IPv4 アドレスの枯渇時に生じる諸課題に適切に対処するための手順書

(Vol.2)

平成２７年４月

目次

第1章　はじめに

本手順書は、平成 25 年度「IPv4 アドレスの枯渇に伴う諸課題への対応推進事業の請負 Technical verification for dealing with IPv4 address exhaustion problems」事業における実証実験の結果に基づき、インターネットサービスに関わる事業者等が、IPv4 アドレスの枯渇時に生じる諸課題に適切に対処するための指針となるよう、まとめたものである。

1.1　背景・目的

Carrier Grade NAT（CGN）を利用した IPv4 アドレスの共同利用環境や IPv4/IPv6 共存環境に関しては、運用、情報セキュリティ対策等に係るノウハウがインターネットサービスに関わる事業者に十分に蓄積・共有されていない。昨年度実施した「IPv4 アドレスの枯渇に伴う情報セキュリティ等の課題への対応に関する実証実験の請負」事業において、主に有線系ネットワークにおける CGN 導入時のセッション制限に係る課題やセキュリティに係る課題についての対策案を整理し、手順書としてまとめた [1]。昨年度の手順書では、セッション制限に係る課題に関して、有線系ネットワークのアプリケーションのセッション数を調査し、通信における公平性 [2]を保つための適切なセッション制限についての課題と対策について解説した。また、情報セキュリティに関わる課題の一つとして、IPv4 アドレス共同利用環境においてサーバサイドでのアクセス制限によってオーバーブロッキングとなってしまう課題と対策を解説した。そして、以上をもとに有線系ネットワークにおける CGN の最適配置と導入のための手順をまとめた。

一方、近年スマートフォン利用者が急激に増えたことにより、モバイル通信事業者において CGN を利用した IPv4 アドレスの共同利用環境の導入が既に進められている。しかし、モバイルネットワークにおいても CGN 導入により有線系ネットワークと同等の課題が存在すると指摘されている。本年度の実証実験では、有線系ネットワークでの検証を踏まえて、モバイルネットワークにおける課題として新たに以下の 4 項目の検証を行い、対策について検討した。

---

[1] 総務省
「IPv4 アドレスの枯渇時に生じる諸課題に適切に対処するための手順書」:
http://www.soumu.go.jp/main_content/000240919.pdf

[2] NAT の一種である CGN は、IP アドレスとポート番号を利用して NAPT 変換を行う。複数ユーザで 1 つの IP アドレスを共有するため、1 つの IP アドレスあたりのポート数(論理的な上限は、65535 ポート)を複数ユーザで分割して使用することとなる。アプリケーションが行う TCP/IP 通信では、1 つの TCP セッションあたり 1 つのポートを利用するため、同一 IP アドレスを共有するユーザの 1 人が大量のセッションを使用するアプリケーションを利用した場合、残りの多数のユーザが使用できるポートが足りなくなることとなり、アプリケーションを利用できなくなるなど、通信を行う際の公平性が失われてしまう。

（1）IPv4 アドレス共同利用技術のモバイルネットワークへの適用性の検証
（2）IPv4 共同利用環境における新技術導入による対策
（3）クラウドサービスに係る IPv4 アドレスの枯渇対策に伴う課題への対処
（4）その他 IPv4 アドレスの共同利用環境において考慮すべき課題への対処

特にモバイルネットワーク固有の課題として、モバイルネットワークのネットワーク遅延やパケットロスが CGN の性能に与える可能性が考えられる。そのため、モバイル端末における通常時・ハンドオフ [3]時のネットワーク遅延やパケットロスを実測し、それらをエミュレーションする検証環境を用いて IPv4 共同利用環境に与える影響について検証を行った。その結果、モバイルネットワークにおける遅延やパケットロスは CGN の性能に影響を与えるが、CGN の導入に問題を与えるほどではないことが確認された。

モバイルネットワークにおいては、使用されるアプリケーションや利用シーンが異なるため、有線系ネットワークとセッション数の傾向が異なる。そのため、新たにモバイルアプリケーションについてセッション数の調査を行い、セッション数制限の最適値および収容効率に基づく CGN の最適配置を求めた。本調査・検証の過程でセッション数の仮説を立て、1000 人規模のカンファレンスネットワークに CGN を導入し、モバイル端末通信の IPv4 アドレス共同利用状況および IPv6 通信対応状況をマクロ的に観測した。その結果、モバイル端末におけるセッション数の実測値を計測できた。また、ユーザが利用するコンテンツの多くが IPv6 通信への対応を始めており、モバイル端末が IPv6 に対応することにより、多くの通信が IPv6 に移行することが確かめられた。

本カンファレンスでは、平均して 30%～40%（最大 61.24%）の通信が IPv6 だった。海外の発表 [4]でも同様の数値を示しており、既に IPv6 が我々の身近にあることがわかる。IPv6 が利用されることによって CGN の負荷が減少することから、このことは、インターネット関連事業者は CGN 等を利用して IPv4 アドレスの延命を図りつつも、早急に IPv4/IPv6 を同時に利用できる導入モデルを検討すべきであるという示唆を含んでいる。

このため本年度は、IPv4/IPv6 を同時に利用できる導入モデルとして CGN+IPv6 モデルにおける CGN 機器の推奨構成、および IPv4 over IPv6 技術（464 技術）の動作検証を行う為の環境構築を行う手順等について手順書としてまとめた。

1.2　対象者

[3] ハンドオフ：モバイル通信において、端末と通信を行う基地局が別の基地局に切り替わること。
[4] Alastair JOHNSON「OPERATIONAL FEEDBACK FROM LARGE SCALE NAT」
https://conference.apnic.net/data/37/2014-02-25-apnic37-nat-panel_1393371165.pdf

IPv4/IPv6 共存環境を既に導入している企業や、IPv4 アドレス環境のみで事業を行っている企業も含め、IPv4 アドレス枯渇への対策が必要と考えられる事業者を対象とし、手順書を作成した。

IPv4 アドレス枯渇への対策が必要と考えられる事業者は以下のとおりである。

・MNO 事業者

・MVNO 事業者

・ISP 事業者

また、関連する領域として、以下の読者を想定する。

・ASP 事業者、コンテンツ・アプリケーション提供者

・データセンタ事業者

・通信機器メーカ　等

1.3　IPv4 枯渇対策の推奨対応について

MNO 事業者・MVNO 事業者・ISP 事業者における IPv4 枯渇対策について、推奨対応を記載する。

■MNO 事業者

NTT ドコモ、KDDI(au)、 Softbank、UQ コミュニケーションズなど、大手の MNO 事業者ではすでに CGN が導入されている。

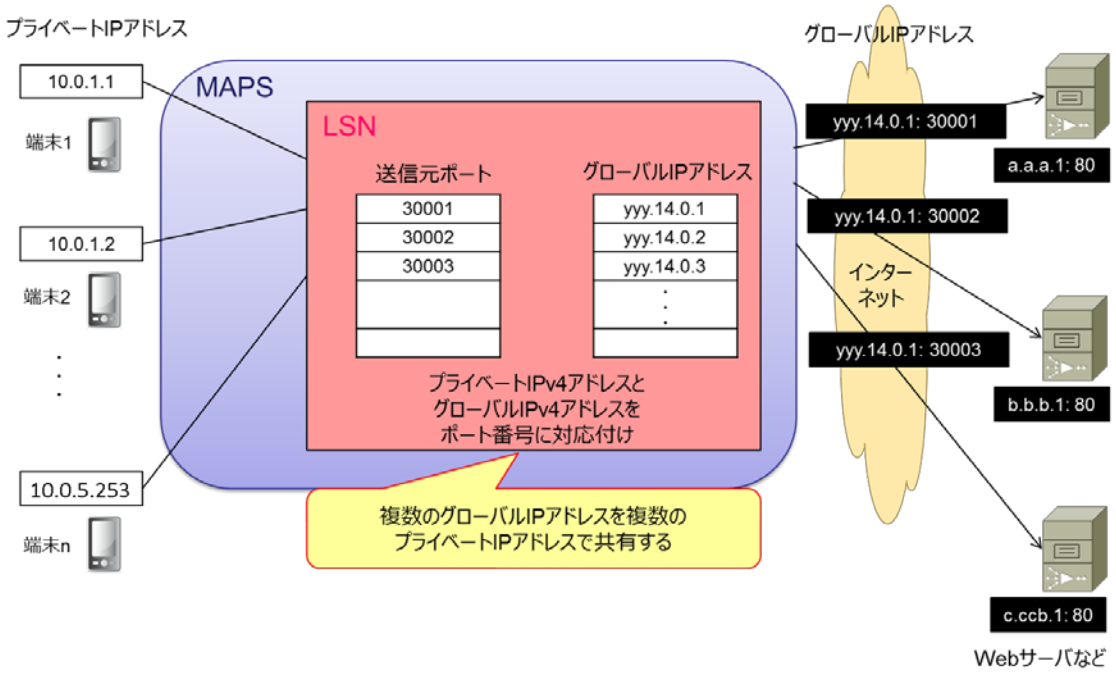


図 1-1　MNO 事業者における CGN の適応事例
(NTT Docomo テクニカルジャーナル VOL.18 NO.3「2010 年スマートフォン新サービス・機能―スマートフォン向けサービス提供基盤―」[5]を参考に作成)

これらの事業者はプライベート IPv4 からグローバル IPv4 アドレスへの変換による IPv4 アドレス共同利用技術(NAT44 モデル)を用いている。しかし、NAT44 モデルの CGN においては、IPv6 とともに導入することが重要である。
理由を以下に挙げる。

- IPv6 対応によって CGN への負荷を減らすことができるため
- CGN によって通信に影響のあるアプリケーションが存在するため [6]
- NAT ログを用いた法的要請対応は、NAT ログ保存・検索のためのコストがかかるため

これらの問題点は IPv6 への移行によって解決する。CGN(NAT44)と同時に IPv6 を提供

---

[5] NTT Docomo「2010 年スマートフォン新サービス・機能」
https://www.nttdocomo.co.jp/binary/pdf/corporate/technology/rd/technical_journal/bn/vol18_3/vol18_3_038jp.pdf
MAPS：Multi-Access Platform System の略。NTTDocomo のプラットフォーム基盤のこと。
LSN：Large Scale NAT の略。Carrier Grade NAT(CGN)と同義。
[6] 総務省「IPv4 アドレスの枯渇時に生じる諸課題に適切に対処するための手順書」：
http://www.soumu.go.jp/main_content/000240919.pdf

して、CGN の負荷を減らしながら IPv6 に移行するモデルを、NAT44+IPv6 モデルと呼ぶ。

NAT44+IPv6 モデルにおける CGN 機器の推奨構成について、2.2 節に記載する。
モバイルネットワークにおけるセッション数について、2.3 節に記載する。

また、CGN の以下の設定に関する推奨対応について、各節に記載する
・セッション数に関する設定(2.4.1)
・アドレスに関する設定(2.4.2)
・NAT タイムアウトに関する設定(2.4.3)
・Full Cone NAT に関する設定(2.4.4)
・Application Level Gateway(ALG)に関する設定(2.4.5)
・ログに関する設定(2.4.6)

MNO 事業者における IPv6 移行に向けた CGN 導入モデルとして、NAT44+IPv6 モデルの他に、464XLAT[7]モデルが考えられる。

IPv6 移行技術の一つである 464XLAT は、IPv6 のみのバックボーンネットワーク上で IPv4 通信を提供する。DNS64/NAT64[8]モデルでは IPv4 リテラル [9]による通信ができないため、Skype など IPv4 リテラルを用いるアプリケーションで通信ができないが、464XLAT ではモバイル端末に Customer-side translator(CLAT)機能 [10]を持たせることでこの問題を解決し、IPv4・IPv6 両方の通信が可能となる。

Android4.3 以降では CLAT の機能が標準提供されており、T-Mobile などが 464XLAT を用いた IPv4 アドレス共有・IPv6 移行を進めている。日本国内のモバイル事業では 464XLAT はまだ導入されていないが、実証のための検証をすることが重要である。

464XLAT の動作検証を行う為の環境構築の手順書を 3.3 節に記述した。

■MVNO 事業者

DTI(ServersMan SIM LTE)、楽天(楽天ブロードバンド LTE)など、MVNO 事業者の一部でも既に CGN が導入されている。MNO 事業者と同様の理由により、CGN 導入済の MVNO 事業者・これから CGN 導入を検討している MVNO 事業者には、IPv6 サービスも同時に提供することを推奨する。

---

[7] IPv6 と IPv4 の共存技術の一つ。詳細は 3.3 節を参照。
[8] 同じく、IPv6 と IPv4 の共存技術の一つ。DNS64 によって IPv4 と IPv6 を関連づける名前解決を行い、NAT64 によって IPv6 から IPv4 へプロトコル変換をする。
[9] 直接 IPv4 アドレスを指定し、名前解決を行わない通信のこと。
[10] CLAT : 加入者側の機器にて IPv4 アドレスと IPv6 アドレスをステートレスに変換(RFC 6145)するトランスレータの名称。

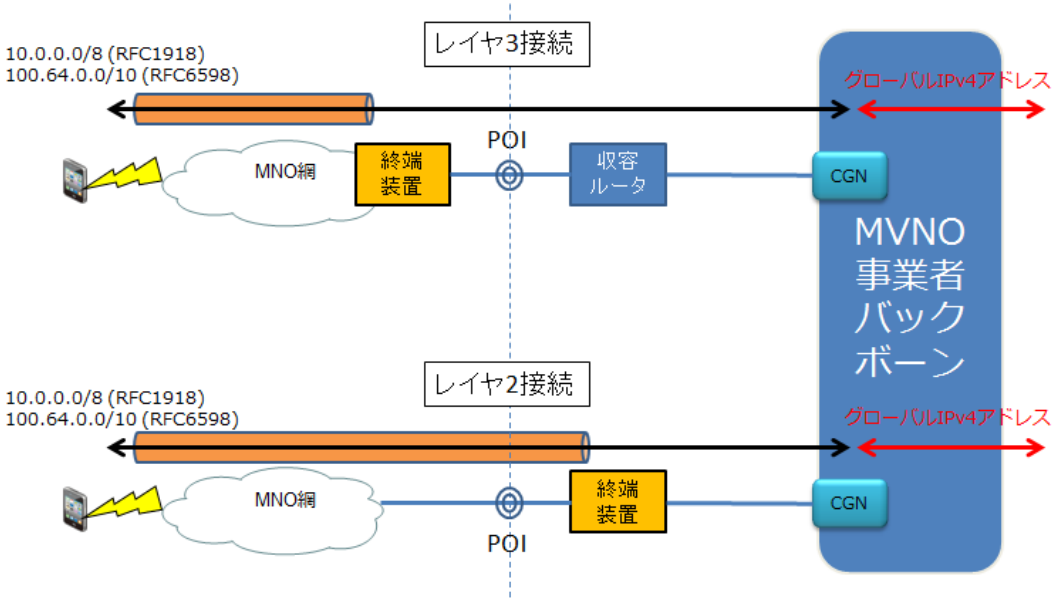


図 1-2　MVNO 事業者における CGN の適応事例

NAT44+IPv6 モデルにおける CGN 機器の推奨構成について、2.2 節に記載する。
モバイルネットワークにおけるセッション数について、2.3 節に記載する。

CGN の以下の設定に関する推奨対応について、各節に記載する

・セッション数に関する設定(2.4.1)
・アドレスに関する設定(2.4.2)
・NAT タイムアウトに関する設定(2.4.3)
・Full Cone NAT に関する設定(2.4.4)
・ALG(Application Level Gateway)に関する設定(2.4.5)
・ログに関する設定(2.4.6)

■ISP 事業者

有線系ネットワークに CGN を導入する手順は平成 25 年度の手順書[11]に記載した。CATV 事業者では CGN を導入しているケースが存在するが、NTT 東日本及び NTT 西日本の PPPoE 接続をアクセスラインとして CGN 導入している事業者はまだいない。固定アクセス網における新規加入者の増加は緩やかであり、すでに所有している IPv4 アドレスによっ

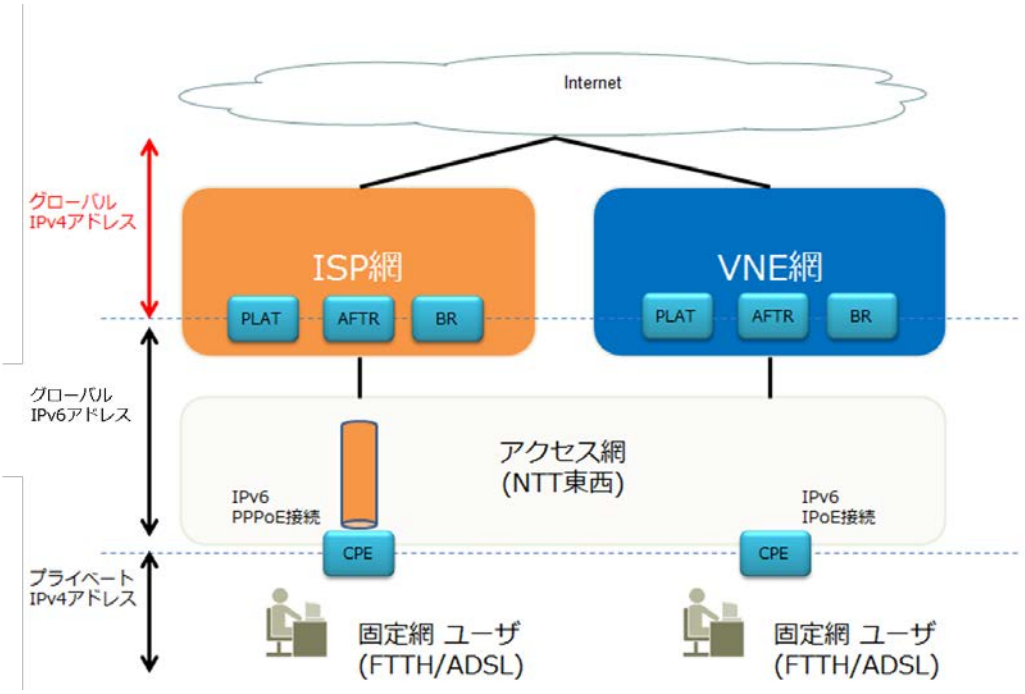
[11] 総務省
「IPv4 アドレスの枯渇時に生じる諸課題に適切に対処するための手順書」:
http://www.soumu.go.jp/main_content/000240919.pdf

