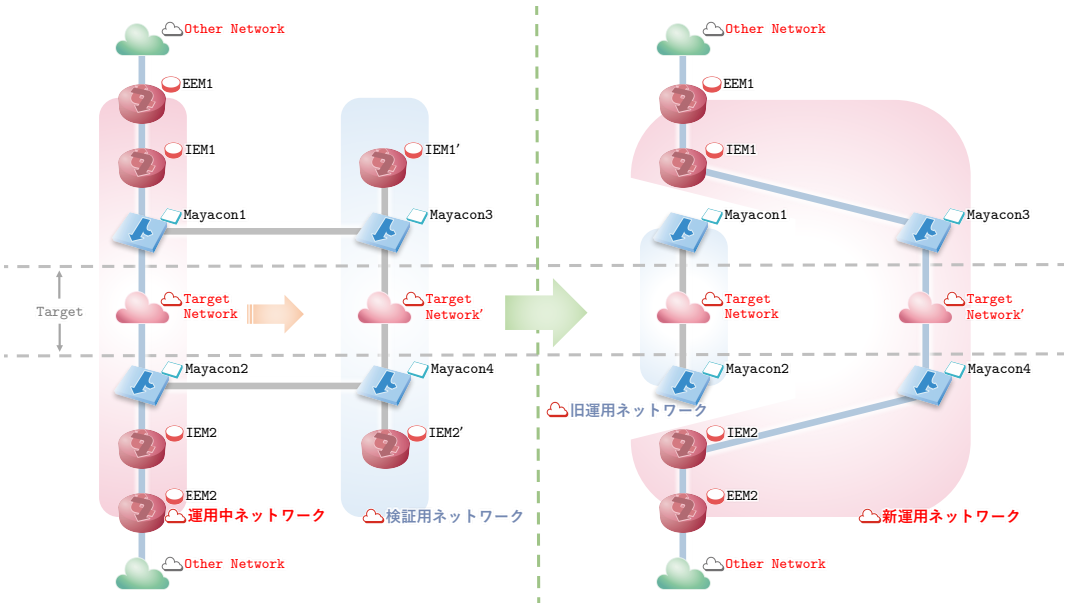

図 4.1: 提案手法の概念図

図 4.1 に、提案手法の概念図を示す。

検証用ネットワーク内では、運用中ネットワークと同等のパケットを提供するため、パケットミラーリングを利用している。

運用中ネットワーク側のパケットは検証用ネットワークにミラーリングされ流入するが、検証用ネットワーク側で発生したパケット及び、運用中ネットワークから流入したパケットは運用中ネットワーク側に流れ出ることはない。これは、検証用ネットワーク内で

の変更や検証に利用したパケットが運用中ネットワークで稼働しているサービスに影響を与えないようにするためである。

また、以下のシステムにおける機器及び、機能の仕組みについて述べる。

Target

Target は、置き換え対象となるネットワークである。

Internal

Internal は、Target ネットワーク及び、Target ネットワークから 1 ホップ外側にあるネットワーク機器を含むネットワークである。また、Internal ネットワークと External ネットワークの境界にあるネットワーク機器を IEM(Internal Edge Machine) とする。

External

External は、Target ネットワーク及び Internal ネットワークに属さないネットワークであり、IEM の外側にあるネットワークである。また、IEM から見て 1 ホップ internal ネットワーク側にあるネットワーク機器を EEM(External Edge Machine) とする。

4.2 Mayacon

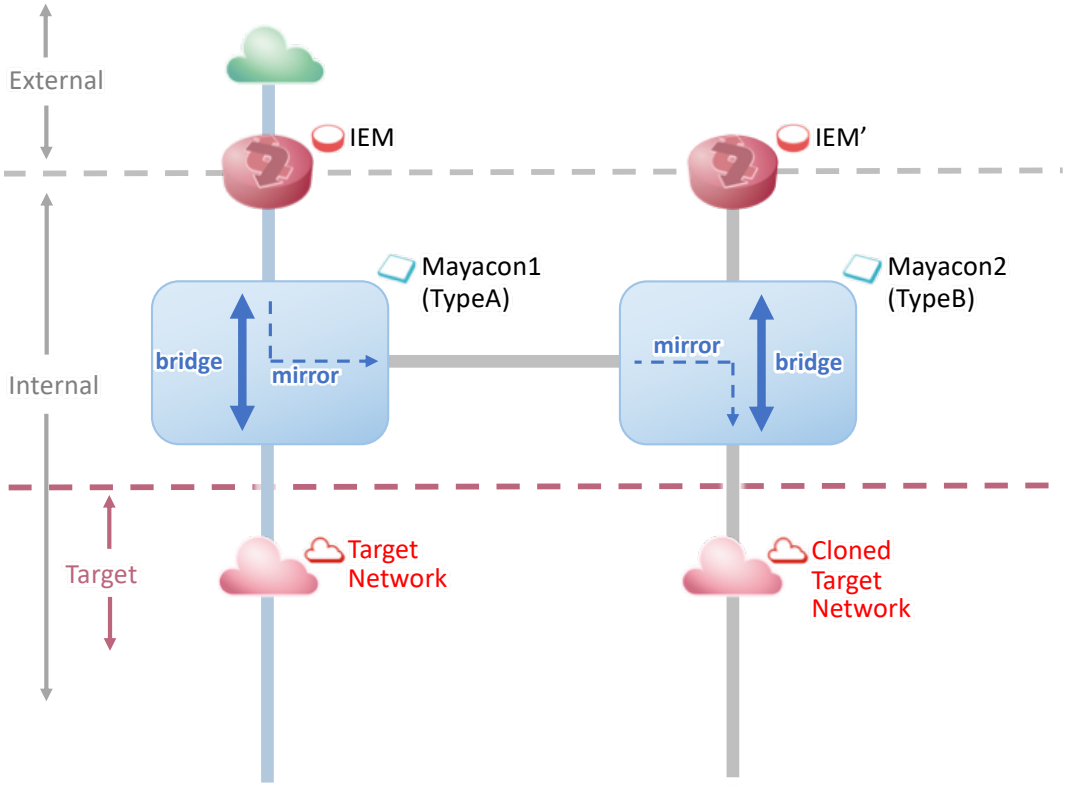

図 4.2: Mayacon の仕組み

Mayacon とは、運用中ネットワーク及び検証用ネットワーク内で用いられる選択的パケットミラーリング機構である。本実装では、DIVINE における運用中ネットワークと同等のパケットを検証用ネットワークでも扱うことができるという要件を、運用中ネットワークのパケットを検証用ネットワークにミラーリングする方法で実現する。Mayacon は、そのための機能を提供している。Mayacon の主な機能は以下の 3 つである。

- 運用中ネットワークから検証用ネットワークへのパケットのミラーリング
- IEM と Target ネットワークを繋ぐブリッジの提供
- ミラーリングとブリッジの制御

検証用ネットワークを新運用ネットワークとして置き換える前の状態では、IEM から流れてきたパケットを検証用ネットワーク側にミラーリングする。検証用ネットワーク側

では、運用中ネットワーク側から受けたパケットをさらに検証用ネットワークの Target ネットワーク側ににミラーリングする。

また、運用中ネットワーク側では、パケットのミラーリングを行う一方で IEM と Target ネットワークを結ぶブリッジを提供することで、現行のサービスが行われているネットワークの流れを確保している。一方、検証用ネットワーク側のブリッジは検証を行う際の起点となる IEM' から検証用ネットワーク側の Target ネットワークのリンクを提供している。検証用ネットワーク側にこのような IEM があることで、パケットジェネレータを利用した様々なパケットの流入検証などを行うことができる。

ネットワークを置き換える際には、置き換え前のミラーリングやブリッジの設定を変更する必要がある。そのため、Mayacon ではその際のミラーリングの解除やブリッジ元とブリッジ先の変更などの制御を行う。制御は、仮想マシンのホスト OS 側からマネジメントネットワーク越しに Mayacon の制御コマンドを実行することで行う。

