

SA46T-AS の StarBED 環境での大規模実証実験

松平 直樹^{1,a)} 蛭子 恵一² 宮地 利幸³ 三輪 信介³ 篠田 陽一⁴

概要：

IPv4 アドレスの枯渇以降の状況を踏まえた以降技術として SA46T-AS を提案している。今回、大規模実証実験を StarBED 環境に於いて実施した。本実験はサーバを対象に IPv4 アドレス共有を行うシステムとして実施し、実験ノード当たり 128VM、合計 1536VM で構築されたサーバでアドレス共有を行った。規模追従性を実現するための自動展開技術を開発し、基本的な通信確認を行った他に、性能測定、負荷試験、VM 多重度に関する試験も行った。これら実験の構成、結果について述べ、これら実験結果の評価を行い、課題を述べる。これら実験を通じて SA46T-AS 技術の有効性を示し、今後の実証実験について述べる。

1. はじめに

IPv4 アドレスの枯渇以後の状況を踏まえた IPv4 から IPv6 への移行技術として SA46T-AS (Stateless Automatic IPv4 over IPv6 Tunneling with Address Sharing) を提案している [1][2]。今回、大規模実証実験を StarBED[3] 環境において実施した。本論では、大規模実証実験環境と、この実験に際して開発した自動化展開技術、及び性能測定について述べ、SA46T-AS の実用性について述べる。

2. SA46T-AS の概要

2.1 SA46T-AS の位置づけ

IPv4 アドレス枯渇が現実となる一方で、IPv6 の普及は本格化しているとも言え難く、円滑な移行にギャップが生じている。

IPv4 から IPv6 への移行技術は (1)Dual Stack、(2)カプセル化、(3)IPv4-IPv6 変換の 3 種あるが、IPv4 アドレスの現場での枯渇後は、Dual Stack の構成が取れなくなり、IPv6 only な状況が自然発生すると予測される。一方、IPv4 の継続利用についても重要性が増すと予測される。IPv4 アドレスの枯渇後、IPv4 の継続利用を可能には、IPv4 アドレ

スの再利用 (Reuse) と共有 (Share) のふたつのアプローチが考えられるが、これらアプローチは、移行技術であるカプセル化や IPv4-IPv6 変換が兼ね備えることが望ましい。

このような認識に基づいて、筆者らは、IPv4 アドレス枯渇後の移行技術として、SA46T[4][5]、SA46T-AS、SA46T-AT[6][7] を提案している。表 1 に SA46T と SA46T-AS、および SA46T-AT の位置づけを示す。

表 1 SA46T-AT の位置づけ

移行技術	IPv4 アドレス	
	再利用 (Reuse)	共有 (Share)
Dual Stack	-	-
カプセル化	SA46T	SA46T-AS
IPv4-IPv6 変換	SA46T-AT	-

SA46T-AS は、IPv4 アドレスの共有を可能とする IPv4 over IPv6 カプセル化技術である。SA46T-AS の技術的なポイントは、カプセル化の outer header のアドレスである IPv6 アドレスのフォーマットにある。このアドレスを SA46T-AS アドレスと呼ぶ。図 1 に SA46T-AS アドレスを示す。

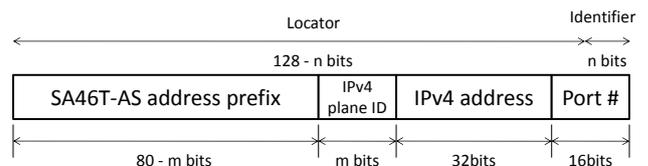


図 1 SA46T-AS アドレス

SA46T-AS アドレスは、SA46T-AS prefix、IPv4 network plane ID、IPv4 アドレス、TCP/UDP の Port 番号から構

¹ 富士通株式会社
FUJITSU LIMITED
² 富士通コンピュータテクノロジーズ
Fujitsu Computer Technologies
³ 独立行政法人 情報通信研究機構 北陸 StarBED 技術センター
National Institute of Information and Communications
Technology (NICT), Hokuriku StarBED Technology Center
⁴ 北陸先端科学技術大学院大学 高信頼ネットワークイノベーション
センター
Japan Advanced Institute of Science and Technology
(JAIST), Dependable Network Innovation Center
a) matsuhira@jp.fujitsu.com

成される。

SA46T-AS アドレスは、SA46T アドレスに Port 番号も含めるようにしたアドレスで、IPv4 アドレスのアイデンティファイア部を Port 番号を用いることにより拡張したアドレスと見ることができる。なお、IPv4 network plane ID を含むロケータ部の拡張は SA46T と同様である。

2.2 SA46T-AS の適用領域

SA46T-AS の適用対象として、クライアントや P2P が考えられるが、まず、サーバ、つまりデータセンターもしくはクラウドを対象とした。これは、サーバでは、IPv4 グローバルアドレスが必須であり、IPv4 アドレス枯渇は直ちに影響する切実な問題となる。