て既存ユーザを収容できているため、新規 IPv4 アドレスのニーズは小さいといえる。

NTT 東西の IPoE 接続においては、IPv4 over IPv6 技術を用いて、IPv6 ネットワーク上に IPv4 接続を提供してする方式(例: Yahoo!BB における「IPv6 高速ハイブリッド IPv6 IPoE + IPv4」)を選択している事業者がある。今後も、以下のような IPv4 over IPv6 技術を利用して IPv4 アドレスを節約しつつ IPv6 通信に移行していく事業者が現れる可能性がある。

- DS-Lite
- 464XLAT
- MAP-E
- MAP-T

以上に挙げた 464 技術の全てに対応しているオープンソースである Vyatta[12]を用いた検証環境の構築手順について、3 章に記述した。



[12] Vyatta : Vyatta 社によりオープンソースで開発されていた Linux ベースのソフトウェアルータ。

図 1-3　ISP 事業者における 464 技術の適応事例（NTT 東西をアクセス網とした場合）
図中の PLAT、AFTR、BR はそれぞれ、464XLAT、DS-lite、MAP-E/MAP-T における
IPv6/IPv4 変換装置の名称である。

第2章　モバイルネットワークにおける CGN 導入の最適モデル

モバイルネットワークを有する MNO 事業者・MVNO 事業者を想定し、NAT44+IPv6 モデルにおける CGN 導入の手順を記述する。

2.1　機器選定のための検証構成・検証手法

CGN の導入にあたっては、CGN 機器の選定を実施し、1 台あたりの収容可能人数と、サービススペックを決定することが重要である。導入に必要な CGN 機器の検証項目を以下に記述する。

■性能検証

- CGN の性能を測定する
- CGN 周辺システムの性能を測定する
- 障害を模擬した切替実施により、High Availability(HA)[13]構成の有効性を確認する

以上の性能について検証で確認し、CGN の配置および収容規模を決定する。

■機能検証

- ポートアサインルールについて
  - 十分な数のプールアドレスを保有できるか
  - 動的・静的なポートアサインルールを設定できるか
  - ユーザごとのポート数上限を設定できるか
- NAT テーブルについて
  - Full Cone NAT として動作させることが可能であるか
  - NAT タイムアウト値が可変であるか
- ルーティングについて
  - IPv4 グローバルアドレスを透過的にルーティング可能であるか
  - IPv6 アドレスを透過的にルーティング可能であるか
  - 動的ルーティングプロトコル(OSPF/BGP)が利用可能であるか
- NAT ログについて
  - ユーザ特定に必要な NAT ログを送出することが可能か

---

[13] 冗長化によって実現できる高可用性のこと。

- NAT ログの内容・フォーマットについて設定可能か

- Application Level Gateway(ALG)について
  - ALG の設定によって CGN を越えた通信が可能となるか

以上の機能について検証で確認し、運用手順を想定しサービススペックを決定する。

2.1.1 性能検証手法

CGN 機器の性能の基準となる設計値は以下の 3 点である。

(A) スループット(Throughput)

(B) 最大同時セッション数(MCS:Max Concurrent Session)

(C) コネクション確立速度(CER: Connection Establishment Rate)

これらの値は、IXIA 社などの測定器メーカによるハードウェア(Optixia XM シリーズ)/ソフトウェア(IxLoad[14])を用いて計測が可能である。また、「IPv4 アドレスの枯渇に伴う諸課題への対応推進事業の請負」において大規模検証環境(StarPorte)を提供しており、申込みにより StarBED における負荷発生ツール(同時セッション数負荷・コネクション確立速度負荷)を用いた検証が可能である。

それぞれの値は独立ではなく、CGN 機器に対して複合的な負荷となる。メーカの公称値(カタログ値)は特定の条件下においてそれぞれ達成された値であり、実網トラフィックにおいては各負荷が合計されるので、実際の収容限界値がカタログ値を下回る可能性が高い点には注意されたい。

一例として、大規模検証環境:StarPorte を利用し、A10 ネットワークス社の CGN 機器、AX3000 を検証した結果を以下に示す。

- スループット: 20G[bps]
- 最大同時セッション数:16M[session]
- コネクション確立速度:130K[cps]

モバイルにおける 1 ユーザ当たりの値は、

- スループット:3K[bps]
- 平均セッション数:50[session]
- コネクション確立速度:0.1 以下[cps]

程度である。

以上より、AX3000 においては、セッション数によって一番始めに性能限界を迎えることが予想され、モバイルユーザで 32 万端末程度を同時収容できると推測できる。

---

[14] IXIA「コンテンツアウェアデバイスのパフォーマンステスト」
http://www.ixiacom.jp/sites/default/files/content/support/library/datasheets/ixload_brochure_japanese.pdf

通信の正常性は、CGN を通るパケットにおけるパケットロス・遅延・ジッタを計測して確認する。CGN が無い場合・CGN で NAT する場合・CGN で NAT されずに転送される場合の 3 つを比較することで、CGN におけるボトルネック・NAPT 変換におけるボトルネックが無いことを確認する。また、CGN 機器自身のカウンタ(インターフェースカウンタ・メモリ使用率・CPU 使用率)を確認する。その他、CGN の特徴として、NAT テーブルの使用状況を確認することが重要である。A10 ネットワークス社の AX シリーズを例として、NAT テーブルの利用状況を確認するコマンドと出力結果を提示する。

■NAT プールアドレスごとの利用中のポート統計情報の確認

ax-cgn#show ip nat lsn pool-statistics

(出力結果)

LSN Address Pool Statistics:

LSN-POOL	Address	Users	ICMP			UDP			TCP			
			Freed	Total	Rsvd	Freed	Total	Rsvd	Freed	Total	Rsvd	
	115.69.229.8	5	0	7	7	1	18093	18094	0	20	12739	12759	0
	115.69.229.9	7	0	27	27	27	11104	11131	0	84	10886	10970	0
	115.69.229.10	8	0	0	0	3	9309	9312	0	35	7427	7462	0
	115.69.229.11	7	0	13	13	0	11037	11037	0	37	6279	6316	0
	115.69.229.12	7	0	45	45	5	10680	10685	0	82	9898	9980	0
	115.69.229.13	7	0	0	0	1	7161	7162	0	51	4396	4447	0
	115.69.229.14	5	0	96	96	9	6240	6249	0	37	4561	4598	0
	115.69.229.15	9	0	3	3	74	10030	10104	0	156	7160	7316	0

※各行が、プールアドレスごとの利用中のポートの統計情報を表し、枠で示した ICMP/UDP/TCP の列は、それぞれのプロトコルでの利用中のポート数を表す。

■ユーザごとの利用ポート数の確認

ax-cgn#show ip nat lsn user-quota-sessions

(出力結果)

LSN User-Quota Sessions:

Inside Address	NAT Address	ICMP	UDP	TCP	Session	Pool LID
100.64.0.188	115.69.229.9	0	3	4	6	LSN-POOL 1
100.64.0.134	115.69.229.10	0	0	2	1	LSN-POOL 1
100.64.0.77	115.69.229.10	0	3	8	11	LSN-POOL 1
100.64.0.165	115.69.229.12	0	1	6	8	LSN-POOL 1
100.64.0.187	115.69.229.10	0	1	4	5	LSN-POOL 1
100.64.0.55	115.69.229.11	0	0	4	3	LSN-POOL 1
100.64.0.85	115.69.229.15	0	21	60	81	LSN-POOL 1
100.64.0.123	115.69.229.8	0	0	2	2	LSN-POOL 1
100.64.0.136	115.69.229.9	0	0	4	4	LSN-POOL 1
100.64.0.145	115.69.229.13	0	0	3	3	LSN-POOL 1
100.64.0.70	115.69.229.10	0	1	1	2	LSN-POOL 1
100.64.0.182	115.69.229.9	0	5	13	16	LSN-POOL 1
100.64.0.129	115.69.229.9	0	1	16	4	LSN-POOL 1
100.64.0.126	115.69.229.11	0	0	1	0	LSN-POOL 1

100.64.0.106		115.69.229.12	0	1	3	4
LSN-POOL	1					
100.64.0.113		115.69.229.14	0	6	3	8
LSN-POOL	1					
100.64.0.161		115.69.229.14	0	0	10	10
LSN-POOL	1					
100.64.0.36		115.69.229.9	0	0	20	19
LSN-POOL	1					
100.64.0.186		115.69.229.13	0	0	12	9
LSN-POOL	1					
100.64.0.108		115.69.229.13	0	0	1	1
LSN-POOL	1					
100.64.0.76		115.69.229.9	0	0	5	5
LSN-POOL	1					
100.64.0.130		115.69.229.15	0	1	2	4
LSN-POOL	1					
100.64.0.192		115.69.229.8	0	0	4	4
LSN-POOL	1					
100.64.0.79		115.69.229.14	0	2	0	2
LSN-POOL	1					
100.64.0.185		115.69.229.11	0	0	6	6
LSN-POOL	1					
100.64.0.190		115.69.229.8	0	0	12	10
LSN-POOL	1					
100.64.0.119		115.69.229.14	0	1	0	1
LSN-POOL	1					
100.64.0.100		115.69.229.9	0	0	1	1
LSN-POOL	1					
100.64.0.189		115.69.229.12	0	1	3	3
LSN-POOL	1					
100.64.0.166		115.69.229.12	0	1	10	10
LSN-POOL	1					
100.64.0.131		115.69.229.13	0	1	8	8
LSN-POOL	1					
100.64.0.175		115.69.229.15	0	19	6	28
LSN-POOL	1					

100.64.0.54		115.69.229.8	0	0	13	13
LSN-POOL	1					
100.64.0.183		115.69.229.9	0	1	1	1
LSN-POOL	1					
100.64.0.118		115.69.229.12	0	0	1	1
LSN-POOL	1					
100.64.0.179		115.69.229.14	0	4	1	5
LSN-POOL	1					
100.64.0.84		115.69.229.12	0	0	16	16
LSN-POOL	1					
100.64.0.93		115.69.229.11	0	0	1	1
LSN-POOL	1					
100.64.0.181		115.69.229.15	0	0	6	6
LSN-POOL	1					
100.64.0.135		115.69.229.12	0	0	12	12
LSN-POOL	1					
100.64.0.180		115.69.229.13	0	1	2	0
LSN-POOL	1					
100.64.0.184		115.69.229.9	0	0	12	12
LSN-POOL	1					
100.64.0.49		115.69.229.15	0	0	3	2
LSN-POOL	1					
100.64.0.97		115.69.229.10	0	0	4	4
LSN-POOL	1					
100.64.0.138		115.69.229.15	0	1	16	9
LSN-POOL	1					
100.64.0.71		115.69.229.13	0	0	5	5
LSN-POOL	1					
100.64.0.117		115.69.229.8	0	2	9	9
LSN-POOL	1					
100.64.0.61		115.69.229.14	0	0	1	1
LSN-POOL	1					
100.64.0.191		115.69.229.13	0	4	8	8
LSN-POOL	1					
100.64.0.56		115.69.229.15	0	0	3	2
LSN-POOL	1					

100.64.0.104		115.69.229.13	0	0	1	1
LSN-POOL	1					
100.64.0.92		115.69.229.10	0	0	1	0
LSN-POOL	1					
100.64.0.173		115.69.229.14	0	2	19	23
LSN-POOL	1					
100.64.0.156		115.69.229.10	0	0	2	1
LSN-POOL	1					
100.64.0.72		115.69.229.12	0	0	1	1
LSN-POOL	1					
100.64.0.82		115.69.229.13	0	0	1	1
LSN-POOL	1					
100.64.0.132		115.69.229.12	0	2	1	3
LSN-POOL	1					
100.64.0.176		115.69.229.11	0	0	11	11
LSN-POOL	1					
100.64.0.178		115.69.229.13	0	0	13	9
LSN-POOL	1					
100.64.0.164		115.69.229.15	0	1	4	5
LSN-POOL	1					

Total User-Quota Sessions Shown: 60

※ユーザごとの、利用プールアドレス、利用ポート数を表す。

■ポート割当に関する統計情報の確認(割当失敗の確認)

ax-cgn#show ip nat lsn statistics

(出力結果)

Traffic statistics for LSN:

Total TCP Ports Allocated	69651
Total TCP Ports Freed	69207
Total UDP Ports Allocated	92018
Total UDP Ports Freed	91962
Total ICMP Ports Allocated	206
Total ICMP Ports Freed	206
Data Session Created	168652
Data Session Freed	168142

User-Quota Created	1369
User-Quota Freed	1310
User-Quota Creation Failed	0
TCP NAT Port Unavailable	0
UDP NAT Port Unavailable	0
ICMP NAT Port Unavailable	0
New User NAT Resource Unavailable	0
TCP User-Quota Exceeded	0
UDP User-Quota Exceeded	0
ICMP User-Quota Exceeded	0
Extended User-Quota Matched	0
Extended User-Quota Exceeded	0
Data Session User-Quota Exceeded	0
TCP Full-cone Session Created	2397
TCP Full-cone Session Freed	2278
UDP Full-cone Session Created	79
UDP Full-cone Session Freed	70
Full-cone Session Creation Failed	0
Hairpin Session Created	40
Self-Hairpinning Drop	0
Endpoint-Independent Mapping Matched	6778
Endpoint-Independent Filtering Matched	40
Endpoint-Dependent Filtering Drop	0
NAT Pool Mismatch Drop	0
TCP Port Overloaded	0
UDP Port Overloaded	0
TCP Port Overloading Session Created	0
UDP Port Overloading Session Created	0
TCP Port Overloading Session Freed	0
UDP Port Overloading Session Freed	0
NAT IP TCP Max Ports Allocated	0
NAT IP UDP Max Ports Allocated	0
Full-cone Inbound Filtering Drop	0
No Class-List Match	0
LSN LID Drop	0
LSN LID Pass-through	0

割当失敗した通信セッションの数を表す。

2.1.2 CGN 周辺システム検証手法

CGN の周辺システムとして、CGN から送出される NAT 変換ログを受信し保存するサーバが必要である。ポートの動的割り当てをしている場合には NAT テーブルの割当・消去のタイミングで、下記のログを取得することが推奨される。

- 送信元 IP アドレス:ポート
- NAT 変換後 送信元 IP アドレス:ポート
- 通信先 IP アドレス:ポート
- Timestamp

ログの形式として、大きく分けて ASCII 形式とバイナリ形式がある。ASCII 形式の場合、このログは 1 レコードで約 120 byte となる。バイナリ形式の場合、1 レコードで約 30byte となり、ASCII 形式と比較しログの量は約 25%となる。

NAT 変換ログの送出方法は

- SYSLOG を用いた方法 [15]
- IPFIX を用いた方法 [16]

が IETF にて標準化のために検討されている。

SYSLOG を用いた方法では、rsyslog など既存の syslogd を用いてデータの受信・保存が可能である。ログの送出は ASCII 形式が一般的だが、NAT 変換ログの量を圧縮するために、バイナリ形式で送出できる CGN 機器も存在する。IPFIX を用いた方法はバイナリ形式で送出するが、IPFIX を受け取れる NAT 変換ログに特化したツールが必要となり、現状では個別に開発が必要となる。

NAT 変換ログ取得の正常性の確認は、CGN 機器のカウンタと、SYSLOG のデータ数を比較することで実施する。CGN 機器のカウンタと、SYSLOG のデータ数が不一致であった場合、SYSLOG は UDP パケットで再送がないため、原因として CGN 機器での送出ができなかった他に、途中のネットワーク機器にてパケットが破棄された可能性や SYSLOG サーバで受信ができなかった可能性が考えられる。切り分けには、途中のネットワーク機器や SYSLOG サーバでパケットキャプチャを行う。切り分けのためにも NAT 変換ログはユーザ通信とは別のインターフェースから送出するのが望ましい。

SYSLOG サーバ側の上限で NAT 変換ログを 1 台で受け取ることができない場合には、SYSLOG サーバの分散が必要となる。CGN 機器で SYSLOG 送出先を複数設定し分散することが可能であれば、SYSLOG サーバを追加するだけでよい。ただし、均等にバランスされて送出することができなければ、特定のサーバに負荷が集中してしまうため、負荷分散手法について検証することを推奨する。また、SYSLOG サーバが複数となることで、NAT

---

[15] IETF「draft-ietf-behave-syslog-nat-logging」
http://tools.ietf.org/wg/behave/draft-ietf-behave-syslog-nat-logging/
[16] IETF「draft-ietf-behave-ipfix-nat-logging」
http://tools.ietf.org/wg/behave/draft-ietf-behave-ipfix-nat-logging/

ログの検索がより困難になることが予想される。対策として、NAT ルールを複数に分けてルールごとに送出するサーバを分ける方法が考えられる。CGN 機器に送出先を複数設定する機能がない場合には、ロードバランサなどのアプライアンス機器を用いて SYSLOG を分散する。

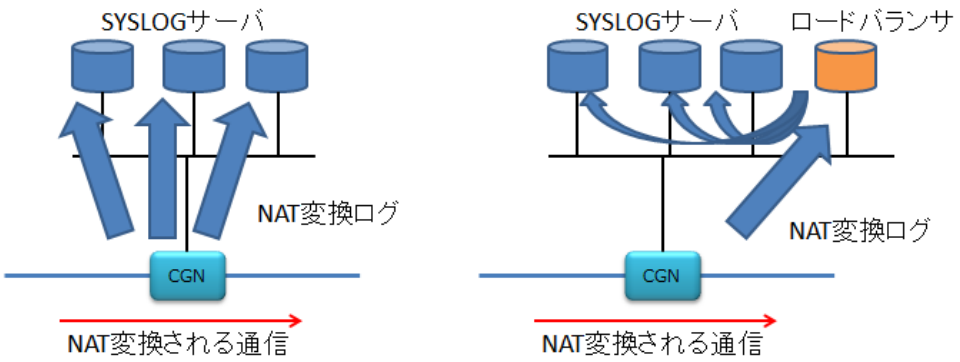


図 2-1　SYSLOG サーバ構成例
複数 SYSLOG サーバに NAT ログを分散する機能が CGN 機器にある場合(左)・ない場合(右)

