
IPsec HOWTO

Ralf Spenneberg <ralf (at) spenneberg.net>

SATOH, Satoru

<ss (at) gnome.gr.jp>

	2003-08-18	
	改訂履歴	
改訂 0.9.2	2003-08-19	RS
	Fixed a typo	
改訂 0.9.1	2003-08-18	RS
	Minor corrections	
改訂 0.9.0	2003-08-15	RS
	Added: Using the OpenBSD isakmpd	
改訂 0.8.3	2003-05-13	RS
	Further typos corrected. Some sentences rephrased.	
改訂 0.8.2	2003-05-03	RS
	Bugfixes	
改訂 0.8.1	2003-04-30	RS
	added chapter covering certificates	
改訂 0.8	2003-04-18	RS
	first draft	

この HOWTO 文書は Linux カーネル 2.4 そして 2.5/2.6 での IPsec を用いて VPN をセットアップする基本的、またより高度なステップについてふれるつもりです。Linux カーネル 2.4 については既に非常に沢山の文書がありますので、この文書では開発版カーネルでの新規の IPsec 機能の方にまず集中します。より後の版では Linux カーネル 2.4 についても含めるようにします。

目次

1. はじめに	2
1.1. この HOWTO を書いたわけ	2
1.2. この文書の形式	2
1.3. この文書の主な貢献者	2
1.4. 法的な情報	2
2. 理論	3
2.1. IPsec とは何か?	4
2.2. IPsec プロトコル	5
2.3. IKE プロトコル	7
3. Linux カーネル 2.2、2.4 -- FreeS/WAN	8
3.1. インストール	8
4. KAME ツールを使っている Linux カーネル 2.5/2.6 の場合	8

4.1. インストール	8
4.2. setkey を使った手動鍵接続	9
4.3. racoon を使った自動鍵接続	13
5. Linux カーネル 2.5/2.6 での OpenBSD isakmpd の利用	17
5.1. インストール	17
5.2. 事前共有鍵 (PSK) の利用	18
5.3. X.509 証明の利用	20
6. より高度な設定	21
6.1. DHCP-over-IPsec	21
6.2. NAT-Traversal	21
6.3. GRE-Tunnel	22
6.4. L2TP	22

1. はじめに

この文書の最新版は常に The Linux Documentation Project と公式ページ <http://www.ipsec-howto.org> に置いてあります。

1.1. この HOWTO を書いたわけ

私は過去に沢山の HOWTO 文書を利用しています。その大部分は私には非常に有用なものでした。Linux カーネルに新しい IPsec 機能が実装されたとき、すぐにそれをいじくり回しはじめ、そして関連文書が極小数のものに限られることがわかりました。それで私はこの HOWTO を書きはじめたのです。

1.2. この文書の形式

この文書は五つの章からなります。

セクション 1: はじめに	このセクション
セクション 2: 理論	IPsec の理論。IPsec プロトコルの基本。
セクション 3: Linux カーネル 2.2 および 2.4	このセクションでは FreeS/WAN の設定方法について説明しています。
セクション 4: Linux カーネル 2.5/2.6	このセクションでは KAME ツール (setkey と racoon) を利用した IPsec VPN の設定方法について説明しています。
セクション 5: より高度な設定	このセクションでは DHCP-over-IPsec、NAT-Traversal などを利用したより高度な設定について説明しています。

1.3. この文書の主な貢献者

- ・ Uwe Beck
- ・ Juanjo Ciarlante
- ・ Ervin Hegedus
- ・ Barabara Kane

1.4. 法的な情報

1.4.1. Copyright

Copyright (c) 2003 Ralf Spenneberg

この文書は任意のフォーマットで複製、配布(有償、あるいは無償で)して構いません。ただし、この文書への訂正、コメントなどは保守者まで返して下さい。以下の条件であなたは派生物をつくり、配布することができます:

- ・ あなたの作業した派生物を(SGML のような最も適切なフォーマットで) LDP (Linux Documentation Project) か the internet のその種の場所に送って下さい。LDP でないならどこでそれが取得可能なのかを LDP まで知らせて下さい。
- ・ 派生物をこれと同じものか GPL でライセンスして下さい。著作権表示と少なくとも利用するライセンスへのポインタを含めるようにして下さい。
- ・ 原作者と主要な貢献者に対して当然与えられるべき敬意をはらって下さい。

翻訳以外の派生物をつくることを考えているなら、現在の保守者とあなたのプランについて議論してみてください。

1.4.2. 免責条項

この文書について作者は暗にも明示的にも何ら責任を負っていませんし、何の保証も与えていません。あなたの犬が死んでしまっても作者は何も責任を負いません!

1.4.3. 訳者コメント

この和訳は私、佐藤 暁 (SATOHI Satoru - ss at gnome.gr.jp)が個人的な興味で訳したものです。訳の間違いなどすべての責任は私の方にありますので原著者に問い合わせたりせず、私の方で連絡をお願いします。原文とは異なり DocBook XML 4.2 形式で DocBook XSL Stylesheet を使って HTML に変換しています。訳は現在一時的に <http://bonobo.gnome.gr.jp/~ss/x/ipsec-howto/> として置いてありますが、最終的には JF に寄贈するのがいいのではないかと考えています。

なおライセンスなどは(当然のことながら)オリジナルに準じています。

最終更新: Tue 23 Sep. 2003

1.4.4. 関連文書

- ・ Networking Overview HOWTO, 同和訳
- ・ Networking HOWTO, 同和訳
- ・ VPN-Masquerade HOWTO,
- ・ VPN HOWTO, 同和訳
- ・ Advanced Routing & Traffic Control HOWTO

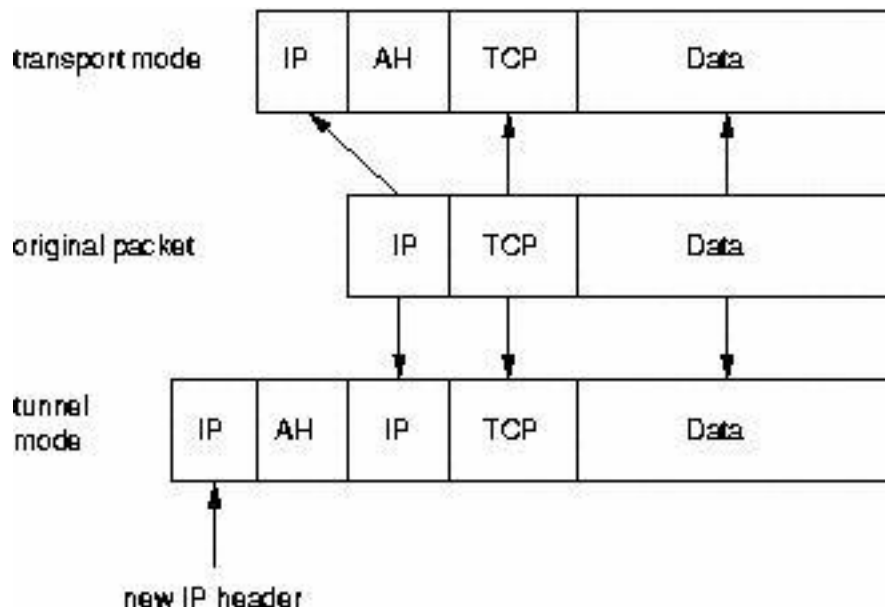
2. 理論

2.1. IPsec とは何か?

IPsec は IP とより上位のプロトコルにセキュリティを提供する IP プロトコルの拡張です。最初は新しい IPv6 標準のために開発され、それから IPv4 に“バックポート”されました。IPsec アーキテクチャは RFC2401 に記載されています [訳注: RFC2401 和訳]。以下の数段落で IPsec の簡略な説明をしたいと思います。