図 2 に、SA46T-AS サーバ IPv4 アドレス共有システムの構成図を示す。これは、SA46T-AS をサーバ環境に適用したもので、サーバの IPv4 アドレスを、TCP/UDP のポート番号が重複しない条件で再利用を可能とするシステムである。

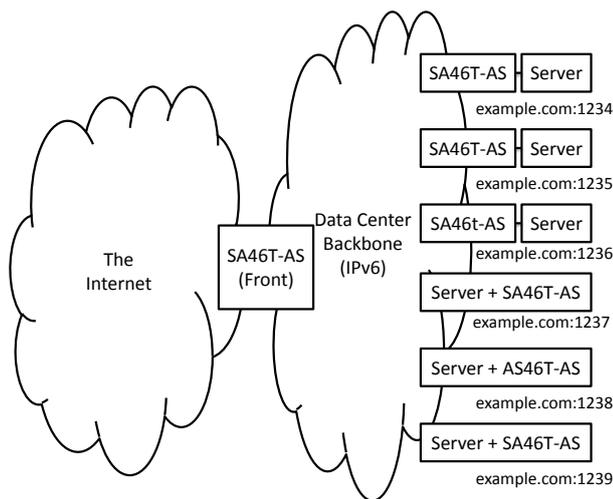


図 2 SA46T-AS サーバ IPv4 アドレス共有システムの構成図

このシステムでは、SA46T-AS は、フロントエンドとバックエンドの二種で構成される。これを、SA46T-AS フロントエンド、SA46T-AS バックエンドと呼ぶ。また、SA46T-AS バックエンドでは、サーバと一体型の構成かサーバ外付けの構成の双方が取れる。以下、SA46T-AS バックエンドと一体のサーバを SA46T-AS バックエンド内蔵アドレス共有サーバ、以後、略して、SA46T-AS 内蔵サーバと呼ぶ。

クライアントからは SA46T-AS フロントエンドがサーバに見えるが、アクセス先の実態は SA46T-AS 内蔵サーバ、もしくは、SA46T-AS に接続されるサーバである。

3. 大規模実験の狙い

3.1 大規模実証実験の必要性と方針

SA46T-AS は IPv4 アドレス共有のための技術であり、IPv4 アドレスの共有度をあげられることが重要である。具体的に、最大数である 65535 台で共有できれば、アドレスの共有度に関して制限のない、十分スケーラビリティを備えた、技術であると評価できる。

一方、データセンターでは、仮想化技術が身近になって以来、プロトタイプの作成が容易になり、VM の新規作成が同時並行的に発生するスプラウティングと呼ばれるような状況や、需要予測を超えた要求への対応も欠かせない。IPv4 アドレスの供給も含むサーバ収容のスケーラビリティ、すなわち、小規模から大規模まで、同一の仕組みで展開できることが重要である。従って、方式が大規模展開可能であることの証明は必須である。

我々は既に、小規模な実証実験を一般公開の場である Interop Tokyo 2012 にて、デモンストレーションとして実施し、基本的な SA46T-AS 方式の有効性を証明できたものと評価している。この第一段階の実証実験の結果を踏まえ、第二段階に行うべきことを検討した結果、(1) 規模の追従性、つまり、大規模に展開した環境で実際に動作することを通じスケーラビリティを有することの実証、(2) 実質的な性能を発揮できることの実証、と考えた。

3.2 大規模実証実験環境

SA46T-AS の動作検証を行うための実験環境として、情報通信研究機構 北陸 StarBED 技術センターに設置されている、大規模ネットワークテストベッド StarBED を利用した。

StarBED には、多数の実験専用の PC とそれを接続するネットワーク機器が設置されており、利用者が自由にこれらのリソースを設定し、実験環境を構築できる。

3.3 今回の実証実験の目的

今期の大規模実証実験の目的を以下に示す。

(1) 規模追従性(スケーラビリティ)

- 自動展開技術の開発と動作の実証
- 基本的なアクセス確認
 - 1 台のクライアントから全ての SA46T-AS 内蔵サーバへのアクセス
 - 複数のクライアントから 1 台の SA46T-AS 内蔵サーバへの同時アクセス

(2) 基本性能測定

- スループット測定と SA46T-AS のオーバヘッド測定

(3) 次の段階の実証実験のための準備実験

- 負荷試験のための準備実験

- VM 多重度に関する準備実験

4. 実験の構成

4.1 StarBED の概要

StarBED は地理的に一カ所に集中して設置されているネットワークテストベッドであり、2013年8月現在で1300台以上のPC（ノード）を実験専用提供している。利用希望者は利用申請をすることで、割り当てられた実験ノードおよびVLAN番号を利用期間中は占有的に利用できる。これらの実験ノードに、OSや必要なアプリケーションソフトウェアのインストールを行うことで、実験ノードのソフトウェア環境を整備し、また、ネットワークスイッチでのVLAN設定および実験ノードでのネットワーク設定を行うことで、目的とする実験環境全体を構築する。これらの手順はSpringOSと呼ばれるStarBEDを操作するためのミドルウェアを通して実施することが可能であるが、ソフトウェアのインストールなどを一括で行うためには、環境の未設定時から実験ノードにアクセスする必要がある。