2.1.3　　HA 検証手法

CGN は NAT 変換という重要な役割を網内において行うことから、Single Point of Failure(SPOF)となる可能性がある。そのため、CGN 機器において可能な 冗長構成を検討し、検証することが重要である。HA 機能の実装は機器によって異なるため一概に述べることはできないが、ここでは A10 ネットワークス社の AX シリーズを例に HA 検証手法を説明する。

AX シリーズは、 Virtual Router Redundancy Protocol(VRRP)によって Active-Standby 構成を取る。Active となっている機器が保持しているセッション情報は、機器間を接続する Heart Beat 線の接続によって Standby 側の機器に常に同期される。通信は VRRP によって常に Active 側の CGN 機器を通ることになるが、障害が発生した時には Standby 側にトラフィックが移るとともに、同期されていたセッション情報を用いて NAT 変換が管理されるため、ほぼ無瞬断で切り替えることが可能である。

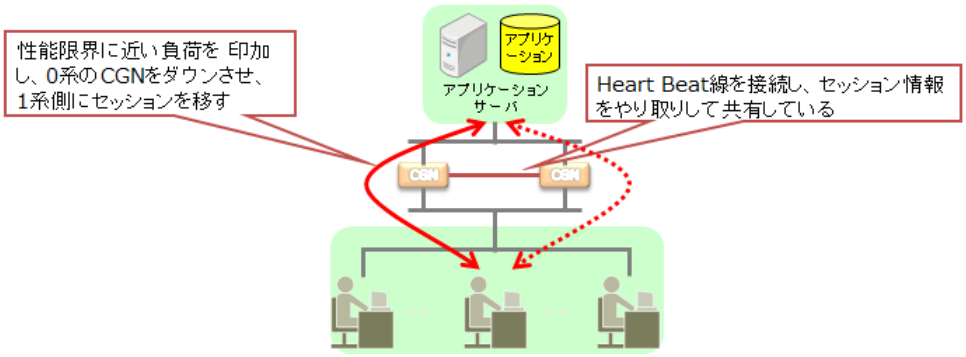


図 2-2　HA 検証構成例(A10 ネットワークス社 AX シリーズの場合)

HA 機能の検証では、性能限界に近い負荷をかけた状態で疑似的な障害を発生させる。正常性の確認は、性能検証と同様に CGN を通るパケットにおけるパケットロス・遅延・ジッタを計測して確認する。また、CGN 機器のセッション情報が失われてしまうと、そのユーザが通信できなくなる障害が発生するため、切り替わりの前後でのセッション情報を確認する。

以上からわかるように、CGN 機器における冗長化では、セッション情報の同期(あるいは引き継ぎ)と冗長化プロトコルの連携が重要である。CGN 機器の中には、OSPF・BGP などを用いることができるものがあり、これらの動的ルーティングプロトコルによる切り替わり条件を網羅的に検証するべきである。特に、往復の通信が別々の筐体を通る非対称通信をサポートできる CGN 機器は少ない。

2.2　機器構成

MNO 事業者および MVNO 事業者に NAT44+IPv6 モデルを適用した場合の機器構成を記述する。

NAT44+IPv6 モデルでの IPv6 通信の経路には、

- CGN 機器を通過する場合
- CGN 機器を通過しない場合

の 2 パターンが存在する。多くの CGN 機器で IPv6 通信を透過的にルーティングが可能であるが、CGN 機器の負荷を減らしていくモデルであることと、障害時の明確な切り分けのために、後者の構成を推奨する。

■MNO 事業者機器構成

多くの MNO 事業者は CGN 導入済の Access Point Name(APN)を有している。ただし、プライベート IPv4 アドレスの払い出しを行うサービスプランは、IPv6 アドレスの払い出しを行っていない。対して、データ通信プランの APN は、グローバル IPv4 アドレス + IPv6 アドレスの組み合わせによるデュアルスタック対応ができている(Docomo:MoperaU、KDDI: LTE NET for DATA 等)。

CGN 導入済の APN についても、データ通信プランの APN と同様に IPv4v6 type によるデュアルスタック対応が可能である。2.3 節で記述するように、IPv4/IPv6 デュアルスタック対応によって、インターネット向けの通信の 30%～40%が IPv6 となることが期待できる。IPv6 対応によって、CGN 機器の負荷が下がり収容効率が高まるほか、NAT ログを用いた法的要請対応のための NAT ログ保存・検索のコストを下げることができる。

しかし、キャリアメールサービス・フィルタリングサービスなど、IPv4 のみで提供しているサービスがあることから、フィルタリングなどの必須のセキュリティサービスが IPv6 対応しないと IPv6 アドレスの払い出しができないという課題が「IPv6 によるインターネットの利用高度化に関する研究会」でも報告されている。

■MVNO 事業者機器構成

図 2-3・図 2-4 にレイヤ 3 接続・レイヤ 2 接続の際の CGN の適用例を示す。接続を 1Gbps とし、10 万 ID～100 万 ID 規模の接続を想定している。レイヤ 3 接続で CGN を導入する場合、MNO 事業者の終端装置においてプライベート IPv4 アドレスの払い出しを行う。そのため、MNO 事業者側でプライベート IPv4 アドレス払い出しに対応している必要がある。

MVNO 事業者と MNO 事業者の間は冗長化されているケースが多く、動的ルーティングプロトコルを利用できない CGN では収容することができない。その場合、収容ルータより上位に CGN を導入することとなる。CGN 機器は図のように VRRP による冗長構成を取るとした。IPv4 通信は CGN でグローバルアドレスに変換されて再び収容ルータに戻り、

MVNO 事業者バックボーンを介してインターネットに接続される。IPv4 通信を CGN 経由とするために、収容ルータにて policy based routing(PBR)または Virtual Routing and Forwarding(VRF)が必要となる。

IPv6 通信は CGN 機器を経由せずに収容ルータから直接、MVNO 事業者バックボーンを介してインターネットに抜けることが可能である。MVNO 事業者は収容ルータおよびバックボーンルータの IPv6 対応をすればよい。しかし、IPv6 アドレスの払い出しは MNO 事業者にて行うため、MNO 事業者が IPv6 払い出し対応をしていなければならない。また、MNO 事業者のルータも IPv6 対応していなければならない。

以上より、レイヤ 3 接続の場合、NAT44+IPv6 モデルを適用するには、MNO 事業者側での対応が必要である。終端装置にてプライベート IPv4 アドレス+IPv6 アドレスの払い出しが可能となっているとともに、MNO 事業者のバックボーンも IPv6 対応していなければならない。対して、レイヤ 2 接続をしている MVNO 事業者は、自前で終端装置を持っていることから、MNO 事業者の対応に依存せずに、プライベート IPv4 アドレスや IPv6 の払い出しを行うことができる。

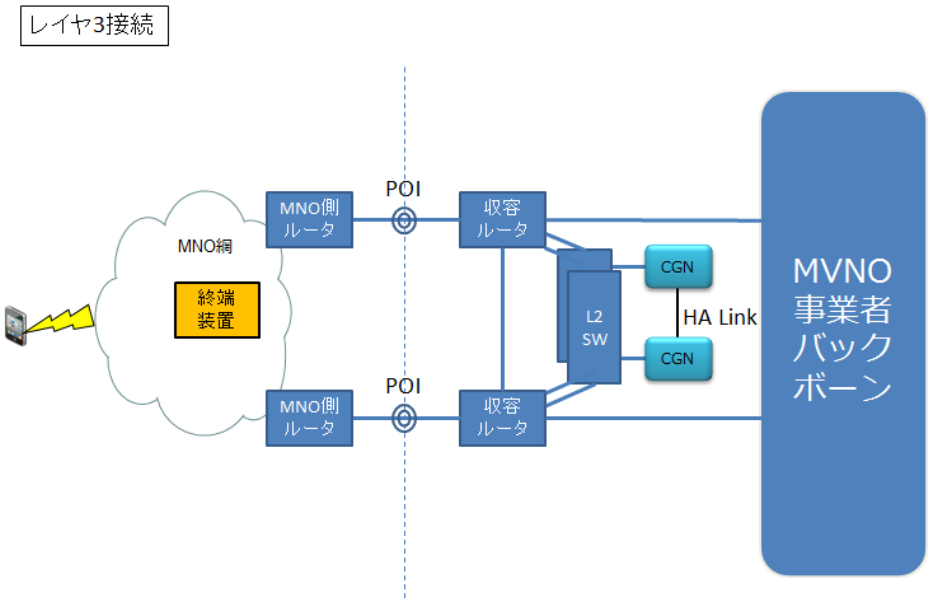


図 2-3　MVNO 事業者レイヤ 3 接続 CGN 構成例

レイヤ 2 接続においては、アドレス払い出しを MVNO 事業者の終端装置で行うため、MNO 事業者の制約がない。そのためレイヤ 2 接続であれば IPv6 対応していない MNO 事業者を利用した MVNO 事業者であっても IPv6 でのサービス提供が可能である。

NAT44+IPv6 モデルにおいては、MVNO 事業者の終端装置でプライベート IPv4 アドレスと IPv6 アドレスの払い出しを行う。インターネット向けの IPv4 通信は、図の集約ルータにて CGN 向けとなり、アドレス変換されて集約ルータに戻る。レイヤ 3 接続の場合と同

様に集約ルータにて PBR または VRF が必要となる。CGN 機器は図のように VRRP による冗長構成を取るとした。

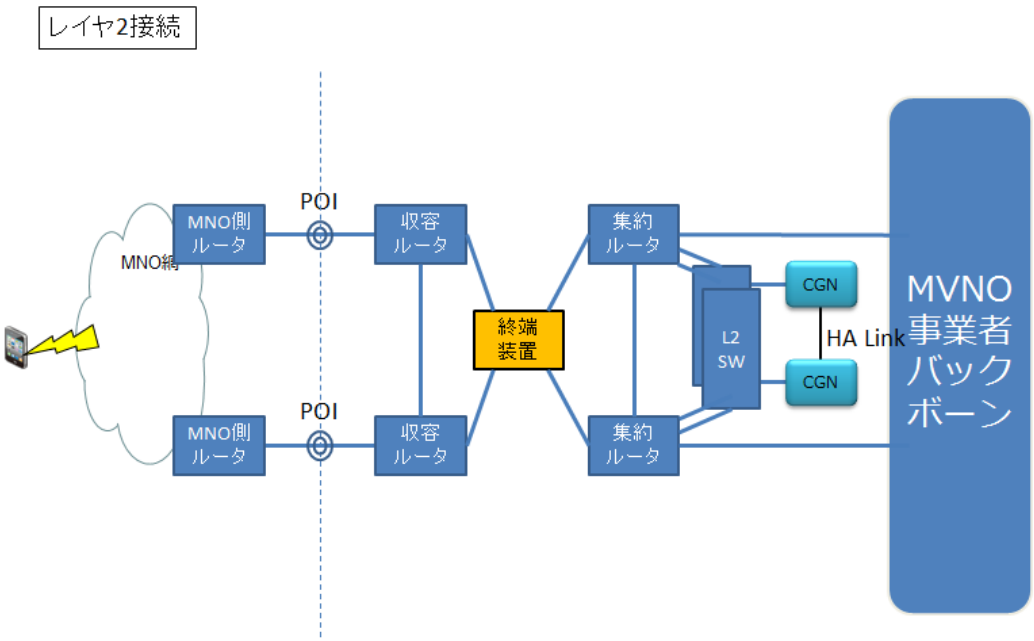


図 2-4　MVNO 事業者レイヤ 2 接続 CGN 構成例

以上の対応によりプライベート IPv4 通信のみ CGN を通過し、IPv6 通信は CGN の制約がないような構成が可能となる。ただし、モバイルサービスにおいては、インターネット到達性のサービスに追加して、帯域制御・オプティマイズ・フィルタリングなどの機能を実装しているケースが多い。それらが MVNO 事業者としてコアとなる機能である場合、IPv6 への移行をスムーズに行うためには、IPv6 と IPv4 で同等のサービス提供ができることが重要である。MNO 事業者と同様に、コアネットワークの IPv6 化と合わせて、付加サービスの IPv6 も同時に進めていくべきである。

2.3　モバイルネットワークにおけるセッション数

スマートフォン等で用いられる各種アプリケーションの特性（同時セッション数、通信期間・頻度等）を分析した結果を表 2-1 に示す。SPDY[17]対応により、Web サイト・アプリケーションごとのセッション数は 30%～70%程度に抑えられている。現在、モバイルネットワークにおいては 1 ユーザあたりのセッション数は以下のように推定される。

モバイル網における推定セッション数

- 平均セッション数：50 ポート /1 IPv4 アドレス
- 最大セッション数：500 ポート /1 IPv4 アドレス

表 2-1　モバイルアプリケーション・Web サイトの NAT エントリ数(TCP/UDP) および SPDY 対応状況

対象アプリケーション	iPhone		Android		SPDY 対応
	TCP	UDP	TCP	UDP	
Line	6	11	-	-	×
Skype	10	177	40	199	×
Google Map	6	0	6	0	△
Gmail	10	0	4	0	△
Google Search(Voice)	18	0	11	0	○
Twitter	11	0	19	0	△
Facebook MSG	10	0	10	0	△
Facebook App	38	0	22	0	○
Dropbox	8	0	21	0	×
Evernote	9	0	21	0	×
Puzzdra	10	0	5	0	×
Navitime	15	0	11	0	×
Apple.com	20	0	15	0	×
Mizuho bank	25	0	30	0	×
Amazon	26	0	28	0	×
iTunes store	43	0	-	-	×

[17] SPDY : Google 社が開発した、HTTP を改良した Web サーバーとユーザー間の通信プロトコル

Yahoo.co.jp	53	0	55	0	×
Rakuten	64	0	48	0	×
Youtube	25	0	32	0	△
Niconico Douga	69	0	30	0	×
Ustream	160	0	123	0	×

また、HTML5 conference (2013/11/30 開催)における観測(図 2-5)や Apricot2014 における CGN に関する Alcatel-Lucent 社の発表から、IPv4/IPv6 のデュアルスタック対応によって 30～40%の通信が IPv6 となると推測される。

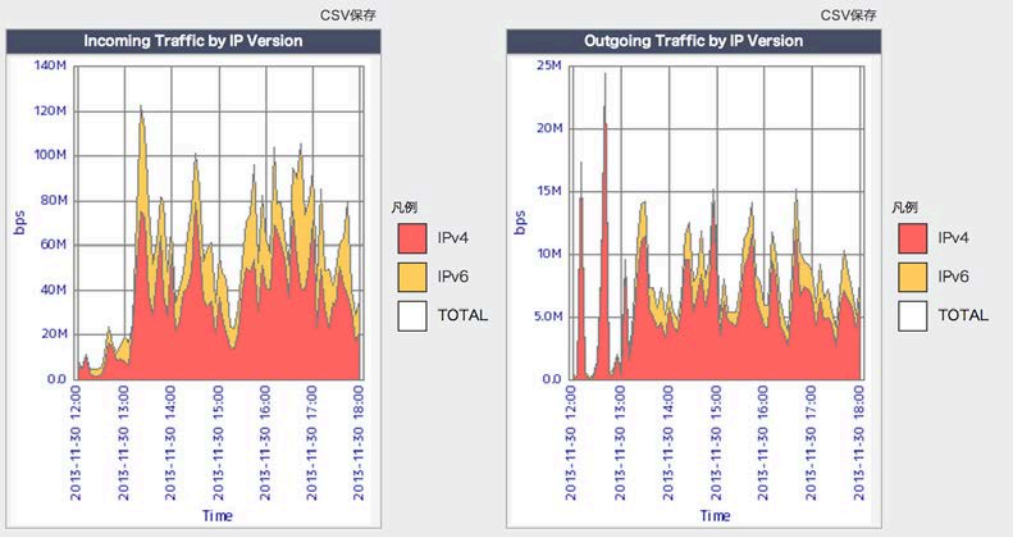


図 2-5　HTML5 Conference IPv4/IPv6 トラフィック推移

NAT44+IPv6 モデルにおけるセッション数を図 2-6 に示す。これより、30%～40%程度通信が IPv6 となっており、CGN における平均セッション数は 30 程度となっていることがわかる。

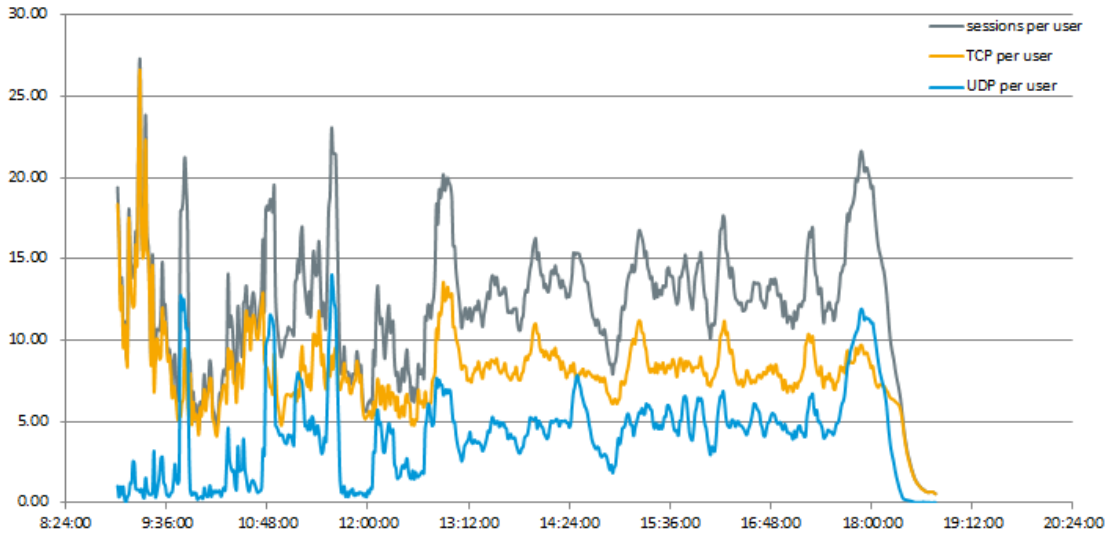


図 2-6　HTML5 Conference 1IPv4 アドレスあたりの平均セッション数

以上より、NAT44+IPv6 モデルにおける、CGN のセッション数(消費ポート数)は以下のように推定される。

モバイル網における推定セッション数(NAT44+IPv6 モデル)

