# USAGI IPv6

神田 充* / USAGI Projcect†

2002 年 8 月 19 日

## 1 はじめに

Linux における IPv6 に関する状況は、各種 BSD や WindowsXP などの OS に比べ出遅れており、また準拠している仕様も古くきちんとメンテナンスがされているという状況ではなかった。そのため USAGI プロジェクトが結成されこれらの状況の改善のために活動を行なってきている。

今回、プロジェクトの進捗及び独自に実装を行なっている IPsec の実装に焦点をあて説明する。

## 2 Linux における IPv6

### 2.1 USAGI プロジェクトによる改善

現状の Linux のカーネル/ライブラリ/アプリケーションにて未実装となっている主な機能の中で USAGI プロジェクトにおいて実装/統合された機能は以下である。

- IPv6 over ATM の実装
- IPv6 over ARCnet の実装
- SNMPv6 の改善
- 近隣探索プロトコルの改善
- アドレス自動設定機能の改善
- ICMP Node Information Query

*(株) 東芝研究開発センター
†http://www.linux-ipv6.org/

- Anycast
- longest prefix match による Source Address Selection
- Privacy Extension
- ISATAP
- IPv6/IPv4 over IPv4 tunnel
- IPv6 IPsec
- HUT MobileIPv6[1] の統合
- IPv6 API に関する library の提供
- 基本的な IPv6 アプリケーションの提供

# 3 USAGI IPsec 実装

Linux における IPsec 実装としては FreeS/WAN プロジェクト[2]が有名であるが、その実装は IPv4 のみでありまたそのカーネル空間の実装は強く IPv4 に依存している。そのため USAGI プロジェクトでは一から実装することにした。現在、ほぼ IPsec の実装を終え安定化、効率化、機能拡張作業に移っている。

# 4 IPsec アーキテクチャ

IPsec を適用するためには IP ヘッダとデータグラムの間に専用のヘッダを挿入して利用する。IP ヘッダを含むパケット全体の完全性を保証するための AH (Authentication Header) ヘッダ[3]とパケットのデータを暗号化及び完全性を保証するために使用される ESP(Encapsulating Security Payload) ヘッダ[4]の 2 つがある。

IPsec 全体のアーキクチャとしては、IPsec を利用するための適用方針を決める Security Policy Database(SPD)、パケットに適用する各種パラメータ (暗号化アルゴリズム、認証アルゴリズム、鍵) である SA(Security Association) を格納する Security Association Database(SADB)、及び実際にそれらをパケットに適用し処理する部分から成る。

[1] http://www.mipl.mediapoli.com/
[2] http://www.freeswan.org/
[3] RFC2402
[4] RFC2406