これを可能とするため、StarBEDの全ての実験ノードは実験に利用するネットワークインターフェース(NIC)以外に管理用のNICが搭載され、このNICは静的に設定が行われているため、OSインストールなどの環境設定時、さらには実験中のトラブル発生時にも実験に利用しているネットワーク以外からのアクセスが可能となっている。図3にStarBEDの物理トポロジを示す。

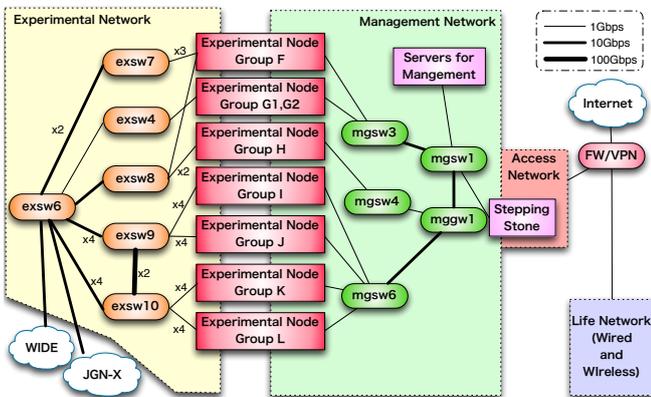


図3 StarBEDの物理トポロジ

StarBEDの実験ノードは、柔軟な実験環境を構築するため、あるいは導入時期の差異によって、さまざまな仕様のものが存在する。それぞれグループとして区別されているが、今回の実験ではグループIと呼ばれるノードの一部を利用した。表2にグループIに属するノードの仕様を示す。

4.2 ハードウェアの構成

実験システムの物理構成を図4に示す。本実験では、SA46T-AS内蔵サーバをまずVMで100台程度で動かし、最終的にその10倍の1000台程度で実験を行う方針で臨ん

表2 グループIノード仕様

Model	Cisco UCS C200 M2
CPU	6-core Xeon X5670 * 2
Memory	48GB
HDD	SATA 500GB * 2
実験用NIC (GbE)	4
管理用NIC (GbE)	1
導入年	2011

だ。1000台なら十分に大規模であり、実験ノード台数としても10数台程度で実施できるとの見通しがあったため、規模感として手頃である。最終的には、クライアント搭載用に1台、DHCPv4サーバに1台、SA46T-ASフロントエンドに1台、SA46T-AS内蔵サーバ搭載用に12台の合計で15台の実験ノードを用いた。

ネットワークは、クライアントとSA46T-ASフロントエンドを接続するIPv4ネットワーク、SA46T-ASのバックボーンネットワークとなるIPv6ネットワーク、そしてマネージメントのIPv4ネットワークの3種を、StarBEDのExperimental Network上にVLANを用いて構築した。これら3種のネットワークで用いるNICは全て同一で、ハードウェアに違いはない。

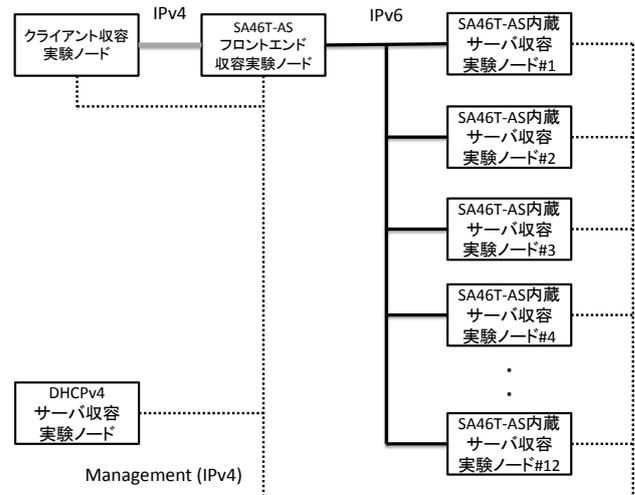


図4 物理構成

4.3 ソフトウェアの構成

実験ノードでは全て共通の仮想化環境とした。具体的には、ハイパーバイザとしてCentOS6.3のKVMを用い、ゲストOSにCentOS5.5を用いた。

ゲストOSは、Diskのサイズが、システム用に2.5GB、データ用に0.7GBの合計3.2GByteで、そして、メモリサイズが256MByte、CPUは1個の割り当てで構成した。なお、ハイパースレッディングを有効としたことにより、論理的なCPUコア数は24である。また、ネットワークドライバにはvirtioを使用した。

SA46T-AS 内蔵サーバを搭載する実験ノードでは、実験ノードあたり 128 台の VM を動作させたが、これら VM が使用する実メモリは 32GByte であり、実験ノードが搭載する 48GByte のメモリで十分余裕のある資源割り当てである。このためスワップは発生しないものと考えられる。

図 5 は SA46T-AS 内蔵サーバを搭載する実験ノードの構成図である。ハイパーバイザーと VM で構成された SA46T-AS 内蔵サーバが 128 台搭載される。

