第 XIII 部

IP パケットの暗号化と認証

第 13 部
IP パケットの暗号化と認証

第 1 章 はじめに

IPsec WG は今年度、春合宿、春研究会、秋合宿と BoF を開催した。本報告では IPsec 鍵交換デーモンである racoon2 の現在のステータス、および NTT の小早川氏による Plug and Play IPsec、ソニーの濱野氏による詐称 IPsec パケットの多量送付による IPsec プロトコルスタックへの影響についての研究発表について報告する。

第 2 章 racoon2

東京大学の鎌田健一氏<kamada@hongo.wide.ad.jp>、横河電気の坂根昌一氏<sakane@tanu.org>、宮澤和紀氏<kazunori@miyazawa.org>および報告者（東芝　神田充<mk@isl.rdc.toshiba.co.jp>）により、IPsec の鍵交換プロトコルである IKEv1 をサポートした鍵交換デーモンである racoon の後継実装として各種 BSD/Linux 上で動作し、複数の鍵交換プロトコルをサポートする鍵交換デーモン racoon2 を開発している。

IPsec[1]において通常使用される IPsec のための鍵交換プロトコルとして IKE（Internet Key Exchange）[2]がある。現在 IETF IPsec WG において IKE の改訂作業が進んでおり IKEv2 と呼ばれている。また、それとは別に Kerberos の認証システムを利用した鍵交換プロトコル（KINK）が、KINK WG において策定が行われている。racoon2 においてはこれら複数の IPsec のための鍵交換プロトコルをサポートすることとし、このためのアーキテクチャが必要となる。

2.1 サポート予定鍵交換プロトコル

racoon2 においては IKEv1、IKEv2 および KINK を鍵交換のプロトコルとしてサポート予定である。

2.2 racoon2 アーキテクチャ

racoon2 において考えられているアーキテクチャを図 2.1 に示す。

我々がサポート予定である各種 BSD/Linux において IPsec のパケット処理部、IPSec セキュリティアソシエーションデータベース（SAD、IPsec Security Association Database）、IPsec セキュリティポリシーデータベース（SPD、IPsec Security Policy Database）はカーネル内部に実装されており、IPsec のための鍵交換デーモン（racoon2）はユーザランドにて実装するため、これらの間において鍵などの設定は PF_KEYv2[3]およびその拡張を用いて設定を行う。

全体の構成としては、racoond と呼ばれるデーモンが IPsec セキュリティポリシーの設定の全てを担当する。各鍵交換デーモンが鍵交換の結果得られた SA（Security Association）は、各デーモンが直接 PF_KEYv2 によってカーネルに設定する。

2.2.1 IKE

IKEv1 と IKEv2 の違いは、主に鍵交換におけるプロトコルによるものが多く、基本的なアーキテクチャに関する違いは少ない。IKEv2 の仕様でカーネルに与える影響が大きいと思われるものは、Trafffic Selector である。Traffic Selector は、IPsec が適用されるトラフィックを識別するものであるが、この Traffic Selector の設定はアドレスの範囲で行う。PF_KEYv2 においてはこのような指定方法で設定できないため、適時 PF_KEYv2 において解釈できるように変更して設定していく必要がある。

