Linux Is IPv6 Ready —IPv6 はそこにある—

吉藤 英明 [*1,*2] **小堺 康之** [*1,*3] **中村 雅英** [*1,*4]

[*1]USAGI/WIDE **プロジェクト**

[*2] **慶應義塾大学**, [*3](**株**) **東芝**, [*4](**株**) **日立コミュニケーションテクノロジー**

概 要

Linux の IPv6 スタックの歴史は古い。しかしながら、その初期の段階のスタックの品質は、決して良いものとはいえなかった。また、セキュリティ機能やモビリティ機能など、その後実用的に必要不可欠とされる、いくつかの重要な機能が不足していた。

USAGI プロジェクトはこのような状況を打破し、Linux システムに高い品質のスタックを実現することを目的に設立された。6 年に及ぶ活動の結果、多くの機能が実装され、品質も大幅に向上した。我々の成果はカーネルの正式頒布物にも統合され、数々の “IPv6 Ready Logo” を取得している。

日々成長するスタックの品質を維持するため、我々は日常的な検査が可能な自動試験システムを開発・導入することで、問題の早期発見及び解決を可能としてきた。

本稿では、Linux における IPv6 およびその関連技術の実装の現状を俯瞰するとともに、国際的認証の取得と品質維持のための技術について述べる。

1 IPv6 の歴史と USAGI プロジェクト

現在の “インターネット” は、1960 年代後半から Internet Protocol Version 4 (IPv4) によって構築・運用されてきた。1980 年代の終わり頃、インターネット技術の標準化団体である IETF (Internet Engineering Task Force) [1] に参加している専門家らは、インターネットの急速な発展に対応するため、新しいバージョンの Internet Protocol が必要であるとの認識に達した。その結果、1992 年、新しいプロトコルは IPng (IP next generaion) と命名され、全面的な技術的議論が開始された。後に IPv6 (IP Version 6) [2] となる IPng は、単なるアドレス空間の拡張といった規模性の拡張のみにとどまらず、パケット転送性能やプロトコルの拡張性、セキュリティやプライバシーといった、従来の IPv4 が抱える種々の問題を解決することを目的として検討・開発が始まった。

このような原則の下、1994 年、IPv6 の基本仕様が制定された。いくつかの試験的実装や運用を経て、現在では、種々のネットワーク機器や、オペレーティングシステム (OS) を含むソフトウェアのベンダーが、対応を推進している。また、ISP (Internet Service Provider) による商用サービスも始まっており、いよいよ、すべてのものをネットワークに接続するための規格として実用段階に入っている。これは、機器に実装される IPv6 スタックは、すべからく、製品品質を備えていることが必要不可欠となったことを意味する。

一方、Linux の IPv6 実装の歴史は長く、1996 年終わりには Pedro Roque による実装が Linux の公式配布物の開発版 2.1 に登場している。しかし、その後しばらくの間、 Linux の IPv6 スタックは積極的に開発されることはなかった。

1998 年、日本における Linux IPv6 ユーザーが集まって発足した Linux IPv6 Users Group JP では、Linux IPv6 プロトコルスタックの実運用に即した種々の検証が行なわれた。次第に種々の相互運用性等の面で問題があることが明らかとなった。具体的には、不完全のソケット API 実装 [3, 4] や、近隣探索 (Neighbor Discovery) [5] 及び静的アドレス自動設定 (Stateless Address Autoconfiguration) [6] の不具合 [7]、必須とされる IP セキュリティ [8]、IP モビリティ [9, 10] 機能の欠如、が挙げられた。

これらの問題を集中的に解決し、世界でもっとも著名なオープンソース OS である Linux によりよい IPv6 スタックを実現し、IPv6 技術を推進するために、USAGI(Universal Playground for IPv6) プロジェクト [11] を発足させることになり、2000 年 10 月より、産学共同の WIDE プロジェクト [12] の支援の下、本プロジェクトが開始された。

USAGI プロジェクトでは、Linux IPv6 スタックの品質を向上させるとともに、従来欠けていた機能の実装を行っている。

2 品質の向上

IPv6 の普及には品質の向上が不可欠である。

2.1 評価手法

IPv6 スタックの品質の評価には、いくつかの基準がある。その中でも、TAHI Conformance Test Suite と、IPv6 Ready Logo Phase-1 および Phase-2 プログラム [13] がもっとも重要なものである。