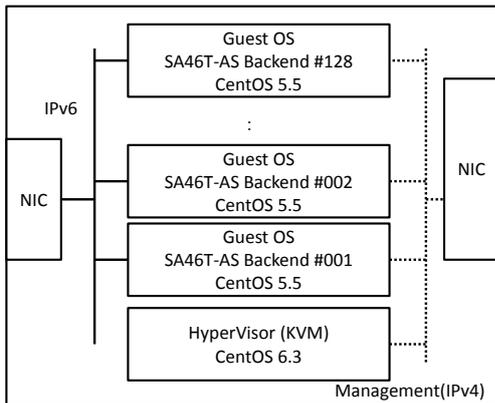


図 5 SA46T-AS 内蔵サーバ搭載実験ノードの構成

なお、SA46T-AS フロントエンドを搭載する実験ノード、クライアントを収容する実験ノード、DHCPv4 サーバもネットワークの収容については各々異なるが、仮想環境であり、同様な構成である。

4.4 論理構成

このような仮想化技術を用いて構成される論理構成図が図 6 である。論理的には、1536 台の SA46T-AS 内蔵サーバが接続されていることとなる。

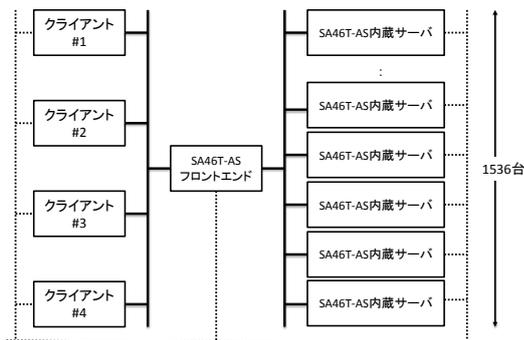


図 6 論理構成図

4.5 ネットワーク関連の設定について

4.5.1 フラグメンテーションについて

SA46T-AS はカプセル化を行うため、フラグメンテーションが発生する。本実験では、IPv4 の Path MTU Discovery

を動作させることとした。なお、TCP/MSS の書き換えは行わずに実施した。

4.5.2 マネージメントネットワークで用いるプロトコルスタックについて

ハイパーバイザーの設定など、主な操作はマネージメントネットワークを介して実施する。現状を勘案すると、マネージメントネットワークでは、アドレスが豊富な IPv6 を使用すべきだが、StarBED へのインターネットを経由したりリモートアクセス手段が IPv4 のみであったことから、IPv4 を用いた。

5. 自動展開技術の開発

今回の実験では、自動化の対象は SA46T-AS 内蔵サーバを搭載する実験ノードのみとし、他の実験ノードについては、基本的にマニュアル設定とした。

5.1 各種パラメータの自動生成

本実証実験では、実験ノードあたり 128VM、12 台の実験ノードで合計 1536VM を立ち上げる必要がある。手動での生成は現実的でないため、VM の作成と SA46T-AS バックエンド内蔵サーバの起動を自動で行うためのスクリプトを作成した。

VM の作成と起動を自動で行うには、これらを構成するパラメータをどのように自動で生成するかを設計する必要がある。図 7 にその相関関係を示す。

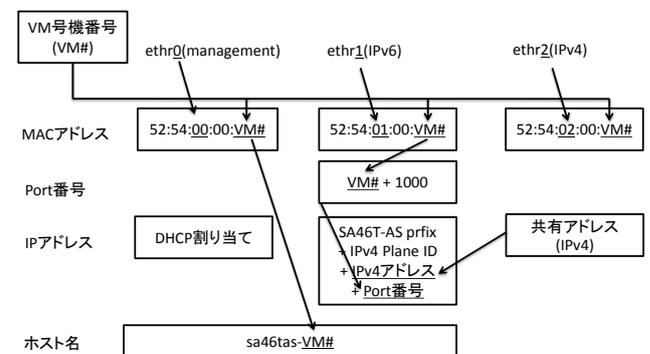


図 7 パラメータの生成方法

VM 作成では、その MAC アドレスと IP アドレスの生成ができることが必要となる。この実験では、VM 号機番号を定義し、これをキーとして、各種パラメータを生成する方式とした。

VM 号機番号は、SA46T-AS 内蔵サーバが動作する VM に対して与えられ、SA46T-AS サーバ IPv4 アドレス共有システムでユニークな番号である。この番号は、実験ノードには依存しない。

MAC アドレスは、イーサネットのデバイス番号*1と VM

*1 物理マシンでは、イーサネットのデバイス番号は OS が任意にナンバリングするが、KVM の環境では、予め意図した通りに採番

号機番号をから自動生成するものとした。

なお、SA46T-AS 内蔵サーバでは、マネジメントネットワークのインタフェースと IPv6 ネットワークの二つのインタフェースが接続されるが、これら以外に、サーバ自身の IPv4 アドレス、すなわち、アドレス共有する IPv4 アドレスの割り当てにも必要になるため、その IP アドレスを割り当てるインタフェースに対する MAC アドレスの生成も必要となる。従って、合計 3 つの MAC アドレスの自動生成を行っている。