- 平均セッション数：30 ポート / 1 IPv4 アドレス
- 最大セッション数：500 ポート / 1 IPv4 アドレス

最大セッション数が NAT44 モデルと変わらないのは、セッションを大量消費するアプリケーションが IPv6 対応をしていない可能性が高いからである。

2.4　推奨対応・対応手順

モバイルネットワークに導入した際の CGN に関連する推奨設定および A10 ネットワークス社 AX シリーズを用いた config 例を記載する。

2.4.1　セッション数に関する設定

NAT プールアドレスを公平に分配するために、CGN では 1 つの IPv4 アドレスが利用することができるセッション数の上限を決めるべきである。ただし、通常のアプリケーションにおいて問題が出ない上限とならなければならない。2.3 節の調査結果から、1 つの IPv4 アドレスあたり 500 セッションを上限と設定することを推奨する。

■1 つの IPv4 アドレスあたりのポート割当上限の設定
(設定コマンド)
lsn-lid 1
 source-nat-pool [POOL-NAME]
 user-quota icmp 500
 user-quota udp 500 reserve 0
 user-quota tcp 500 reserve 0

※1 ユーザ当たりの上限=500 ポートとしている。
reserve <num> は、事前に確保するポートの数である。
reserve 0 とすることで、動的ポートアサインとなる。

2.4.2 アドレスに関する設定

■NAT プールアドレス数設計

NAT プールアドレスの数は 1 つの IPv4 アドレスあたりの同時セッション数から設計する。

モバイル網における推定セッション数は、平均セッション数：50 ポート / 1IPv4 アドレスである。100 万 ID を払い出している場合、

1,000,000 * 50 = 50,000,000 [port]

必要となる。これは、

50,000,000 [port] / (65536– 1024) [port/アドレス] = 775 [アドレス]

に相当する。なお、0～1023 までは Wellknown ポートとして払い出しから除外されるため、1 つの IPv4 アドレスあたり使用可能なポート数は、65536 – 1024 = 64512 port となる。

■セッション数上限の設定
(設定コマンド)
system resource-usage l4-session-count [SESSION 上限数]
※セッション数上限の拡張が必要な場合に投入する

■NAT プールアドレス数の上限の設定
(設定コマンド)
system resource-usage nat-pool-addr-count [POOL ADDRESS 上限数]
※NAT プールアドレス数上限の拡張が必要な場合に投入する

■NAT プールアドレスの設定
(設定コマンド)
ip nat pool [POOL1] x.x.0.0 x.x.1.255 netmask /23　lsn ha-group-id 1 lsn
ip nat pool [POOL2] x.x.2.0 x.x.3.255 netmask /23　lsn ha-group-id 1 lsn
ip nat pool [POOL3] x.x.4.0 x.x.5.255 netmask /23　lsn ha-group-id 1 lsn
ip nat pool [POOL4] x.x.6.0 x.x.7.255 netmask /23　lsn ha-group-id 1 lsn
ip nat pool [POOL5] x.x.8.0 x.x.9.255 netmask /23　lsn ha-group-id 1 lsn
ip nat pool [POOL6] x.x.10.0 x.x.11.255 netmask /23　lsn ha-group-id 1 lsn
ip nat pool [POOL7] x.x.12.0 x.x.13.255 netmask /23　lsn ha-group-id 1 lsn
ip nat pool [POOL8] x.x.14.0 x.x.15.159 netmask /23　lsn ha-group-id 1 lsn

■ユーザ割当アドレス設計

ユーザの IPv4 アドレスは ISP 共有アドレス[RFC6598] 100.64.0.0/10 の中から払い出しを行うべきである。

■NAT 対象のアドレスの設定
(設定コマンド)
class-list [CLASS-NAME]
 100.64.0.0 /10 lsn-lid 1
ip nat inside source class-list [CLASS-NAME]

2.4.3　NAT タイムアウトに関する設定

RST または FIN が観測されていない TCP セッションや、UDP、ICMP の通信は一定時間 NAT テーブルを保持すべきである。TCP の場合、FIN が観測されてもアプリケーションの挙動を担保するために 2Maximum Segment Life(MSL)時間だけ NAT テーブルを保持してから NAT テーブルを解放すべきである。これらの NAT テーブル保持時間を変更可能である機器を選定することが望ましい。

NAT タイムアウト値を短くすることによって、TCP/UDP のポートをなるべく早く、多くの人数で使いまわすことができ、CGN の収容効率を高めることができるが、アプリケーションによっては放置しておくとタイムアウトにより接続が切れてしまうため、短い NAT タイムアウト値はネットワークの品質低下と見られてしまう場合がある。実際に NAT タイムアウト値を短くする際にはアプリケーションの挙動を阻害しないか注意深く行う必要がある。

UDP は TCP と異なり、FIN や RST などのセッションの終了を通知する仕組みがないため、必ず一定時間 NAT テーブルを保持しなければならない。特に、DNS トラフィックは

通信ごとに発生するため、DNS パケットのタイムアウト値が長いと CGN の性能に影響を与えてしまうことが検証で実証されている。それに対する対策として、DNS 通信については、ポート番号(UDP/53)に基づいてタイムアウト値小さくすることで性能への影響を抑えられることが実証された。また、DNS パケットについてのみ応答を見て NAT テーブルをクリアする CGN 機器も存在し、タイムアウト値を小さくする対策と同様に CGN の性能への影響を少なくできることが確認された。

特筆すべきこととして、モバイルアプリケーションにおいては、アプリケーションの更新の有無を確認するセッションが存在し、定期的に特定のパケットが通信されて、セッションを維持し続ける仕組みがある。この通信は、Facebook においては約 50 秒間隔であるが、Google Cloud Messaging では約 15 分となっている。そのため、CGN の導入によって、Google Cloud Messaging を利用したアプリケーション(グーグルハングアウト等)が利用不可になることが報告されている [18]。対策として、Google Cloud Messaging の使用ポート(TCP/5528～5530)のみ、NAT タイムアウト値を長くする方法が考えられる。CGN を導入している MNO 事業者・MVNO 事業者の一部では、実際に Google Cloud Messaging の使用ポートのタイムアウト値を長くしていることが観測された。

WebSocket や HTTP/2 などの次世代 Web 技術ではサーバからの PUSH 通知が規定されているが、PUSH の間隔が長い場合、その間に CGN の NAT テーブルのタイムアウトが発生してしまうと、通信影響が発生してしまう。そのため、今後もアプリケーションに合わせて TCP/443 番ポートなど特別な扱いが必要なポートが発生する可能性がある。

以上より、タイムアウト値の推奨設定と設定方法を記載する。

- TCP アイドル 300 秒
- UDP 300 秒
- DNS 3 秒
- ICMP 2 秒
- Google Cloud Messaging(TCP/5528～5530) 900 秒以上

■NAT タイムアウト値の確認方法 ax-cgn#show ip nat timeouts
(出力結果)
NAT Timeout values in seconds:
SYN　　TCP　　UDP　　ICMP

60　　300　　300　　fast
Service 53/udp has fast-aging configured

[18] https://groups.google.com/forum/#!topic/android-gcm/IC2ovQhgBJA

※以上はデフォルトであり、ICMP:2 秒、DNS:3 秒となっている。

■NAT タイムアウト値変更コマンド
(設定コマンド)
ip nat translation tcp-timeout [sec]
ip nat translation udp-timeout [sec]

■特定ポートのタイムアウト値変更コマンド
(設定コマンド)
ip nat translation service-timeout [tcp/udp] [#port] to [#port] [sec]
※例：Google Cloud Messaging の場合
ip nat translation service-timeout tcp 5528 to 5530 900

2.4.4 Full Cone NAT に関する設定

Endpoint Independent Mapping(EIM)とは、通信先が異なっていても、同一の送信元 IP アドレス/ポートに対しては同一のプール IP アドレス/ポートを割り当てる挙動である。

Endpoint Independent Filtering(EIF)とは、EIM によって作成された NAT テーブルにおけるプール IP アドレス/ポート向けの通信は、外部のどのホストからの通信でも受け入れる挙動である。EIM かつ EIF であるとき、Full Cone NAT と呼ばれる。Full Cone NAT は NAT テーブルが作成されていれば外部の全ての宛先からの通信を受け入れるため、P2P 通信が可能となる。

モバイル網において、LINE や Skype などのメッセンジャーアプリなど P2P 通信を行うアプリケーションの利用が広まっていることから、CGN の透過性を高めるために、可能な限り Full Cone NAT となっていることが望ましい。しかし、Full Cone NAT の NAT テーブルは TCP セッション終了後も保持する必要があるため、タイムアウト値を別に管理するために通常の NAT テーブルと別に保存される。そのため、Full Cone NAT 用の NAT テーブルの収容限界がある。検証した CGN 機器の一部では、全ての通信を Full Cone NAT とすると、この収容限界が原因で著しく性能が低下した。そのため、対策として Full Cone NAT とするポート番号の範囲を選択する必要がある。一般的に 1-1023 番の wellknown ポートについては主にサーバ-クライアント型の通信のため、Full Cone NAT としない対応が推奨される。実測により 1024 番以降のポートを利用した通信は、最大で全体の 60%程度のセッション数となることが確認された。そのため、全体の 60%程度の通信を収容できる Full Cone 用の NAT テーブルを保持できることが望ましい。

■1024 番ポート以降を Full Cone NAT とする設定
(設定コマンド)

ip nat lsn full-cone default
※上記の設定によりデフォルトで 1-1023 番の通信は Full Cone NAT とならない。

2.4.5 Application Level Gateway(ALG)に関する設定

CGN 機器の透過性を高めるために、可能な限り全ての ALG を有効化することが望ましい。しかし、ALG の有効化によって性能面でのインパクトがあることから、実際に有効化するかについては、事業者ごとに負荷状況に応じた判断が必要である。そのため、ALG の種類が多く、また ON/OFF が個別な機器を選定すべきである。

また、プライベート IPv4 アドレスを払い出すサービスのスペックとして、ALG を強いて有効化しない方策も考えられる。その場合、アプリケーションを救う方法として IPv6 への移行に誘導することが可能である。

■ALG をアプリケーションごとに enable にする設定
(設定コマンド)
ip nat lsn alg esp enable
ip nat lsn alg ftp enable
ip nat lsn alg pptp enable
ip nat lsn alg rstp enable
ip nat lsn alg sip enable
ip nat lsn alg tftp enable

2.4.6 ログに関する設定

事業者には、特定電気通信役務提供者の損害賠償責任の制限及び発信者情報の開示に関する法律(プロバイダ責任制限法)により、法的要請に基づいて発信者に関する情報を開示提供する義務が課せられている。しかし、CGN によってグローバル IPv4 アドレスを共有するケースでは、発信者のプライベート IPv4 アドレスを特定するために通信先のホストからの申告と NAT 変換ログを突合する必要が出てくる。

ポートの動的割り当てをしている場合には NAT テーブルの割当・消去のタイミングで、下記の内容のログを取得しなければならない。

- Timestamp
- 送信元 IP アドレスとポート番号
- NAT 変換後 送信元 IP アドレスとポート番号

ポートの静的割り当てをしている場合には、ユーザごとの NAT テーブルの割当が静的に決められているため、上記のログの取得が不要となる。

ただし、通信先ホストからの申告に送信元ポートの情報が含まれていない場合は、上記

の情報だけではユーザの特定が不可能なため、動的割り当てと静的割り当てのどちらの場合においても、

- 通信先 IP アドレスとポート番号

を取得しなければならない。しかし、現状では送信元ポートの情報の取得に対応しているアプリケーションサーバは少ない。そのため、実質上は通信先 IP アドレスとポート番号情報の取得は必須である。

静的割り当てにおいては、ユーザごとにセッション数上限と同じポート数(500 ポート)を静的に割り当てなければならないため、100 万 ID を払い出している場合、

1,000,000 * 500 = 500,000,000 [port]

必要となる。これは、

500,000,000 [port] / (65536– 1024) [port/アドレス] = 7751 [アドレス]

に相当する。動的割り当てと比較して 10 倍のプールアドレスが必要となる。

IPv4 アドレスの利用効率が高いことから、動的割り当ての設定とすることを推奨する。

■NAT ログにおける通信先情報の取得の設定
(設定コマンド)
```
ip nat template logging [TEMPLATE-NAME]
 include-destination
```

■NAT ログの形式の指定の設定
(設定コマンド)
```
ip nat template logging [TEMPLATE-NAME]
 format default
```

■NAT ログ取得サーバの指定の設定
(設定コマンド)
```
slb server [NAME01] [IPaddress]
    port 514   udp
slb server [NAME02] [IPaddress]
    port 514   udp
slb service-group [GROUP-NAME] udp
     member [NAME01]:514
     member [NAME02]:514
```
※以上の設定は 2 台の syslog サーバをグループ化し、分散して NAT ログを送付している。

以上、モバイルネットワークを想定し、CGN 導入の推奨対応と手順を記述した。

第3章　Vyatta ASAMAP を用いた小規模な 464 技術検証環境の構築手順

本文書では Vyatta ASAMAP を用いて以下の 四つのの 464 技術の動作検証を行う為の環境構築を行う手順について説明する。

- DS-Lite
- 464XLAT
- MAP-E
- MAP-T

DS-Lite は RFC6333（IETF「Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion」）[19] で仕様が定められている。カスタマ側装置(B4)とキャリア側装置(AFTR)で IPv4 パケットをカプセル化、カプセル化解除を行い、キャリア内ネットワークを IPv6 のみで構築することを可能とする技術である。MAP-E や MAP-T と異なりカスタマ側装置では NAPT 機能を持たず、キャリア側装置で NAPT 機能を持つ。

464XLAT は RFC6877(IETF「464XLAT: Combination of Stateful and Stateless Translation Abstract」) [20]で仕様が定められている。カスタマ側装置(CLAT)で IPv4 パケットを IPv6 パケットに変換(IP ヘッダ書き換え)し、キャリア側装置(PLAT)で IPv6 パケットを IPv4 パケットに変換する、いわゆるダブル・トランスレーションを行うことでキャリア内ネットワークを IPv6 のみで構築することを可能とする技術である。 DS-Lite と同様 カスタマ側装置では NAPT 機能を持たず、キャリア側装置で NAPT 機能を持つ。

MAP-E は draft-ietf-softwire-map(IETF「Mapping of Address and Port with Encapsulation (MAP)」)[21] で標準化が進められている。 DS-Lite と同じように、カスタマ側装置(CE)とキャリア側装置(BR)で IPv4 パケットをカプセル化、カプセル化解除を行い、キャリア内ネットワークを IPv6 のみで構築することを可能とする技術である。但し

---

[19] IETF「Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion」 http://tools.ietf.org/html/rfc6333
[20] IETF「464XLAT: Combination of Stateful and Stateless Translation Abstract」 http://tools.ietf.org/html/rfc6877
[21] IETF「Mapping of Address and Port with Encapsulation (MAP)」 http://tools.ietf.org/html/draft-ietf-softwire-map

DS-Lite とは異なり、キャリア側装置で NAPT 機能を持たず、カスタマ側装置で NAPT 機能を持つ。

MAP-T は draft-ietf-softwire-map-t(IETF「Mapping of Address and Port using Translation (MAP-T)」)[22] で標準化が進められている。464XLAT と同じように、カスタマ側装置(CE)で IPv4 パケットを IPv6 パケットに変換(IP ヘッダ書き換え)し、キャリア側装置(BR)で IPv6 パケットを IPv4 パケットに変換する、いわゆるダブル・トランスレーションを行うことでキャリア内ネットワークを IPv6 のみで構築することを可能とする技術である。但し 464XLAT とは異なり、キャリア側装置で NAPT 機能を持たず、カスタマ側装置で NAPT 機能を持つ。

『3.1 共通の設定項目 (31 ページ)』では各 464 技術に共通する設定項目について説明する。特にカスタマ側装置は仮想化環境で容易にスケールアウトさせることができるようアドレス体系を工夫しているため、その点を詳しく説明する。

『3.2 DS-Lite の環境構築(39 ページ)』では Vyatta ASAMAP を DS-Lite AFTR/B4 として検証環境を構築するための手順を説明する。

『3.3 464XLAT の環境構築 (47 ページ)』では Vyatta ASAMAP を 464XLAT PLAT/CLAT として検証環境を構築するための手順を説明する。

『3.4 MAP-E と MAP-T の環境構築 (57 ページ)』では Vyatta ASAMAP を MAP-E/MAP-T の BR/CE として検証環境を構築するための手順を説明する。 MAP-E と MAP-T は一つの設定パラメータが異なるだけでほとんど同じ設定内容となるため 一つの節で説明を行う。

	トランスレーション	トンネリング
キャリア側装置に NAPT 機能	464XLAT	DS-lite
カスタマ側装置に NAPT 機能	MAP-T	MAP-E

[22] IETF「Mapping of Address and Port using Translation (MAP-T)」
http://tools.ietf.org/html/draft-ietf-softwire-map-t

