Oracle® Solaris 11.2 でのネットワークのセキュリティー保護

Part No: E53811-02
2014 年 9 月

ORACLE

目次

表目次

例目次

このドキュメントの使用法

- **概要** – ネットワークセキュリティーを提供する方法を説明します。リンク保護、チューニング可能なネットワークパラメータ、ファイアウォール保護、IPsec と IKE、および Web サーバーの SSL カーネル保護が含まれます。
- **対象読者** – ネットワークセキュリティー管理者。
- **必要なナレッジ** – サイトのセキュリティー要件。

製品ドキュメントライブラリ

この製品に関する最新情報および既知の問題については、ドキュメントライブラリ (http://www.oracle.com/pls/topic/lookup?ctx=E56342) に記載されています。

Oracle サポートへのアクセス

Oracle ユーザーは My Oracle Support から電子サポートにアクセスできます。詳細は、http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info を参照してください。聴覚に障害をお持ちの場合は、http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs を参照してください。

フィードバック

このドキュメントに関するフィードバックを http://www.oracle.com/goto/docfeedback からお聞かせください。

♦ ♦ ♦ 第 1 章

1 仮想化環境でのリンク保護の使用

この章では、リンク保護と Oracle Solarisシステムでのその構成方法について説明します。この章の内容は次のとおりです。

Oracle Solaris 11.2 のネットワークセキュリティーの新機能

このセクションでは、このリリースでの重要な新しいネットワークセキュリティーに関する既存のお客様のための情報を紹介します。

IKE バージョン 2 (IKEv2) は、最新バージョンの IKE プロトコルを使用する IPsec の自動鍵管理を実現します。IKEv2 および IPsec は、Oracle Solaris の暗号化フレームワーク機能の暗号化アルゴリズムを使用します。

注記 - Oracle Solaris の暗号化フレームワーク機能は、FIPS 140-2 レベル 1 で検証されています。IKE での FIPS 140 モードの使用については、表8-1「Oracle Solaris での IKEv2 および IKEv1 の実装」を参照してください。ハードウェアとソフトウェアの詳細は、Oracle FIPS 140 ソフトウェア検証 (http://www.oracle.com/technetwork/topics/security/fips140-software-validations-1703049.html)を参照してください。

IKE バージョン 1 (IKEv1) のサポートは引き続き使用可能です。詳細は、第8章「インターネット鍵交換について」を参照してください。

リンク保護について

システム構成で仮想化の導入が増えることによって、ホスト管理者は、ゲスト仮想マシン (VM) に物理リンクまたは仮想リンクへの排他的アクセス権を付与することができます。この構成では、仮想環境のネットワークトラフィックを、ホストシステムによって送受信される幅広いトラフィックから分離できるため、ネットワークパフォーマンスが向上します。同時に、この構成はシステムとネットワーク全体をゲスト環境で生成される可能性のある有害なパケットのリスクにさらす可能性があります。

リンク保護は、潜在的に悪意のあるゲスト VM がネットワークに対して引き起こす可能性のある損害を回避することを目的としています。この機能は、次の基本的な脅威から保護します。

- IP、DHCP、および MAC のなりすまし
- Bridge Protocol Data Unit (BPDU) 攻撃などの L2 フレームのなりすまし

注記 - リンク保護は、特に複雑なフィルタリング要件のある構成の場合に、ファイアウォールの展開に置き換わるものではありません。

リンク保護のタイプ

Oracle Solaris でのリンク保護メカニズムは、次の保護のタイプがあります。

`mac-nospoof`

システムの MAC アドレスのなりすましに対する保護を有効にします。リンクがゾーンに属する場合、`mac-nospoof` を有効にすると、ゾーンの所有者がそのリンクの MAC アドレスを変更することを妨げます。

`ip-nospoof`

IP なりすましに対する保護を有効にします。デフォルトで、DHCP アドレスとリンクローカル IPv6 アドレスを含むアウトバウンドパケットが許可されます。

`allowed-ips` リンクプロパティーを使用して、アドレスを追加できます。IP アドレスの場合、パケットのソースアドレスは `allowed-ips` リスト内のアドレスに一致する必要があります。ARP パケットの場合、パケットの送信者のプロトコルアドレスが `allowed-ips` リスト内に存在する必要があります。

`dhcp-nospoof`

DHCP クライアントのなりすましに対する保護を有効にします。デフォルトでは、ID がシステムの MAC アドレスと一致している DHCP パケットを使用できます。

allowed-dhcp-cids リンクプロパティーを使用して、許可されるクライアントを追加できます。allowed-dhcp-cids リスト内のエントリは、dhcpagent(1M) のマニュアルページに指定されているとおりに書式設定されている必要があります。

restricted

送信パケットを IPv4、IPv6、および ARP に制限します。この保護のタイプは、潜在的に有害な L2 制御フレームがリンクから生成されるのを妨ぐよう設計されています。