SA46T-AS 内蔵サーバに割り当てるポート番号については、今回はサーバあたり 1 ポートの割り当てであるため、VM 号機番号に 1000 を加えた値とした。

SA46T-AS アドレスは、静的な値である、SA46T-AS プレフィックスと IPv4 Network Plane ID、及び IPv4 アドレスと、自動生成されたポート番号と組み合わせて生成するものとした。

なお、VM へのアクセスを容易にするため、ホスト名についても自動生成の対象に加えた。こちらは、VM 号機番号をそのまま利用した。

5.2 マネジメントネットワークへのアドレス自動割り当て

各種設定やコピーのためには、予めマネジメントネットワークが立ち上がっている必要がある。

マネジメントネットワークは全ての実験ノードや、そこで動くハイパーバイザ及び VM の全てが接続されるネットワークである。マネージメントネットワークでは、実験ノードに常に同じ IPv4 アドレスを割り当てられるよう、MAC アドレスと IPv4 アドレスを静的に括り付けることとし、そのアドレスを DHCP で配布することとした。

この括り付けの設定を自動で生成する、`mk_dhcp_conf` と呼ぶスクリプトを開発した。本スクリプト、VM 号機番号を範囲指定することにより、その範囲分の IP アドレスを MAC アドレスに対し自動的に割り当てる。生成したファイルは DHCP サーバの構成情報として使用される。

5.3 フロントエンドの設定自動化

SA46T-AS フロントエンドでは、各 SA46T-AS バックエンドへの経路情報をルーティングテーブルに保持する必要がある。今回の実験では、これら情報をダイナミックルーティングプロトコルで取得するのではなく、ステティック設定とした。これは、SA46T-AS バックエンドの経路数分であり、今回、SA46T-AS あたり 1 経路となるので、この設定が膨大となる。そこで、この設定の自動生成を含む、SA46T-AS フロントエンドの初期化を行う、`mk_init-sa46t` と呼ぶスクリプトを開発した。

可能である。

5.4 VM の自動生成と起動及び SA46T-AS バックエンドの自動設定

以下に SA46T-AS バックエンドの展開について記載する。

- (1) StarBED のマネージメントネットワーク上にあるストレージから、KVM、SA46T-AS 内蔵サーバのテンプレートとなる VM、使用するスクリプトを実験ノードのディスクに手動でコピーする。
- (2) 実験ノードの KVM 環境にて、`create_vm` を実行し、VM をコピーする。この際、コマンドのパラメータとして、VM 号機番号を範囲指定する。コマンドは、MAC アドレスを VM 号機番号を元書き換え、かつ、VM 起動時に用いる XML で記載された定義ファイルを書き換える。
- (3) 実験ノードの KVM 環境にて、`kick_vm` コマンドを実行し、VM を起動する。
- (4) VM 起動後、VM にて、`startup_sa46t` が起動される。VM 内の各種設定及び SA46T-AS バックエンド用の初期化スクリプトが生成、実行される。なお、HTTP アクセスの際、正しい SA46T-AS 内蔵サーバにアクセスできているかを確認できるようにしているが、`index.html` にて VM 番号を表示できるようにしているが、`index.html` の書き換えもここで行っている。

なお、これ以外に、VM を停止するための、`shutdown_vm` と呼ぶスクリプトも開発した。

5.5 テストツール

クライアントからサーバにアクセスする際も、膨大なポート番号へのアクセス行う必要があるため、`access_chk` と呼ぶスクリプトを作成した。これを含め、今回作成したスクリプト群を表 3 に示す。

6. 実験項目とその結果

6.1 規模追従性

6.1.1 自動展開技術の開発と動作の実証

`create_vm` で実験ノード上で 128 個の VM を作成し、その後、`kick_vm` で実験ノード内の VM の起動を順次行う。このとき、ハイパーバイザーから取得できるゲスト OS の状態*2、に、OS が立ち上がったことを示す状態が存在しないため、OS の立ち上がりを確認して、その後次の VM を起動するといった立ち上げができない。そこで、`kick_vm` では、等間隔で VM の起動を行うこととした。

当初、30 秒間隔で VM の起動を行ったが、`httpd` デーモンの起動に失敗する VM が 2-3 割あった。間隔を 60 秒に変更したところで起動の失敗が無くなり、正常に起動した。

*2 `virsh domstate` コマンドで VM の状態を取得できるが、この状態は全部で、`no state`, `running`, `blocked`, `paused`, `shutting down`, `shut off`, `crashed` の 7 種であり、OS が立ち上がったことを示す状態が存在しない。