IPsec は通信の認証、完全性、機密性を保証するために二つの異なるプロトコル - AH と ESP - を使います。IP データグラム全体だけでなく、より上位のプロトコルだけに対しても保護することができます。トンネルモードとトランスポートモードという二つのモードがあります。トンネルモードでは IP データグラムは IPsec プロトコルを使った新しい IP データグラムに完全にカプセル化されます。トランスポートモードでは IP データグラムのペイロードだけが IPsec プロトコルによって処理され、IP ヘッダとより上位のプロトコルのヘッダとの間に IPsec ヘッダが挿入されます (図 1. 「IPsec トンネル、トランスポートモード」)。

図 1. IPsec トンネル、トランスポートモード



IP データグラムの完全性を守るために IPsec プロトコルはハッシュメッセージ認証コード (HMAC) を利用します。この HMAC を得るために IPsec プロトコルは MD5 や SHA のようなハッシュアルゴリズムを使い、秘密鍵と IP データグラムの内容に基づいてハッシュ値を計算します。そしてこの HMAC は IPsec プロトコルヘッダに含められ、パケットの受け取り手は秘密鍵にアクセスできれば HMAC の値を検証することができます。

IP データグラムの機密性を守るために、IPsec プロトコルは標準的な対称暗号アルゴリズムを利用します。IPsec 標準は NULL および DES の実装を要求しています。今日通常は 3DES、AES そして Blowfish といったより強力な暗号アルゴリズムが使われています。

サービス拒否攻撃(DOS)から守るために IPsec プロトコルはスライドウィンドウを利用します。各パケットには一連の数字が割当てられ、パケットの数字がそのウィンドウかより新しいものの中にある場合だけ受け入れられます。より古いパケットは即座に破棄されます。このプロトコルは攻撃者がオリジナルパケットを記録し、それを使って連続攻撃をしかけた場合に対抗するものです。

ピアが IPsec パケットをカプセル化し、またそれを元に戻すためには何らかの方法でその通信における秘密鍵とアルゴリズム、IP アドレスを保持しておく必要があります。IP データグラムを守るた

めに必要なこれらすべてのパラメータはセキュリティアソシエーション (SA) 内に、セキュリティアソシエーションはセキュリティアソシエーションデータベース (SAD) 内に、というように順ぐりに保存されています。

各セキュリティアソシエーションは以下のパラメータを定義しています:

- ・ 結果生成される IPsec ヘッダの送信元および送信先 IP アドレス、すなわちパケットを保護する IPsec ピアの IP アドレス。
- ・ IPsec プロトコル (AH あるいは ESP)。時には圧縮 (IPCOMP) もサポートされています。
- ・ IPsec プロトコルが利用するアルゴリズムと秘密鍵。
- ・ セキュリティパラメータ索引 (SPI)。これはセキュリティアソシエーションを識別するための 32 bit の数字です。

いくつかのセキュリティアソシエーションデータベースの実装ではさらにパラメータを保持しておくことができます:

- ・ IPsec モード (トンネルあるいはトランスポート)
- ・ 連続攻撃に対する保護のためのスライドウィンドウの大きさ
- ・ セキュリティアソシエーションの有効期限

セキュリティアソシエーションは送信元と送信先の IP アドレスを定義するので、全二重 IPsec 通信の内一方しか保護することができませんので、双方向の IPsec を保護するためには二つの単方向セキュリティアソシエーションが必要となります。

セキュリティアソシエーションは IPsec がどのように通信を保護するかを指定しているだけです。いつどの通信を保護するかを定義するための補足情報が必要となります。これは、セキュリティポリシーデータベース (SPD) に保存される、セキュリティポリシー (SP) に保存されています。

セキュリティポリシーには通常以下のパラメータを指定します:

- ・ 保護すべきパケットの送信元と送信先。トランスポートモードではこれらは SA 内で同一ですが、トンネルモードでは異なります!
- ・ 保護するプロトコル(とポート)。いくつかの IPsec 実装では保護する指定プロトコルを指定することができません。この場合指定した IP アドレスのすべての通信が保護されます。
- ・ パケットの保護に利用するセキュリティアソシエーション。

セキュリティアソシエーションの手動設定は非常に誤りを起しやすく、安全ではありません。秘密鍵と暗号化アルゴリズムは VPN 内のすべてのピアの間で共有されなければなりません。特に鍵の交換はシステム管理者に深刻な問題をもたらします。まだ暗号化できないのにどうやって対称な秘密鍵を交換すれば良いのでしょうか?

この問題を解決するためにインターネット鍵交換プロトコル (IKE) が開発されました。このプロトコルは phase 1 でピアを認証し、phase 2 でセキュリティアソシエーションをネゴシエーションし、対称秘密鍵を Diffie Hellmann 鍵交換法を使って選択します。それからなお IKE プロトコルはそれらの機密性を確保するために定期的に秘密鍵を変更します。

2.2. IPsec プロトコル

IPsec プロトコルファミリーは認証ヘッダ (Authentication Header, AH) とカプセル化セキュリティペイロード (Encapsulated Security Payload, ESP) の二つのプロトコルで構成されています。両方とも独立した IP プロトコルです。AH は IP プロトコル 51、ESP は IP プロトコル 50 (/etc/protocols を参照)です。以下の二つのセクションでこれらプロトコルについて簡単に述べます。

2.2.1. AH – Authentication Header (認証ヘッダ)

AH プロトコルは IP データグラム完全性を確保します。AH プロトコルは完全性を確保するために HMAC を計算します。HMAC を計算する際に AH プロトコルは秘密鍵、パケットペイロード、そして IP アドレスのような IP ヘッダの不変部分をベースにします。それからパケットに AH ヘッダを付加します。AH ヘッダは 図 2. 「パケットの完全性を確保する AH ヘッダ」のようになります。

図 2. パケットの完全性を確保する AH ヘッダ

Next Header	Payload Length	Reserved
Security Parameter Index (SPI)		
Sequence Number (Replay Defense)		
Hash Message Authentication Code		