注記 - リンク保護のためにドロップされたパケットは、4 つの保護のタイプ (mac_spoofed、dhcp_spoofed、ip_spoofed、restricted) についてのカーネル統計によって追跡されます。これらのリンクごとの統計情報を取得するには、20 ページの「リンク保護の構成と統計情報を表示する方法」を参照してください。

これらの保護タイプの詳細については、dladm(1M) のマニュアルページを参照してください。

リンク保護の構成

リンク保護を使用するには、リンクの protection プロパティーを設定します。保護のタイプがほかの構成ファイルと連携する場合 (ip-nospoof と allowed-ips または dhcp-nospoof と allowed-dhcp-cids など)、2 つの一般的なアクションを実行します。まず、リンク保護を有効にします。次に、構成ファイルをカスタマイズして、通過させるその他のパケットを特定します。

注記 - 大域ゾーンでリンク保護を構成する必要があります。

次のタスクマップは、Oracle Solaris システムで、リンク保護を構成するための手順を示しています。

表 1-1 リンク保護の構成のタスクマップ

タスク	説明	参照先
リンク保護を有効にします。	リンクから送信されるパケットを制限し、なりすましからリンクを保護します。	18 ページの「リンク保護を有効にする方法」
リンク保護を無効にします。	リンク保護を解除します。	18 ページの「リンク保護を無効にする方法」
IP リンク保護タイプを指定します。	リンク保護メカニズムを通過できる IP アドレスを指定します。	19 ページの「IP なりすましに対して保護するように IP アドレスを指定する方法」
DHCP リンク保護タイプを指定します。	リンク保護メカニズムを通過できる DHCP アドレスを指定します。	20 ページの「DHCP なりすましから保護するように DHCP クライアントを指定する方法」
リンク保護構成を表示します。	保護されているリンクおよび例外を一覧表示し、実施統計情報を表示します。	20 ページの「リンク保護の構成と統計情報を表示する方法」

▼ リンク保護を有効にする方法

この手順では、送信パケットのタイプを制限し、リンクのなりすましを防ぎます。

始める前に Network Link Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **使用可能なリンク保護のタイプを表示します。**

```
# dladm show-linkprop -p protection
LINK     PROPERTY        PERM VALUE        EFFECTIVE    DEFAULT   POSSIBLE
net0     protection      rw   --           --           --        mac-nospoof,
                                                                   restricted,
                                                                   ip-nospoof,
                                                                   dhcp-nospoof
```

可能なタイプの説明については、16 ページの「リンク保護のタイプ」および `dladm`(1M) のマニュアルページを参照してください。

2. **1 つまたは複数の保護タイプを指定してリンク保護を有効にします。**

```
# dladm set-linkprop -p protection=value[,value,...] link
```

次の例では、`vnic0` リンク上の 4 つすべてのリンク保護のタイプが有効にされています。

```
# dladm set-linkprop \
-p protection=mac-nospoof,restricted,ip-nospoof,dhcp-nospoof vnic0
```

注意 - 各保護の値は、有効にする前に単独でテストします。システムの構成が間違っていると、通信できなくなる可能性があります。

3. **リンク保護が有効にされていることを確認します。**

```
# dladm show-linkprop -p protection vnic0
LINK     PROPERTY        PERM VALUE        EFFECTIVE    DEFAULT   POSSIBLE
net0     protection      rw   mac-nospoof  mac-nospoof  --        mac-nospoof,
                              restricted   restricted   --        restricted,
                              ip-nospoof   ip-nospoof   --        ip-nospoof,
                              dhcp-nospoof dhcp-nospoof --        dhcp-nospoof
```

▼ リンク保護を無効にする方法

この手順では、リンク保護をデフォルト値のリンク保護なしにリセットします。

始める前に Network Link Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **リンク保護を無効にするには、`protection` プロパティーをそのデフォルト値にリセットします。**

```
# dladm reset-linkprop -p protection link
```

2. **リンク保護が無効になっているかどうかを確認します。**

```
# dladm show-linkprop -p protection vnic0
LINK     PROPERTY        PERM VALUE        EFFECTIVE    DEFAULT   POSSIBLE
net0     protection      rw   --           --           --        mac-nospoof,
                                                                  restricted,
                                                                  ip-nospoof,
                                                                  dhcp-nospoof
```