また、Mayacon には 2 つのタイプがあり、それぞれ「Mayacon typeA」、「Mayacon typeB」とする。

4.2.1 Mayacon TypeA

Mayacon typeA は運用中ネットワーク側で動作する。検証用ネットワークに運用中ネットワークを流れるパケットをミラーリングする段階では、IEM と Target ネットワークを繋ぐブリッジを提供している。それと同時に、検証用ネットワーク側の IEM から対象ネットワークに流れてくるパケットを検証用ネットワーク側にミラーリングする。この時、ミラーリングには tc(Traffic Control) を利用しているためフィルタを設定することで検証用ネットワークに流れるパケットを制御することができる。

4.2.2 Mayacon TypeB

一方 Mayacon typeB は、検証用ネットワーク側で動作する。Mayacon typeA とは L2 で繋がっており、Mayacon typeA から受け取ったパケットをミラーリングして検証用ネットワーク側に流す。また、検証用ネットワーク側の IEM から受け取ったパケットを置き換え対象のネットワークに流すためのブリッジも行う。

4.3 検証用ネットワークの構築

前章 3.2 項で述べた検証用ネットワーク構築の実装方法について述べる。DIVINE が定義する検証用ネットワークを実現するためには、運用中ネットワークの設定及び構成を複製する仕組みと、運用中ネットワークに流れるパケットと同等のパケットを検証用ネットワーク内に流す仕組みが必要である。本研究ではそれぞれ、VirtualBox の仮想マシンのクローン機能及び、tc のパケットミラーリング機能をを用いて実装を行った。それらの実装の詳細を 4.3.1 項及び、4.3.2 項で詳しく述べる。

4.3.1 VirtualBox のクローンを用いた複製

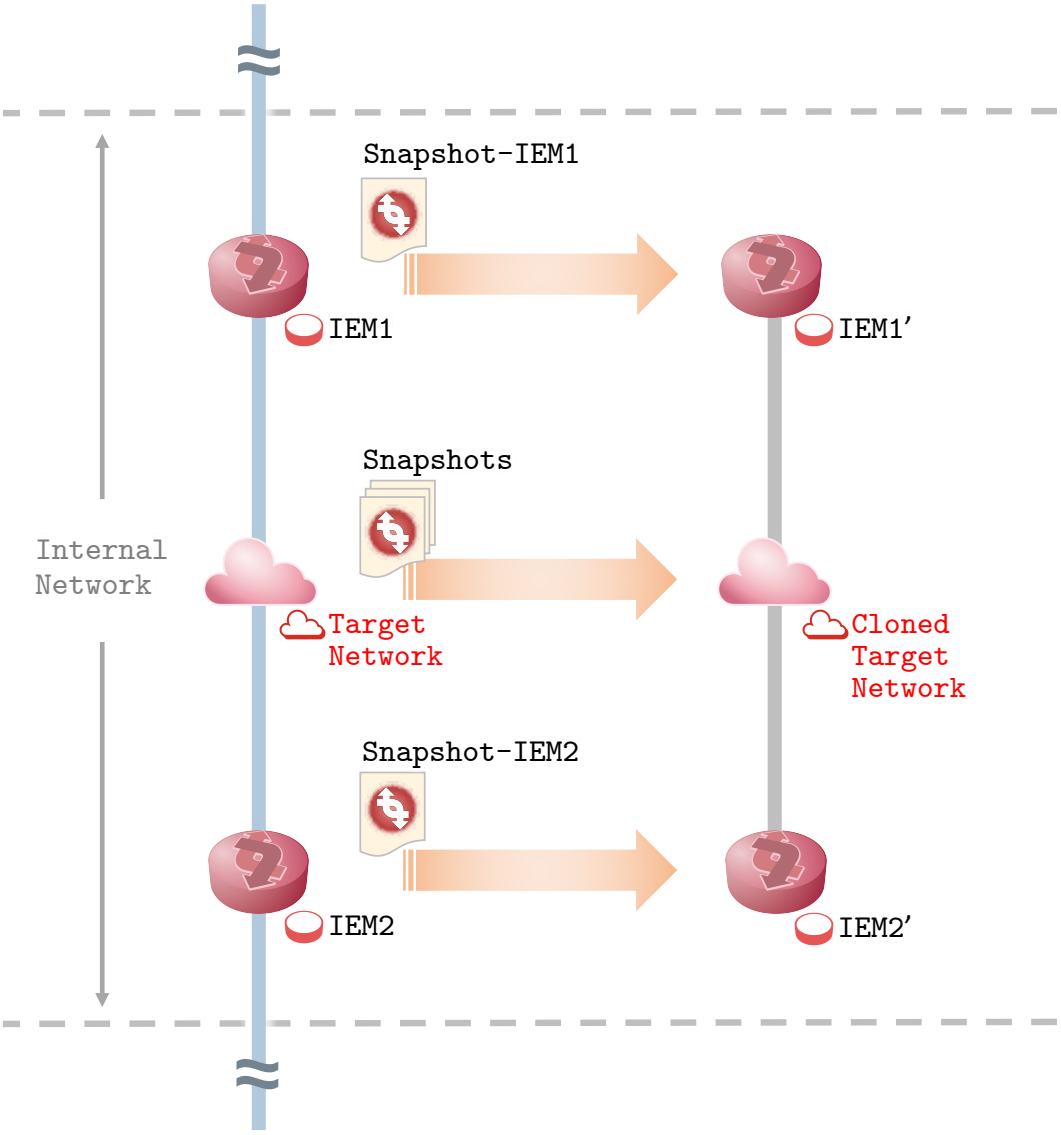

図 4.3: ネットワーク複製のイメージ

VirtualBox のクローン機能を用いた運用中ネットワークの複製方法について述べる。運用中ネットワーク複製の大まかな流れは以下の通りである。

1. 運用中ネットワーク上で動作している仮想マシンの中で、検証用ネットワーク構築に必要な仮想マシンすべてのスナップショットを生成する

2. 1 で作成したスナップショットを元にそれぞれの仮想マシンの複製を行う

VirtualBox のクローン機能とは、複製したい仮想マシンイメージに対してクローンを実行することで、複製したい対象と同じ仮想マシンを作ることができる。複製される仮想マシンは、複製元のマシンの設定及び、仮想ディスクと同じデータでかつ、複製時の複製元マシンの状態と同じ状態で生成される。また複製の際には、複製されるマシンの NIC が持つ MAC アドレスを複製元のマシンと異なるアドレスをつけることができる。この時付与されるアドレスは、VitualBox が自動で決定する。

仮想マシンの複製は、VirtualBox の CUI を利用して行う。VirtualBox におけるクローンは 4.1 に示す操作で行うことができる。

コマンド 4.1: クローン操作の例

```
vboxmanage clonevm <srcVM> --name <clonedVM> --register
```

コマンド 4.1 の”srcVM”、“clonedVM” はそれぞれ、クローン元の仮想マシンの名称、複製される仮想マシンの名称を示す。また、register コマンドは、仮想マシンの複製と同時に VirtualBox における仮想マシンの管理リストに追加するためのコマンドである。

しかし、コマンド 4.1 で示した操作を利用した複製は停止状態の仮想マシン及び、仮想マシンのスナップショットに対してのみ実行可能であるため、運用中ネットワーク上の仮想マシンをそのまま複製することは難しい。しかし、スナップショットは起動中の仮想マシンに対しても実行可能である。したがって、本研究の実装では運用中ネットワーク上の仮想マシンのスナップショットを保存し、そのスナップショットを元にクローン操作を行う。VirtualBox の CUI を用いたスナップショット操作及び、クローン操作の例をコマンド 4.2 に示す。

コマンド 4.2: スナップショット操作及びクローン操作の例

```
[スナップショット操作]
vboxmanage snapshot <srcVM> take <snapshotName>

[クローン操作]
boxmanage clonevm <srcVM> --snapshot <srcSnapshot> --options \
> keepallmacs --name <clonedVM> --register
```

コマンド 4.2 の “srcVM”、“snapshotName” はそれぞれ、スナップショットの元になる仮想マシン、新しく生成されるスナップショットの名称である。

スナップショットから複製をするクローン操作については、“srcVM”、“srcSnapshot”、

“clonedVm” はそれぞれ、複製元の仮想マシンの名称、複製元のスナップショットの名称、複製される仮想マシンの名称である。option コマンドでは keepallmacs を指定することで、複製元の仮想マシンが持つ NIC の MAC アドレスもそのまま複製することができる。