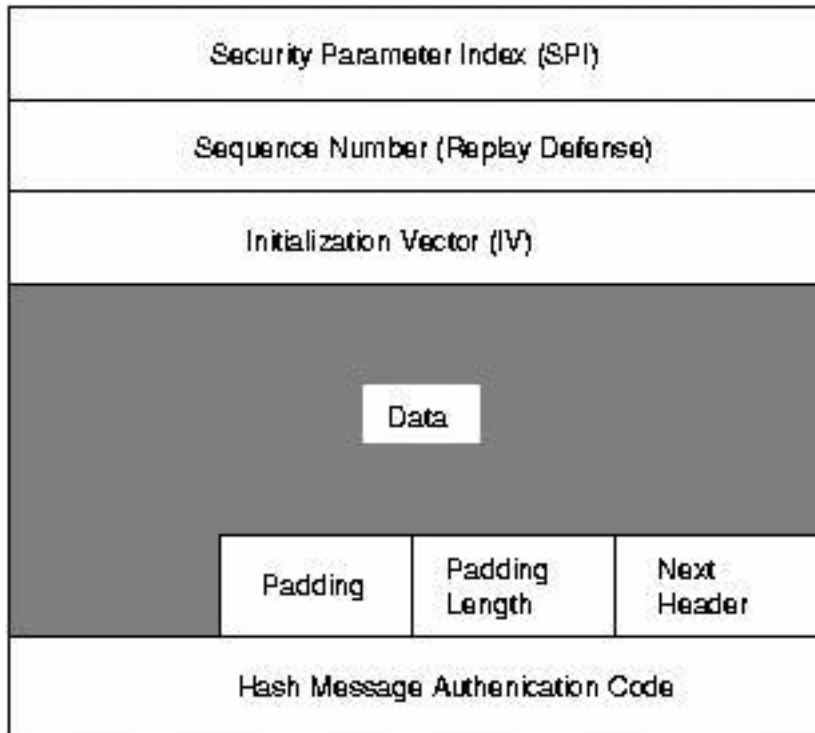
AH ヘッダの長さは 24 byte です。第一バイトは 後続ヘッダ (Next Header) フィールドです。このフィールドで後に続くヘッダのプロトコルを指定します。トンネルモードでは IP データグラムは完全にカプセル化されるので、この値は 4 となります。トランスポートモードで TCP データグラムをカプセル化するときこの値は 6 となります。次のバイトでペイロードの長さを指定します。このフィールドの後には二つの予約されたバイトが続きます。次の double word で 32 bit 長のセキュリティパラメータ索引 (SPI) を指定します。SPI はパケットのカプセル化を解くために使用するセキュリティアソシエーションを指示します。32 bit シーケンス番号 は連続攻撃に対する防御です。最後の 96 bit は ハッシュメッセージ認証コード (HMAC) を保持します。HMAC はピアだけが知っている秘密鍵から生成されるので、これを検証することでパケットの完全性を保証することができます。

AH プロトコルは、IP データグラムに含まれる IP ヘッダの中の IP アドレスのような改変不能な部分を保護しているので、NAT が使えません。ネットワークアドレス変換 (NAT) は IP ヘッダ内の IP アドレス (通常は送信元 IP) を異なる IP アドレスで置き換えます。この変換の後ではもはや HMAC は正しいものでなくなってしまいます。IPsec プロトコルの NAT-Traversal 拡張はこの制限の解決方法を実装しています。

2.2.2. ESP – カプセル化セキュリティペイロード

ESP プロトコルは HMAC を使ったパケットの完全性と暗号化による機密性を同時に実現できます。パケットを暗号化し HMAC を計算した後で ESP ヘッダが生成されパケットに付加されます。ESP ヘッダは二つの部分で構成され、図 3. 「ESP ヘッダ」に示されるようになっています。

図 3. ESP ヘッダ



ESP ヘッダの最初の double word には セキュリティパラメータ索引 (SPI) を指定します。SPI は ESP パケットのカプセル化を解くために使う SA を指示します。二番目の double word はシ連続番号で、これは連続攻撃から保護するのに使います。三番目の double word は暗号化プロセスで利用する 初期化ベクタ (IV) を指定しています。もし IV がないと対称暗号アルゴリズムは周期攻撃(a frequency attack)を許してしまいます。IV は異なる暗号化ペイロードに至る二つの識別可能なペイロードを保証しています。

IPsec は暗号化プロセスでブロック暗号を使うので、ペイロード長が複数ブロック長に達していない場合はペイロードにパディングが必要となります。それからパディング長が加えられます。パディング長に続いて 2 byte 長の 後続ヘッダ フィールドで後続ヘッダを指定します。最後に 96 bit 長の HMAC が ESP ヘッダに加えられ、パケットの完全性を保証します。HMAC はパケットのペイロードだけを考慮しています。IP ヘッダは計算プロセスに含まれていません。

よって NAT は ESP プロトコルを壊すことにはなりません、それでもなお多くの場合 NAT は IPsec と一緒に使うことができません。NAT-Traversal はこのような場合に UDP パケット内に ESP パケットをカプセル化するという解決策を提供します。

2.3. IKE プロトコル

IKE プロトコルは、セキュアな通信の設定において最も重要なピアの認証と対称鍵の交換の問題を解決します。それからセキュリティアソシエーションを生成し、SAD に移します。IKE プロトコルは通常ユーザ空間のデーモンを必要とし、OS 内には実装されません。IKE プロトコルはその通信に 500/UDP を利用します。

IKE プロトコル機能には二つの phase があります。phase 1 で インターネットセキュリティアソシエーション鍵管理セキュリティアソシエーション (ISAKMP) を確立します。phase 2 で ISAKMP SA を IPsec SA のネゴシエーションとセットアップに利用します。

phase 1 でのピアの認証は普通事前共有鍵 (PSK) と RSA 鍵、X.509 証明 (racoon はさらに Kerberos もサポートしています) に基づいて行うことができます。

phase 1 は通常二つの異なるモード、メインモードとアグレッシブモード、をサポートしています。どちらのモードもピアを認証し、ISAKMP SA をセットアップしますが、アグレッシブモードではそのた

めに半分のメッセージだけで済ませます。しかしこのために欠陥を持つことになり、というのはアグレッシブモードは識別保護をサポートしていないので、事前共有鍵と組合せて使ったときに中間一致攻撃 (man-in-the-middle attack) に対して脆弱となってしまいます。一方、これがアグレッシブモードの唯一の目的でもあります。メインモードの内部機能は不明なピアについて異なる事前共有鍵を使うことをサポートしていないからです。アグレッシブモードは識別保護機能をサポートしないので、クライアントの識別が明確になります。そういうわけでピアは認証を行う前に互いについて知り、異なるピアに対して異なる事前共有鍵を使うことができます。

phase 2 では、IKE プロトコルはセキュリティアソシエーション提案を交換し、ISAKMP SA に基づいてセキュリティアソシエーションを取り決めます。ISAKMP SA は中間一致攻撃 (man-in-the-middle attack) から防御するための認証法を提供しています。phase 2 はクイックモードを利用します。

普通、二つのピアは一つの ISAKMP SA だけについてやり取りし、その SA はそれからいくつか(少くとも二つ)の単方向 IPsec SA のネゴシエーションに利用されます。

3. Linux カーネル 2.2、2.4 -- FreeS/WAN

ToDo

3.1. インストール

...