3.1　共通の設定項目

本文書で構築する検証環境の基本的な構成図を図 3-1 に示す。

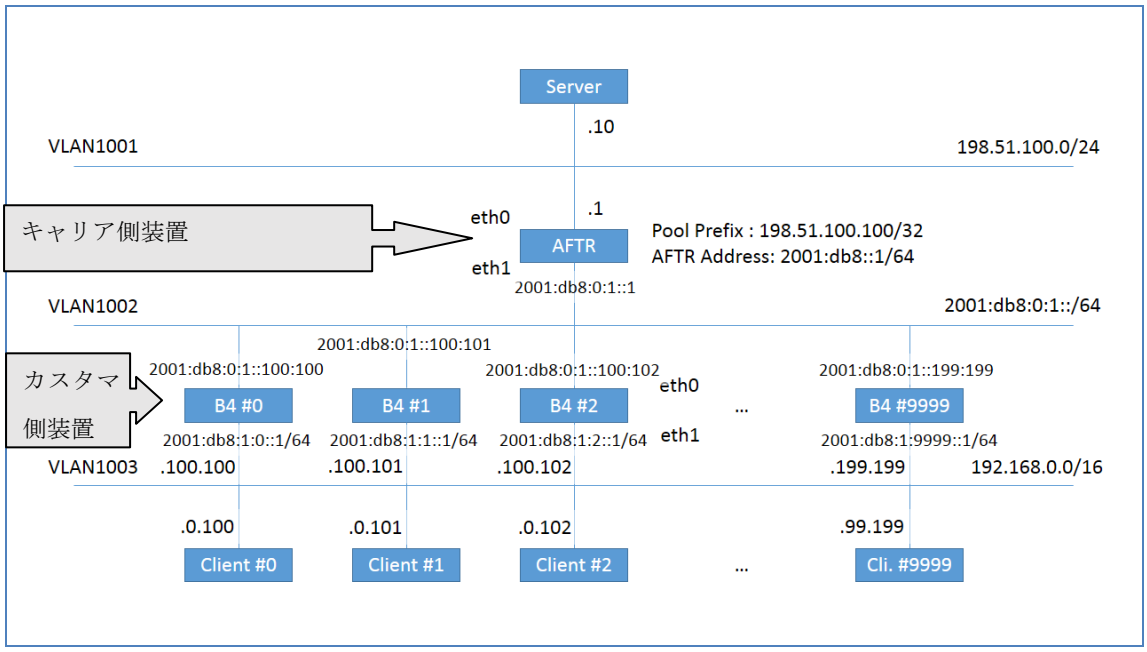


図 3-1　本文書で構築する検証環境の基本的な構成図

図 3-1 は DS-Lite の例となっているが以下の 3 点を除き、検証に必要な環境は他の 464 技術と同様である。

- カスタマ側装置とキャリア側装置の名称が異なること [23]
- カスタマ側が共有する IPv4 Prefix の名称が異なること [24]
- MAP-E と MAP-T には Rule IPv6 Prefix や EA-Bits Length といったパラメータが追加されていること

本構成ではカスタマ側装置に 0 から 9999 までの最大 10000 個の ID を振ることとする。

---

[23]名称が異なるとは、DS-Lite の場合は「AFTR」と「B4」となり、MAP-E および MAP-T の場合は「BR」と「CE」となるように、カスタマ側装置とキャリア側装置の名称がそれぞれの方式で異なることを表す。

[24]名称が異なるとは、DS-Lite や 464XLAT では「Pool Prefix」と呼称するが、MAP-E および MAP-T では「Rule IPv4 Prefix」と呼称するように、カスタマ側が共有する IPv4 Prefix の名称はそれぞれの方式で異なることを表す。

まず、本構成では以下の 3 つのネットワークを作成する。

- VLAN1001
  - ✧ インターネット側 IPv4 ネットワーク(198.51.100.0/24)
- VLAN1002
  - ✧ キャリア内 IPv6 ネットワーク(2001:db8:0:1::/64)
- VLAN1003
  - ✧ カスタマ内 IPv4 ネットワーク(192.168.0.0/16)
  - ✧ カスタマ内 IPv6 ネットワーク(カスタマ側装置毎に異なる 2001:db8:1:0::/64 から 2001:db8:1:9999::/64 までの 10000 個)
  - ✧ 本検証環境では VLAN1003 は 1 つの VLAN を共有する。

但し、カスタマ内 IPv6 ネットワークはトンネル終端アドレスや NAT46 変換のために全カスタマ側装置で異なる必要があるため別々のものを設定する。なお、本検証ではカスタマ側装置のカスタマ内 IPv6 ネットワークのプレフィクスはトンネル終端アドレスや NAT46 変換用のプレフィクスとしてのみ用いるものとする。

以下、設定方法について説明する。

3.1.1 キャリア側装置の設定内容

キャリア側装置のインターネット側 IPv4 ネットワークに接続されたインターフェースには 192.0.2.1/24 を設定する。

キャリア側装置のキャリア内 IPv6 ネットワークに接続されたインターフェースには 2001:db8:0:1::1/64 を設定する。

3.1.2 カスタマ側装置の設定内容

今回の検証に関しては、カスタマ側装置のアドレスは図 3-2 のように設定する。

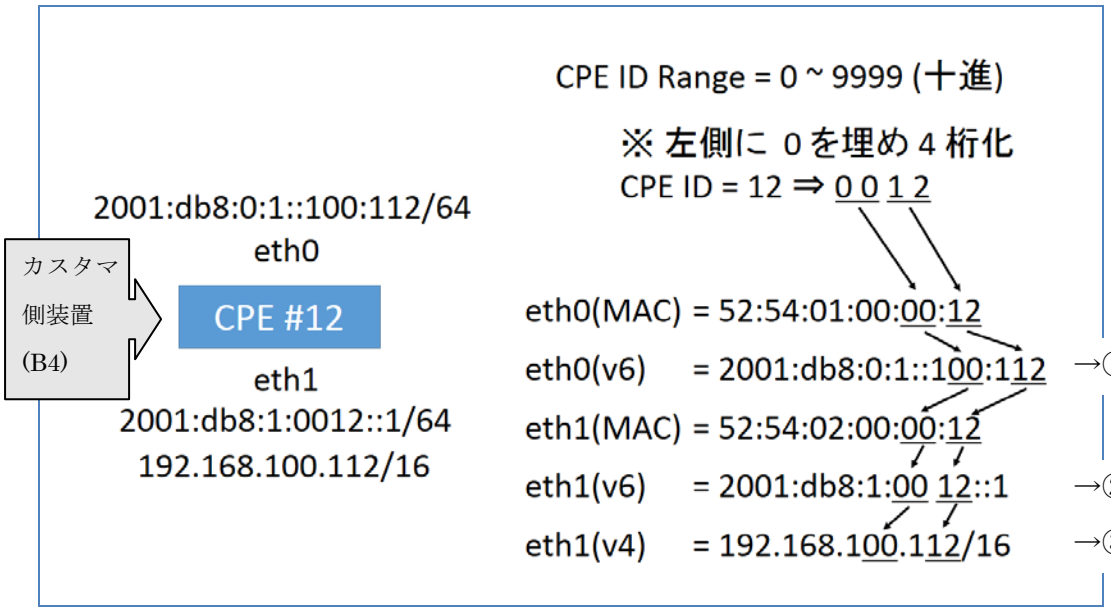


図 3-2　カスタマ側装置のアドレス設定例（CPE #12 の例）

まずカスタマ側装置の ID を文字列 4 文字となるよう左側に 0 を埋める（例: ID が 12 の場合は 0012 とする）。

① カスタマ側装置のキャリア内 IPv6 ネットワークに接続されたインターフェースには 2001:db8:0:1::/64 に :1(ID の前半 2 文字):1(ID の後半 2 文字) を最後の 4 オクテットに設定したものを用いる（例: ID が 12 の場合は 2001:db8:0:1::100:112/64 とする）。

② カスタマ側装置のカスタマ内 IPv6 ネットワーク側インターフェースには 2001:db8:1::/48 に :(ID 4 文字) を第 7 から第 8 オクテットに設定し、最後の 1 オクテットを 1 に設定したものを用いる（例: ID が 12 の場合は 2001:db8:1:12::1/64 とする）。

③ カスタマ側装置のカスタマ内 IPv4 ネットワーク側インターフェース 192.168.0.0/16 に .1(ID の前半 2 文字).1(ID の後半 2 文字) を設定したものを用いる（例: ID が 12 の場合は 192.168.100.112/16 とする）。

カスタマ側装置のキャリア内 IPv6 ネットワーク側インターフェースとカスタマ内ネットワーク側インターフェースの MAC アドレスも同様に図 3-2 のように計算し設定する。但し MAC アドレスの設定については 2 通り方法が考えられるため後述する。

3.1.3 MAC アドレスの設定

MAC アドレスの設定方法には以下の 2 通りの方法が考えられる。

- libvirtd [25]の設定ファイルの MAC アドレス欄を直接書き換える方法
- Vyatta のコマンドで MAC アドレスを書き換える方法

3.1.3.1 libvirtd の設定ファイルの MAC アドレス欄を直接書き換える方法

この方法は QEMU[26] を実行する libvirtd の設定ファイルの MAC アドレス欄を書き換えることで仮想 NIC の MAC アドレスを書き換える方法である。仮想化環境上のゲスト OS からみると仮想 NIC の MAC アドレスが書き換えられたものになる。

Vyatta インストール後にこの方法で MAC アドレスを書き換えた場合、仮想環境上のゲスト OS である Vyatta からみると、NIC が交換されたように認識されるため NIC の番号が再計算されるという不都合がある（例: eth0, eth1 として認識されていたものが MAC アドレス書き換え後に再起動すると eth2, eth3 として認識される等）。ただし Vyatta の設定で以下のコマンドを実行しておくことでこの問題を回避することは可能である。

```
# delete interface ethernet eth0 hw-id（古い MAC アドレス）
# set interface ethernet eth0 hw-id（新しい MAC アドレス）
```

後述する『Vyatta のコマンドで MAC アドレスを書き換える方法』とは異なり、この方法は virtio-net [27]仮想 NIC でも実現可能なためパケット転送性能が求められる場合はこちらの方式の方が好ましい。

具体的には以下の手順で書き換える。

1. 「virsh edit (VM の名前)」 を実行し libvirtd の設定ファイルの編集モードに入る
2. MAC アドレスを書き換えたい仮想 NIC の MAC アドレス欄を書き換える

例）

```
<domain type='kvm'>
```

[25] libvirtd：仮想マシン管理用の共通 API を提供するオープンソースのライブラリ
[26] QEMU：Linux, Windows, OS X に対応したオープンソースのエミュレータ
[27] virtio-net：準仮想化ドライバと呼ばれる仮想 NIC

```
…
    <interface type='bridge'>
      <mac address= ‘(この部分を新しい MAC アドレスに書き換える)’ />
      <source bridge='br'/>
      <model type='e1000'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
    </interface>
```

3.1.3.2 Vyatta のコマンドで MAC アドレスを書き換える方法

仮想 NIC が MAC アドレスの書き換えに対応する場合、 Vyatta で以下のコマンドを実行することで、一時的に MAC アドレスを変更することができる。これは仮想化環境上のゲスト OS からみると仮想 NIC の物理アドレスはそのままで異なる MAC アドレスをソフト的に設定するような意味となる。

ただしこの方法は仮想 NIC によってはサポートしておらず、この方法を用いたい場合は e1000(1000BASE-T NIC) を仮想 NIC として選択すべきである。

具体的には以下の手順で書き換える。

```
# set interface ethernet eth0 mac (新しい MAC アドレス)
```

本手順書では、カスタマ側装置の 仮想マシンをクローンしてインストールの手間を省くことを考慮し、『Vyatta のコマンドで MAC アドレスを書き換える方法』で MAC アドレスの設定を行う。

3.1.4 ICMP Redirect の送信抑制の設定

MAP-E と MAP-T の場合は必ず以下の設定を行う。

```
# set firewall send-redirects disable
```

MAP-E と MAP-T では同一の IP アドレスを NAPT を用いずに複数のカスタマ側装置で共有するため、上記の設定を行わないと誤った ICMP Redirect を送信してしまう場合がある。そのため ICMP Redirect の送信を無効化するために上記コマンドを実行する。

ただし DS-Lite と 464XLAT でも ICMP Redirect は不要と思われるため手順書上は全環境構築の際に上記のコマンドを実行するようにしている。

3.1.5 キャリア側装置のインターネット側インターフェースの Proxy ARP 機能の有効化

特に DS-Lite や 464XLAT といった(MAP-E や MAP-T と比較し)少ない IPv4 アドレスでサービスを提供可能な 464 技術の場合、キャリア側装置のインターネット側インターフェースに設定したネットワークに含まれるアドレスを NAPT のプールアドレスとして設定したいケースがある。

例えば図 3-1 の場合、キャリア側装置のインターネット側インターフェースには 198.51.100.1/24 を設定しているが プールには 192.51.100.0/24 内に含まれる 198.41.100.100/32 を設定しようとしている。

この場合 Server が Client への応答パケットを返そうとした際に 198.51.100.100 の ARP 解決を試みるが、Vyatta ASAMAP はそのままでは ARP 応答を行わないため、応答パケットを返すことができない。

このような構成で検証を行いたい場合は、以下のようにキャリア側装置のインターネット側インターフェースで Proxy ARP を行うよう設定する必要がある。

```
# set interface ethernet eth0 vif 1001 ip enable-proxy-arp
```

3.1.6 フラグメント処理時の最大 IPv6 パケットサイズの設定

464 技術では IPv4 パケットを IPv6 化する際に IPv6 の MTU に収まらなくなる可能性がある。その場合でかつ IPv4 の Don't Fragment フラグが設定されていない場合、何らかの形でフラグメント処理を行い IPv6 化する。

Vyatta ASAMAP では以下のコマンドにより IPv6 化する際の最大 IPv6 パケットサイズを設定することができる。

```
# set interfaces map map0 ipv6-fragment-size 1500
```

デフォルトでは IPv6 の最小 MTU サイズである 1280 となる。
本手順書では最大 IPv6 パケットサイズを 1500 に設定する。

3.1.7　カプセル化方式におけるフラグメント処理時のスタック指定の設定

カプセル化方式を採用する DS-Lite と MAP-E では IPv6 化してフラグメント処理を行う際に IPv6 スタック側でフラグメントを行う(IPv4 パケットはそのままで IPv6 fragment option header を用いて IPv6 でフラグメントを行う)か、IPv4 側でフラグメント処理を行い IPv6 パケットには IPv6 fragment option header がつかないようにするかの 2 通りの方法が考えられる。

Vyatta ASAMAP では以下のコマンドによりどちらのスタックでフラグメント処理を行うかを設定することができる。

```
# set interfaces map map0 ipv4-fragment-inner false
```

ipv4-fragment-inner を false に設定した場合、IPv6 スタック側でフラグメント処理が行われる。
ipv4-fragment-inner を true に設定した場合、IPv4 スタック側でフラグメント処理が行われる。

本手順書では IPv4 スタック側でフラグメント処理を行うよう統一している。

3.1.8　カスタマ側装置における TCP MSS オプション書き換えの設定

DS-Lite/464XLAT/MAP-E/MAP-T ではいちど IPv4 パケットをカプセル化したり IPv6 パケットに変換したりすることでキャリア内 IPv6 ネットワークを通過する。 IPv6 ヘッダの大きさはオプション無し IPv4 ヘッダと比較し 20 オクテット大きいため、場合によっては IPv4 から IPv6 にする際にフラグメントが必要になる可能性がある。

カスタマ側装置で TCP MSS オプションの書き換えを行うことでこの問題を回避することができる。

具体的にはカスタマ側装置で以下のコマンドを実行することで TCP SYN パケットの MSS オプション値が 1200 に書き換えられ TCP 通信についてはフラグメントの必要がないサイズでエンド・エンド間通信が行われるようになる。

```
# set policy route mssclamp rule 1 protocol tcp
# set policy route mssclamp rule 1 tcp flags SYN
# set policy route mssclamp rule 1 set tcp-mss 1200
```

```
# set interfaces ethernet eth1 vif 1003 policy route mssclamp
```

DS-Lite や MAP-E といったカプセル化方式の 464 技術の場合、 TCP MSS オプションを 1200 にすると最大 IPv6 パケットサイズは 1280 オクテットとなる。これは IPv4 の最小 MTU 値となるためどのようなキャリア内 IPv6 ネットワークにおいても確実にフラグメントが発生しない設定となる。

もしキャリア内 IPv6 ネットワークの MTU 値が 1500 であることが保証できる場合は TCP MSS オプションの値に 1420 を設定することで不要なフラグメントを回避しスループットが向上するする可能性がある。

464XLAT や MAP-T といったダブル・トランスレーション方式の 464 技術の場合、TCP MSS オプションを 1220 にすると最大 IPv6 パケットサイズは 1280 オクテットとなる。

同様にキャリア内 IPv6 ネットワークの MTU 値が 1500 であることが保証できる場合は TCP MSS オプションの値に 1440 を設定することで不要なフラグメントを回避しスループットが向上するする可能性がある。

本手順書では DS-Lite と MAP-E の場合に TCP MSS オプション値として 1420 を、464XLAT と MAP-T の場合に 1440 をそれぞれ用いる。

3.2　DS-Lite の環境構築

本節で構築する構成を図 3-3 に示す。

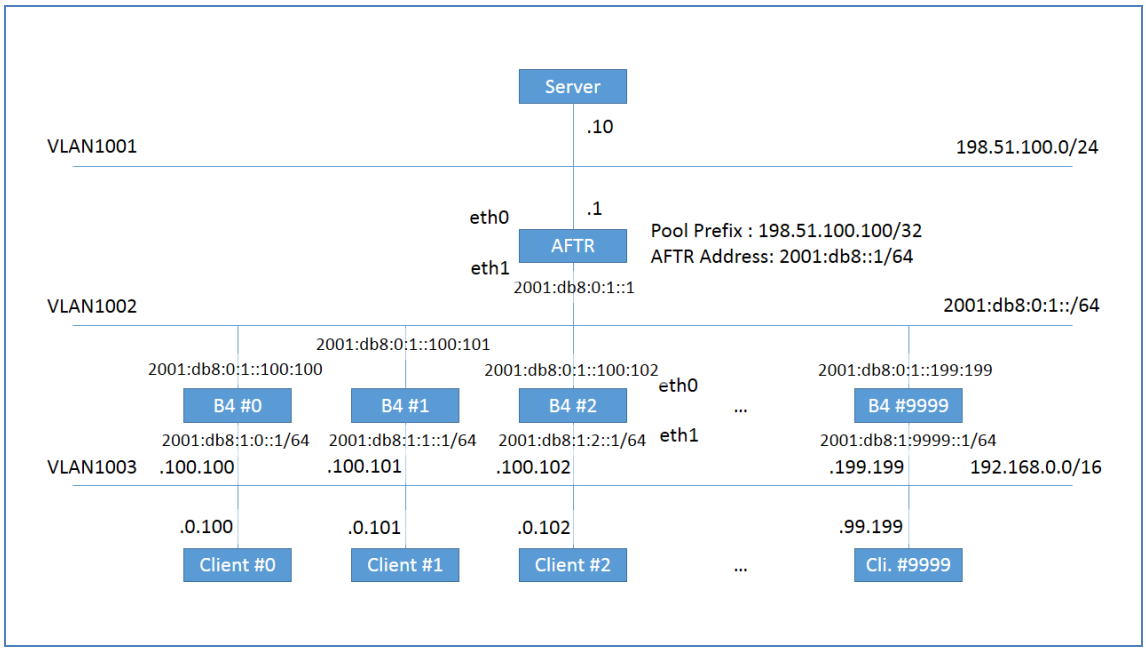


図 3-3　DS-Lite の環境構成

Client から Server へ通信を行う場合、まず Client からパケットを受け取った B4 は IPv4 ヘッダの情報を特に書き換えることなく IPv6 パケットにカプセル化し AFTR(2001:db8::1) へ転送する。