4.3.2 運用中ネットワーク内パケットのミラーリング

運用中ネットワークから検証用ネットワークへのミラーリングの具体的な手法について述べる。本実装におけるミラーリングは、運用中ネットワーク及び検証用ネットワーク内のそれぞれの Mayacon で tc コマンドを用いることで実現している。Mayacon 内で使用している tc コマンドをコマンド 4.3 に示す。srcIF は全て共通でミラーリング元の NIC の名称を表し、dstIF はミラーリング先の NIC を表す。

また、運用中ネットワークと検証用ネットワークでミラーリング元とミラーリング先が異なっている。運用中ネットワークでは、IEM から検証ネットワーク側の Mayacon へパケットをミラーリングしており、検証用ネットワーク側では、運用中ネットワーク側の Mayacon から受け取ったパケットをさらに検証ネットワーク側の Target ネットワークに転送する。

コマンド 4.3: ミラーリングコマンド

```
tc qdisc add dev <srcIF> ingress
tc filter add dev <srcIF> parent ffff: protocol all u32 match u8 0 0 action
    mirred egress mirror dev <dstIF>
tc filter show dev <srcIF> parent ffff:
```

4.4 ハイパーバイザを介した置き換え手法

本節では、DIVINE におけるハイパーバイザの機能を利用した、運用中ネットワークと検証用ネットワークの置き換えの実装方法について述べる。

以下は、ハイパーバイザの機能を利用した置き換え手法の手順を表したものである。はじめに、大まかな置き換え手順について述べる。

1. ネットワークの複製
2. Mayacon の挿入

3. 運用中ネットワークのパケットをミラーリング

4. ネットワークの移行

5. Mayacon の抜去

1. の「ネットワークの複製」に関しては、項で述べた方法を用いる。

2. の「Mayacon の挿入」では、Mayacon(後述) という選択的ミラーリング機構を持ったスイッチマシンを運用中ネットワーク及び、検証用ネットワークに挿入する。これを行う理由は、Mayacon を 3. でパケットのミラーリング及び、各ネットワーク内のパケットの制御に用いるためである。

3. の「運用中ネットワークのパケットをミラーリング」では、Mayacon を用いて運用中ネットワークに流れるパケットを検証用ネットワーク側にミラーリングする。このミラーリングは各ネットワークに Mayacon 挿入完了後自動的に始まる。

4. の「ネットワークの移行」では、運用中ネットワークから検証用ネットワークへの移行を行う。

5. の「Mayacon の抜去」では、旧運用ネットワーク (運用中ネットワーク) ごと置き換え可能ネットワークから切り離す。

4.4.1 Mayacom 挿入

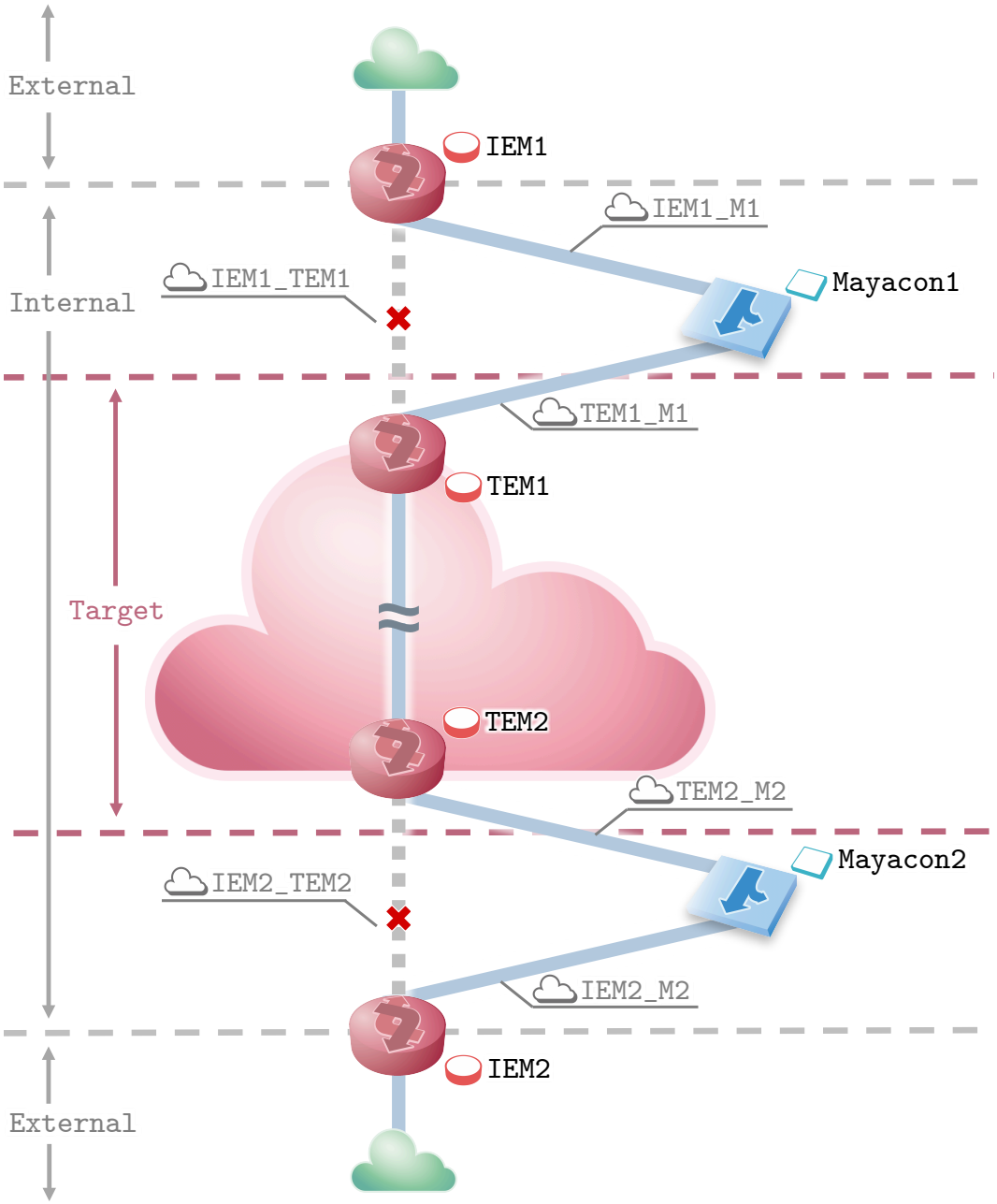

図 4.4: Mayacon の挿入

本提案の実装では、検証用ネットワークに運用中ネットワークを流れるパケットをミラーリングするため、運用中ネットワークに Mayacon を挿入することが必要である。そのため、図 4.4 のように Mayacon の挿入を行う。Mayacon の挿入は、VirtualBox の操作で

ネットワーク名を変更することで実現する。なお、この命令は仮想マシンが動作中でもマシンの電源を切らずに行うことができる。図 4.4 中の TEM(Target network Edge Machine) とは、置き換えの対象とするネットワークの最も外側にあるマシンでのうち、IEM のネクストホップにあるマシンを示す。図 4.4 の IEM1 - TEM1 間への MayaconX の挿入を例にとると、IEM1 及び TEM1 に対して VirtualBox の以下の操作を実行して、IEM1 のネットワーク名を"IEM1_TEM1" から"IEM1_M1" に変更する。同様に TEM1 でもネットワーク名を"IEM1_TEM1" から"TEM1_M1" に変更する。なお、Mayacon1 には Mayacon1 生成の際に二つの仮想 NIC にネットワーク名を設定しておく (ここではそれぞれ、"IEM1_M1" 及び"TEM_M1")。IEM1 及び TEM1 で行った操作の一般的な例をコマンド 4.4 に示す。