4. KAME ツールを使っている Linux カーネル 2.5/2.6 の場合

この章では Linux カーネル >2.5.47、2.6.* のネイティブの IPsec スタックの使い方について説明します。IPsec スタックのインストールと設定は FreeS/WAN とは大きく異なり、FreeBSD や NetBSD、OpenBSD のような *BSD 系に似ています。

最初にまず Linux カーネルとユーザ空間ツールのインストールおよび設定について取り上げようと思います。それからトランスポート、トンネルモードでの手動鍵接続の設定について説明しようと思います。さらに事前共有鍵、X.509 証明を利用した自動鍵接続の設定について取り上げます。roadwarrior サポートについては最後に説明します。

4.1. インストール

インストールには最低でも Linux カーネルバージョン 2.5.47 か 2.6.* が必要となります。カーネルソースは <http://www.kernel.org> からダウンロード可能です。ソースをダウンロードしたら展開、設定、コンパイルしなければなりません。

```
cd /usr/local/src
tar xvjf /path-to-source/linux-<version>.tar.bz2
cd linux-<version>
make xconfig
make bzImage
make modules
make modules_install
make install
```

これらは Linux カーネルを設定し、コンパイルするためによく使われるコマンド群です。もし特別な設定が必要なら Kernel-HOWTO (JF の Kernel-HOWTO 和訳) を参照して下さい。

カーネルを設定するときに、以下の機能を有効にすることが重要です:

```
Networking support (NET) [Y/n/?] y
*
* Networking options
*
PF_KEY sockets (NET_KEY) [Y/n/m/?] y
IP: AH transformation (INET_AH) [Y/n/m/?] y
IP: ESP transformation (INET_ESP) [Y/n/m/?] y
IP: IPsec user configuration interface (XFRM_USER) [Y/n/m/?] y

Cryptographic API (CRYPTO) [Y/n/?] y
HMAC support (CRYPTO_HMAC) [Y/n/?] y
Null algorithms (CRYPTO_NULL) [Y/n/m/?] y
MD5 digest algorithm (CRYPTO_MD5) [Y/n/m/?] y
SHA1 digest algorithm (CRYPTO_SHA1) [Y/n/m/?] y
DES and Triple DES EDE cipher algorithms (CRYPTO_DES) [Y/n/m/?] y
AES cipher algorithms (CRYPTO_AES) [Y/n/m/?] y
```

カーネルのバージョンによっては IPv6 サポートもまた有効にしなければならないかもしれません。

カーネルをコンパイル、インストールしたらユーザ空間ツールをインストールします。現在そのツールは <http://ipsec-tools.sourceforge.net/> で保守されています。パッケージを手作業でコンパイルする場合はカーネルヘッダの場所を指定する必要があるでしょう。このパッケージは少なくともカーネルバージョン 2.5.47 以降のカーネルヘッダを必要としています。

```
./configure --with-kernel-headers=/lib/modules/2.5.47/build/include
make
make install
```

さて準備が整いました。

4.2. setkey を使った手動鍵接続

手動鍵接続という意味は接続の設定に必要なすべてのパラメータが管理者によって与えられるということです。ピアの自動認証とパラメータのやり取りに IKE プロトコルを利用しません。管理者はどのプロトコル、アルゴリズム、鍵を使ってセキュリティアソシエーションをつくり、付随してセキュリティアソシエーションデータベース (SAD) に移していくか決定します。

4.2.1. トランスポートモード

この節では最初にトランスポートモードでの手動鍵接続の設定についてふれます。設定するのにもっとも単純な接続方法なので、ここからはじめるのがおそらく一番良いでしょう。ここでは IP アドレス 192.168.1.100 と 192.168.2.100 のマシンが IPsec を使って通信する場合について考えます。

すべてのパラメータは SAD と SPD に保存され、setkey コマンドによって変更できます。このコマンドには非常に網羅的なマニュアルページがありますので、ここではトランスポートモードで接続を設定するのに必要なオプションについてだけふれます。setkey は `setkey -f /etc/ipsec.conf` というように実行するとファイルからコマンドを読み込みます。以下は適切な `/etc/ipsec.conf` の例です:

```
#!/usr/sbin/setkey -f

# Configuration for 192.168.1.100
```

```
# Flush the SAD and SPD
flush;
spdf flush;

# Attention: Use this keys only for testing purposes!
# Generate your own keys!

# AH SAs using 128 bit long keys
add 192.168.1.100 192.168.2.100 ah 0x200 -A hmac-md5 ¥
0xc0291ff014dccdd03874d9e8e4cdf3e6;
add 192.168.2.100 192.168.1.100 ah 0x300 -A hmac-md5 ¥
0x96358c90783bbfa3d7b196ceabe0536b;

# ESP SAs using 192 bit long keys (168 + 24 parity)
add 192.168.1.100 192.168.2.100 esp 0x201 -E 3des-cbc ¥
0x7aeaca3f87d060a12f4a4487d5a5c3355920fae69a96c831;
add 192.168.2.100 192.168.1.100 esp 0x301 -E 3des-cbc ¥
0xf6ddb555acfd9d77b03ea3843f2653255afe8eb5573965df;

# Security policies
spdadd 192.168.1.100 192.168.2.100 any -P out ipsec
      esp/transport//require
      ah/transport//require;

spdadd 192.168.2.100 192.168.1.100 any -P in ipsec
      esp/transport//require
      ah/transport//require;
```

もしあなたがテスト目的でなしに任意の用途に手動鍵接続を使いたいということならこのスクリプトの鍵を何か別の鍵で置き換える必要があるでしょう。鍵を作成するには次のようなコマンドを使います:

```
$ # 128 Bit long key
$ dd if=/dev/random count=16 bs=1 | xxd -ps
16+0 Records ein
16+0 Records aus
cd0456eff95c5529ea9e918043e19cbe

$ # 192 Bit long key
$ dd if=/dev/random count=24 bs=1 | xxd -ps
24+0 Records ein
24+0 Records aus
9d6c4a8275ab12fbfdcaf01f0ba9dcfb5f424c878e97f888
```

ランダム鍵であることを確実にするために、鍵を作成するときにはデバイス /dev/random を使って下さい。

このスクリプトはまずセキュリティアソシエーションデータベース (SAD) とセキュリティポリシーデータベース (SPD) を初期化します。それから AH SA と ESP SA を作成します。コマンド add はセキュリティアソシエーションを SAD に追加するもので、送信元および送信先 IP アドレス、IPsec プロトコル (ah)、SPI (0x200)、そしてアルゴリズムの指定が必要です。認証アルゴリズムは -A (暗号化には -E を、圧縮には -C を、ただし IP 圧縮はまだサポートされていません) で指定します。アルゴリズム指定に続いて必ず鍵の指定が必要となります。鍵は二重引用符でくられた "ASCII" か 0x ではじまる十六進数でなければなりません。

Linux は AH および ESP について次のアルゴリズムをサポートしています: hmac-md5、hmac-sha

、des-cbc そして 3des-cbc。ごく近い将来におそらく次のアルゴリズムがサポートされるようになるでしょう: simple (暗号化なし)、blowfish-cbc、aes-cbc、hmac-sha2-256 そして hmac-sha2-512。[訳注: 2.6.0-test5 ではこれらすべてに加えて twofish、serpent、cast5、cast6 がサポートされているようです]

spdadd は SPD にセキュリティポリシーを追加します。これらのポリシーは、どのパケットが IPsec によって保護されどのプロトコルと鍵を利用するかを定義しています。このコマンドには、保護すべきパケット送信元および送信先 IP アドレス、保護すべきプロトコル(とポート)、そして使用するポリシー(-P)の指定が必要です。ポリシーには、方向(in/out)、適用すべきアクション(ipsec/discard/none)、プロトコル(ah/esp/ipcomp)、モード(transport)、そしてレベル(use/require)を指定します。