2.1.1 TAHI Conformance Test Suite

TAHI Conformance Test Suite は、TAHI プロジェクト [14] によって開発された、実装が IPv6 仕様に適合しているかを主に試験、確認するための枠組みである。以下のような分野別にテストの詳細がテストスクリプトとして記述されている。

- IPv6
- ICMPv6
- Neighbor Discovery

- Stateless Address Autoconfiguration
- Path MTU Discovery
- Tunneling
- Robustness
- IPsec

この試験スイートはIPv6スタックの仕様適合性を評価するために広く使われており、LinuxのIPv6スタックの評価にも、この試験スイートを利用することができる。

2.1.2 IPv6 Ready Logoプログラム

IPv6 Ready LogoプログラムはIPv6の相互接続性に関する国際的な認証活動である。この認証を取得するには、申請者は定められた自己テスト (Self Test) の結果と、試験シナリオ (Test Scenario) に基づく相互接続性テスト結果を添えて申請し、審査を受ける。自己テストのツールとしては、IPv6普及・高度化推進協議会 [15] とTAHIプロジェクトによるTest Suite for IPv6 Ready Logoが認められている。

Phase-1

IPv6に必要不可欠なプロトコルを含み、他のIPv6機器と相互接続可能であることを示す。

自己テストは、IPv6 core、ICMPv6、Neighbor DiscoveryおよびStateless Address Autoconfigurationで必須とされる機能を対象とする。また、相互接続性試験には比較的単純なものが実施される。

Phase-2

Phase-2のロゴは、IPv6の特定の機能それぞれに規定される、より強い要求事項を満たしていることを示す。“Core Protocols” のロゴは、全ての基礎となるもので、IPv6 core、Neighbor Discovery、Stateless Address Autoconfiguration、Path MTU DiscoveryおよびICMPv6を対象とし、仕様の “MUST” と “SHOULD” で示されるような必須条件が評価される。また、相互接続性試験もPhase-1に比べてかなり複雑なものである。“Core Protocols” のロゴを取得した上で、より高度な機能、たとえば、IPsecやMobile IPv6に関するロゴも提供されており、また、Multicast Listener Discovery Version 2などに関する検討もされている。

2.2 Linux IPv6スタックの問題と改善点

USAGIプロジェクト以前は、LinuxでIPv6を利用可能であったものの、品質、特に、標準への準拠という面では、とても満足のいくものとはいえず、とりわけ、Neighbor DiscoveryやStateless Address Autoconfigurationで、残念ながら全く良い成績を残すことができなかった [7]。

USAGIプロジェクトでは、問題の解析を進めた結果、以下のような種々の問題があることがわかり、修正した [16, 17]。

不正確な状態遷移

Neighbor Discoveryの状態遷移マシンに不的確なものがあった。

→複雑だった排他制御や依存関係を整理し、保守性を向上させ、解決した。

不正確な時間管理

Neighbor Discovery、Stateless Address Autoconfigurationやカーネル内経路選択[1]において、有効期間などが正確でなかった。

→前項と同様、保守性の向上と正確な時間管理のために論理構造上の整理を行った。

入力検査の不足

Neighbor Discoveryなどにおいて、外部からの入力に対する検査が甘く、不正な入力に対して不適切な反応をしていた。

→検査のための共通基盤を実装するなど、構造を整理して解決した。

Linuxカーネル2.4向けに行われた改造は大規模であったため、そのまま直ちにカーネルの公式頒布物に統合することはできなかった。しかし、USAGIプロジェクトは利用者を含めたコミュニティーを形成し、その品質に一定の評価を得たことで統合への機運が高まり、多岐にわたる成果を整理・分割することで、徐々に公式頒布物に反映されるようになった。

これらの貢献もあり、2003年5月より、筆者の一人である吉藤が、公式頒布物のネットワーキング [IPv4/IPv6] 部門のメンテナーの一員に加わっており、よりグローバルな視点でネットワークスタックの改善に関与している。

なお、当初は上記TAHI Conformance Test Suiteを品質基準にしていたが、近年はIPv6 Ready Logoを基準として利用している。Linux公式頒布物では2.6.11-rc2がIPv6 Ready Logo Phase-1を取得しており、IPv6 Ready Logo Phase-2の取得も予定している。