表 3 スクリプト一覧

mk_dhcpd_conf	マネージメントネットワークの MAC アドレスと IP アドレスの対応を指定された号機分、DHCP サーバの設定形式で自動生成する
create_vm	ハイパーバイザー上で動作し、指定された号機分の VM を作成し virtmanager に登録する
kick_vm	ハイパーバイザー上で動作し、ハイパーバイザーに登録されている VM を指定された号機分起動する
shutdown_vm	ハイパーバイザー上で動作し、ハイパーバイザーに登録されている VM を指定された号機分停止する
mk_init-sa46t	SA46T-AS フロントエンドで動作し、SA46T-AS バックエンドの数を指定して、SA46T-AS フロントエンドで必要となる経路情報を含む SA46T-AS フロントエンドの初期化スクリプトを生成、実行する
startup_sa46t	SA46T-AS 内蔵サーバで動作し、VM 内の各種設定および SA46T-AS 内蔵サーバ用の SA46T-AS の初期化スクリプトを生成、実行する
access_chk	クライアント上で動作し、指定された範囲のポートに対して順次 wget でアクセスを行う

このとき、128VM の立ち上げに 130 分かかった。

6.1.2 基本的なアクセス確認

1 台のクライアントから wget コマンドを使用して順次アクセスし、1536 個全ての SA46T-AS 内蔵サーバにアクセスできるかを確認した。結果は、128 秒で全ての SA46T-AS 内蔵サーバにアクセスできた。VM あたりに換算すると、1 台あたり 83msec となる。

また、4 台のクライアントから 1 台の VM への同時にアクセスを行った。小さなファイルサイズでは同時のアクセスを確認できないので、500MByte のファイルを wget で獲得した。apache のログで確認し、同時アクセス可能であることを確認した。

これらを通じ、基本的なアクセスが成功したことを確認した。

6.2 性能測定

次に、1 クライアントへの 500Mbyte のファイル転送時間を計測した。こちらも、wget コマンドを用いて計測した。こちらは、30 回程度計測を行い、最大値と最小値を除外し、平均値を求めた。

また、SA46T-AS を経由した場合と、SA46T-AS を経由しない場合の双方で測定した。なお、SA46T-AS を経由しない場合のデータ採取は、マネージメントのネットワークを用いて行った。

測定の結果と参考値 [8] を表 4 に示す。

6.3 次の段階の試験へのデータ採取

6.3.1 負荷試験のための準備実験

基本的な性能が取れたため、負荷をかけてみた。具体的に、4 つのクライアントから順次 1536VM から 500MByte

表 4 性能測定結果と参考値

SA46T-AS 経由有無	転送速度
経由あり	58.2MB/s (465.6Mbps)
経由なし	72.7MB/s (581.6Mbps)
参考値	596Mbps

のファイル転送を行った。全ての転送が終わるのに、36 時間を要した。

本試験は、一つのクライアントから、VM に対してシーケンシャルにアクセスするもので、クライアントの起動もほぼ同時であるため、4 つのクライアントが同一の VM に対して同時にアクセスし、完了後、次の VM にアクセスするというものである。

クライアント #1 で 13VM、クライアント #2 で 12VM、クライアント #3 で 11VM、クライアント #4 で 13VM に対し、リトライタイムアウト^{*3}で転送が失敗していた。また、1VM のみ、httpd がダウンしていたものがあった。これは、実験内のある 3 時間に発生した現象で、なおかつ、同一の実験ノードで発生したので、この実験ノード固有の問題の可能性が高いものと考えられる。

6.3.2 VM 多重度に関する準備実験

今回、実験ノードあたり 128 台の VM を起動させることとしたが、実際に、何台くらいの VM が妥当なのかの指標は必要になるものと考えられる。このための準備実験として、SA46T-AS 内蔵サーバに対してアクセスするクライアント数を変化させて性能を測定する実験を行った。

具体的に、SA46T-AS 内蔵サーバに対し、アクセスするクライアント数を 1 台から 4 台に変化させて測定を行った。この際、クライアントが搭載される実験ノード及び SA46T-AS フロントエンドが搭載される実験ノードの負荷が同じになるよう、異なる実験ノードに搭載される SA46T-AS 内蔵サーバに、他のクライアントからのアクセスを行うようにした。また、この際、ファイルサイズを変化させて実験を行った。

グラフ化した実験結果を図 8 に示す。

ファイルサイズが小さい場合にスループットの違いが顕著になり、SA46T-AS 内蔵サーバにアクセスするクライアント数が少ないほど、スループットが大きい。ファイルサイズが大きくなると、アクセスするクライアント数の違いによらず、スループットが収束する傾向が見られた。

7. 実験の評価と課題

7.1 規模追従性

7.1.1 自動展開技術の開発と動作の実証

自動展開技術が機能し、その有効性が確認できた。実験ノードあたり、128VM を 130 分で展開できた。立ち上げは実験ノードで同時に行うことができるので、SA46T-AS