この設定ファイルは IPsec 接続の両方のピアで作成する必要があります。上のリストはピア 192.168.1.100 では何ら変更なしに動作しますが、ピア 192.168.2.100 ではパケットの方向の差異を反映させるために少々変更が必要です。これを行う最も簡単な方法はセキュリティアソシエーションとセキュリティポリシー内の IP アドレスを交換することです。これを以下に示します:

```
#!/usr/sbin/setkey -f

# Configuration for 192.168.2.100

# Flush the SAD and SPD
flush;
spdflush;

# Attention: Use this keys only for testing purposes!
# Generate your own keys!

# AH SAs using 128 bit long keys
add 192.168.2.100 192.168.1.100 ah 0x200 -A hmac-md5 ¥
0xc0291fff014dccdd03874d9e8e4cdf3e6;
add 192.168.1.100 192.168.2.100 ah 0x300 -A hmac-md5 ¥
0x96358c90783bbfa3d7b196ceabe0536b;

# ESP SAs using 192 bit long keys (168 + 24 parity)
add 192.168.2.100 192.168.1.100 esp 0x201 -E 3des-cbc ¥
0x7aeaca3f87d060a12f4a4487d5a5c3355920fae69a96c831;
add 192.168.1.100 192.168.2.100 esp 0x301 -E 3des-cbc ¥
0xf6ddb555acfd9d77b03ea3843f2653255afe8eb5573965df;

# Security policies
spdadd 192.168.2.100 192.168.1.100 any -P out ipsec
    esp/transport//require
    ah/transport//require;

spdadd 192.168.1.100 192.168.2.100 any -P in ipsec
    esp/transport//require
    ah/transport//require;
```

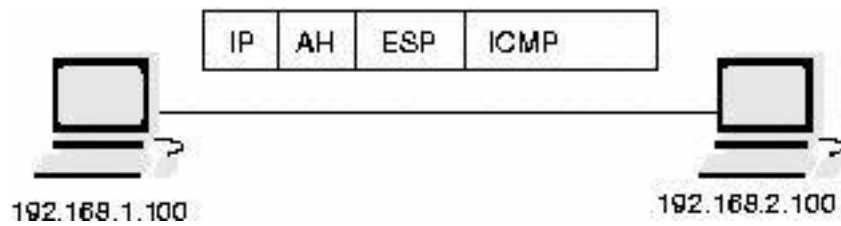
ピアにこれら設定ファイルを置けば、setkey -f /etc/ipsec.conf で読み込むことができます。読み込みに成功しているかどうかは SAD と SPD を表示してみればわかります:

```
# setkey -D
# setkey -DP
```

設定は 図 4. 「AH および ESP を用いた二つのマシン間のトランスポートモード接続」のようにな

ります。

図 4. AH および ESP を用いた二つのマシン間のトランスポートモード接続



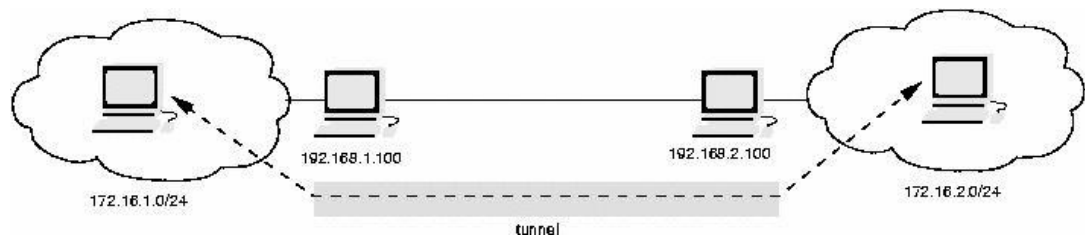
一方のピアから他方へ ping を打ってみると通信が暗号化され、tcpdump で次のようなパケットを表示されることでしょう:

```
12:45:39.373005 192.168.1.100 > 192.168.2.100: AH(spi=0x00000200, seq=0x1):
ESP(spi=0x00000201, seq=0x1) (DF)
12:45:39.448636 192.168.2.100 > 192.168.1.100: AH(spi=0x00000300, seq=0x1):
ESP(spi=0x00000301, seq=0x1)
12:45:40.542430 192.168.1.100 > 192.168.2.100: AH(spi=0x00000200, seq=0x2):
ESP(spi=0x00000201, seq=0x2) (DF)
12:45:40.569414 192.168.2.100 > 192.168.1.100: AH(spi=0x00000300, seq=0x2):
ESP(spi=0x00000301, seq=0x2)
```

4.2.2. トンネルモード

トンネルモードは、IPsec を使いゲートウェイとして働く二つのピアが、相互のネットワーク間の通信を保護するのに使います (図 5. 「二つのネットワーク間の通信を保護する二つのピア」)。元の IP パケットはゲートウェイによって暗号化され、カプセル化されて他方のピアに転送されます。ピアはパケットのカプセル化を解き、元の保護されていないパケットを渡すことになります。

図 5. 二つのネットワーク間の通信を保護する二つのピア



以下に示されるトンネルモードのためのセキュリティアソシエーションとポリシーの設定はトランスポートモードでのそれと似ています。

```
#!/usr/sbin/setkey -f

# Flush the SAD and SPD
flush;
spdf flush;

# ESP SAs doing encryption using 192 bit long keys (168 + 24 parity)
# and authentication using 128 bit long keys
add 192.168.1.100 192.168.2.100 esp 0x201 -m tunnel -E 3des-cbc ¥
0x7aeaca3f87d060a12f4a4487d5a5c3355920fae69a96c831 ¥
```

```

-A hmac-md5 0xc0291ff014dccdd03874d9e8e4cdf3e6;

add 192.168.2.100 192.168.1.100 esp 0x301 -m tunnel -E 3des-cbc ¥
0xf6ddb555acfd9d77b03ea3843f2653255afe8eb5573965df ¥
-A hmac-md5 0x96358c90783bbfa3d7b196ceabe0536b;

# Security policies
spdadd 172.16.1.0/24 172.16.2.0/24 any -P out ipsec
      esp/tunnel/192.168.1.100-192.168.2.100/require;

spdadd 172.16.2.0/24 172.16.1.0/24 any -P in ipsec
      esp/tunnel/192.168.2.100-192.168.1.100/require;

```

この例では ESP プロトコルだけを使っています。ESP プロトコルによって完全性と機密性を保証できます。この場合 ESP アルゴリズムの順番が重要です。最初に暗号化アルゴリズムと鍵を、次に認証アルゴリズムと鍵を定義します。

BSD IPsec 実装との違いは、Linux ではセキュリティアソシエーションはトランスポート、トンネルモード両方に利用できるということだけです。トランスポートモードが既定のモードなので、もしトンネルモードが必要なら、セキュリティアソシエーションは `-m tunnel` で定義する必要があります。

さてセキュリティポリシーで保護するネットワークの IP アドレスを定義しました。これらの送信元および送信先 IP アドレスを持つパケットは IPsec によって保護されます。トンネルモードを使うときは常に、セキュリティポリシーでトンネルと保護すべき実際のピアのアドレス群を指定しなければなりません。これは適切な IPsec SA をみつけるのに必要となります。