3 IPセキュリティの実現

IPセキュリティ（IPsec）[8] の機能は、IPv6仕様上必須とされているにもかかわらず、カーネル配布物の中にIPsecの実装は含まれていなかった。一方、IPv4についてはFreeS/WANプロジェクト [18] がその実装を独自に提供し、IPv6についても、このFreeS/WANプロジェクトのソースコードを一部利用していたものが存在していただけであった[2]。

3.1 USAGI実装

USAGIプロジェクトの実装は、開発当初はFreeS/WANを出発点としたものの、IPsecのアーキテクチャーの中でIPバージョンに関わらず共通化できる部分であるSAD (Secutirty Association Database) やSPD (Security Pol-

[1] 一部Stateless Address Autoconfigurationに含まれる

[2] これは、米国の暗号輸出規制が影響していたことは否めない。

icy Database) などを汎用化した上で、FreeS/WAN が採用していた仮想デバイスによるトンネリング方式でなく、IP 層に統合された IPsec スタックを実装した。また、暗号や認証アルゴリズムは、IPsec 固有のものでないため、cryptoapi [19] の枠組みを利用した。

この USAGI IPsec 実装は、Linux 開発版 2.5.x において、公式頒布物に提案されたが、そのままは採用されなかった。

3.2 XFRM 実装

Linux 開発版 2.5 では、USAGI プロジェクトの IPsec 実装が契機の一つとなり、IP セキュリティ機能の実装が検討された。その結果、XFRM (Transform) と呼ばれる IP 層内 IPsec スタックが David S. Miller や Alexey N. Kuznetsov らにより実装された。これは、その後、USAGI プロジェクトと石黒邦宏により IPv6 対応がなされた後、IPv4 と IPv6 に共通なパケット変換 (Transform) のための枠組みとなった。

現在、Linux 2.6 は、IPv6 Ready Logo Phase-2 の IPsec 分野のうち、基礎 (Basic) 項目およびほぼ全て[3]の発展 (Advanced) 項目の試験をパスしており、ロゴを申請中である。

4 パケットフィルタ

Linux カーネルにおけるパケットフィルタは Netfilter Project[20] を中心として開発と保守が行われている。Netfilter Project が Linux に実装したパケットフィルタは既に IPv6 に対応しており、パケット内のデータのみでパケットの通過、遮断を判断する基本的なフィルタリングが可能であった。しかし、ネットワークフローの状態を追跡するコネクショントラッキングと呼ばれる機能が IPv6 に対応していなかった。そのため、例えばネットワークフローの状態に基づいたフィルタリングが不可能であり、ユーザーが IPv6 の利用を躊躇する理由のひとつとなっていた。そこで、USAGI プロジェクトは Netfilter Project と協力して IPv4、IPv6 両方のパケットを処理可能なコネクショントラッキング (以下 nf_conntrack と記す) を実装した [21]。以下の節では、nf_conntrack の機能と開発状況について説明する。

4.1 コネクショントラッキングの機能

nf_conntrack はパケット内の IP アドレスやポート番号で識別できるネットワークフローをコネクションと定義している。nf_conntrack の機能は、パケットを解析して以下のコネクションに関する情報を更新し、他のモジュールがそれらの情報を利用できるように保持することである。

- コネクションの状態。コネクションが確立されたか、他のコネクションと関係があるかなどを表す。

[3]未統合の AES-128-XCBC-96 サポートを除く

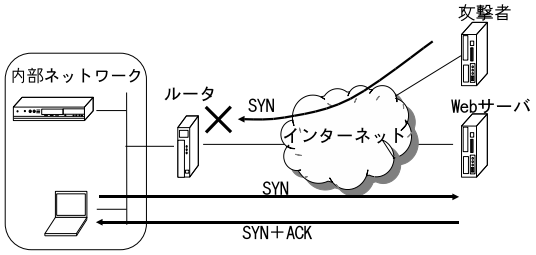

図 1: コネクショントラッキングの利用例

- トランスポートプロトコルに関する情報。例えば、TCP コネクションの状態などが追跡される。
- アプリケーションプロトコルに関する情報。例えば、FTP の PORT コマンドを含んだパケットを検知すると、PORT コマンドに記された IP アドレス、ポート番号を保持する。これにより、そのアドレス宛先としたファイル転送用のコネクションが生成されることを予期しておくことができる。
- コネクションのタイマ。これが時間切れになるとコネクションが切断したと判断される。
- コネクション中を流れたパケット数やバイト数