^{*3} 20 秒のタイムアウトで 3 回リトライを実施で合計 60 秒。

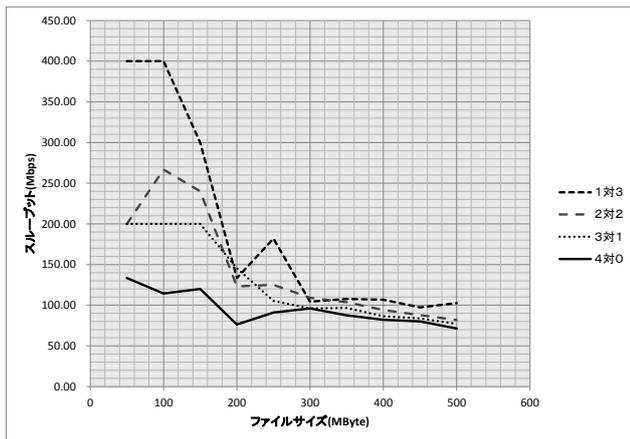


図 8 ファイルサイズとスループット

サーバ IPv4 アドレス共有システム全体でも、130 分で展開できる。今回は、実験ノードを 12 台用いたので、総数 1536VM の展開を行ったが、実験ノードを 512 台用いれば、ポート共有の最大数となる 65536VM の展開も 130 分で可能であるとの見通しを得られた。

一方、立ち上げ時間の妥当性には課題が残った。OS の立ち上がりを確認できれば、その確認を終えた直後に次の VM の展開が可能となるので、最低限の時間で立ち上げが可能になる。VM のアプリケーションからハイパーバイザに情報伝達する仕組みを作れば可能性があり、今後の検討課題である。

7.1.2 基本的なアクセス確認

基本的なアクセス確認については問題なく、課題も特にないと評価できる。

7.2 性能測定

まず、SA46T-AS を経由しない性能について評価を行う。

参考値は、同一ホスト内のゲスト OS 間の通信性能だが、仮想化技術を用いない場合のサーバ間の通信性能が 3.88Gbps と 6.5 倍以上の性能が得られていることから、異なるホスト内のゲスト OS 間の通信性能も、ほぼ同等になると推察され、今回の結果の参考に値するデータと考えられる。

これをふまえ、SA46T-AS 経由なしの性能と比較すると、参考値とほぼ同様の性能を示しており、本性能測定結果は概ね妥当なものであると評価できる。

次に、SA46T-AS を経由した性能について評価を行う。SA46T-AS を経由することにより転送性能は約 80% に性能劣化した。

性能劣化の原因は、(1) カプセル化／デカプセル化の処理時間、(2) フラグメント化に伴うパケット数の増大及び IPv6 ヘッダ追加に伴う送信データ量の増加、(3) Path MTU Discovery の学習の際のエラー通知及び再送処理の 3 つと考えられる。

カプセル化／デカプセル化の処理時間については、さほど費やすステップ数は多くなく、20%の性能劣化を説明するには不十分である。

フラグメンテーションによる影響は、パケット数は 3%増加し、IPv4 での転送量は、0.07%増加でほぼ変わらず、IPv6 パケットの転送量は 3%増加で、20%の性能劣化を説明するには不十分である。

Path MTU Discovery では、Path MTU の値が定期的にクリアされるが、経験則では、10 分程度の周期でクリアされるようである。500MByte の転送には約 8 秒で完了するため、Path MTU Discovery の動作学習は 1 度のみで、再学習は行われないため、無視してよいものと考えられる。

これらから、上記の要因では 20%の性能劣化は説明できない。念のため、SA46T-AS のカプセル化に伴ってパケットデータのコピーが発生していないかを確認したが、発生は確認されなかった。

本性能劣化の原因調査は今後の課題だが、仮想環境特有の現象かどうかも含めた確認、つまり、物理環境での性能劣化がどの程度かとの比較などを行った調査が必要と考えられる。

7.3 次の段階の試験へのデータ採取

7.3.1 負荷試験のための準備実験

1536 台へのアクセスに 36 時間と長時間を要したが、SA46T-AS 内蔵サーバあたり 100 秒、クライアントあたり 25 秒である。負荷を掛けなかった際は約 8.6 秒で 500MByte を転送できるので、ボトルネックがどこかに生じているものと考えられる。また、タイムアウトしたことについては、その再現性の有無の確認が必要である。

ボトルネックがどこにあるのかの調査は今後の課題である。また、今回は、簡単に負荷をかけられる方法で実施したが、本格的な負荷試験のためには、専用の負荷試験ツールの使用が必要になると考えられる。

7.3.2 VM 多重度に関する準備実験

サーバへのアクセス数の変動を見ようとしたが、仮想環境ではリアルタイムクロックの精度に問題があり、さらに、計測はリアルタイムクロックを用いたため秒単位での結果しか得られなかったため、採取したデータの信頼性に問題のあることがわかった。

このため、ファイルサイズが小さいときは、所要時間も小さくなるため、時間精度の誤差が相対的に大きくなり、データの信頼性は低くなる。その一方、ファイルサイズが大きいときは、所要時間も大きくなり、時間精度の誤差が相対的に小さくなり、データの信頼性も比較的高いと言える。従って、500MByte の転送結果のみを用いて行う。

500MByte の転送性能は、無負荷時の基本性能である 465.6Mbps に比べると低いが、アクセス数の違いによって大きな性能変動は確認できなかったと言える。たかだか 4

クライアント程度からのアクセスには、サーバは十分に対応できると考えて良さそうである。一方で、きちんとした評価を行うには、クライアント台数を多くする必要があるとも考えられる。サーバの性能が十分であるなら、ボトルネックは別のところにある可能性もあり、こちらも今後の課題である。