また IKEv1 と IKEv2 においてはデフォルトで使用するポート番号は 500 である。このため IKE デー

1 RFC2401
2 RFC2409
3 RFC2367

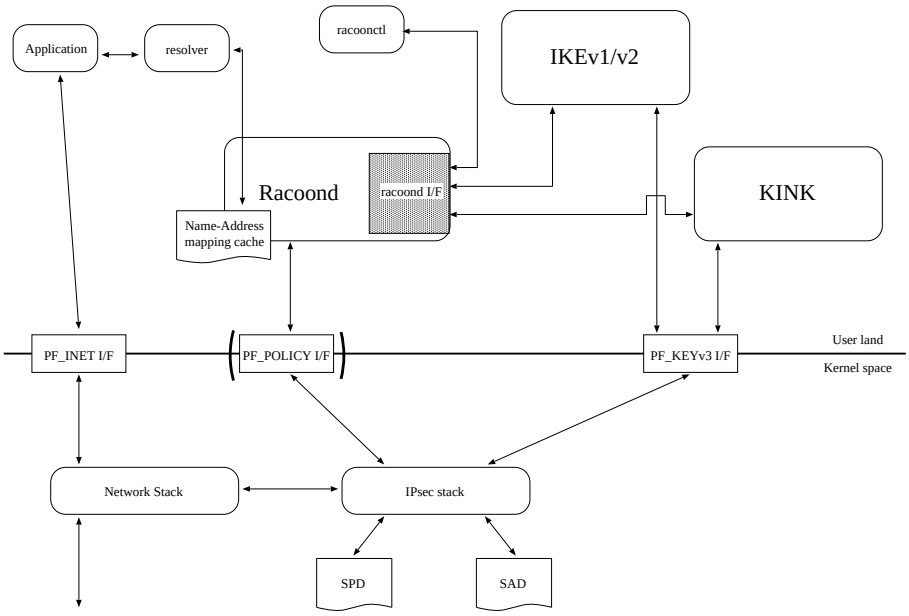

図 2.1.　racoon2 アーキテクチャ図

モンは来たパケットの中身を判断して IKEv1 のパケットであるか IKEv2 のパケットであるかを判断し各処理部に引き渡す必要がある。

2.2.2 KINK

KINK は、お互いの認証と通信の暗号化に Kerberos を用いるプロトコルである。IPsec SA 折衝のメッセージには、IKEv1 でも用いられている ISAKMP ペイロード[4]を再利用している。Kerberos では、プリンシパル名とレルム名と呼ばれる識別子によって、相互に認証が行われる。KINK で用いるプリンシパル名は、FQDN から導出することが決められている。そのため IPsec との整合をとるためには FQDN と IP アドレスの関連づけが必要になる。

尚、KINK がデフォルトで使用するポートはまだ決まっていない。

2.2.3 FQDN のサポート

先程述べたように KINK では、プリンシパル名とレルム名に基づいて認証する。プリンシパル名は FQDN にマップされる。しかし、IPsec は、IP 層に実装される技術であるため IP アドレスに基づいて処理される。このため直接 FQDN を扱うことはできない。よって IPsec の各種処理を行うためには FQDN と IP アドレスの変換が重要となってくる。これらの問題は KINK だけでなく、IKE にて行う IPsec の初期設定を FQDN で指定できるようにする時にも生じる問題である。

例えば、あるホスト宛の通信を IPsec を用いて保護する場合で、アプリケーションが IP アドレスを用いる場合を考える。

カーネル内の SPD にはあらかじめ IP アドレスで指定された IPsec セキュリティポリシーがある。アプリケーションは、IPsec セキュリティポリシーに設定されているものと同じ IP アドレスを指定して通信を行う。このとき、カーネル内の SAD に適切な IPsec SA が無ければ、ユーザーランドの鍵交換デーモンに IPsec SA の設定（のための鍵交換）を要求する。この要求は、IP アドレスに基づいて行われるため FQDN は使用されない（使用できない）。

そのため FQDN を用いる場合には、以下のような問題点がある

1. IPsec セキュリティポリシーの設定

   FQDN を識別子として IPsec を使用する場合、IPsec セキュリティポリシーは初期設定時に決めることができない。なぜならば、IPsec セキュリティポリシーは IP アドレスを用いて指定する必要があり、IP アドレスは FQDN を DNS などによって名前解決した時に初めて決定されるからである。

4　RFC2408

2. IPsec セキュリティポリシーとアプリケーションが使用する実際の IP アドレス
   FQDN と IP アドレスの対応が決定したとしても、必ずしもアプリケーションがその IP アドレスを使用するとは限らない。
3. FQDN による設定と PF_KEYv2 の SADB_ACQUIRE メッセージ
   FQDN を用いて IPsec セキュリティポリシーを指定し、IP アドレスが決定してカーネルに設定されたとしても、カーネルから鍵交換デーモンへの IPsec SA 設定要求メッセージである SADB_ACQUIRE には識別子として IP アドレスが入るために、そのままでは元の FQDN との対応ができない。