また、nf_conntrack は解析したパケットとコネクションとの関係が以下のいずれに当たるかを割り出す。

NEW
解析したパケットを含めて片方向のパケットしか検出していないことを示し、未だコネクションが確立していないことを示す。

ESTABLISHED
解析したパケットもしくはそれ以前のパケットによりコネクションが確立されたことを示す。ここで、「確立」は両方向のパケットを検知したことを示す。

RELATED
NEW と同様だが、解析したパケットが他のコネクションと関係あることを示す。例えば FTP でファイルを転送する場合、ファイル転送用のコネクションに属すパケットはコントロール用のコネクションと関係があるとみなされる。

INVALID
解析したパケットが破損しているか解析前のコネクションの状態と矛盾することを示す。

パケットとコネクションとの関係を利用すると、例えば図 1 で示すようなパケットフィルタをルータ上で実現できる。図 1 において、突然インターネット上の機器からルータに到来した TCP SYN パケットと、それが属すコネクションの関係は、NEW になる。しかし、インターネット上の機器から到来したパケットであっても、それが内部ネットワークからインターネットへ送信された TCP SYN パケットに対して応答するパケットならば、コネクションとの関係は ESTABLISHED となる。そこで、イ

ンターネット上からルータに到来するパケットの内、コネクションとの関係が NEW であるパケットを破棄するよう設定すれば、突然インターネット上の機器から到来したパケットを破棄することができる。

4.2 現在の開発状況

nf_conntrack は既に Linux 2.6.15 以降のカーネルに導入されている。また、パケットとコネクションとの関係を利用したパケットフィルタを可能とする state モジュールが Linux 2.6.16 以降のカーネルに導入されており、バージョン 1.3.4 以降の iptables コマンドで利用可能である。今後、Netfilter Project では、より高速なコネクション情報の管理方法の実現や、従来のコネクショントラッキングを利用した IPv4 NAT 機能の nf_conntrack への移行などが予定されている。また、IPv6 パケットフィルタに関係する長期的な開発項目として、Netlink [22] を利用したパケットフィルタの API や、より効率的なパケットフィルタ機構の実現などが予定されている。

なお、USAGI プロジェクトでは、IPv6 パケットフィルタ関連の機能も含めて nf_conntrack の保守を継続しており、筆者の一人である小堺が、Netfilter Project にコアメンバーとして参加している。

5 モビリティ機能の実現

モバイル IPv6 (Mobile IPv6) は、IPv6 で注目されている機能の 1 つであり、IPv6 の機能を活かした効率的な通信が可能である。

IPv6 におけるモビリティの実現は、当初、ヘルシンキ工科大学 (HUT) の Go-Core プロジェクトによる Mobile IPv6 の実装 (MIPL; Mobile IPv6 for Linux)[23] を基に、仕様適合性や動作の安定性を向上させて公開していた。カーネル 2.6 にむけた開発の段階で、MIPL の頒布物への統合が提案されたが、必要以上にカーネルに依存しており、ソースコードの変更量が大きすぎるなどの問題があり、採用されなかった。数々の議論の後、Go-Core プロジェクトと USAGI プロジェクトは、カーネルの公式頒布物に統合できるよう、新しい設計の MIPL2 の開発に着手した。

新設計では、仕様 [9][10] を分析し、カーネルのソースコードの変更量が少なくなるように努めた。シグナリングの交換や、モバイルノードの移動検知など、ユーザー空間で可能な処理は極力デーモンで行うようにし、実際の通信パケットの操作部分をカーネル内に残すようにした。MIPL2 開発中は、TAHI Conformance Test Suite の実施や他実装との相互接続性確認により品質向上に注力した。

これらの活動の結果、既に MIPL2 バージョン 2.0.1 がリリースされ、デーモンとカーネルパッチは入手可能となっている。現在は、カーネルの変更を公式頒布物へ統合できるように準備中である。また、Mobile IPv6 は、IPsec や鍵交換 (IKE; Internet Key Exchange) [24, 25] との連携が大きな課題の一つであり、既存の IKE 実装の変更が軽微になるように、API[26] の標準化を提案中である。