4.3. racoon を使った自動鍵接続

KAME IKE デーモン `racoon` もまた Linux に移植されています。このデーモンは自動鍵設定による IPsec 接続を可能にします。`racoon` は事前共有鍵、X.509 証明書、そして Kerberos による認証をサポートしています。デーモンは IKE の phase 1 でメインモード、アグレッシブモードそしてベースモードを利用できます。この章ではメインモードで事前共有鍵と X.509 証明書 (TODO: Kerberos) を使って `racoon` を設定する場合について説明します。最後に `roadwarriors` シナリオについても簡単にふれます。

4.3.1. 事前共有鍵

`racoon` を使って最も簡単に認証するのは事前共有鍵を利用する方法です。鍵は `/etc/psk.txt` 内に定義しておく必要があります。このファイルは権限のないユーザは読めないようにしておくべき (`chmod 400 /etc/psk.txt`) で、次のような構文となっています:

```

# IPv4 Adressen
192.168.2.100      simple psk
5.0.0.1           0xe10bd52b0529b54aac97db63462850f3
# USER_FQDN
ralf@spenneberg.net  This is a psk for an email address
# FQDN
www.spenneberg.net   This is a psk

```

このファイルは行毎に区切られています。最初の行は事前共有鍵によって認証されるピアの指定です。第二行からはじまるのがすべて PSK です。

`racoon` の設定は簡単です。以下は典型的な `/etc/racoon.conf` 設定ファイルの内容です:

```
path pre_shared_key "/etc/psk.txt";
```

```

remote 192.168.2.100 {
    exchange_mode main;
    proposal {
        encryption_algorithm 3des;
        hash_algorithm md5;
        authentication_method pre_shared_key;
        dh_group modp1024;
    }
}

sainfo address 172.16.1.0/24 any address 172.16.2.0/24 any {
    pfs_group 768;
    encryption_algorithm 3des;
    authentication_algorithm hmac_md5;
    compression_algorithm deflate;
}

```

設定ファイルではまず最初に racoon がどこで事前共有鍵をみつけられるかを定義しています。それからピア 192.168.2.100 と IKE ネゴシエーションの phase 1 で使うパラメータを定義しています。二つ目の段落ではセキュリティアソシエーションの設定に使うパラメータを指定しています。この指定は、ほとんどの場合定義された IP アドレスに固有ですが、IP アドレスの代わりに anonymous を使えば汎用に使うこともできます。ここで SA のために使う暗号化、認証、そして圧縮アルゴリズムを定義します。racoon の起動時のエラーを避けるためにこれら3つすべてを定義する必要があります。

IKE デーモン racoon は起動時にすぐにトンネルネゴシエーションを開始するわけではありません。そうではなくトンネルが必要になるまで racoon は待ちます。それを通知するためにはカーネルはいつ racoon に通知するべきか知っている必要があります。そのために管理者は適切なセキュリティアソシエーションを持たないセキュリティポリシーを定義しておく必要があります。Linux カーネルは、セキュリティポリシーに関連づけられたパケットを保護する必要があり、セキュリティアソシエーションが利用可能でないときはいつも、racoon を呼出し、必要なセキュリティアソシエーションを尋ねます。racoon はそれから IKE ネゴシエーションを開始し、終了時に SA を作成します。そして Linux カーネルはパケットを送れるようになります。

以下は 192.168.1.100 について必要なポリシーの設定を想定したものです:

```

#!/usr/sbin/setkey -f
#
# Flush SAD and SPD
flush;
spdf flush;

# Create policies for racoon
spdadd 172.16.1.0/24 172.16.2.0/24 any -P out ipsec
        esp/tunnel/192.168.1.100-192.168.2.100/require;

spdadd 172.16.2.0/24 172.16.1.0/24 any -P in ipsec
        esp/tunnel/192.168.2.100-192.168.1.100/require;

```

setkey -f /etc/ipsec.conf を使ってポリシーが読み込まれると racoon は起動できるようになります。テスト目的のためには racoon を racoon -F -c /etc/racoon.conf というように起動させるべきです。また他のピアの設定は方向の違いを反映させるために変更する必要があります。ファイル /etc/psk.txt、/etc/ipsec.conf そして /etc/racoon.conf 内の IP アドレスもまた変更が必要となります。

トンネルの初期化は以下のログのようになります:

```
2003-02-21 18:11:17: INFO: main.c:170:main(): @(#)racoon 20001216 20001216
sakane@kame.net
2003-02-21 18:11:17: INFO: main.c:171:main(): @(#)This product linked Open
SSL 0.9.6b [engine] 9 Jul 2001 (http://www.openssl.org/)
2003-02-21 18:11:17: INFO: isakmp.c:1365:isakmp_open(): 127.0.0.1[500] use
d as isakmp port (fd=7)
2003-02-21 18:11:17: INFO: isakmp.c:1365:isakmp_open(): 192.168.1.100[500]
used as isakmp port (fd=9)
2003-02-21 18:11:37: INFO: isakmp.c:1689:isakmp_post_acquire(): IPsec-SA r
equest for 192.168.2.100 queued due to no phasel found.
2003-02-21 18:11:37: INFO: isakmp.c:794:isakmp_ph1begin_i(): initiate new
phase 1 negotiation: 192.168.1.100[500]<=>192.168.2.100[500]
2003-02-21 18:11:37: INFO: isakmp.c:799:isakmp_ph1begin_i(): begin Identit
y Protection mode.
2003-02-21 18:11:37: INFO: vendorid.c:128:check_vendorid(): received Vendor
ID: KAME/racoon
2003-02-21 18:11:37: INFO: vendorid.c:128:check_vendorid(): received Vendor
ID: KAME/racoon
2003-02-21 18:11:38: INFO: isakmp.c:2417:log_ph1established(): ISAKMP-SA es
tablished 192.168.1.100[500]-192.168.2.100[500] spi=6a01ea039be7bac2:bd288f
f60eed54d0
2003-02-21 18:11:39: INFO: isakmp.c:938:isakmp_ph2begin_i(): initiate new p
hase 2 negotiation: 192.168.1.100[0]<=>192.168.2.100[0]
2003-02-21 18:11:39: INFO: pfkey.c:1106:pk_recvupdate(): IPsec-SA establish
ed: ESP/Tunnel 192.168.2.100->192.168.1.100 spi=68291959(0x4120d77)
2003-02-21 18:11:39: INFO: pfkey.c:1318:pk_recvadd(): IPsec-SA established:
ESP/Tunnel 192.168.1.100->192.168.2.100 spi=223693870(0xd554c2e)
```

4.3.2. X.509 証明書

racoon は認証プロセスのために X.509 証明書を使うことをサポートしています。これらの証明書は認証局 (CA) で検証できます。設定は認証部分を除けば PSK 設定に似ています:

```
path certificate "/etc/certs";

remote 192.168.2.100 {
    exchange_mode main;
    certificate_type x509 "my_certificate.pem" "my_private_key.pem";
    verify_cert on
    my_identifier asn1dn;
    peers_identifier asn1dn;
    proposal {
        encryption_algorithm 3des;
        hash_algorithm md5;
        authentication_method rsasig;
        dh_group modp1024;
    }
}

sainfo address 172.16.1.0/24 any address 172.16.2.0/24 any {
    pfs_group 768;
    encryption_algorithm 3des;
    authentication_algorithm hmac_md5;
    compression_algorithm deflate;
}
```