これらを解決（FQDN をサポート）するために、racoon2 では racoond と呼ばれる中央デーモンを置く。racoond は racoon2 の初期設定ファイルより FQDN で指定されている部分を読みこみ記憶しておく。そして racoond は、システム内で動作するアプリケーションのための DNS proxy サーバとして動作することにより、アプリケーションがネットワークを介した通信を行う場合にその名前解決の結果を（該当する FQDN ならば）キャシュしておく。これによって鍵交換デーモンが、ある IP アドレスが設定ファイルにて指定されている FQDN に対応するものであるかどうかを racoond に問い合わせることにより、FQDN と IP アドレスとの対応解決を行う。

2.3 現在のステータス

2.3.1 libracoon

IPsec セキュリティポリシーや、IPsec セキュリティアソシエーションの設定および設定ファイルの読み込み部分など、各デーモン共通部分は共通ライブラリとして作成予定であるがまだ実装されていない。

2.3.2 racoond モジュール

FQDN と IP アドレスとの対応関係をとるための、DNS proxy サーバ機能（hosts ファイルなどを読み込む機能を含む）は実装済みである。また、各鍵交換デーモンとの通信や、管理者が設定などを行うための通信処理も実装済みである。後は初期設定ファイルの読み込み部分と、カーネルへの IPsec セキュリティポリシー設定部分であるが、これらは共通ライブラリを用いて実装する予定であるため、これらが完成次第実装を行う予定である。

2.3.3 KINK モジュール

現在、draft-ietf-kink-kink-05.txt をもとに実装を行っている。KINK（および Kerberos）のメッセージ交換モードは、client-server モードによる optimistic approach のみが実装されている。user-to-user モードおよび 3-way handshake は、まだ実装されていない。

KINK メッセージは、CREATE、REPLY メッセージの送受信が可能で、鍵交換および期限の切れた鍵の再交換ができる。3-way handshake が実装されていないため、ACKREQ ビットは処理されていない。

KINK ペイロードは、KINK_AP_REQ、KINK_AP_REP、KINK_KRB_ERROR、KINK_ISAKMP、KINK_ENCRYPT、KINK_ERROR の送受信が可能である。user-to-user モードが実装されていないため、KINK_TGT_REQ、KINK_TGT_REP は未実装である。internet-draft では KINK_ISAKMP ペイロードは複数個存在することができるが、現在は 1 つのみ処理可能である。KINK_KRB_ERROR ペイロードに関しては、

1. Kerberos のサービスキーが利用可能な状況では、完全性を守る必要があるが、まだ実装されていない
2. Responder では、受信した KINK_KRB_ERROR が認証されていない場合には、適切に処理する必要があるが、まだ行われていない

など、課題が残っている。

DPD（Dead Peer Detection）は、CREATE または REPLY メッセージによって受動的に発見される状況のみ実装されている。すなわち、通常のメッセージ交換が行われる際に、通信相手の epoch が変化しているのを発見すると、その相手に関する既存の SA を消去する。ICMP メッセージや IPsec 通信の消失など、他の要因による DPD 処理の開始は実装されていない。

ISAKMP

ISAKMP に関しては、Security Association ペイロード、Nonce ペイロードの処理のみが racoon(v1) から取り込まれている。

PF_KEY

PF_KEYv2 インターフェイスを介して、SADB_GETSPI、SADB_ACQUIRE、SADB_EXPIRE メッセージの受信、SADB_GETSPI、SADB_ADD メッセージの送信が可能である。sadb_msg_satype メンバ、SPD、設定ファイルなどを参照して、要求されたパラメータの SA を作る機能はまだ実装されていない。現在は ESP のみを作成することができる。

racoond インターフェイス

racoond インターフェイスは、LOGIN、QUIT メッセージによるセッションの開始・終了、および、FQDN QUERY メッセージによる IP アドレスから FQDN の問い合わせが実装されている。