性能測定全般では、IPv4 では http でしかアクセスできないため、アクセスするためのツールが wget のみで、iperf などの性能測定のためのツールが使えなかった。これは、IPv4 アドレスを共有するシステム特有の課題である。この測定方法についても、今後の課題である。

また、SA46T 内蔵サーバは仮想環境であることが望ましいものの、クライアントや SA46T フロントエンドは仮想環境である必然性は無い。仮想環境でなければ、リアルタイムクロックの精度に関する課題は無くなる。この実験構成についても、今後の課題である。

8. SA46T-AS 方式の評価

今回、大規模展開のための基本的な自動化技術を確認したことから、SA46T-AS は十分な実用性があるものと評価できる。

今回は、実験ノードあたり 128VM での実験を行い、128VM が起動した環境でアクセス確認や性能測定を問題なく実施できたので、128VM が動作したとの実績とすることが可能と考えられる。サーバ負荷の影響については今後の課題だが、リファレンスたる具体的な数値が得られたことは成果と考えられる。さらに多くの VM を搭載できるのか、あるいは実効性能を考慮した際、もっと少ない VM 数が現実的であるのかは今後の課題と言えるが、今回の実験で実用性があると評価できると考えられる。

全てのポート番号に対応するには 512 台の実験ノードが必要になるが、この台数は決して少なくはないものの非現実的な数でもなく、また、サーバハードウェアの性能向上やコア数の増加なども進んでおり、実用性があると評価できるであろう。

性能については、20%の性能劣化の説明は課題であるが、使い物にならないほどの性能劣化とは言えず、実用的であると言えるだろう。また、負荷試験にも、12 台の実験ノード中問題が出たのは 1 台のみで、他の 11 台は十分耐えたので、特定の実験ノード固有の問題の可能性が高く、実用的と評価できる。

以上の結果から、SA46T-AS 方式は、十分実用性があるものと評価できる。

9. 今後の実証実験について

次の第三段階以降の実証実験の試験項目案を以下に示す。小規模で実施して精査すべき項目と大規模で実施すべき項目があり、今回の課題解決も含め、実施計画が重要となる。

- web サーバの試験ツール等を用いた負荷試験
- VM 多重度を変化させたときの性能評価
- VM マイグレーション連携
- SA46T-AS の最大構成 (65535VM) での動作確認と展開時間等のデータ採取
- 実利用環境での実験及びユーザによる実利用

10. まとめ

SA46T-AS の大規模実証実験を StarBED 環境にて実施し、データセンター環境でのシステム構築を実際に行い、自動展開技術を開発し、動作を確認した。そして、この環境下で基本的なアクセスが問題なく実施できたことを確認し、基本的な通信性能を測定した。また、負荷試験や、VM へのアクセスクライアント数を変化させた性能も測定した。これら実験を通じ、SA46T-AS 方式が十分実用性がある方式であることを実証した。

StarBED 環境で実際に構築した実験環境は、SA46T-AS サーバ IPv4 アドレス共有システムのプロトタイプである。今回このリファレンスを構築できたことも、成果である。

IPv6 の普及状況及び IPv4 アドレスの枯渇の状況を見ながら、インターネットの健全な発展に寄与すべく、今後も引き続き標準化、実用化、そして普及に向けて取り組んでいく所存である。

参考文献

- [1] 松平直樹 *SA46T-AS: アドレスを用いずに IPv4 アドレス共有を実現するカプセル化技術* FIT2013, 2013
- [2] N. Matsuhira “*Stateless Automatic IPv4 over IPv6 Tunneling with IPv4 Address Sharing*” draft-matsuhira-sa46t-as-05.txt, Internet-Draft, July 2013
- [3] 宮地利幸, 中田潤也, 知念賢一, ラズバン・ベウラン, 三輪信介, 岡田崇, 三角真, 宇多仁, 芳炭将, 丹康雄, 中川晋一, 篠田陽一 *StarBED: 大規模ネットワーク実証環境情報処理*, Vol.49, Number 1, pp.51-70, 情報処理学会, Jan, 2008
- [4] 松平直樹 “*SA46T: IPv4 アドレス枯渇後の IPv6 移行と IPv4 継続利用を両立するカプセル化技術*” IC2012, 2012 年 11 月
- [5] N. Matsuhira “*Stateless Automatic IPv4 over IPv6 Encapsulation / Decapsulation Technology: Specification*” draft-matsuhira-sa46t-spec-07.txt, Internet-Draft, July 2013
- [6] 松平直樹, 上野幸杜, 堀場勝彦, 中村修 *SA46T-AT: SA46T アドレスを用いた IPv4-IPv6 変換技術* IC2013, 2013
- [7] N. Matsuhira, K. Horiba, Y. Ueno, O. Nakamura “*SA46T Address Translator*” draft-matsuhira-sa46t-at-02.txt, Internet-Draft, September 2013
- [8] 山岸 祐大, 田崎 創, 重近 範行 *KVM のネットワークパフォーマンス改善* IC2010, 2010