証明と秘密鍵はパス /etc/certs に保存されます。このパスは設定ファイル内にオプション path certificate で設定します。証明と無効証明リストは openssl で生成、PEM 形式で保存されます。証明の作成については X.509 証明書の章を参照して下さい。ピアの証明が認証局で検証される (verify_cert on が既定値) と CA の証明もまたこのディレクトリに保存されます。このファイルは OpenSSL がその証明をみつけられるように、ハッシュ名を使って名前を変更するかリンクを張っておく必要があります。

```
ln -s CAfile.pem `openssl x509 -noout -hash < CAfile.pem`.0
```

もし補足的に証明が無効証明リスト (CRL) についても検証されるなら、CRL も同じディレクトリに保存され、同様にハッシュ名でリンクされている必要があります。

```
ln -s CRLfile.pem `openssl crl -noout -hash < CAfile.pem`.r0
```

証明と秘密鍵を保存する際に重要なのは racoon が秘密鍵を復号化できないということです。従って秘密鍵は復号化したテキストファイル形式で保存しなければならず、暗号化していれば復号化する必要があります。

```
# openssl rsa -in my_private_key.pem -out my_private_key.pem
read RSA key
Enter PEM pass phrase: password
writing RSA key
```

4.3.3. Roadwarrior

roadwarrior は不明な動的 IP アドレスを使って VPN ゲートウェイに接続するクライアントです。racoon と組合せるとこれは二つの問題を引き起こします:

- ・ IP アドレスは不明なので racoon の設定ファイルでも /etc/psk.txt ファイルでも指定できません。クライアントを識別する他の方法をみつけなければなりません。事前共有鍵を使っている場合はアグレッシブモードが解決策となりますが、最適な方法は X.509 証明書を使うことです。
- ・ 送信先 IP アドレスが不明なので、racoon のためのセキュリティポリシーを作成できません。racoon は接続の初期化時にセキュリティポリシーとセキュリティアソシエーションを作成しなければなりません。

これを実現するには /etc/racoon.conf にいくつかの変更が必要となります:

```
path certificate "/etc/certs";

remote anonymous {
    exchange_mode main;
    generate_policy on;
    passive on;
    certificate_type x509 "my_certificate.pem" "my_private_key.pem";
    my_identifier asnldn;
    peers_identifier asnldn;
    proposal {
        encryption_algorithm 3des;
        hash_algorithm md5;
        authentication_method rrsasig;
        dh_group modp1024;
    }
}
```

```

}

sainfo anonymous {
    pfs_group modp1024;
    encryption_algorithm 3des;
    authentication_algorithm hmac_md5;
    compression_algorithm deflate;
}

```

オプション `generate_policy on` は `racoon` に新規接続の初期化時に適切なポリシーを生成するように指示します。オプション `passive on` は `racoon` に受動的であり続け、外からの新規接続開始を待ち続けるように指示します。`racoon` は接続開始しません。

しかし最も重要な変更点は `remote` および `sainfo` 行の `anonymous` 定義です。これは `racoon` にどこからの接続についても受け付けるように指示します。

5. Linux カーネル 2.5/2.6 での OpenBSD isakmpd の利用

Thomas Walpuski は OpenBSD の IKE デーモンを Linux に移植しています (<http://bender.thinknerd.de/~thomas/IPsec/isakmpd-linux.html>)。現在 `isakmpd` は Linux カーネル 2.5.47 以上と 2.6.x で IPsec VPN のセットアップに利用することができます。この章ではこの `isakmpd` のインストールと設定について説明するつもりです。

5.1. インストール

もしあなたが RPM ベースのディストリビューションか Debian を使っているのならインストールは適切なパッケージツールを使えば完了です。著者は Linux カーネル 2.6.0-test2 に対する `isakmpd` の RPM パッケージ (http://www.spenneberg.org/VPN/Kernel-2_6_IPsec) をビルドしました。カーネル内部の ABI は度々変更されているのでこのパッケージは他のバージョンでは動かないかもしれないということに注意して下さい。Debian プロジェクトでは `apt-get install isakmpd` でインストールできるパッケージが用意されています。

ソースからインストールする場合、もし X.509 証明を使いたいのなら `keynote` パッケージ (<http://www1.cs.columbia.edu/~angelos/keynote.html>) が必要となります。さらに Linux カーネル 2.5.47 以上か 2.6.x が必要です。

`isakmpd` ソースを取得する際は Thomas Walpuski のウェブページに述べられているステップに従います。それから `GNUmakefile` を適宜編集し、`OS=linux` 行を有効にします。もし Linux カーネルソースを `/usr/src/linux` に置いていないなら、`sysdep/linux/GNUmakefile.sysdep` もそれに合せて変更する必要があるでしょう。

コンパイルはコマンド `make` で完了できます。

`isakmpd` には二つのコマンド `keyconv` と `certpatch` が付属しています。これらのツールはサブディレクトリ `apps/` にあり、手作業でコンパイルする必要があるでしょう (RPM パッケージでは一部として含まれています)。`keyconv` が DNSSEC を `openssl` 鍵に変換するのに対し、`certpatch` は既存の証明に `SubjectAltName` を追加することができます。

最後に一つ注意: すべてのマニュアルページは Latin1 形式となっています。Red Hat 9 はこれらのマニュアルページを表示することができません。読めるように変換しなければなりません (RPM パッケージでは対応しています): `iconv --from-code LATIN1 --to-code UTF-8 --output isakmpd2.8 isakmpd.8`

isakmpd をコンパイルしたら、必須のディレクトリ階層を作成します:

```
mkdir /etc/isakmpd
mkdir /etc/isakmpd/ca
mkdir /etc/isakmpd/certs
mkdir /etc/isakmpd/keynote
mkdir /etc/isakmpd/crls
mkdir /etc/isakmpd/private
mkdir /etc/isakmpd/pubkeys
```

5.2. 事前共有鍵 (PSK) の利用

isakmpd は 設定ファイル (/etc/isakmpd/isakmpd.conf) とポリシーファイル (/etc/isakmpd/isakmpd.policy) を使います。設定ファイルはよく知られた .INI スタイルの形式となっています。各セクションは次のようにして始まっています:

```
[section]
```

セクション内ではタグに値を割当てることができます:

```
tag=value
```

値が長すぎて一行に収まらなければバックスラッシュテクニックを使っていくつかの行に分けることができます。コメントはハッシュマーク # を使えばどこにでも置けます。

最初に認証のために事前共有鍵を使う簡単な設定について見てみましょう。セットアップのためのトンネルについての以後のセクションを見て下さい。

```
[General]
Listen-on=                192.168.1.100

[Phase 1]
192.168.2.100=            ISAKMP-peer-west

[Phase 2]
Connections=              IPsec-east-west

[ISAKMP-peer-west]
Phase=                     1
Local-address=             192.168.1.100
Address=                   192.168.2.100
Authentication=           ThisIsThePassphrase

[IPsec-east-west]
Phase=                     2
ISAKMP-peer=               ISAKMP-peer-west
Configuration=             Default-quick-mode
Local-ID=                  Net-east
Remote-ID=                 Net-west

[Net-west]
ID-type=                   IPV4_ADDR_SUBNET
Network=                   172.16.2.0
```

```

Netmask=                255. 255. 255. 0

[Net-east]
ID-type=                IPV4_ADDR_SUBNET
Network=                172. 16. 1. 0
Netmask=                255. 255. 255. 0

[Default-quick-mode]
DOI=                    IPSEC
EXCHANGE_TYPE=         QUICK_MODE
Suites=                 QM-ESP-3DES-MD5-PFS-SUITE