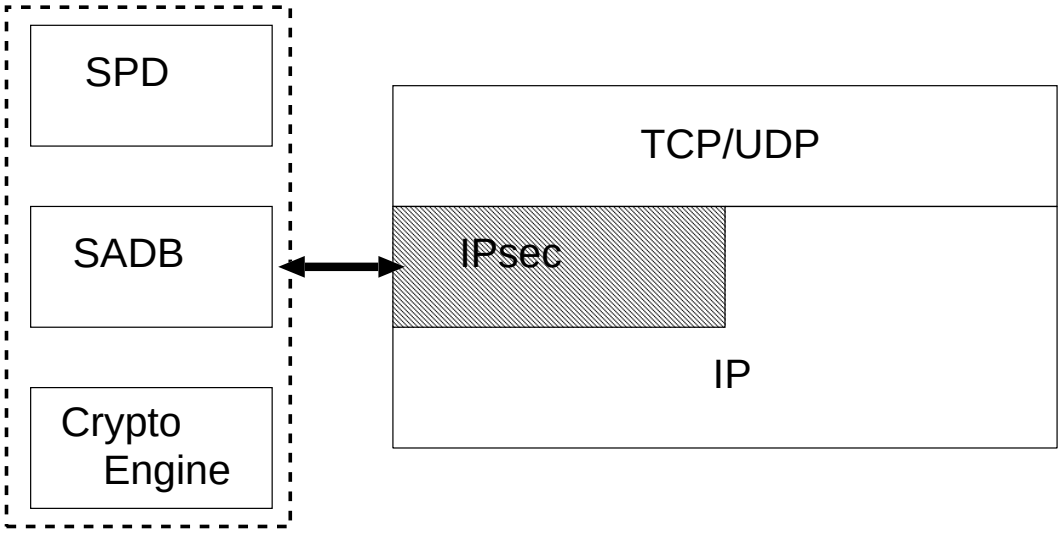


図 1: USAGI IPsec アーキテクチャ

またモードとしてはトンネルモードとトランスポートモードの 2 種類が定義されている[5]。

USAGI 実装では、IPsec の処理部分は IP スタック内に実装している (図 1)。

## 4.1 SPD と SADB

SPD と SADB は本質的に IP のバージョンを問わない。そのため、IPv4 及び IPv6 の両方に対応できるようにするためカーネル内に sockaddr_storage 構造体を導入することにより両方のバージョンに対応した。

設定のためにユーザ空間とカーネル空間とのやりとりには、PF_KEY[6] プロトコルが使用されるが、RFC に規定されているだけでは不十分なため各実装において独自に拡張が行なわれているのが現状である。USAGI プロジェクトでは PF_KEY の拡張部分は、互換性を考慮して FreeS/WAN プロジェクト実装と同じになるようにしてある。

また、手動で Security Policy 及び Security Association を設定するために pkfey コマンドを用意している。

[5] RFC2401
[6] RFC2367

## 4.2 暗号アルゴリズム、認証アルゴリズム

IPsec において使用する暗号アルゴリズム及び認証アルゴリズムは CryptoAPI[7] の枠組みを導入し使用している。この枠組みは暗号化、復号化などが関数ポインタ化されており、これにより各種アルゴリズム操作用のインターフェースを抽象化している。

現在、USAGI IPsec 実装にてサポートしているアルゴリズムは以下の通りである。

暗号化アルゴリズム:

- DES-CBC
- 3DES-CBC
- AES
- NULL

認証アルゴリズム:

- HMAC-MD5
- HMAC-SHA-1

## 4.3 IPsec トランスポートモードパケット処理

Linux カーネルの IP(v6) スタックにおけるパケット送出ルーチンは一つではなく上位プロトコル毎に複数の関数に分かれている。そのため TCP 用の ip6_xmit() 関数、UDP/ICMP 用の ip6_build_xmit() 関数、近隣探索で使用される ndisc_send_na() 関数、ndisc_send_ns() 関数、ndisc_send_rs() 関数それぞれに IPsec 処理ルーチンを導入している。

受信ルーチンについては、ip6_input_finish() 関数て統一されているのでこの関数に IPsec 処理ルーチンを導入するだけで済んでいる。

### 4.3.1 送信処理

IPsec を適用すべきパケットの送信処理の流れは以下のようになっている:

1. IPsec Policy のチェック
2. IPsec SA の検索
3. IPsec の適用 (暗号化等のパケットに対する具体的処理)

[7] http://www.kerneli.org/cryptoapi/

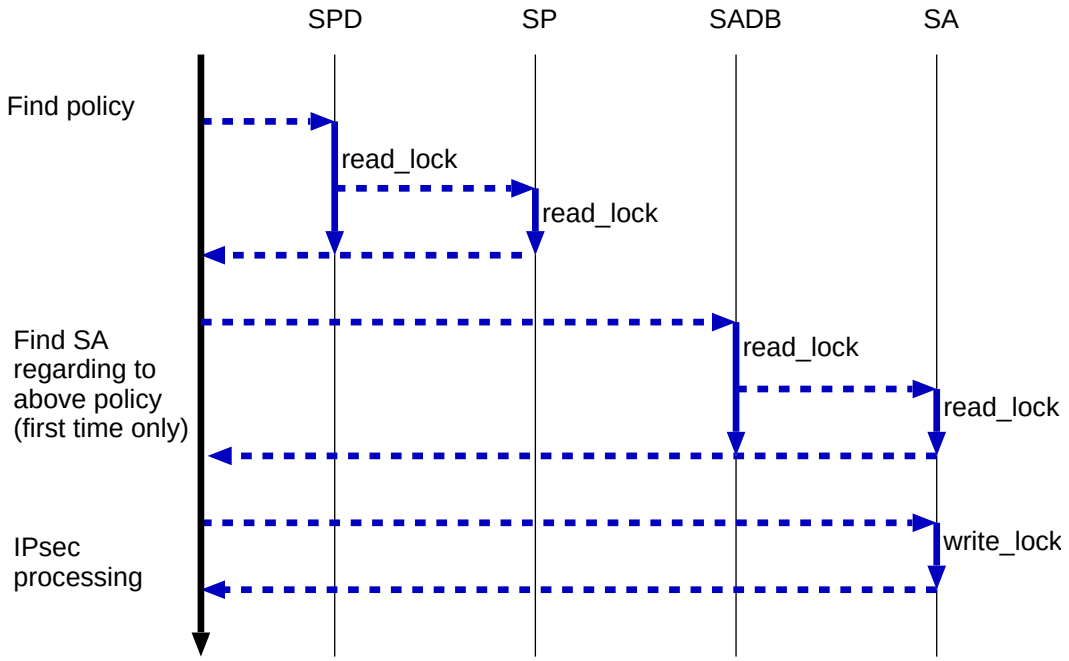


図 2: 送信処理の流れとロック

4. 送信

上記の流れと SPD、SADB 及びそれぞれの各要素のロックの状態を図 2 に示す。

### 4.3.2 受信処理

IPsec パケットの受信処理の流れは以下のようになっている:

1. 受信
2. AH/ESP ヘッダの SPI(Security Parameter Index) より IPsec SA の検索
3. AH ヘッダの場合完全性のチェック、ESP ヘッダの場合復号化を行う
4. IPsec Policy をチェック

上記の流れと SPD、SADB 及びそれぞれの各要素のロックの状態を図 3 に示す。

## 4.4 IPsec トンネルモードパケット処理

IPsec トンネルモードは IPv6 over IPv6 トンネルデバイスを使用して実現している。この方法の利点は基本的に外側 IP ヘッダの処理と内側 IP ヘッダの処理を

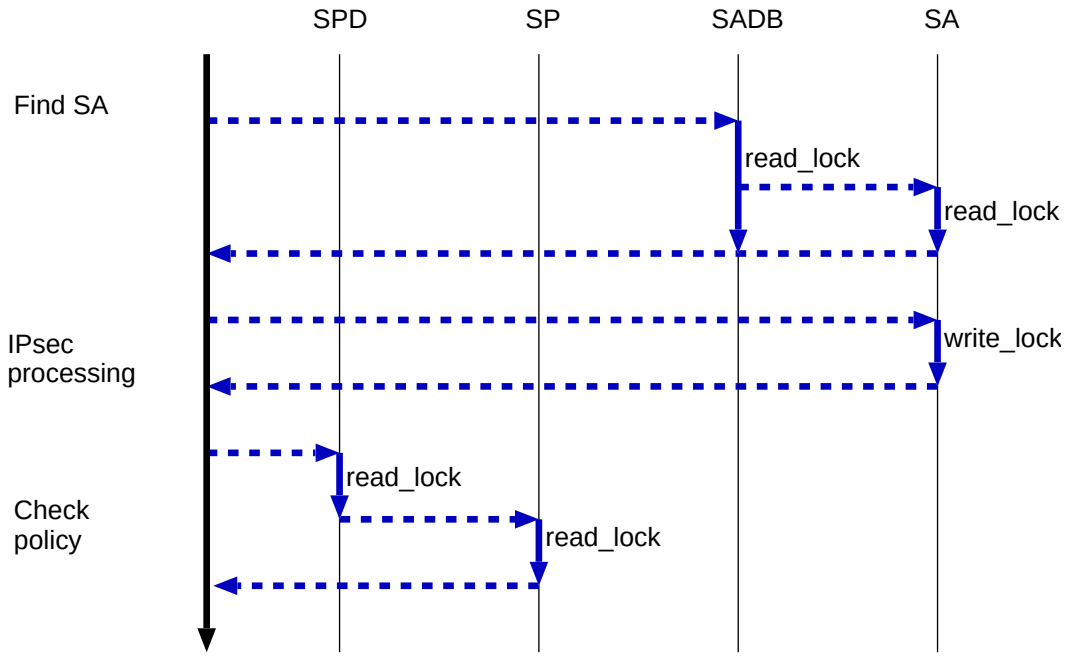


図 3: 受信処理の流れとロック

IPsec トランスポートと同じルーチンで実行できるため簡潔に実現できることである。

### 4.5 IKE

IPsecで利用するためのSAを自動的に交換するためのプロトコルであるIKE(Internet Key Exchange)については、FreeS/WANプロジェクトの実装であるPlutoをIPv6化し動作するように変更した。