2.3.4 IKE モジュール

racoon2 の IKE モジュールは、IKEv1 と IKEv2 をサポートする。基本的な構成は、racoon に準じ、タイマベースのスケジューラに基づいて動作する。IKE モジュールは、libracoon による設定ファイルデータベース、PF_KEY インタフェース、socket インタフェースとそれぞれのプロトコル処理部分から構成される。現状は、旧 racoon アーキテクチャをベースに IKEv2 を試験的に実装した段階であり、新しい racoon2 のアーキテクチャへ移行できていない。IKEv2 には、EAP などオプショナルな機能がいくつかあるが、これらの実装は、基本部分の実装が完了してからになる予定である。IKEv1 も同様に、racoon2 アーキテクチャへの移行ができていない。

第 3 章　Plug and Play IPsec

一般の人々が IPsec を利用する上での一番の障害になっていると思われる IPsec の鍵の設定を家電のように簡単に扱えるようにするための仕組み（IPv6 上）が、NTT Communications の小早川知昭氏 <tomoaki@nttv6.jp> と宮川晋氏 <miyakawa@nttv6.jp> により発表された。

3.1 仕様

必要なプレイヤーは通信するためのノードと認証サーバである。ノードと認証サーバとは pre-shared key にて暗号化通信を行う。

ノード間の通信 phase は 3 つ：

1. phase1　生通信（IPsec 暗号化なし）
2. phase2　未認証暗号化（IPsec 暗号化のみ）
3. phase3　（サーバによる）認証済通信

3.2 IPsec Discovery option の導入

IPv6 Extension header に IPsec Discovery option header を導入し、この中にノードがサポートする認証サーバ名を埋めこみ、通信ノード同士が使用できる認証サーバを介して互いを認証する仕組みを導入する。また、鍵交換そのものは IKE に準じたプロトコルを使用し、まず暗号化の鍵を交換し、（必要なら）認証サーバを通した認証を行う。

第 4 章　IPsec DoS 脆弱性評価と対抗策の提案

存在する IPsec SA に該当する不正な IPsec packet を大量に送信することにより、攻撃対象ノードの無駄な暗号化処理を増やすタイプの攻撃に対する防御法として、preclude function を用いた IP オプションを定義して使用する方法が、濱野淳史氏 <atsu@sm.sony.co.jp>より秋合宿の IPsec BOF にて発表された。

4.1 IPsec DoS 脆弱性/脅威モデルの定義

まず、「IPsec DoS 脆弱性とは何か」の定義として

- 詐称 IPsec パケットの送付

を考えたが、詐称パケットか否かは暗号を解くまでわからない。このため脅威モデル（Threat Model）として、

- 被害ノードで、正当な通信相手との IPsec SA が設定されている
- 攻撃者はその IPsec SA に該当する<dstIP, SPI, protocol>とその現在のシーケンス番号を知っている

という前提条件のもとで、攻撃者は IPsec SA に該当する不正な IPsec packet を送付することにより

- 被害 node の CPU 資源の消費（問題点 1）

- 正当な通信相手からのパケットの処理あふれ（問題点 2）

を起こす可能性があるとした。

4.2 脆弱性の定量評価

定量評価のために 1,500 Byte の詐称 IPsec packet を実際に送付する実験を行った。実験環境として

1. OS: NetBSD1.6
2. CPU: Pentuim4 2 GHz
3. Memory: 512 MByte
4. NIC: GbE

を利用した。図 4.1 に、あるプロセスの実行時間の、攻撃トラフィックが無い状態との比率を示す。

この結果より、3DES CBC を利用した ESP の場合は 80 Mbps、HMAC SHA-1 を利用した AH の場合には 150 Mbps 付近で、実行時間が発散していることがわかり、IPsec のための暗号処理時間が飽和してユーザプロセスがほとんど停止しているといえる。