AFTR はカプセル化解除した後、 NAT44 処理を行い IPv4 ヘッダを書き換え Server へ転送する。

AFTR は Server からの返信パケットを NAT44 処理し、 IPv6 パケットにカプセル化し B4 へ転送する。

B4 はカプセル化解除した後、 Client へ転送する。
B4 の設定例として 図 3-3 の CPE #0 の設定を行う。

3.2.1 AFTR の設定

まず基本的な IP アドレスとルーティングの設定を行う。

```
[edit]
vyatta@vyatta# set interfaces ethernet eth0 vif 1001 address 198.51.100.1/24
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1002 address 2001:db8:0:1::1/64
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:0::/64 next-hop
2001:db8:0:1::100:100
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

上記では CPE#0 のための IPv6 ルーティングの設定を行っているが、以下のように同様に CPE の数だけ設定を行う必要がある。数が多いためスクリプトでコマンドを生成し一括で実行する。

```
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:0::/64 next-hop
2001:db8:0:1::100:100
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:1::/64 next-hop
2001:db8:0:1::100:101
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:2::/64 next-hop
2001:db8:0:1::100:102
...
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:9999::/64 next-hop
2001:db8:0:1::199:199
```

次に ICMP Redirect を無効化する。

```
[edit]
vyatta@vyatta# set firewall send-redirects disable
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次にインターネット側 IPv4 ネットワークで Proxy ARP を有効にする設定を行う。

```
[edit]
vyatta@vyatta# set interfaces ethernet eth0 vif 1001 ip enable-proxy-arp
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に AFTR の仮想インターフェース作成とパラメータ設定を行う。

```
[edit]
vyatta@vyatta# set interfaces map map0 role br
[edit]
vyatta@vyatta# set interfaces map map0 br-address 2001:db8::1/64
[edit]
vyatta@vyatta# set interfaces map map0 default-forwarding-mode encapsulation
[edit]
vyatta@vyatta# set interfaces map map0 pool 1 pool-prefix 198.51.100.100/32
[edit]
vyatta@vyatta# set interfaces map map0 ipv6-fragment-size 1500
[edit]
vyatta@vyatta# set interfaces map map0 ipv4-fragment-inner true
[edit]
vyatta@vyatta#   set   protocols   static   interface-route     198.51.100.100/32
```

```
next-hop-interface map0
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

上記の設定が完了したら AFTR としての設定は完了である。

以下のコマンドを実行しパラメータが正しく設定されていることを確認する。

```
[edit]
vyatta@vyatta# run show interfaces map map0

    Interface name          : map0
    Role                    : BR
    Tunnel source           : (null)
    BR address              : 2001:db8::1/64
    IPv4 pool               :
        IPv4 pool #0000         : 198.51.100.100/32
    Default forwarding mode : encapsulation
    Default forwarding rule : true
    IPv6 fragment size      : 1500
    IPv4 fragment inner     : true
    NAPT always             : true
    NAPT force recycle      : false
    Basic mapping rule      : (null)
    MAP IPv6 address        : 2001:db8::1/64
    Shared IPv4 address     : 0.0.0.0
    Assigned port-set ID    : 0x0/0
    Port-set                :
        Port-set #0000          :   4096(0x1000) -  65535(0xffff)

[edit]
vyatta@vyatta#
```

3.2.2 B4 の設定

B4 #0 を例に設定する。

まず eth0 と eth1 の MAC アドレスを変更する。

```
[edit]
vyatta@vyatta# set interfaces ethernet eth0 mac 52:54:01:00:00:00
[edit]
vyatta@vyatta# set interfaces ethernet eth1 mac 52:54:02:00:00:00
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に基本的な IP アドレスとルーティングの設定を行う。

```
[edit]
vyatta@vyatta# set interfaces ethernet eth0 vif 1002 address
2001:db8:0:1::100:100/64
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1003 address 2001:db8:1:0::1/64
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1003 address 192.168.100.100/16
[edit]
vyatta@vyatta# set protocols static route6 2001:db8::/64 next-hop 2001:db8:0:1::1
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に ICMP Redirect を無効化する。

```
[edit]
vyatta@vyatta# set firewall send-redirects disable
```

```
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に TCP MSS オプションを書き換える設定を行う。

```
[edit]
vyatta@vyatta# set policy route mssclamp rule 1 protocol tcp
[edit]
vyatta@vyatta# set policy route mssclamp rule 1 tcp flags SYN
[edit]
vyatta@vyatta# set policy route mssclamp rule 1 set tcp-mss 1420
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1003 policy route mssclamp
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に B4 の仮想インターフェース作成とパラメータ設定を行う。

```
[edit]
vyatta@vyatta# set interfaces map map0 role ce
[edit]
vyatta@vyatta# set interfaces map map0 tunnel-source eth1.1003
[edit]
vyatta@vyatta# set interfaces map map0 br-address 2001:db8::1/64
[edit]
vyatta@vyatta# set interfaces map map0 default-forwarding-mode encapsulation
[edit]
vyatta@vyatta# set interfaces map map0 ipv6-fragment-size 1500
[edit]
vyatta@vyatta# set interfaces map map0 ipv4-fragment-inner true
```

```
[edit]
vyatta@vyatta# set protocols static interface-route 0.0.0.0/0 next-hop-interface
map0
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

上記の設定が完了したら B4 としての設定は完了である。

以下のコマンドを実行しパラメータが正しく設定されていることを確認する。

```
[edit]
vyatta@vyatta# run show interfaces map map0

    Interface name          : map0
    Role                    : CE
    Tunnel source           : eth1.1003
    BR address              : 2001:db8::1/64
    IPv4 pool               :
    Default forwarding mode : encapsulation
    Default forwarding rule : true
    IPv6 fragment size      : 1500
    IPv4 fragment inner     : true
    NAPT always             : true
    NAPT force recycle      : false
    Basic mapping rule      : (null)
    MAP IPv6 address        : 2001:db8:1:0:c0:0:200:0/128
    Shared IPv4 address     : 0.0.0.0
    Assigned port-set ID    : 0x0/0
    Port-set                :
        Port-set #0000          :   4096(0x1000) -  65535(0xffff)

[edit]
vyatta@vyatta#
```

3.2.3　確認

Client から Server に対して ping や ssh 接続を実施してみて動作を確認する。

DS-Lite では AFTR が NAPT 処理を行う。 AFTR で以下のコマンドを実行することで現在の通信内容(NAPT テーブル・エントリの内容)を確認することができる。

```
[edit]
vyatta@vyatta# run show interfaces map map0 napt

 Proto: 'I' = ICMP, 'T' = TCP, 'U' = UDP.
 Flags: SynOut, SynAckIn, AckOut, FinOut, FinAckIn, FinIn, FinAckOut, Rst.
        '!' = Up, '.' = Down.

 Last used, Local IPv6 address, Local address:port, Mapped address:port, Remote
address:port, Proto, Flags.
 06:09:55 2001:db8:1:0:c0:0:200:0                              192.168.0.100:36257
198.51.100.100:(0xe261)57953    198.51.100.10:22    T !!!.....
 06:09:49 2001:db8:1:0:c0:0:200:0                               192.168.0.100:2201
198.51.100.100:(0xe9e0)59872    198.51.100.10:0     I ........

[edit]
vyatta@vyatta#
```

上記の例では 192.168.0.100 から 198.51.100.10 に対する ICMP と TCP/22 が行われていることがわかる。

3.3　464XLAT の環境構築

本節で構築する構成を図 3-4 に示す。

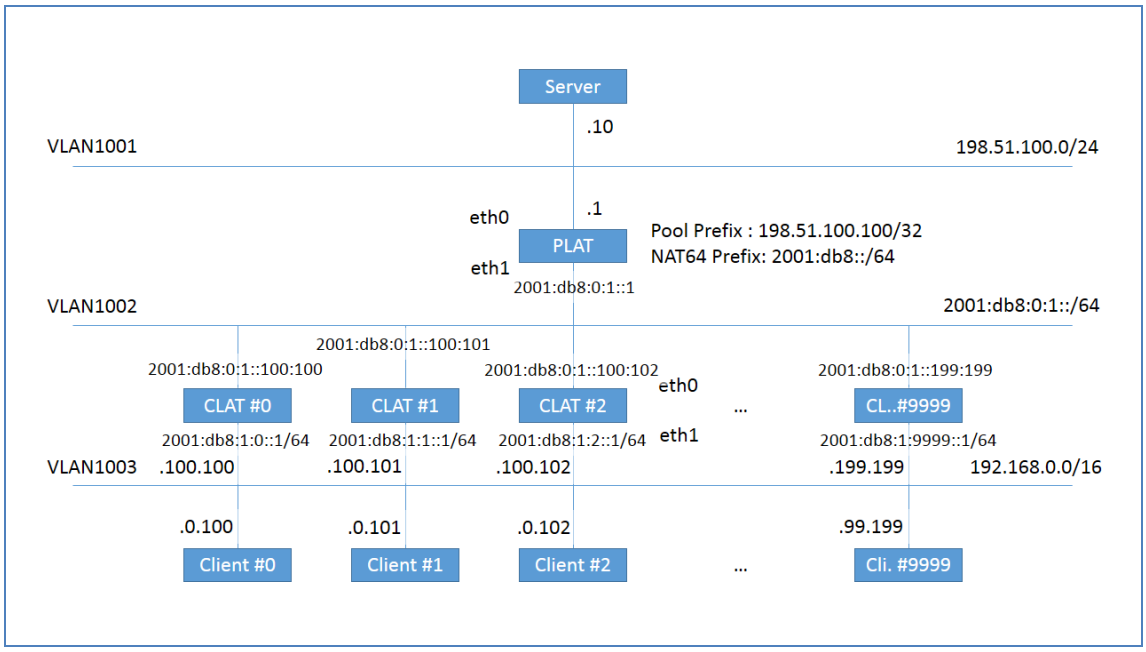


図 3-4　464XLAT の環境構成

Client から Server へ通信を行う場合、まず Client からパケットを受け取った CLAT は IPv4 ヘッダの情報元に IPv6 パケットに変換し PLAT の NAT64 プレフィクス (2001:db8::/64) へ転送する。

PLAT は NAT64 プレフィクス宛の IPv6 パケットを IPv4 パケットに変換し、NAT44 処理を行い IPv4 ヘッダを書き換え Server へ転送する。

PLAT は Server からの返信パケットを NAT44 処理し、 IPv6 パケットに変換し CLAT へ転送する。

CLAT は PLAT から受け取った IPv6 パケットを IPv4 パケットに変換した後、Client へ転送する。

CLAT の設定例として 図 3-4 の CPE #0 の設定を行う。

3.3.1 PLAT の設定

まず基本的な IP アドレスとルーティングの設定を行う。

```
[edit]
vyatta@vyatta# set interfaces ethernet eth0 vif 1001 address 198.51.100.1/24
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1002 address 2001:db8:0:1::1/64
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:0::/64 next-hop
2001:db8:0:1::100:100
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

上記では CPE#0 のための IPv6 ルーティングの設定を行っているが、以下のように同様に CPE の数だけ設定を行う必要がある。数が多いためスクリプトでコマンドを生成し一括で実行する。

```
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:0::/64 next-hop
2001:db8:0:1::100:100
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:1::/64 next-hop
2001:db8:0:1::100:101
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:2::/64 next-hop
2001:db8:0:1::100:102
...
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:9999::/64 next-hop
2001:db8:0:1::199:199
```

次に ICMP Redirect を無効化する。

```
[edit]
vyatta@vyatta# set firewall send-redirects disable
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次にインターネット側 IPv4 ネットワークで Proxy ARP を有効にする設定を行う。

```
[edit]
vyatta@vyatta# set interfaces ethernet eth0 vif 1001 ip enable-proxy-arp
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に PLAT の仮想インターフェース作成とパラメータ設定を行う。

```
[edit]
vyatta@vyatta# set interfaces map map0 role br
[edit]
vyatta@vyatta# set interfaces map map0 br-address 2001:db8::1/64
[edit]
vyatta@vyatta# set interfaces map map0 default-forwarding-mode translation
[edit]
vyatta@vyatta# set interfaces map map0 pool 1 pool-prefix 198.51.100.100/32
[edit]
vyatta@vyatta# set interfaces map map0 ipv6-fragment-size 1500
[edit]
vyatta@vyatta# set interfaces map map0 ipv4-fragment-inner true
[edit]
```

```
vyatta@vyatta#  set  protocols  static  interface-route     198.51.100.100/32
next-hop-interface map0
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

br-address で設定するプレフィクスが NAT64 プレフィクスとなる。

NAT64 プレフィクスの設定方法には /64 のほかに /96 を指定する場合がある。

/64 を指定した場合、 IPv4 から IPv6 へ変換を行う際、 IPv6 アドレスの Interface ID 部分(後ろの 64 bit) は以下のようになる。

- (8 bit の 0)(IPv4 アドレス)(24 bit の 0)

例えば IPv4 アドレスが 192.168.0.100 の場合 Interface ID は c0:a800:6400:0 となる。

/96 を指定した場合、 IPv4 から IPv6 へ変換を行う際、 IPv6 アドレスの Interface ID 部分(後ろの 64 bit) は以下のようになる。

- (16 bit の 0)(16 進数で 6464)(IPv4 アドレス)
  - CLAT 側の Interface ID(Vyatta ASAMAP 固有の値)
- (32 bit の 0)(IPv4 アドレス)
  - PLAT 側の Interface ID

例えば CLAT 側の IPv4 アドレスが 192.168.0.100 の場合 CLAT 側の Interface ID は 0:6464:c0a8:64 となる。

br-address で設定するプレフィクスで指定するプレフィクス長は PLAT と CLAT で統一されている必要がある。

上記の設定が完了したら PLAT としての設定は完了である。

以下のコマンドを実行しパラメータが正しく設定されていることを確認する。

```
[edit]
vyatta@vyatta# run show interfaces map map0

    Interface name          : map0
    Role                    : BR
    Tunnel source           : (null)
    BR address              : 2001:db8::1/64
    IPv4 pool               :
        IPv4 pool #0000         : 198.51.100.100/32
    Default forwarding mode : translation
    Default forwarding rule : true
    IPv6 fragment size      : 1500
    IPv4 fragment inner     : true
    NAPT always             : true
    NAPT force recycle      : false
    Basic mapping rule      : (null)
    MAP IPv6 address        : 2001:db8::1/64
    Shared IPv4 address     : 0.0.0.0
    Assigned port-set ID    : 0x0/0
    Port-set                :
        Port-set #0000          :   4096(0x1000) -  65535(0xffff)

[edit]
vyatta@vyatta#
```

3.3.2 CLAT の設定

CLAT #0 を例に設定する。

まず eth0 と eth1 の MAC アドレスを変更する。

```
[edit]
vyatta@vyatta# set interfaces ethernet eth0 mac 52:54:01:00:00:00
[edit]
vyatta@vyatta# set interfaces ethernet eth1 mac 52:54:02:00:00:00
```

```
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に基本的な IP アドレスとルーティングの設定を行う。

```
[edit]
vyatta@vyatta#    set    interfaces    ethernet    eth0    vif    1002    address
2001:db8:0:1::100:100/64
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1003 address 2001:db8:1:0::1/64
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1003 address 192.168.100.100/16
[edit]
vyatta@vyatta# set protocols static route6 2001:db8::/64 next-hop 2001:db8:0:1::1
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に ICMP Redirect を無効化する。

```
[edit]
vyatta@vyatta# set firewall send-redirects disable
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に TCP MSS オプションを書き換える設定を行う。

```
[edit]
```

```
vyatta@vyatta# set policy route mssclamp rule 1 protocol tcp
[edit]
vyatta@vyatta# set policy route mssclamp rule 1 tcp flags SYN
[edit]
vyatta@vyatta# set policy route mssclamp rule 1 set tcp-mss 1440
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1003 policy route mssclamp
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に CLAT の仮想インターフェース作成とパラメータ設定を行う。

```
[edit]
vyatta@vyatta# set interfaces map map0 role ce
[edit]
vyatta@vyatta# set interfaces map map0 tunnel-source eth1.1003
[edit]
vyatta@vyatta# set interfaces map map0 br-address 2001:db8::1/64
[edit]
vyatta@vyatta# set interfaces map map0 default-forwarding-mode translation
[edit]
vyatta@vyatta# set interfaces map map0 ipv6-fragment-size 1500
[edit]
vyatta@vyatta# set interfaces map map0 ipv4-fragment-inner true
[edit]
vyatta@vyatta# set protocols static interface-route 0.0.0.0/0 next-hop-interface
map0
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

br-address で設定するプレフィクスが NAT64 プレフィクスとなる。

NAT64 プレフィクスの設定方法には /64 のほかに /96 を指定する場合がある。

/64 を指定した場合、 IPv4 から IPv6 へ変換を行う際、 IPv6 アドレスの Interface ID 部分(後ろの 64 bit) は以下のようになる。

- (8 bit の 0)(IPv4 アドレス)(24 bit の 0)

例えば IPv4 アドレスが 192.168.0.100 の場合 Interface ID は c0:a800:6400:0 となる。/96 を指定した場合、 IPv4 から IPv6 へ変換を行う際、 IPv6 アドレスの Interface ID 部分(後ろの 64 bit) は以下のようになる。