コマンド 4.4: ネットワーク名変更を行う VirtualBox の操作

```
vboxmanage controlvm <VMname> <NICnum> intnet <NWname>
```

vboxmanage は、VirtualBox の CUI(Command Line Interface) であり、controlvm オプションは、実行中の仮想マシンに対して各種変更や電源管理等を行うためのものである。ここでは、ネットワーク名の変更を行うためまず VMname に対象の仮想マシンの名前を指定し、ネットワーク名を変更する NIC の番号を指定する。この操作における NIC の番号とは、1 から 8 までの VirtualBox が管理に利用している番号であり、showvminfo オプションなどで知ることができる。NIC の番号を指定したあとは、ネットワークのタイプと新しく所属するネットワークの名前を指定する。今回は、ネットワークタイプを intnet とし、ネットワーク名は IEM1 であれば"IEM1_M1"、TEM1 であれば"TEM1_M1" とする。intnet は、VirtualBox 上の仮想マシンだけで構成される内部ネットワークであである。また、TEM2 - IEM2 間及び、Mayacon2 でもこれと同様の操作を行うことで Mayacon の挿入が完了する。Mayacon の挿入は、運用中ネットワーク及び検証用ネットワークのどちらでも同様の操作であり、違いは挿入する Mayacon のタイプのみである。

4.4.2 ネットワークの移行

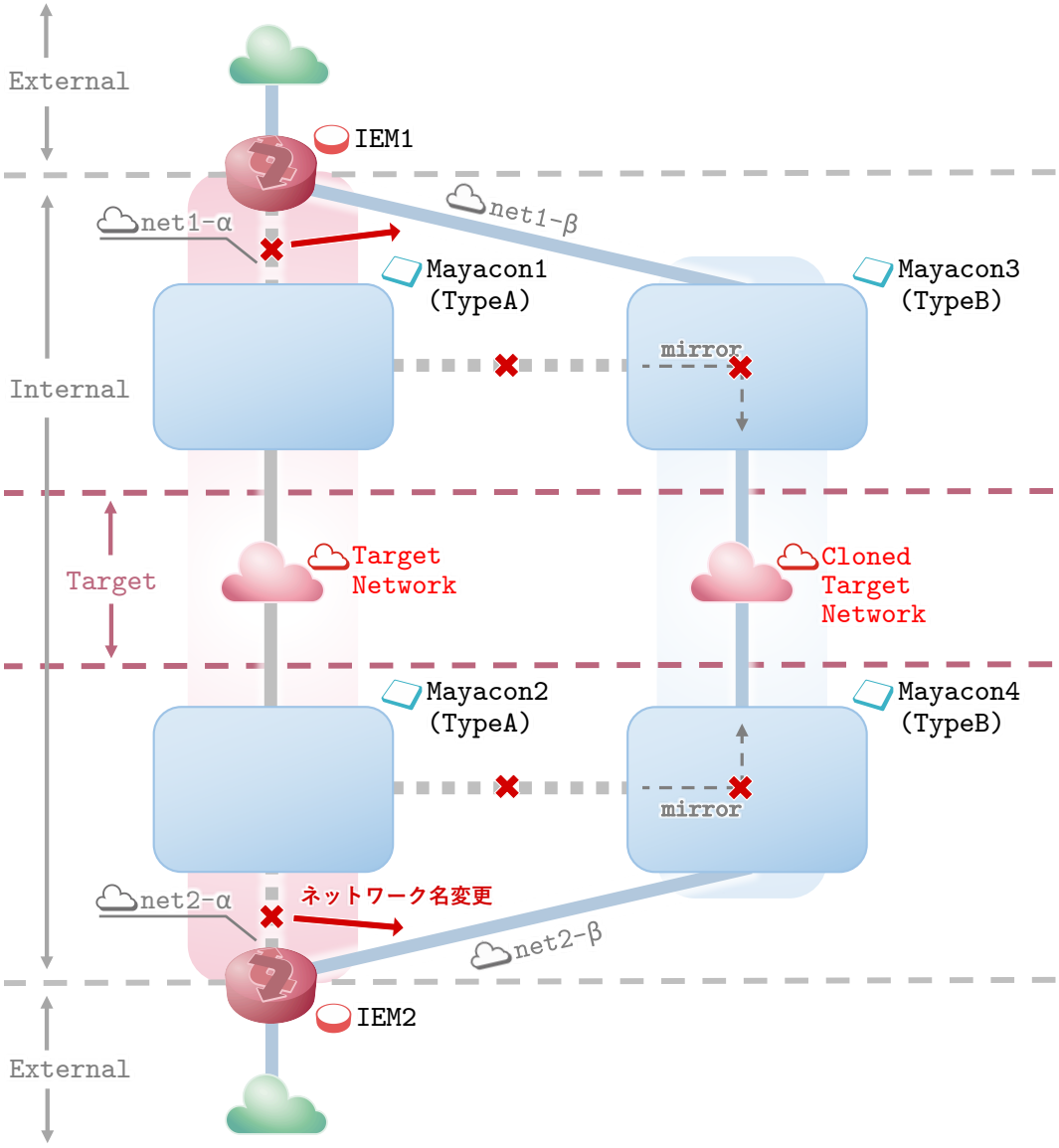

図 4.5: ネットワークの移行

本論文では、ネットワーク置き換え時に各ネットワークに Mayacon 挿入後に実際にサービスを提供するネットワークを検証用ネットワークに切り替えることを特に、ネットワークの移行と呼ぶ。ネットワークの移行では主に以下の 2 つの段階で移行を行う。

Step1. 検証用ネットワーク側へのミラーリングの解除と Mayacon の NIC の再設定

Step2. それぞれの IEM の属するネットワーク名称の変更

Step1 については、Mayacon の制御コマンドを使って次の手順で設定を行う。

手順 1. Mayacon typeB のミラーリングを無効にする (Mayacon コマンド)

手順 2. Mayacon typeB の IEM 側の仮想 NIC が属するネットワークの名称を変更する (VirtrualBox コマンド)

手順 3. Mayacon typeA のミラーリングを無効にする (Mayacon コマンド)

Step2 については、Step1 の手順 2 で変更したネットワークの名称に合わせて各 IEM のネットワーク名称を変更する。

4.4.3 Mayacon 抜去

運用中ネットワークから検証用ネットワークに移行が完了すると、運用中ネットワーク側に残る Mayacon が不要になる。そのため Mayacon の抜去が必要になるが、項のネットワークの移行が完了した段階で運用中ネットワーク内の Mayacon 及び、その間の置き換え対象のネットワークの切り離しができているため、ネットワークの移行が完了すると同時に Mayacon の抜去も完了する。この時点でネットワークの置き換えが完了する。切り離された Mayacon 及び置き換え対象のネットワークは、そのままの状態で保持し、ネットワーク置き換え後にサービスインした検証用ネットワーク側で不具合が起きた場合などに置き戻せるようにする。

第5章　動作実験

本章では、前章で述べたDIVINEをあるネットワークに適用して動作実験を行う。

5.1　動作実験

DIVINEの動作実験では、単純なネットワークを用意してそのネットワークを対象に実験を行った。

5.1.1　動作実験で用いた環境

動作実験で想定したネットワーク構成を図5.1に示す。

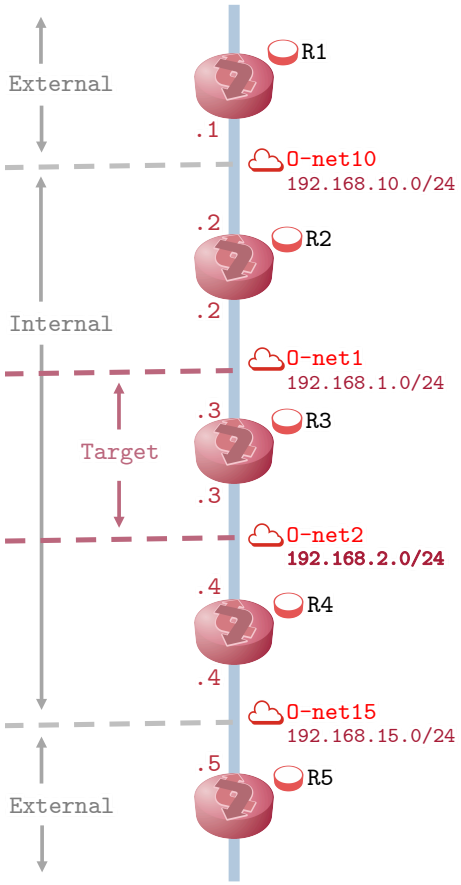

図 5.1: 実験用ネットワーク

ネットワーク内のルーティングは RIP のみを用いている。実験ネットワークのそれぞれの役割は、以下の通りである。

External

図 5.1 の R1 及び R5 が External に該当する。External は、置き換えを行うネットワークを持つ組織外のネットワークを想定している。そのため、本実験では External のネットワークへの設定変更や機器追加等の変更を行わない。