この状態をモデル化した式は以下の通りとなる。

$$ratio = \frac{1}{1 - (T_d + T_p + T_c)R_i}$$

T_d: データリンクでの処理時間
T_p: プロトコルの処理時間
T_c: 暗号処理時間
R_i: 攻撃レート

4.3 解法

既存の手法での解決としてはRED（Random Early Detection）、RTOS、受信キューをわけるなどが考えられるが、問題点 1 の「他の処理に CPU が割り当てられなくなる」は解決できるが問題点 2 を解決することは難しい。そもそもの問題は、被害ノードで要求されるよりも少ない計算量で攻撃側がパケット送付できてしまうということから、解決の糸口として

- 詐称パケットだと判定するまでに必要な処理量を下げる（要件 1）
- 送受信者のみ知る秘密情報を IPsec packet にいれる（要件 2）

を考慮にして以下のような解法を設計した。

要件 2 を満足するために、図 4.2 に示す Preclude Function を用いる。Preclude Function では、送受信者であらかじめ疑似乱数生成器の seed を共有し、導出された疑似乱数を一方向関数に通してやることにより、不可逆性を保証する。ここで必要とされる処理量が、暗号処理と比較して絶対的に少ないため、要件 1 を満足できる。そして、Preclude Function により導出された値（reduced pseudo-random number）と一緒にシーケンス番号を送信することにより、受信側で検証すべき reduced pseudo-random number を導出できる。

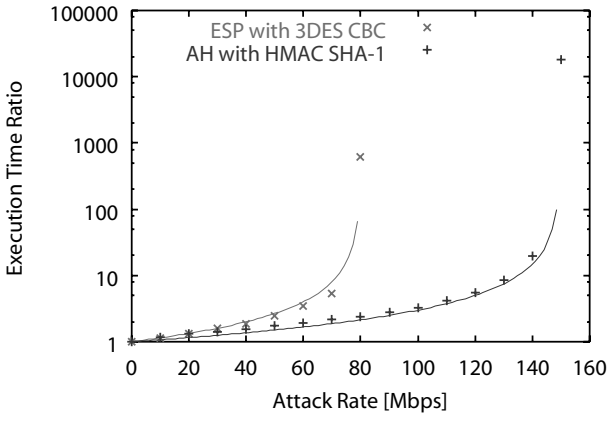

図 4.1. 定量評価結果

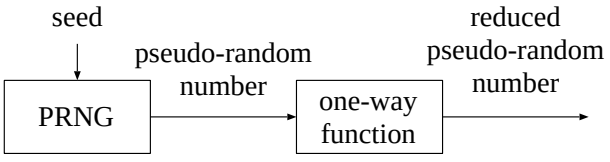

図 4.2. Preclude Function

以上により、4.2 節においてモデル化した式は、以下の通りとなる。

$$ratio = \frac{1}{1 - (T_d + T_p + T_I)R_i}$$

T_d: データリンクでの処理時間

T_p: プロトコルの処理時間

T_I: IPsec DoS 対策（Preclude Function）のための処理時間

R_i: 攻撃レート

暗号処理時間 T_c が、IPsec DoS 対策（Preclude Function）のための処理時間 T_I で置き換えられ、T_I が T_c と比較して絶対的に小さいので、*ratio* を抑えることができる。

4.4 実装

4.3 節において提案した解法を、NetBSD 1.6 に IPv4 オプションとして実装した。Preclude Function としては、送受信者であらかじめ共有しておいた乱数を seed として、線形合同法（32 bit）により導出された疑似乱数の上位 16 bit を取得することにより reduced pseudo-random number とした。そして、シーケンス番号とともに、図 4.3 に示した IPv4 オプションに含めて送信するようにした。

受信側でこのオプションを受信した際には、オプションに含まれている reduced pseudo-random number の妥当性を検証した後に、正当だと判断された場合のみ、IPsec の受信処理を行うこととした。