- (16 bit の 0)(16 進数で 6464)(IPv4 アドレス)
  - CLAT 側の Interface ID(Vyatta ASAMAP 固有の値)
- (32 bit の 0)(IPv4 アドレス)
  - PLAT 側の Interface ID

例えば CLAT 側の IPv4 アドレスが 192.168.0.100 の場合 CLAT 側の Interface ID は 0:6464:c0a8:64 となる。

br-address で設定するプレフィクスで指定するプレフィクス長は PLAT と CLAT で統一されている必要がある。

上記の設定が完了したら CLAT としての設定は完了である。

以下のコマンドを実行しパラメータが正しく設定されていることを確認する。

```
[edit]
vyatta@vyatta# run show interfaces map map0

    Interface name          : map0
    Role                    : CE
    Tunnel source           : eth1.1003
    BR address              : 2001:db8::1/64
    IPv4 pool               :
    Default forwarding mode : translation
    Default forwarding rule : true
    IPv6 fragment size      : 1500
    IPv4 fragment inner     : true
    NAPT always             : true
```

```
    NAPT force recycle      : false
    Basic mapping rule      : (null)
    MAP IPv6 address        : 2001:db8:1::/72
    Shared IPv4 address     : 0.0.0.0
    Assigned port-set ID    : 0x0/0
    Port-set                :
        Port-set #0000          :   4096(0x1000) -  65535(0xffff)

[edit]
vyatta@vyatta#
```

3.3.3 確認

Client から Server に対して ping や ssh 接続を実施してみて動作を確認する。

464XLAT では PLAT が NAPT 処理を行う。 PLAT で以下のコマンドを実行することで現在の通信内容(NAPT テーブル・エントリの内容)を確認することができる。

```
[edit]
vyatta@vyatta# run show interfaces map map0 napt

 Proto: 'I' = ICMP, 'T' = TCP, 'U' = UDP.
 Flags: SynOut, SynAckIn, AckOut, FinOut, FinAckIn, FinIn, FinAckOut, Rst.
        '!' = Up, '.' = Down.

 Last used, Local IPv6 address, Local address:port, Mapped address:port, Remote
address:port, Proto, Flags.
 07:27:38  2001:db8:1:0:c0:a800:6400:0                          192.168.0.100:36259
198.51.100.100:(0xc53b)50491    198.51.100.10:22    T !!!.....
 07:25:34  2001:db8:1:0:c0:a800:6400:0                           192.168.0.100:2213
198.51.100.100:(0x1b99)7065     198.51.100.10:0     I ........

[edit]
vyatta@vyatta#
```

上記の例では 192.168.0.100 から 198.51.100.10 に対する ICMP と TCP/22 が行われていることがわかる。

3.4　MAP-E と MAP-T の環境構築

本節で構築する構成を図 3-5 に示す。

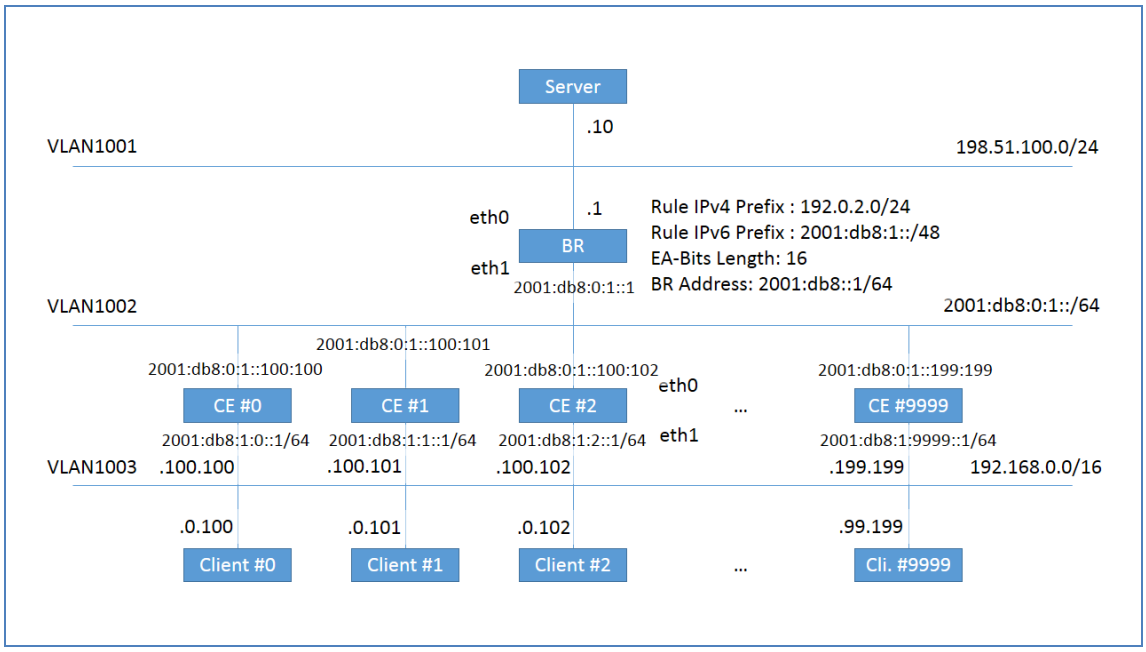


図 3-5　MAP-E と MAP-T の環境構成

Client から Server へ通信を行う場合、まず Client からパケットを受け取った CE は MAP Rule で割り当てられたポート番号を用いた NAT44 処理をし IPv4 ヘッダの情報を書き換え、 MAP-E の場合は IPv6 にカプセル化し、 MAP-T の場合は IPv6 へ変換し BR へ転送する。

BR はカプセル化解除(MAP-E の場合) or IPv4 へ変換(MAP-T の場合)した後 Server へ転送する。この際 DS-Lite の AFTR や 464XLAT の PLAT と異なり BR は NAPT 機能を適用しない。

BR は Server からの返信パケットを IPv6 パケットにカプセル化(MAP-E の場合) or IPv6 パケットに変換(MAP-T の場合)し CE へ転送する。

CE はカプセル化解除(MAP-E の場合) or IPv4 へ変換(MAP-T の場合)した後、NAT44 処理をし Client へ転送する。

CE の設定例として 図 3-5 の CPE #0 の設定を行う。

3.4.1 BR の設定

まず基本的な IP アドレスとルーティングの設定を行う。

```
[edit]
vyatta@vyatta# set interfaces ethernet eth0 vif 1001 address 198.51.100.1/24
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1002 address 2001:db8:0:1::1/64
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:0::/64 next-hop
2001:db8:0:1::100:100
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

上記では CPE#0 のための IPv6 ルーティングの設定を行っているが、以下のように同様に CPE の数だけ設定を行う必要がある。数が多いためスクリプトでコマンドを生成し一括で実行する。

```
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:0::/64 next-hop
2001:db8:0:1::100:100
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:1::/64 next-hop
2001:db8:0:1::100:101
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:2::/64 next-hop
2001:db8:0:1::100:102
...
[edit]
vyatta@vyatta# set protocols static route6 2001:db8:1:9999::/64 next-hop
2001:db8:0:1::199:199
```

次に ICMP Redirect を無効化する。

```
[edit]
vyatta@vyatta# set firewall send-redirects disable
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に BR の仮想インターフェース作成とパラメータ設定を行う(MAP-E の場合)。

```
[edit]
vyatta@vyatta# set interfaces map map0 role br
[edit]
vyatta@vyatta# set interfaces map map0 br-address 2001:db8::1/64
[edit]
vyatta@vyatta# set interfaces map map0 default-forwarding-mode encapsulation
[edit]
vyatta@vyatta# set interfaces map map0 rule 1 ea-length 16
[edit]
vyatta@vyatta# set interfaces map map0 rule 1 ipv4-prefix 192.0.2.0/24
[edit]
vyatta@vyatta# set interfaces map map0 rule 1 ipv6-prefix 2001:db8:1::/48
[edit]
vyatta@vyatta# set interfaces map map0 ipv6-fragment-size 1500
[edit]
vyatta@vyatta# set interfaces map map0 ipv4-fragment-inner true
[edit]
vyatta@vyatta# set protocols static interface-route 192.0.2.0/24 next-hop-interface
map0
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

MAP-T の場合は default-forwarding-mode を以下のように translation で設定する。

```
[edit]
vyatta@vyatta# set interfaces map map0 default-forwarding-mode translation
```

上記を設定したら BR としての設定は完了である。

以下のコマンドを実行しパラメータが正しく設定されていることを確認する(MAP-E の場合)。

```
[edit]
vyatta@vyatta# run show interfaces map map0

    Interface name          : map0
    Role                    : BR
    Tunnel source           : (null)
    BR address              : 2001:db8::1/64
    IPv4 pool               :
    Default forwarding mode : encapsulation
    Default forwarding rule : true
    IPv6 fragment size      : 1500
    IPv4 fragment inner     : true
    NAPT always             : true
    NAPT force recycle      : false
    Basic mapping rule      : (null)
    MAP IPv6 address        : 2001:db8::1/64
    Shared IPv4 address     : 0.0.0.0
    Assigned port-set ID    : 0x0/0
    Port-set                :
        Port-set #0000          :   4096(0x1000) -  65535(0xffff)

[edit]
vyatta@vyatta# run show interfaces map map0 rule
```

```
 Mode: 'E' = Encapsulation, 'T' = Translation. FMR: 'T' = FMR, '-' = Not FMR.

 IPv6 prefix, IPv4 prefix, PSID prefix, EA-bits length, PSID offset, Mode, FMR.
       0:              2001:db8:1::/48        192.0.2.0/24 0x0000/0  16  6 E F

[edit]
vyatta@vyatta#
```

MAP-T の場合は encapsulation と書かれた箇所が translation に置き換わって表示される。

3.4.2 CE の設定

CE #0 を例に設定する。

まず eth0 と eth1 の MAC アドレスを変更する。

```
[edit]
vyatta@vyatta# set interfaces ethernet eth0 mac 52:54:01:00:00:00
[edit]
vyatta@vyatta# set interfaces ethernet eth1 mac 52:54:02:00:00:00
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に基本的な IP アドレスとルーティングの設定を行う。

```
[edit]
vyatta@vyatta# set interfaces ethernet eth0 vif 1002 address
2001:db8:0:1::100:100/64
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1003 address 2001:db8:1:0::1/64
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1003 address 192.168.100.100/16
```

```
[edit]
vyatta@vyatta# set protocols static route6 2001:db8::/64 next-hop 2001:db8:0:1::1
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に ICMP Redirect を無効化する。

```
[edit]
vyatta@vyatta# set firewall send-redirects disable
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

次に TCP MSS オプションを書き換える設定を行う(MAP-E の場合)。

```
[edit]
vyatta@vyatta# set policy route mssclamp rule 1 protocol tcp
[edit]
vyatta@vyatta# set policy route mssclamp rule 1 tcp flags SYN
[edit]
vyatta@vyatta# set policy route mssclamp rule 1 set tcp-mss 1420
[edit]
vyatta@vyatta# set interfaces ethernet eth1 vif 1003 policy route mssclamp
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

MAP-T の場合は tcp-mss の値を以下のように設定する。

```
[edit]
vyatta@vyatta# set policy route mssclamp rule 1 set tcp-mss 1440
```

次に CE の仮想インターフェース作成とパラメータ設定を行う(MAP-E の場合)。

```
[edit]
vyatta@vyatta# set interfaces map map0 role ce
[edit]
vyatta@vyatta# set interfaces map map0 tunnel-source eth1.1003
[edit]
vyatta@vyatta# set interfaces map map0 br-address 2001:db8::1/64
[edit]
vyatta@vyatta# set interfaces map map0 default-forwarding-mode encapsulation
[edit]
vyatta@vyatta# set interfaces map map0 rule 1 ea-length 16
[edit]
vyatta@vyatta# set interfaces map map0 rule 1 ipv4-prefix 192.0.2.0/24
[edit]
vyatta@vyatta# set interfaces map map0 rule 1 ipv6-prefix 2001:db8:1::/48
[edit]
vyatta@vyatta# set interfaces map map0 ipv6-fragment-size 1500
[edit]
vyatta@vyatta# set interfaces map map0 ipv4-fragment-inner true
[edit]
vyatta@vyatta# set protocols static interface-route 0.0.0.0/0 next-hop-interface
map0
[edit]
vyatta@vyatta# commit
[edit]
vyatta@vyatta#
```

MAP-T の場合は default-forwarding-mode を以下のように translation で設定する。

```
[edit]
vyatta@vyatta# set interfaces map map0 default-forwarding-mode translation
```

上記の設定が完了したら CE としての設定は完了である。

以下のコマンドを実行しパラメータが正しく設定されていることを確認する(MAP-E の場合)。

```
[edit]
vyatta@vyatta# run show interfaces map map0

    Interface name          : map0
    Role                    : CE
    Tunnel source           : eth1.1003
    BR address              : 2001:db8::1/64
    IPv4 pool               :
    Default forwarding mode : encapsulation
    Default forwarding rule : true
    IPv6 fragment size      : 1500
    IPv4 fragment inner     : true
    NAPT always             : true
    NAPT force recycle      : false
    Basic mapping rule      :
        Rule IPv6 prefix        : 2001:db8:1::/48
        Rule IPv4 prefix        : 192.0.2.0/24
        Rule PSID prefix        : 0x0/0
        EA-bits length          : 16
        PSID offset             : 6
        Forwarding mode         : encapsulation
        Forwarding rule         : true
    MAP IPv6 address        : 2001:db8:1::c000:200:0/128
    Shared IPv4 address     : 192.0.2.0
    Assigned port-set ID    : 0x0/8
```

```
Port-set                :
    Port-set #0000      :    1024(0x0400) -    1027(0x0403)
    Port-set #0001      :    2048(0x0800) -    2051(0x0803)
    Port-set #0002      :    3072(0x0c00) -    3075(0x0c03)
    Port-set #0003      :    4096(0x1000) -    4099(0x1003)
    Port-set #0004      :    5120(0x1400) -    5123(0x1403)
    Port-set #0005      :    6144(0x1800) -    6147(0x1803)
    Port-set #0006      :    7168(0x1c00) -    7171(0x1c03)
    Port-set #0007      :    8192(0x2000) -    8195(0x2003)
    Port-set #0008      :    9216(0x2400) -    9219(0x2403)
    Port-set #0009      :   10240(0x2800) -   10243(0x2803)
    Port-set #0010      :   11264(0x2c00) -   11267(0x2c03)
    Port-set #0011      :   12288(0x3000) -   12291(0x3003)
    Port-set #0012      :   13312(0x3400) -   13315(0x3403)
    Port-set #0013      :   14336(0x3800) -   14339(0x3803)
    Port-set #0014      :   15360(0x3c00) -   15363(0x3c03)
    Port-set #0015      :   16384(0x4000) -   16387(0x4003)
    Port-set #0016      :   17408(0x4400) -   17411(0x4403)
    Port-set #0017      :   18432(0x4800) -   18435(0x4803)
    Port-set #0018      :   19456(0x4c00) -   19459(0x4c03)
    Port-set #0019      :   20480(0x5000) -   20483(0x5003)
    Port-set #0020      :   21504(0x5400) -   21507(0x5403)
    Port-set #0021      :   22528(0x5800) -   22531(0x5803)
    Port-set #0022      :   23552(0x5c00) -   23555(0x5c03)
    Port-set #0023      :   24576(0x6000) -   24579(0x6003)
    Port-set #0024      :   25600(0x6400) -   25603(0x6403)
    Port-set #0025      :   26624(0x6800) -   26627(0x6803)
    Port-set #0026      :   27648(0x6c00) -   27651(0x6c03)
    Port-set #0027      :   28672(0x7000) -   28675(0x7003)
    Port-set #0028      :   29696(0x7400) -   29699(0x7403)
    Port-set #0029      :   30720(0x7800) -   30723(0x7803)
    Port-set #0030      :   31744(0x7c00) -   31747(0x7c03)
    Port-set #0031      :   32768(0x8000) -   32771(0x8003)
    Port-set #0032      :   33792(0x8400) -   33795(0x8403)
    Port-set #0033      :   34816(0x8800) -   34819(0x8803)
    Port-set #0034      :   35840(0x8c00) -   35843(0x8c03)
```

```
        Port-set #0035          :  36864(0x9000) -  36867(0x9003)
        Port-set #0036          :  37888(0x9400) -  37891(0x9403)
        Port-set #0037          :  38912(0x9800) -  38915(0x9803)
        Port-set #0038          :  39936(0x9c00) -  39939(0x9c03)
        Port-set #0039          :  40960(0xa000) -  40963(0xa003)
        Port-set #0040          :  41984(0xa400) -  41987(0xa403)
        Port-set #0041          :  43008(0xa800) -  43011(0xa803)
        Port-set #0042          :  44032(0xac00) -  44035(0xac03)
        Port-set #0043          :  45056(0xb000) -  45059(0xb003)
        Port-set #0044          :  46080(0xb400) -  46083(0xb403)
        Port-set #0045          :  47104(0xb800) -  47107(0xb803)
        Port-set #0046          :  48128(0xbc00) -  48131(0xbc03)
        Port-set #0047          :  49152(0xc000) -  49155(0xc003)
        Port-set #0048          :  50176(0xc400) -  50179(0xc403)
        Port-set #0049          :  51200(0xc800) -  51203(0xc803)
        Port-set #0050          :  52224(0xcc00) -  52227(0xcc03)
        Port-set #0051          :  53248(0xd000) -  53251(0xd003)
        Port-set #0052          :  54272(0xd400) -  54275(0xd403)
        Port-set #0053          :  55296(0xd800) -  55299(0xd803)
        Port-set #0054          :  56320(0xdc00) -  56323(0xdc03)
        Port-set #0055          :  57344(0xe000) -  57347(0xe003)
        Port-set #0056          :  58368(0xe400) -  58371(0xe403)
        Port-set #0057          :  59392(0xe800) -  59395(0xe803)
        Port-set #0058          :  60416(0xec00) -  60419(0xec03)
        Port-set #0059          :  61440(0xf000) -  61443(0xf003)
        Port-set #0060          :  62464(0xf400) -  62467(0xf403)
        Port-set #0061          :  63488(0xf800) -  63491(0xf803)
        Port-set #0062          :  64512(0xfc00) -  64515(0xfc03)

[edit]
vyatta@vyatta# run show interfaces map map0 rule

 Mode: 'E' = Encapsulation, 'T' = Translation. FMR: 'T' = FMR, '-' = Not FMR.