また、現在 IETF[8] の IP Security Protocol(ipsec) ワーキンググループにおいて次期鍵交換プロトコルの仕様策定が始められている[9]。USAGI プロジェクトでは、仕様が策定され次第実装に移る予定である。

## 5 IPsec Conformance Test

仕様の適合性を確認するため TAHI[10]プロジェクトのテストツールを使用した。テストに使用した USAGI linux カーネルのソースコードは 2002 年 7 月 30 日の CVS 版である。テスト結果の概要を表 1 に示す。いくつかを除いてほぼ PASS し

[8] http://www.ietf.org/
[9] 現在の IKE とは互換性がない。
[10] http://www.tahi.org/

ている。FAIL している項目は Destination ヘッダとの組み合わせのテストであった。この部分は IPsec というよりも IPv6 スタックの Destination ヘッダの処理に起因するものであるため改善の必要がある。

| Host Transport mode | |
|---|---|
| PASS | 58 |
| FAIL | 2 |
| WARN | 1 |
| Router Tunnel mode | |
| PASS | 53 |
| FAIL | 3 |
| WARN | 1 |

表 1: TAHI テスト結果 (IPsec)

### 5.1 その他

我々の IPsec アーキテクチャは基本的に IP のバージョン依存/非依存の部分をきちんと分離してあるため、参考にではあるが IPv4(トランスポートモード) も実装してある。

## 6 カーネルへのマージ

現在、Linux カーネルにコミットされたパッチは細かいバグフィックスが中心である。ただ、先日行なわれたカーネルサミットにて開発版カーネルの新機能取り込み期限 (code freeze) が今年の 10 月末に設定されたため、どれだけ受け入れられるかはこれからの努力次第であるが少くとも 10 月初めまでに送付する予定である。

## 7 今後の予定

現在、PF_KEY の拡張にて IPsec Policy を設定しているがこれを Netlink を利用して設定できるようにすることを考えている (図 4)。また、IKEv2 の仕様が策定され次第、実装を始める予定である。

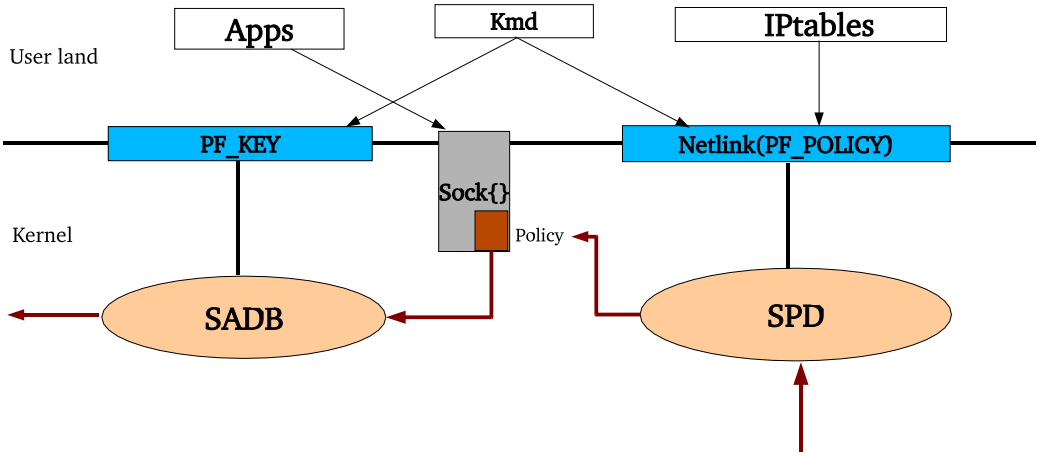


図 4: Netlink を使用した Security Policy の設定