4.5 評価結果

本方式の処理オーバーヘッドと有効性を、4.4 節において実装したものを利用して 4.2 節に示した環境で評価した。

処理オーバヘッドについては、1 KByte の UDP パケットの転送に要する時間を Pentium clock counter により測定した。本方式を実装していないものと比較した処理オーバーヘッドは、以下の通りである。この結果から、処理オーバーヘッドは微小であり、実用に耐え得るものであるといえるだろう。

出力処理	1.5–3.3%
入力処理	1.7–2.0%

次に、4.2 節において行った脆弱性の定量評価を、本方式に対しても行うことにより、有効性の検証を行った。図 4.1 と同じ条件での評価結果を図 4.4、図 4.5 に示す。なお、これらの図には、図 4.1 のデータも含まれている。

これらの結果より、あるプロセスの実行時間の、攻撃トラフィックが無い状態との比率が劇的に改善されていることがわかる。したがって、本方式は有効であるといえるだろう。

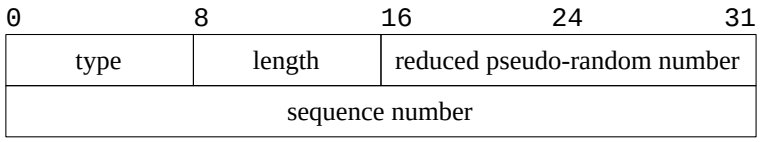

図 4.3. IPv4 option for Preclude Function

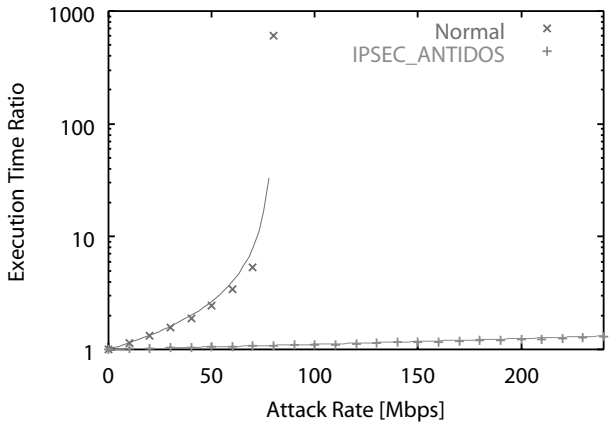

図 4.4. 3DES CBC を利用した ESP の場合の定量評価結果

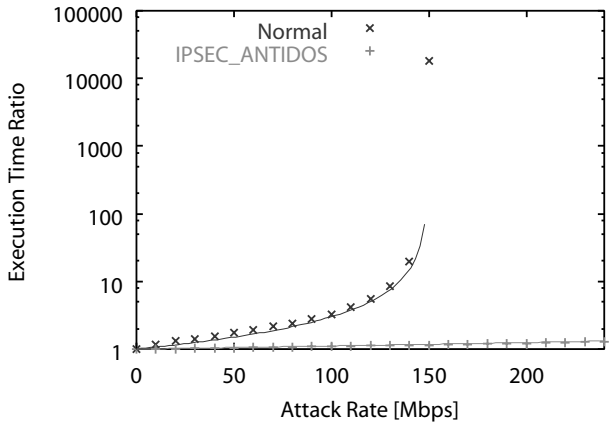

図 4.5. HMAC SHA-1 を利用した AH の場合の定量評価結果

4.6 考察

- Preclude Function の改良
  4.4 節において実装したものでは、疑似乱数生成器として線形合同法を採用した。しかしながらこの方式では、連続する 2 つの reduced pseudo-random number から全数探索により疑似乱数列を推測できてしまう。また Out-of-Sequence のパケットにも対応できないなどの問題点がある。そこで、再設計として keystream generator の採用を検討している。送受信者で（Seed ではなく）鍵を共有して、シーケンス番号を RC4 や AES CTR-mode で捻ることにより、攻撃者による推測にも耐え得り、また Out-of-Sequence にも対応できる。この方式については、現在実装中である。

第 5 章　まとめ

racoon2 に関しては 2004 年度も引き続き作業を進め、今年中に動作できるものの公開をめざしていきたい。