6 自動試験システムの構築

USAGI プロジェクトは、より高機能で高品質の Linux IPv6 スタックの実現を図ってきた。その成果が Linux 公式頒布物などを通じて広く Linux コミュニティに受け入れられた現在、さらなる開発活動に加え、既に開発したコードの品質維持も重要である。

特に、Linux カーネルは日々改良・修正が行われており、メンテナーなど多くの開発者は問題が起こらないように目を光らせているものの、それでも、変更によって副作用が生じる可能性は常につきまとう。

重要なのは、開発を止めたり逆行させることではなく、問題を早期に発見、特定し、修正して、開発を継続することである。

そこで、原則として毎日リリースされているカーネル公式頒布物に対し、IPv6 機能を中心に自動試験を行うシステムを開発した。これには、TAHI Conformance Test Suite の枠組み (Test Tool) を利用している。この自動試験システムにより、IPv6 機能等に問題が発生した場合、それを発見し修正することが容易になっている。なお、このシステムは USAGI Testlab として公開されている [27]。

6.1 システム動作

システムは、待ち受け、ビルド、テストの工程で構成され、繰り返される (図 2)。

テスト実行されていない時は新しいカーネルのリリースを待ち受ける (待ち受け段階)。テスト対象にはリリース版だけでなく、テスト候補である rc 版や、毎日リリースされる git 版も含まれる。

新しいリリースを発見すると、システムはビルド段階に移行し、自動的に新しい設定を行って構築を行ない、できあがったイメージをインストールする。

テスト段階では、システムは同一のカーネルに対して、複数のテストセットの試験を行えるように設計されている。このため、新しいテストセットを開始する毎に、テスト対象システム (NUT; Node Under Test) を適切なモードや設定で起動した上で試験を行う。各テストセットの試験が完了すると、リリースの種類に応じて前回の鍵となるようなリリースとの差分情報を含め、結果を表形式にまとめる。この差分情報により、変更の結果の副作用の有無を確認することができる。

全てのテストが終了すると、予め決められたカーネルで起動し、最初の待ち受け段階に戻る。

6.2 収集データとアクセス

システムは、テスト結果、以前の結果との差分、使われたソースとバイナリ、各段階の実行ログなど、種々の

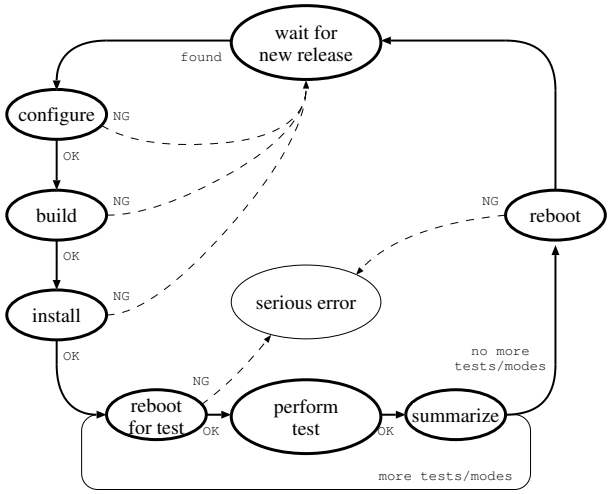

図 2: 自動試験システムの動作フロー

表 1: 自動試験システムの実行テストセット

テストセット
Phase-1 (host) Phase-1 (router)
Phase-2 (host) Phase-2 (router) Phase-2 IPsec (end-node) Phase-2 IPsec (securty-gateway)
TAHI Conformance Test (IPsec, end-node) TAHI Conformance Test (IPsec, security-gateway)

データを収集する。

それぞれのデータは HTTP サーバにより公開され、web ブラウザを利用して閲覧することができる (図 3)。

6.3 今後の計画

表 1 に現在自動試験を行っているテストセットを示す。

今後の計画としては、Mobile IPv6 機能といったさらなるテスト仕様への対応が考えられる。

7 過渡期に対する対策

新しい技術の経常的な採用には、その技術そのものの開発のみならず、旧技術との共存と、新技術への円滑な移行の問題が存在する。IPv6 は当初からそれらを強く意識して開発されてきたため、通常、IPv6 技術を導入することで問題が起こることは考えにくい。したがって、エラーケース（異常系）での検討は必ずしも十分でなかった。