 IPv6 prefix, IPv4 prefix, PSID prefix, EA-bits length, PSID offset, Mode, FMR.
      0:              2001:db8:1::/48        192.0.2.0/24 0x0000/0  16  6 E F
```

```
[edit]
vyatta@vyatta#
```

MAP-T の場合は encapsulation と書かれた箇所が translation に置き換わって表示される。

3.4.3 確認

Client から Server に対して ping や ssh 接続を実施してみて動作を確認する。

MAP-E や MAPT-T では CE が NAPT 処理を行う。 CE で以下のコマンドを実行することで現在の通信内容(NAPT テーブル・エントリの内容)を確認することができる。

```
[edit]
vyatta@vyatta# run show interfaces map map0 napt

 Proto: 'I' = ICMP, 'T' = TCP, 'U' = UDP.
 Flags: SynOut, SynAckIn, AckOut, FinOut, FinAckIn, FinIn, FinAckOut, Rst.
        '!' = Up, '.' = Down.

 Last used, Local address:port, Mapped port, Remote address:port, Proto, Flags.
 08:00:18   192.168.0.100:36260  13312(0x3400)   198.51.100.10:22    T !!!.....
 08:00:13   192.168.0.100:2216   24576(0x6000)   198.51.100.10:0     I ........

[edit]
vyatta@vyatta#
```

上記の例では 192.168.0.100 から 198.51.100.10 に対する ICMP と TCP/22 が行われていることがわかる。

第4章　NAT44 相当の IPv4/IPv6 グローバルアドレスに対するファイアウォールフィルタリングルールの設定例

本実証実験の請負事業にてクラウド事業者やオンラインゲーム事業者にインタビューを行った際、IPv4 では NAT（NAT44）を用いてセキュリティ運用の負荷を軽減しているが、IPv6 には NAT44 のように運用を行えないため、NAT44 相当のファイアウォールのフィルタリングルールの推奨設定例を公開して欲しいとの要望があった。そこで、本実証実験の請負事業では NAT44 相当のフィルタリングルールの推奨設定例の検討を行った。NAT44 相当の IPv6 ファイアウォールの設定を検討するにあたり、奈良先端科学技術大学院大学のファイアウォールのルールを参考にした。奈良先端科学技術大学院大学では IPv4 global / IPv6 global のデュアルスタック環境を提供しており、ファイアウォールのルールを IPv4 と IPv6 に対して基本的に同じポリシーで設計している。外部からの IPv4/IPv6 アクセスは DMZ セグメントのみを許しているというポリシーのため、ポリシーとしては IPv4 NAT の仕様から達成されている、外部からのアクセス遮断を実現しているファイアウォールでのフィルタリングルールとなっている。そこで juniper 社のファイアウォールルールを基にした、NAT 相当の IPv4 global / IPv6 global におけるフィルタリングルールの推奨設定の一例を掲載したい。

4.1　設定例

以下に、フィルタリングルールの推奨設定の一例を示す。

```
firewall {
    family inet {
        filter internal-in {
            term accept-to-dmz-from-external {
                from {
                    source-address {
                        0.0.0.0/0;
                        0.0.0.0/8 except;
                        10.0.0.0/8 except;
```

```
                127.0.0.0/8 except;
                169.254.0.0/16 except;
                172.16.0.0/12 except;
                192.168.0.0/16 except;
                224.0.0.0/3 except;
            }
            destination-address {
                133.4.1.0/24;
            }
        }
        then accept;
    }
    term discard-invalid-source-address {
        from {
            source-address {
                0.0.0.0/0;
                133.1.0.0/16 except;
            }
        }
        then {
            count discard-invalid-source-address;
            log;
            discard;
        }
    }
    term log-prohibited-fragment {
        from {
            fragment-offset 1;
        }
        then {
            count log-prohibited-fragment;
            log;
            next term;
        }
    }
    term accept-last {
```

```
            then accept;
        }
    }
    filter controller-in {
        term accept-trusted-networks {
            from {
                source-address {
                    133.1.0.0/16;
                    /* control-pc1 */
                    192.168.0.11/32;
                    /* control-pc2 */
                    192.168.0.250/32;
                }
            }
            then accept;
        }
        term reject-udp {
            from {
                protocol udp;
                destination-port 0-49151;
            }
            then {
                count reject-udp;
                log;
                reject;
            }
        }
        term reject-tcp {
            from {
                protocol tcp;
            }
            then {
                count reject-tcp;
                log;
                reject tcp-reset;
            }
```

```
        }
        term accept-last {
            then accept;
        }
    }
    filter external-in {
        term discard-invalid-source-address {
            from {
                source-address {
                    0.0.0.0/8;
                    10.0.0.0/8;
                    127.0.0.0/8;
                    169.254.0.0/16;
                    172.16.0.0/12;
                    192.168.0.0/16;
                    224.0.0.0/3;
                }
            }
            then {
                count discard-invalid-source-address;
                discard;
            }
        }
        term discard-internal-source-address {
            from {
                source-address {
                    133.1.0.0/16;
                }
            }
            then {
                count discard-internal-source-address;
                discard;
            }
        }
        term discard-icmp-directed-broadcast {
            from {
```

```
            destination-prefix-list {
                directed-broadcast;
            }
            protocol icmp;
        }
        then {
            count discard-icmp-directed-broadcast;
            discard;
        }
    }
    term log-prohibited-fragment {
        from {
            fragment-offset 1;
        }
        then {
            count log-prohibited-fragment;
            log;
            next term;
        }
    }
    term reject-udp-prohibited-services-to-all {
        from {
            protocol udp;
            destination-port [ 161 162 111 ];
        }
        then {
            count reject-udp-prohibited-services-to-all;
            reject;
        }
    }
    term reject-tcp-prohibited-services-to-all {
        from {
            protocol tcp;
            destination-port [ 111 135 ];
        }
        then {
```

```
            count reject-tcp-prohibited-services-to-all;
            reject;
        }
    }
    term accept-to-dmz {
        from {
            destination-address {
                133.1.6.0/24;
            }
        }
        then accept;
    }
    term accept-from-dmz-to-external {
        from {
            source-address {
                133.1.6.0/24;
            }
            destination-address {
                0.0.0.0/0;
                133.1.0.0/16 except;
            }
        }
        then accept;
    }
    term accept-icmp {
        from {
            protocol icmp;
        }
        then accept;
    }
    term accept-from-stepstone-server {
        from {
            source-address {
                133.1.10.10/32;
            }
            protocol tcp;
```

```
            destination-port [ 20 21 22 23 25 110 513 6667 ];
        }
        then accept;
    }
    term accept-dns {
        from {
            destination-address {
                133.1.6.13/32;
            }
            protocol udp;
            destination-port 53;
        }
        then accept;
    }
    term accept-dns-response {
        from {
            destination-address {
                133.1.6.13/32;
            }
            protocol udp;
            source-port 53;
            destination-port 1024-65535;
        }
        then accept;
    }
    term accept-ntp {
        from {
            destination-address {
                133.1.6.13/32;
            }
            protocol udp;
            destination-port 123;
        }
        then accept;
    }
    term accept-ntp-from-dmz {
```

```
        from {
            source-address {
                133.1.0.0/16;
            }
            protocol udp;
            destination-port 123;
        }
        then accept;
    }
    term accept-ntp-response-from-public-servers {
        from {
            source-address {
                133.243.238.163/32;
                133.243.238.164/32;
                133.243.238.243/32;
                133.243.238.244/32;
                210.171.226.40/32;
                /* ntp1.jst.mfeed.ad.jp */
                210.173.160.27/32;
                /* ntp2.jst.mfeed.ad.jp */
                210.173.160.57/32;
                /* ntp3.jst.mfeed.ad.jp */
                210.173.160.87/32;
                /* time.apple.com nwk */
                17.151.16.0/26;
                /* time.apple.com st1 */
                17.171.4.0/26;
                /* time.asia.apple.com sng */
                17.82.253.7/32;
                /* time.asia.apple.com tok */
                17.83.253.7/32;
                /* time.euro.apple.com */
                17.72.148.48/28;
            }
            protocol udp;
            source-port 123;
```

```
            destination-port [ 123 49152-65535 ];
        }
        then {
            count accept-ntp-response-from-public-servers;
            accept;
        }
    }
    term reject-udp-prohibited-services {
        from {
            protocol udp;
            destination-port [ 0-1023 1434 1900 2049 ];
        }
        then {
            count reject-udp;
            reject;
        }
    }
    term accept-udp-last {
        from {
            protocol udp;
        }
        then accept;
    }
    term accept-tcp-established {
        from {
            protocol tcp;
            tcp-established;
        }
        then accept;
    }
    term accept-ident {
        from {
            protocol tcp;
            destination-port 113;
        }
        then {
```

```
            count accept-ident;
            accept;
        }
    }
    term accept-mail {
        from {
            destination-address {
                13.1.6.72/32;
            }
            protocol tcp;
            destination-port smtp;
        }
        then accept;
    }
    term accept-to-webmail-and-imap {
        from {
            destination-address {
                133.1.6.150/32;
            }
            protocol tcp;
            destination-port [ https 993 995 ];
        }
        then accept;
    }
    term accept-to-smtp-over-tls {
        from {
            destination-address {
                133.1.6.73/32;
            }
            protocol tcp;
            destination-port 587;
        }
        then accept;
    }
    term accept-tcp-dns {
        from {
```

```
            destination-address {
                133.1.6.13/32;
            }
            protocol tcp;
            destination-port 53;
        }
        then accept;
    }
    term accept-www-http {
        from {
            destination-prefix-list {
                www-http-servers;
            }
            protocol tcp;
            destination-port http;
        }
        then accept;
    }
    term accept-www-https {
        from {
            destination-prefix-list {
                www-https-servers;
            }
            protocol tcp;
            destination-port https;
        }
        then accept;
    }
    term reject-tcp {
        from {
            protocol tcp;
        }
        then {
            count reject-tcp;
            log;
            reject;
```

```
            }
        }
        term accept-last {
            then accept;
        }
    }
    filter external-out {
        term accept-lsp-ping {
            from {
                destination-address {
                    127.0.0.0/8;
                }
                protocol udp;
                destination-port 3503;
            }
            then {
                count accept-lsp-ping;
                accept;
            }
        }
        term reject-invalid-destination-address {
            from {
                destination-address {
                    0.0.0.0/8;
                    10.0.0.0/8;
                    127.0.0.0/8;
                    169.254.0.0/16;
                    172.16.0.0/12;
                    192.168.0.0/16;
                    240.0.0.0/4;
                }
            }
            then {
                count reject-invalid-destination-address;
                log;
                reject network-prohibited;
```

```
        }
    }
    term accept-dns-response {
        from {
            source-address {
                133.1.6.13/32;
            }
            protocol udp;
            source-port 53;
        }
        then accept;
    }
    term reject-udp-prohibited-services {
        from {
            source-address {
                133.1.0.0/16;
            }
            protocol udp;
            destination-port [ 7 19 69 109 111 137-139 167 177 2049 ];
        }
        then {
            count reject-udp;
            log;
            reject;
        }
    }
    term reject-tcp-prohibited-services {
        from {
            source-address {
                133.1.0.0/16;
            }
            protocol tcp;
            destination-port [ 135 137-139 445 ];
            tcp-initial;
        }
        then {
```

```
                count reject-tcp;
                log;
                reject;
            }
        }
        term accept-last {
            then accept;
        }
    }
}
family inet6 {
    filter controller-in6 {
        term accept-trusted-networks {
            from {
                source-address {
                    fe80::/64;
                    2001:df0:2ee::/48;
                }
            }
            then accept;
        }
        term reject-udp {
            from {
                next-header udp;
                destination-port 0-49151;
            }
            then {
                count reject-udp;
                log;
                reject;
            }
        }
        term reject-tcp {
            from {
                next-header tcp;
            }
```

```
            then {
                count reject-tcp;
                log;
                reject tcp-reset;
            }
        }
        term accept-last {
            then accept;
        }
    }
    filter internal-in6 {
        term discard-invalid-source-address {
            from {
                source-address {
                    ::/0;
                    ::/128 except;
                    fe80::/64 except;
                    2001:df0:2ee::/48 except;
                }
            }
            then {
                count discard-invalid-source-address;
                log;
                discard;
            }
        }
        term accept-last {
            then accept;
        }
    }
    filter external-in6 {
        term discard-invalid-source-address {
            from {
                source-address {
                    ::/96;
                    ::/128 except;
```

```
                ::ffff:0.0.0.0/96;
                ff00::/8;
            }
        }
        then {
            count discard-invalid-source-address;
            discard;
        }
    }
    term discard-internal-source-address {
        from {
            source-address {
                2001:df0:2ee::/48;
            }
        }
        then {
            count discard-internal-source-address;
            discard;
        }
    }
    term reject-udp-prohibited-services-to-all {
        from {
            next-header udp;
            destination-port [ 111 161 162 ];
        }
        then {
            count reject-udp-prohibited-services-to-all;
            reject;
        }
    }
    term reject-tcp-prohibited-services-to-all {
        from {
            next-header tcp;
            destination-port [ 111 135 ];
        }
        then {
```

```
            count reject-tcp-prohibited-services-to-all;
            reject;
        }
    }
    term accept-to-dmz {
        from {
            destination-address {
                2001:df0:2ee::/64;
            }
        }
        then accept;
    }
    term accept-from-dmz-to-external {
        from {
            source-address {
                2001:df0:2ee::/64;
            }
            destination-address {
                ::/0;
                2001:df0:2ee::/48 except;
            }
        }
        then accept;
    }
    term accept-icmp {
        from {
            next-header icmp;
        }
        then accept;
    }
    term accept-from-stepstone {
        from {
            source-address {
                2001:df0:2ee:a::10/128;
            }
            next-header tcp;
```

```
            destination-port 22;
        }
        then accept;
    }
    term accept-dns {
        from {
            destination-address {
                2001:df0:2ee:50::13/128;
            }
            next-header udp;
            destination-port 53;
        }
        then accept;
    }
    term accept-dns-response {
        from {
            destination-address {
                2001:df0:2ee:50::13/128;
            }
            next-header udp;
            source-port 53;
            destination-port 1024-65535;
        }
        then accept;
    }
    term accept-ntp {
        from {
            destination-address {
                2001:df0:2ee:50::13/128;
            }
            next-header udp;
            destination-port 123;
        }
        then accept;
    }
    term accept-ntp-from-dmz {
```

```
        from {
            source-address {
                2001:df0:2ee::/48;
            }
            next-header udp;
            destination-port 123;
        }
        then accept;
    }
    term accept-ntp-response-from-public-servers {
        from {
            source-address {
                2001:2f8:29:100::fff3/128;
                2001:2f8:29:100::fff4/128;
                2400:3000:20:100::40/128;
                /* ntp1.v6.mfeed.ad.jp */
                2001:3a0:0:2001::27:123/128;
                /* ntp2.v6.mfeed.ad.jp */
                2001:3a0:0:2005::57:123/128;
                /* ntp3.v6.mfeed.ad.jp */
                2001:3a0:0:2006::87:123/128;
            }
            next-header udp;
            source-port 123;
        }
        then {
            count accept-ntp-response-from-public-servers;
            accept;
        }
    }
    term reject-udp-prohibited-services {
        from {
            next-header udp;
            destination-port [ 0-1023 1434 1900 2049 ];
        }
        then {
```

```
            count reject-udp;
            reject;
        }
    }
    term accept-udp-last {
        from {
            next-header udp;
        }
        then accept;
    }
    term accept-tcp-established {
        from {
            next-header tcp;
            tcp-established;
        }
        then accept;
    }
    term accept-ident {
        from {
            next-header tcp;
            destination-port 113;
        }
        then {
            count accept-ident;
            accept;
        }
    }
    term accept-to-webmail-and-imap {
        from {
            destination-address {
                2001:df0:2ee:50::50/128;
            }
            next-header tcp;
            destination-port [ https 993 995 ];
        }
        then accept;
```

```
}
term accept-to-smtp-over-tls {
    from {
        destination-address {
            2001:df0:2ee:50::73/128;
        }
        next-header tcp;
        destination-port 587;
    }
    then accept;
}
term accept-tcp-dns {
    from {
        destination-address {
            2001:df0:2ee:50::13/128;
        }
        next-header tcp;
        destination-port 53;
    }
    then accept;
}
term accept-www-http {
    from {
        destination-prefix-list {
            www6-http-servers;
        }
        next-header tcp;
        destination-port http;
    }
    then accept;
}
term accept-www-https {
    from {
        destination-prefix-list {
            www6-https-servers;
        }
```

```
                next-header tcp;
                destination-port https;
            }
            then accept;
        }
        term reject-tcp {
            from {
                next-header tcp;
            }
            then {
                count reject-tcp;
                log;
                reject;
            }
        }
        term accept-last {
            then accept;
        }
    }
    filter external-out6 {
        term reject-invalid-destination-address {
            from {
                destination-address {
                    ::/96;
                    ::ffff:0.0.0.0/96;
                }
            }
            then {
                count reject-invalid-destination-address;
                log;
                reject administratively-prohibited;
            }
        }
        term reject-udp-prohibited-services {
            from {
                source-address {
```

```
                2001:df0:2ee::/48;
            }
            next-header udp;
            destination-port [ 7 19 69 109 111 137-139 167 177 2049 ];
        }
        then {
            count reject-udp;
            log;
            reject;
        }
    }
    term accept-dns-response {
        from {
            source-address {
                2001:df0:2ee:50::13/128;
            }
            next-header udp;
            source-port 53;
        }
        then accept;
    }
    term reject-tcp-prohibited-services {
        from {
            source-address {
                2001:df0:2ee::/48;
            }
            next-header tcp;
            destination-port [ 135 137-139 445 ];
            tcp-initial;
        }
        then {
            count reject-tcp;
            log;
            reject;
        }
    }
```

```
                term accept-last {
                    then accept;
                }
            }
        }
    }
```