Internal

図 5.1 の R2、R3、R4 が Internal のネットワークに該当する。Internal は、置き換えを行うネットワークを持つ組織内のネットワークを想定している。

Target

図 5.1 の R3 が Target のネットワークに該当する。Target は、置き換えを行う対象のネットワークである。このネットワーク内の機器を複製して作った検証用ネットワークの機器に対して、新たな機器の追加や既存の機器に対する設定変更などを行う。

本実験は、以下の構成の物理/仮想マシン及び、仮想化ソフトウェアを使用して行った。また、仮想マシンは実験用ネットワークで使用した全てのマシンが共通の構成である。

ホスト PC(物理マシン)

- OS: Ubuntu 16.04.3 LTS
- System type: x86 64bit
- CPU: Intel(R) Xeon(R) CPU E3-1231 v3 @ 3.40GHz(4 core)
- Memory: 8GB DDR3 1600MHz *2(total 16GB)

ゲスト PC(仮想マシン)

- OS:VyOS 1.1.7 (helium)
- System type: x86 64bit
- CPU: 1 CPU
- Memory: 512GB

仮想化ソフトウェア

- Software: Virtual Box 5.2.2r119230

5.1.2 実験の目的

運用中ネットワークに対して Mayacon の挿入が可能であること、同一のネットワーク資源を使って検証用ネットワークが構築できること及び、それらのネットワークの置き換えが可能であることを実証する。

5.1.3 実験シナリオ

実験シナリオは以下の通りである。

実験の前提

実験の前提条件は以下の通りである。

- 実験用ネットワーク [運用](図 5.1) はあらかじめ構築済みである
- 実験用ネットワーク [運用] は全て RIP でルーティングされる
- 実験用ネットワーク [運用] の各ノード間の疎通は事前に確認されているものとする
- 本実験は、運用中のネットワークに対する検証環境の構築及び、置き換えを目的にしているため、ノード間でパケットのやりとりが行われている状態で実験を行う