しかし、IPv6 移行の過渡期に稀に起こりうる問題点に対処するため、WIDE プロジェクトでは、2004 年 9 月から、この問題を具体的に調査・分析し、解決することを目的として、IPv6-Fix [28] と呼ばれる活動を行っている [29]。この中では主に、IPv6 に関係すると思われる種々の具体的問題事例を収集し、分析している。

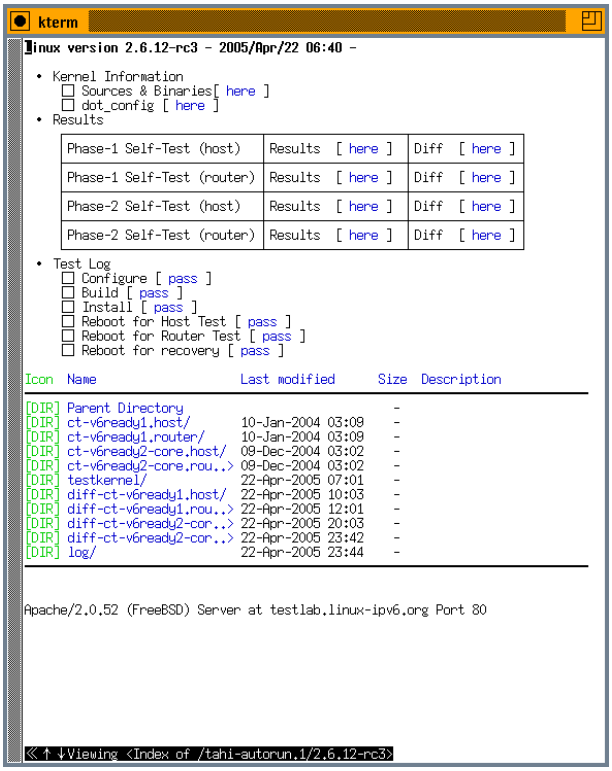

図 3: 自動試験システムへのアクセス例

たとえば、IPv6 接続性がないノードから IPv6 に対応した相手に接続する際、IPv4 での接続に移行するのに数秒以上の長い時間を要する事象が指摘されていたが、これは、“On-link Assumption” と呼ばれる Neighbor Discovery の規定が原因であることが判明した。

原因となった、(デフォルト) 経路がない場合には宛先は同一リンクにあるものと見做す、という規定は、新しい Neighbor Discovery の仕様では削除される予定であり、現在の Linux カーネルではこの問題は起こらなくなっている。

このように、IPv6 と IPv4 の共存期における利用シナリオで起こりうる問題に関しては、関係各機関等と協力の上、その解決を図り、移行を円滑に促進するために必要な対策を検討しており、問題は起こらなくなってきている。しかし、万が一、IPv6 に関係すると思われる事象に遭遇した場合、本稿の読者各位には、是非 IPv6-Fix プロジェクトまでご一報をお願いしたい。

8 態勢充分な Linux IPv6

前述したように、IPv6 Ready Logo は相互接続性の国際的な認証の活動である。2006 年 5 月現在において、300 を超すの製品が、基本的な相互接続性を示す、Phase-1 の認証を取得している。

USAGI プロジェクトも IPv6 Ready Logo プログラムには積極的に参加している。2004 年には安定版 2.6 の派生物で、2005 年には改訂した 2.6 ともう一つの安定版 2.4

それぞれの派生物で Phase-1 のロゴを取得した。

品質向上の活動が一段落し、成果が正式頒布物に統合された 2005 年には、カーネル 2.6.11-rc2 でも同プログラムに参加し、ロゴを取得している。Linux カーネルの中で、IPv6 は長い間試験的 (EXPERIMENTAL) と位置づけられてきたが、これを区切りに、デフォルトでモジュールとして組み込まれるようになった。

IPsec、Mobile IPv6、MLD といった、より高度な機能を対象とする IPv6 Ready Logo Phase-2 プログラムが開始されてから、我々の注目は Phase-2 に移っている。Linux の IPv6 スタックは、高い品質をもっており、これらの認証も近い将来に取得できるであろうと見込まれている。

9 まとめ

本稿では、USAGI プロジェクトと、Linux における IPv6 およびその関連技術の実装の現状を俯瞰するとともに、国際的認証である IPv6 Ready Logo 取得と品質維持のための技術について述べた。

現在の Linux の IPv6 実装は、基本機能のみならず、IP セキュリティやコネクショントラッキング、モビリティ機能なども実現されており、プロジェクト発足当初に比べ、飛躍的に高度化している。そして、文字通り「製品レベル」の品質を実現している。

今後の課題としては、ポリシーに基づく経路制御機能、モビリティ機能およびマルチキャスト転送機能のカーネル公式頒布物への統合が挙げられる。また、カーネルのみならず、ユーザー空間ライブラリやツールを含め、システム全体として、より使いやすい環境を構築することも課題である。

参考文献

[1] Internet Engineering Task Force, <http://www.ietf.org>.