```

この設定ファイルは二つのゲートウェイ 192.168.1.100 と 192.168.2.100 の間のトンネルについて記述されています。このトンネルは 172.16.1.0/24 と 172.16.2.0/24 で使うことができます。この設定ファイルはゲートウェイ 192.168.1.100 について固有なものです。

個々のセクションについて見ていきましょう。最初のセクション [General] では一般的な設定がなされています。ここでは isakmpd が起動時に特定の IP アドレス群に結びつけられるべきであると指定しています。これはもし VPN ゲートウェイ上にいくつかの IP アドレス群がある場合には推奨します。

セクション [Phase 1] には IP アドレス 192.168.2.100 を使ったピアのための設定が記述されています。もしピアの IP アドレスが不明 (roadwarrior のように) ならば既定値が代りに使われます。

セクション [Phase 2] には Phase 1 の認証が済んだ後で作成するトンネルについて記述されています。isakmpd から積極的に接続を開始できないのなら代りに Passive-connections を使います。

さてこれから Phase 1、2 セクション内で参照する名前を定義しなければなりません。最初に ISAKMP-peer-west を定義します。これは Phase 1 で使われ、Local-address とリモート Address を知っている必要があります。もしリモートアドレスが不明ならば、単にこのタグを削除します。Authentication はクリアテキストで与えられる事前共有鍵を使ってなされるべきです。

次に IPsec-east-west を定義します。これは Phase 2 で使われ、ISAKMP-peer ISAKMP-peer-west との間に確立されることとなります。[訳注: 要校正]。それから接続の Configuration と付随的なトンネルの ID (Local-ID と Remote-ID) を定義します。

これらの ID は再度参照されるので必ず定義しなければなりません。ID-type として可能な値は IPV4_ADDR、IPV6_ADDR、IPV4_ADDR_SUBNET そして IPV6_ADDR_SUBNET です。

最後に少なくともトンネルの記述でふれているクイックモード設定を定義しなければなりません。DOI (既定値: IPSEC) と EXCHANGE_TYPE (既定値: QUICK_MODE)、そして利用する Suites を指定します。これが QuickMode-Encapsulated-Security-Payload-3DES-Encryption-MD5-HMAC-Perfect-Forward-Secrecy です。いくつかのスイートをカンマで区切って指定することができます。すべての可能な変換とスイートについてはマニュアルページを参照して下さい。

isakmpd.policy ファイルはもっと短かいです。以下はその例です:

```

KeyNote-Version: 2
Conditions: app_domain == "IPsec policy" &&
            esp_present == "yes" &&
            esp_enc_alg == "3des" &&
            esp_auth_alg == "hmac-md5" -> "true";

```

接続をテストするには isakmpd を次のようにして起動します:

```
isakmpd -d -4 -DA=90
```

こうすると `isakmpd` はフォアグラウンドで (-d) IPv4 を使って (-4) デバッグレベル 90 で起動します。

一度接続が開始されると、サブネットから他のサブネットに ping できるようになるはずです。ipsec-tools をインストールしていればコマンド `setkey` を使って `isakmpd` によって追加されたポリシーとセキュリティアソシエーションを見ることができます。フォアグラウンドで走っている `isakmpd` を `ctrl-c` で kill しても SAD と SPD は初期化されません。コマンド `setkey` を使って手作業でしなければなりません。コマンド `kill -TERM` を使って `isakmpd` を kill すると、SAD と SPD は初期化されます。

5.3. X.509 証明の利用

`isakmpd` はまた X.509 証明を認証プロセスに使うこともできます。証明は通常のツールで作成でき、VPN を構成する各マシン毎に以下のファイルが必要となります:

- ・ `/etc/isakmpd/private/local.key` マシンの秘密鍵 (.pem 形式)。許可属性は 600 でなければなりません。
- ・ `/etc/isakmpd/ca/ca.crt` 信頼する証明オーソリティの証明。
- ・ `/etc/isakmpd/certs/ip-address.crt` ローカルマシンの証明。

証明を `isakmpd` がみつけて使うために `SubjectAltName` を含めておかなければなりません。この X.509v3 拡張は証明の作成時に、あるいはコマンド `certpatch` を使って指定できます。このコマンドは CA の秘密鍵を必要とし、証明を展開、拡張を追加し、再度証明に署名します。

```
certpatch -i ip-address -k ca.key originalcert.crt newcert.crt
```

`certpatch` は IP アドレス、FQDN あるいは UFQDN を証明に追加することができます。

これらのファイルを適切なディレクトリに保存し、適切な許可属性を与えておけば、設定およびポリシーファイルを作成することができます。設定ファイル内では単に Authentication 行を削除し、行 `ID=East` を `ISAKMP-peer-west` セクションに追加し、それから `East` を定義します。付随的に X.509 ディレクトリを指定しなければなりません。完全な設定ファイルは以下のようになります:

```
[General]
Listen-on=                192.168.1.100

[Phase 1]
192.168.2.100=            ISAKMP-peer-west

[Phase 2]
Connections=             IPsec-east-west

[ISAKMP-peer-west]
Phase=                    1
Local-address=            192.168.1.100
Address=                  192.168.2.100
ID=                       East

[East]
ID-type=                  IPV4_ADDR
Address=                  192.168.1.100

[IPsec-east-west]
Phase=                    2
```

```
ISAKMP-peer=           ISAKMP-peer-west
Configuration=         Default-quick-mode
Local-ID=              Net-east
Remote-ID=             Net-west

[Net-west]
ID-type=               IPV4_ADDR_SUBNET
Network=               172.16.1.0
Netmask=               255.255.255.0

[Net-east]
ID-type=               IPV4_ADDR_SUBNET
Network=               172.16..2.0
Netmask=               255.255.255.0

[Default-quick-mode]
DOI=                   IPSEC
EXCHANGE_TYPE=         QUICK_MODE
Suites=                 QM-ESP-AES-SHA-PFS-SUITE

[X509-certificates]
CA-directory=          /etc/isakmpd/ca/
Cert-directory=        /etc/isakmpd/certs/
Private-key=           /etc/isakmpd/private/local.key
```

ポリシーファイルについても変更が必要となります。信頼する CA によって署名された証明を使ったピアだけに許可したいので、Authorizer 行の後に次の内容を追加します。完全なポリシーファイルは以下のようになります:

```
KeyNote-Version: 2
Authorizer: "POLICY"
Licensee: "DN:/C=DE/ST=NRW/L=Steinfurt/O=Spenneberg.Com/OU=VPN/CN=RootCA"
Conditions: app_domain == "IPsec policy" &&
            esp_present == "yes" &&
            esp_enc_alg == "3des" &&
            esp_auth_alg == "hmac-md5" -> "true";
```

DN: の後のテキストは CA 証明の subject 行と一致していなければなりません:

```
openssl x509 -in ca/ca.crt -noout -subject
```

さあこれで通常と同じように isakmpd を起動して、設定をテストできるようになりました。

6. より高度な設定

6.1. DHCP-over-IPsec

...

6.2. NAT-Traversal

...

6.3. GRE-Tunnel

...

6.4. L2TP

...