実験の手順

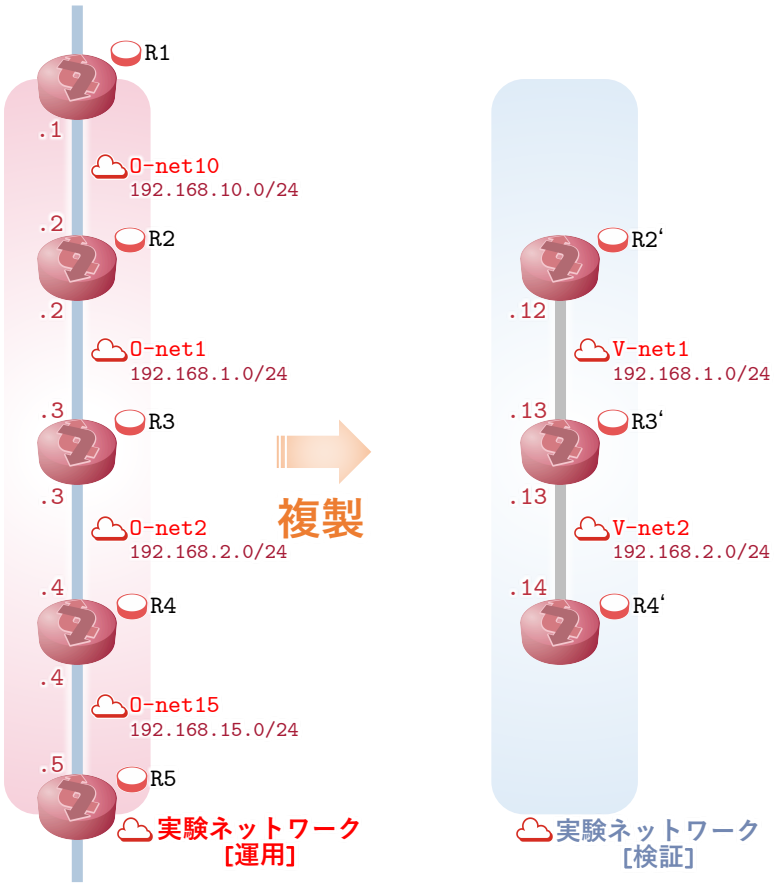

図 5.2: 運用中ネットワークの複製

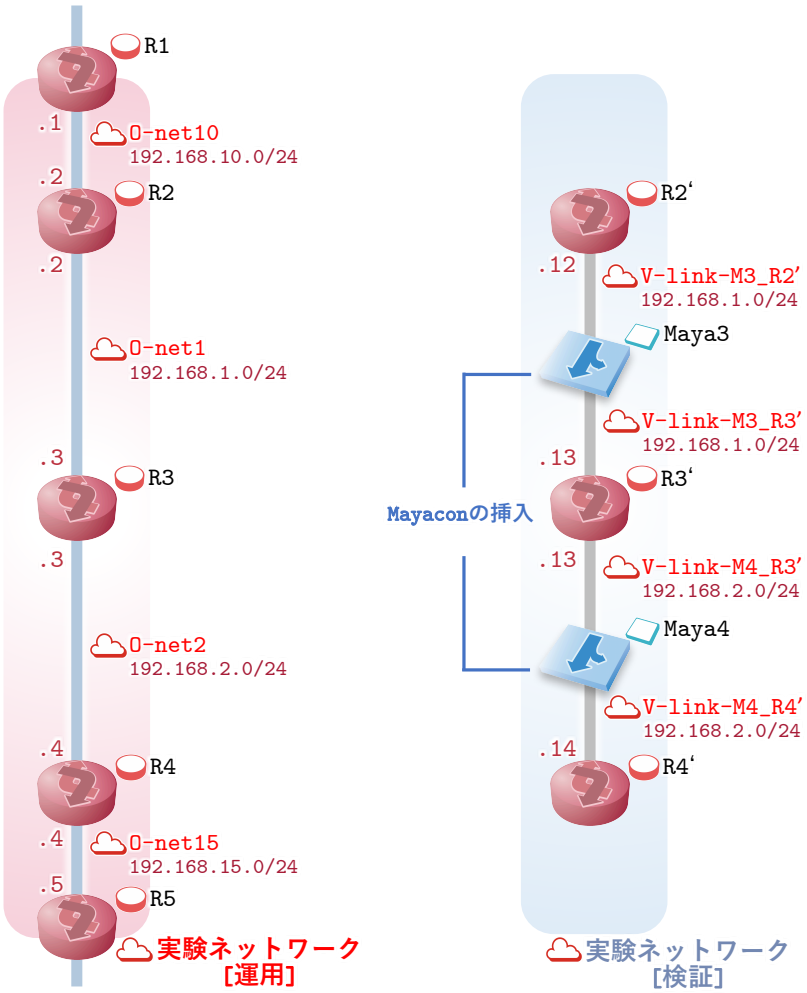

図 5.3: 実験用ネットワーク [検証] への Mayacon の挿入

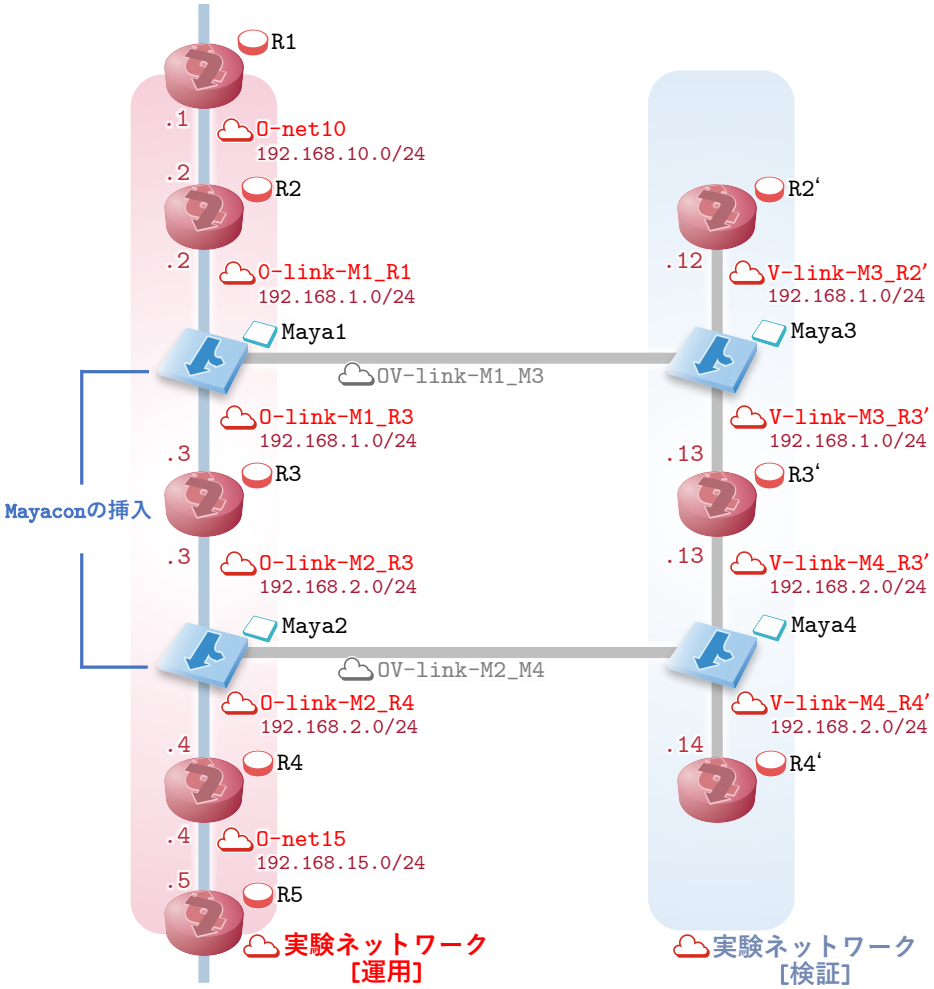

図 5.4: 実験用ネットワーク [運用] への Mayacon の挿入

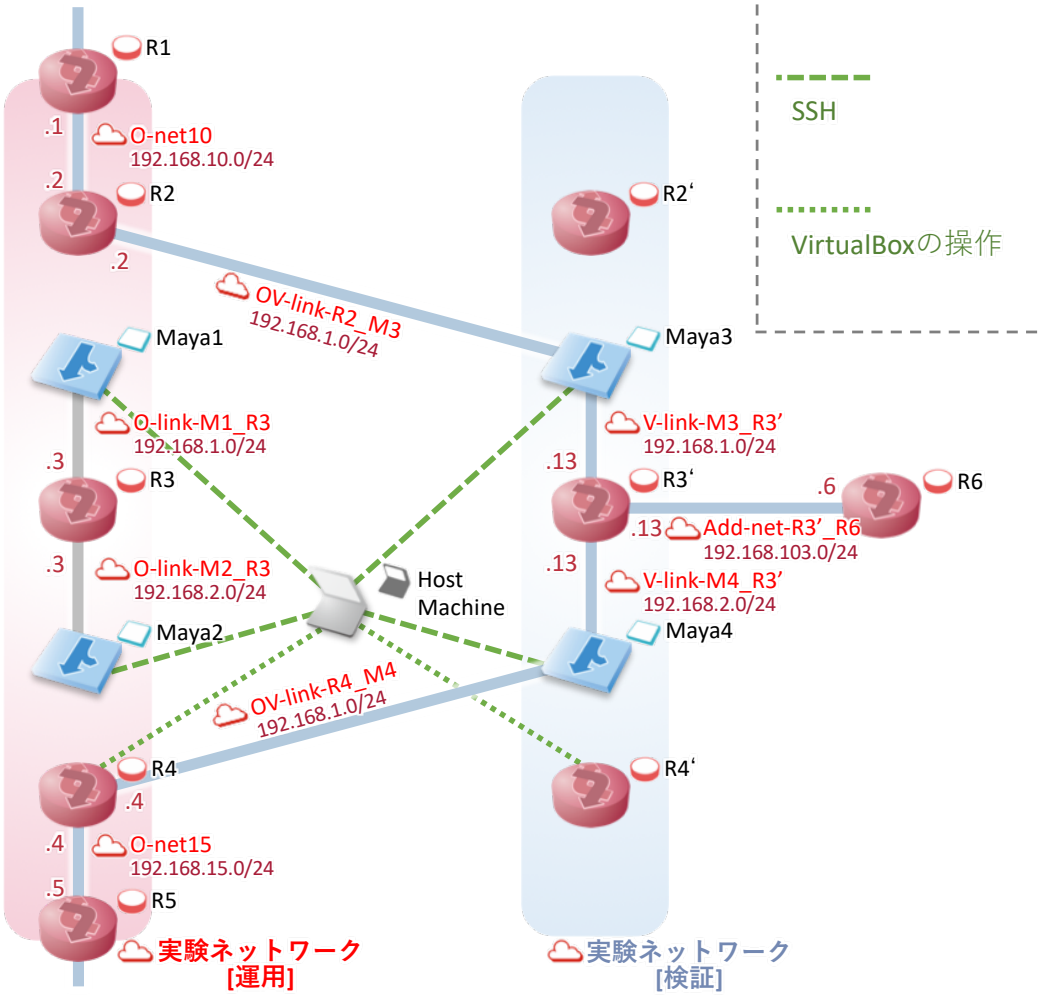

図 5.5: 置き換えの操作

1. 構築済みの実験用ネットワーク [運用] を複製する (図 5.2)。複製は、R2、R3、R4 のマシンイメージを VirtualBox の機能を利用する。複製後のマシンはそれぞれ、R2’、R3’、R4’ とする。置き換えの対象でない R2、R4 を複製するする理由は、検証用のパケットや新たなサービスを R2 及び R4 を通して検証することを想定しているためである。なお、本実験は検証用ネットワークの複製及び運用中ネットワークと検証用ネットワークの置き換えを行えるかを目的としているため、検証用ネットワーク内での具体的な検証は行わない。

2. 実験用ネットワーク [検証] を構築する (図は検証用ネットワーク構築後の実験環境全体を示したものである) 検証用ネットワークの複製後、複製したネットワークに対して Mayacon TypeB を挿入する。挿入箇所は、R2’　R3’ 間及び、R3’　R4’ 間の 2

箇所であり、挿入した Mayacon はそれぞれ Maya3、Maya4 とする (図)。Maya3 及び Maya4 の設定をコンフィグ 5.1、コンフィグ 5.2 に示す。

コンフィグ 5.1: Maya3 の設定

```
// R2'側インターフェス
ethernet eth1 {
    bridge-group {
        bridge br0
    }
    duplex auto
    hw-id 00:00:00:00:b3:1a
    smp_affinity auto
    speed auto
}

// Maya1 側インターフェイス
ethernet eth5 {
    duplex auto
    hw-id 00:00:00:00:b3:b1
    smp_affinity auto
    speed auto
}

// R3'側インターフェイス
ethernet eth6 {
    bridge-group {
        bridge br0
    }
    duplex auto
    hw-id 00:00:00:00:b3:2a
    smp_affinity auto
    speed auto
}

// eth5 から eth1 へのミラー(eth3 の tc filter show の結果)
filter protocol all pref 49152 u32
filter protocol all pref 49152 u32 fh 800: ht divisor 1
```

```
filter protocol all pref 49152 u32 fh 800::800 order 2048 key ht 800 bkt
    0 terminal flowid ???
  match 00000000/00000000 at 0
        action order 1: mirred (Egress Mirror to device eth1) pipe
        index 10 ref 1 bind 1
```