[2] S. Deering and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification,” RFC2460, 1998.

[3] R. Gilligan, S. Thomson, J. Bound and W. Stevens, “Basic Socket Interface Extensions for IPv6,” RFC2553, 1999.

[4] W. Stevens and M. Thomas, “Advanced Sockets API for IPv6,” RFC2292, 1998.

[5] T. Narten, E. Nordmark and W. Simpson, “Neighbor Discovery for IP Version 6 (IPv6),” RFC2461, 1998.

[6] S. Thomson and T. Narten, “IPv6 Stateless Address Autoconfiguration,” RFC2462, 1998.

[7] 福原一朗, 国武功一, 吉藤英明, 岡村耕二, 楠根雄志, 関谷勇司, “Linux でつなぐ IPv6 の世界,” Linux Conference ’99, 1999.

[8] S. Kent, R. Atkinson, “Security Architecture for the Internet Protocol,” RFC2401, 1998.

[9] D. Johnson, ”Mobility Support in IPv6,” RFC3775, 2004.

[10] J. Arkko, ”Using IPsec to Protect Mobile IPv6 Signaling Between Mobile Nodes and Home Agents,” RFC3776, 2004.

[11] USAGI Project, <http://www.linux-ipv6.org>.

[12] WIDE Project, <http://www.wide.ad.jp>.

[13] IPv6 Ready Logo Program, <http://www.ipv6ready.org>.

[14] TAHI Project, <http://www.tahi.org>.

[15] IPv6 普及・高度化推進協議会, <http://www.v6pc.jp>.

[16] Y. Sekiya, H. Yoshifuji, M. Kanda and K. Miyazawa, “Evaluation and Improvement of IPv6 Protocol Stack by USAGI Project,” Proceedings of Ottawa Linux Symposium 2002, pp.496-504, 2002.

[17] 吉藤英明, 神田充, 高宮紀明, 関谷勇司, 江崎浩, 村井純, “USAGI プロジェクトによる IPv6 基本ソフトウェアの開発,” 電子情報通信学会論文誌 B, Vol. J85-B, No.8, pp. 1139-1346, 2002.

[18] FreeS/WAN Project, <http://www.freeswan.org>

[19] International Kernel Crypto API for GNU/Linux, <http://sourceforge.net/projects/cryptoapi/>.

[20] Netfilter Project, <http://www.netfilter.org>

[21] 小堺康之, 吉藤英明, 江崎浩, 村井純, “Linux における IP バージョン非依存なコネクショントラッキングの設計と実装,” 電子情報通信学会 技術研究報告, Vol.104, No.690, pp.29-33, 2005.

[22] J. Salim, H. Khosravi, A. Kleen and A. Kuznetsov, “Linux Netlink as an IP Services Protocol,” RFC3549, 2003.

[23] Mobile IPv6 for Linux, <http://www.mobile-ipv6.org>

[24] D. Harkings, D. Carrel, “The Internet Key Exchange (IKE),” RFC2409, 1998.

[25] C. Kaufman, Ed., “Internet Key Exchange (IKEv2) Protocol,” RFC4306, 2005.

[26] S. Sugimoto, ”PF_KEY Extension as an Interface between Mobile IPv6 and IPsec/IKE,” draft-sugimoto-mip6-pfkey-migrate-02.txt, 2006.

[27] USAGI Testlab, <http://testlab.linux-ipv6.org>, USAGI Project.

[28] IPv6-Fix Project, <http://v6fix.net>.

[29] 吉藤英明, 廣海緑里, “IPv6-Fix: IPv6 普及期に向けた取り組み,” 情報処理学会 情報処理, Vol.46 No.8, pp.887-893, Aug. 2005.