コンフィグ 5.2: Maya4 の設定

```
// R3' ~ R4'間のブリッジ
bridge br0 {
    aging 300
    hello-time 2
    max-age 20
    priority 0
    stp false
}

// R3'側インターフェイス
ethernet eth0 {
    duplex auto
    hw-id 00:00:00:00:b2:1a
    smp_affinity auto
    speed auto
}

// R4'側インターフェイス
ethernet eth2 {
    duplex auto
    hw-id 00:00:00:00:b2:3a
    smp_affinity auto
    speed auto
}

// Maya2側インターフェイス
ethernet eth3 {
    duplex auto
    hw-id 00:00:00:00:b2:b2
    smp_affinity auto
```

```
    speed auto
}

// eth3から eth0へのミラー(eth3の tc filter showの結果)
filter parent ffff: protocol all pref 49152 u32
filter parent ffff: protocol all pref 49152 u32 fh 800: ht divisor 1
filter parent ffff: protocol all pref 49152 u32 fh 800::800 order 2048
    key ht 800 bkt 0 terminal flowid ???
  match 00000000/00000000 at 0
        action order 1: mirred (Egress Mirror to device eth0) pipe
        index 1 ref 1 bind 1
```

3. 構築済みの実験用ネットワーク [運用] に項 4.4 の方法で図 5.1 の R2 と R3 の間及び、R3 と R4 の間に Mayacon を挿入する

4. 2 で構築した検証用ネットワークの R3' に対して、新たな機器 R6' を追加、R2' と R6' の通信に必要な設定をそれぞれの機器に行う (設定は以下の通りである)

   なお、R6 を追加するにあたり R3' と R6 が属するサブネットワークを 192.168.103.0/24 とし、ルーティングプロトコルは RIP とする。また、VirtualBox 上でのネットワーク名は Add-net-R3'_R6 とする。

コンフィグ 5.3: R2' におけるインターフェイスの設定

```
ethernet eth4 {
    address 192.168.103.1/24
    duplex auto
    hw-id 00:00:00:00:01:a3
    smp_affinity auto
    speed auto
}
```

コンフィグ 5.4: R2' における RIP の設定

```
rip {
    network 192.168.103.0/24
    network 192.168.1.0/24
    network 192.168.2.0/24
```

```
    redistribute {
        connected {
        }
    }
}
```

コンフィグ 5.5: R6 におけるインターフェイスの設定

```
 ethernet eth0 {
    address 192.168.103.2/24
    duplex auto
    hw-id 00:00:00:00:04:a3
    smp_affinity auto
    speed auto
}
```

コンフィグ 5.6: R6 における RIP の設定

```
 rip {
    network 192.168.103.0/24
    redistribute {
        connected {
        }
    }
}
```

5. 実験用ネットワーク [運用] と実験用ネットワーク [検証] を置き換える。置き換えは、ホストマシンからの SSH 経由でのコマンド実行と VirtualBox のコマンド実行をまとめたスクリプトを実行する。

6. WNW の Mayacon の抜去

置き換えの観測方法

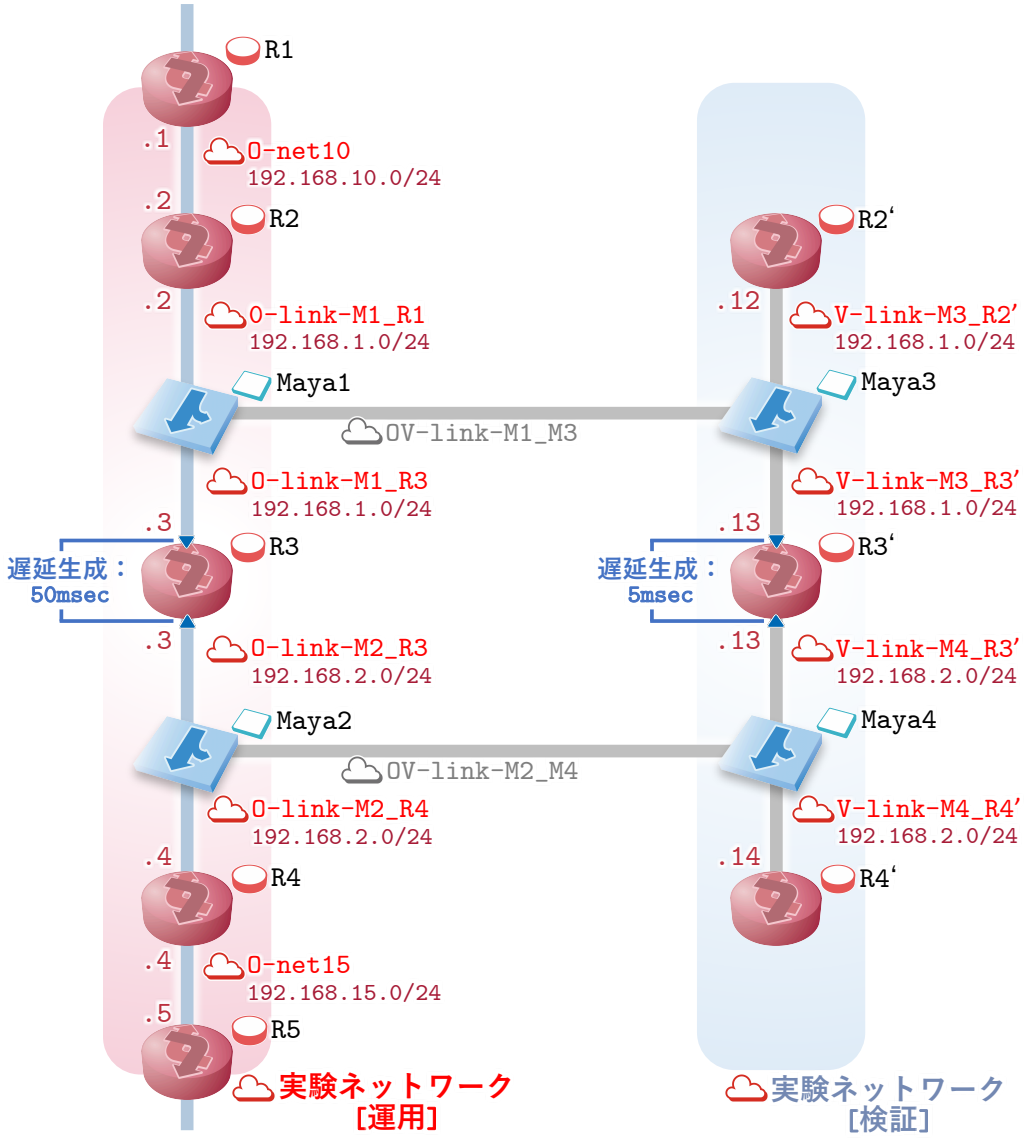

図 5.6: 遅延の生成

本実験の置き換え時の観測手法について述べる。本実験では、ネットワークの置き換えが可能であることを実証するため「ネットワークの置き換えが行われたこと」及び、「置き換えた先の新ネットワークで通信がおこないること」を観測する必要がある。「ネットワークの置き換えが行われたこと」を観測する方法として、図 5.6 のように R2 - R4 間及び R2’ - R4’ 間の通信速度に差をつけ、R1 から R5 宛に ping を継続的に実行し、ping パケットの往復にかかる時間の違いで観測する。R2 - R4 間、R2’ - R4’ 間の通信速度の差

は、R3 でコマンド 5.7 の tc コマンドを実行しそれぞれ遅延を発生させる。本実験では、R2 - R4 間の遅延を 100ms とし、R2' - R4' 間を 10ms とする。あえて遅延の差を利用した観測を行う理由は、MAC アドレスの再送で置き換え時にダウンタイムが発生することを避けるため実験用ネットワーク [運用] と実験用ネットワーク [検証] で同じ MAC アドレスを持つ機器があるため、MAC アドレスでの判別が難しいためであり、ping の実行のみで観測ができるためである。

コマンド 5.7: 遅延設定のために利用した tc コマンド

```
[R3 における設定]
vyos@r3# sudo tc qdisc add dev eth1 root netem delay 50ms
vyos@r3# sudo tc qdisc add dev eth2 root netem delay 50ms

[R3'における設定]
vyos@r3p# sudo tc qdisc add dev eth1 root netem delay 5ms
vyos@r3p# sudo tc qdisc add dev eth1 root netem delay 5ms
```

5.2 実験結果

実験の結果について述べる。実験の結果、運用中ネットワークを元に仮想マシンを複製を利用した検証用ネットワークの構築及び、Mayacon を利用した運用中ネットワークから検証用ネットワークへのパケットのミラーリング、両ネットワーク置き換えは可能であることが分かった。

virtualbox でマシンをクローンする際にクローン元の仮想マシンが一時的に停止する。今回実験で利用した仮想マシンでは、数 ms であった。

また、ネットワークを置き換える際に数十 s の停止時間があった。これは、virtualbox の設計に起因するものであると考えられる。

第6章　考察

本研究における実験結果などを踏まえ、提案手法の有用性や適用性などについて述べる。

本提案手法の検証用ネットワークを構築する前提として、運用中のネットワーク資源を利用するということがある。このため、運用中ネットワークで利用されている計算機器は現在稼働中のマシンイメージと同等のマシンをもう一台動かすための資源が必要である。したがって、常に CPU やメモリ、記憶領域が半分以上利用されている機器が動くネットワークには向いていない。反対に、時節的な要因や社会的な要因で一時的に負荷が高くなるために初めから多めに資源を確保しているネットワーク や運用中に資源を動的に変更可能なネットワークでは有用であると考えられる。

実験の結果から、稼働中のネットワークを元にネットワークが複製が可能であることが分かった。しかし同時に、VirtualBox においてはスナップショットを保存する際に百数十 m から数十 ms の遅延が発生することがわかった。これは、スナップショット保存の際に仮想マシンの状態 (メモリの状態など) を保存するコピーするためである。そのため、数十 msec の精度が必要とされるようなネットワークでは、本提案をそのまま利用するとサービスに対する影響が大きくなることが考えられる。しかし、運用中のネットワークが冗長構成を取っている場合、冗長構成になっているネットワーク一方づつ更新を行うなどの工夫をすることで、本提案が適用可能になる可能性がある。

次に、運用中ネットワークからミラーリングによって検証用ネットワーク側に供給されるパケットの有用性について述べる。ミラーリングされたパケットは稼働中のサービスで実際に流れているパケットを扱えるため、パケットの流量の変化や種類の分布などに関連する検証などには適していると言える。しかし、どのようなパケットが流れてくるか予測が困難であるため、問題の切り分けやある状態を狙った検証が難しい。この問題は、運用中ネットワークからミラーリングされるパケットを Mayacon で選択的に制御できるようにすることで解決することができると考えられる。具体的には、検証用ネットワーク側へのパケット流入量の調整や流入パケットのプロトコル等の特性によるフィルタリングなどである。

最後に、本提案におけるネットワークの置き換えについて述べる。実験の結果から本提案手法による運用中ネットワークと検証用ネットワークの置き換えが可能であることが分かった。

第7章　おわりに

本章では、本論文の全体のまとめと今後の課題と展望を述べる。

7.1　まとめ

本論文では、運用中のネットワーク資源を利用した検証環境の構築及び、運用中ネットワークと検証用ネットワークの切り替え手法の提案を行った。新サービス導入の際に、実環境へ投入後の不具合を事前に再現するのが難しいという問題を、運用中ネットワークの資源を利用して解決することを目的とした。提案手法では、運用中ネットワークを構成する計算機器の仮想マシンイメージを複製し検証用ネットワークを構築し、運用中ネットワークと検証用ネットワークの置き換えを行う。また、検証用ネットワークでは運用中ネットワークに流れるパケットがミラーリングされる。この提案手法に基づき、設計及び実装を行った。実際に実験用ネットワークを使って動作実験を行った結果、検証用ネットワークの構築、ネットワーク置き換えを行うことができた。

7.2　今後の課題と展望

本論文では、実験では、置き換え対象ネットワークのエッジに冗長構成を持たず、RIPのみのルーティングを行っている単純なネットワークを対象にした。実験用ネットワークにおいて提案手法が適用可能であることが分かった。しかし、本研究で提案する検証手法は可能な限り多様なネットワークに適用できることが望ましい。この実験のみではあらゆるネットワークに適用できるとは言いうのは難しい。そのため、ネットワーク構成やルーティングプロトコルをはじめとした多様な特性を持つネットワークで適用可能性を模索し、特性毎の適用のための条件を明らかにする必要がある。これらの条件を洗い出すことで、本提案の適用可能な条件の一般化できるようになると考えられる。ここで一般化され

た情報は、多様なネットワークで置き換え可能な検証用ネットワーク構築を行う際の基礎マニュアルとして有用であると考えられる。

謝辞

本研究を行うにあたり、多くの方から多大なご助言やご助力を頂きました。それらの方々のご協力がなければ、本研究は成り立ちませんでした。ここに深く感謝し、心から厚くお礼申し上げます。

本研究を進めるにあたり、指導教員である篠田陽一教授には様々な助言、適切な御指導を賜りました。心から深く感謝します。また、助言を頂いた主テーマ審査員である丹康雄教授、副テーマ指導教員である内平直志教授、西本一志教授に感謝致します。

本研究室の知念賢一特任准教授、宇多仁助教には、研究に関して活発な議論や多大なご指導を賜りました。心から感謝致します。

情報通信機構北陸リサーチセンターの高野祐輝氏、井上朋哉氏、三浦良介氏、明石邦夫氏には研究に関して様々な助言、ご指導を賜りました。心から感謝致します。

本研究室の博士後期課程の太田悟史氏、阿部博氏には、研究に関して活発な議論、ご指導を賜りました。心から感謝致します。

本研究室修了生の廣瀬真人氏、三木晶司氏には、研究に関して活発な議論、ご指導を賜りました。また、研究生活を送る上で様々なご助力を頂きました。心から感謝致します。

本研究室の博士前期課程の可児友邦氏、押川侑樹氏、橋本光世氏、阿波史和氏、砂川真範氏、淺葉祥吾氏、広瀬太志氏、三島航氏、宮崎駿氏、小松源氏、山口礼央氏には活発な議論や、研究生活を送る上で様々なご助力を頂きました。心から感謝致します。

最後に研究や生活で支えてくれた家族へ心から感謝致します。

参考文献

[1] 高度情報通信ネットワーク社会推進戦略本部.「ｅ－Ｊａｐａｎ戦略」.
http://www.kantei.go.jp/jp/singi/it2/kettei/010122honbun.html,
(accessed: 2018-01-10).

[2] 高度情報通信ネットワーク社会推進戦略本部.「世界最先端 IT 国家創造宣言について」.
http://www.kantei.go.jp/jp/singi/it2/kettei/pdf/20130614/siryou1.pdf,
(accessed: 2018-01-10).

[3] Richard Alimi, Ye Wang, Y. Richard Yang. 「Shadow configuration as a network management primitive」. SIGCOMM'08. 2008, p. 111-122.
http://www.sigcomm.org/sites/default/files/ccr/papers/2008/October/1402946-1402972.pdf,
(accessed: 2018-01-09).

[4] 上野幸杜. 「検証用論理ネットワーク構築手法の提案・実装」.
https://www.kri.sfc.keio.ac.jp/report/mori/2014/c-059/,
(accessed: 2018-01-09).

[5] vyos.
https://vyos.io/,
(accessed: 2018-01-23).

[6] Oracle.「VirtualBox」.
http://www.oracle.com/technetwork/jp/server-storage/virtualbox/overview/index.html,
(accessed: 2018-01-29).

[7] Martin Fowler.「BuleGreenDeployment」.
https://martinfowler.com/bliki/BlueGreenDeployment.html,
(accessed: 2018-01-29).

[8] Junichi Niino.「Immutable Infrastructure（イミュータブルインフラストラクチャ）と捨ててしまえるコンポーネント」 チャド・ファウラー氏.
http://www.publickey1.jp/blog/14/immutable_infrastructure.html.
(accessed: 2018-01-29).

[9] NTT PC Communications Incorporated.「ステージング」.
https://www.nttpc.co.jp/yougo/ステージング環境.html,
(accessed: 2018-01-30).

[10] Amazon Web Services, Inc..「Blue/Green デプロイとは？」.
http://aws.typepad.com/sajp/2015/12/what-is-blue-green-deployment.html,
(accessed: 2018-02-07).