▼ IP なりすましに対して保護するように IP アドレスを指定する方法

始める前に How to Enable Link Protectionに示すように、18 ページの「リンク保護を有効にする方法」保護のタイプを有効にします。

Network Link Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IP なりすましに対して保護を有効にしていることを確認します。**

```
# dladm show-linkprop -p protection link
LINK     PROPERTY        PERM VALUE        EFFECTIVE    DEFAULT   POSSIBLE
link     protection      rw   ip-nospoof   ip-nospoof   --        mac-nospoof,
                                                                  restricted,
                                                                  ip-nospoof,
                                                                  dhcp-nospoof
```

2. **`allowed-ips` リンクプロパティーのデフォルト値のリストに IP アドレスを追加します。**

```
# dladm set-linkprop -p allowed-ips=IP-addr[,IP-addr,...] link
```

次の例に、IP アドレス `10.0.0.1` および `10.0.0.2` を `vnic0` リンクの `allowed-ips` プロパティーに追加する方法を示します。

```
# dladm set-linkprop -p allowed-ips=10.0.0.1,10.0.0.2 vnic0
```

詳細は、dladm(1M) のマニュアルページを参照してください。

▼ DHCP なりすましから保護するように DHCP クライアントを指定する方法

始める前に How to Enable Link Protectionに示すように、18 ページの「リンク保護を有効にする方法」保護のタイプを有効にします。

Network Link Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **DHCP なりすましに対して保護を有効にしていることを確認します。**

```
# dladm show-linkprop -p protection link
LINK     PROPERTY        PERM VALUE        EFFECTIVE    DEFAULT   POSSIBLE
link     protection      rw   dhcp-nospoof dhcp-nospoof --        mac-nospoof,
                                                                  restricted,
                                                                  ip-nospoof,
                                                                  dhcp-nospoof
```

2. **allowed-dhcp-cids リンクプロパティーに ASCII フレーズを指定します。**

```
# dladm set-linkprop -p allowed-dhcp-cids=CID-or-DUID[,CID-or-DUID,...] link
```

次の例に、vnic0 リンクの allowed-dhcp-cids プロパティーの値として、文字列 hello を指定する方法を示します。

```
# dladm set-linkprop -p allowed-dhcp-cids=hello vnic0
```

詳細は、dladm(1M) のマニュアルページを参照してください。

▼ リンク保護の構成と統計情報を表示する方法

始める前に Network Link Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **リンク保護のプロパティー値を表示します。**

```
# dladm show-linkprop -p protection,allowed-ips,allowed-dhcp-cids link
```

次の例に、vnic0 リンクの protection、allowed-ips、および allowed-dhcp-cids プロパティーの値を示します。

```
# dladm show-linkprop -p protection,allowed-ips,allowed-dhcp-cids vnic0
LINK    PROPERTY            PERM  VALUE          EFFECTIVE    DEFAULT POSSIBLE
vnic0   protection          rw    mac-nospoof    mac-nospoof --      mac-nospoof,
                                  restricted     restricted           restricted,
                                  ip-nospoof     ip-nospoof           ip-nospoof,
                                  dhcp-nospoof   dhcp-nospoof         dhcp-nospoof
vnic0   allowed-ips         rw    10.0.0.1,      10.0.0.1,    --      --
                                  10.0.0.2       10.0.0.2
vnic0   allowed-dhcp-cids   rw    hello          hello        --      --
```

注記 - allowed-ips プロパティーは、EFFECTIVE の下に一覧表示されているように、ip-nospoof が有効にされている場合にのみ使用されます。allowed-dhcp-cids プロパティーは dhcp-nospoof が有効にされている場合にのみ使用されます。

2. **リンク保護の統計を表示します。**

dlstat コマンドの出力がコミットされるため、このコマンドはスクリプトに適しています。

```
# dlstat -A
...
 vnic0
  mac_misc_stat
               multircv                      0
              brdcstrcv                      0
               multixmt                      0
              brdcstxmt                      0
          multircvbytes                      0
           bcstrcvbytes                      0
          multixmtbytes                      0
           bcstxmtbytes                      0
               txerrors                      0
             macspoofed                      0  <----------
              ipspoofed                      0  <----------
            dhcpspoofed                      0  <----------
             restricted                      0  <----------
               ipackets                      3
                 rbytes                    182
...
```

出力は、なりすましや制限されたパケットが通過を試みていないことを示しています。

kstat コマンドを使用できますが、その出力はコミットされません。たとえば、次のコマンドは dhcpspoofed 統計情報を検出します。

```
# kstat vnic0:0:link:dhcpspoofed
module: vnic0                           instance: 0
name:   link                            class:    vnic
```

```
        dhcpspoofed                     0
```

詳細は、dlstat(1M) および kstat(1M) のマニュアルページを参照してください。

♦ ♦ ♦ 第 2 章

2 ネットワークのチューニング

この章では、Oracle Solaris のセキュリティーに影響するネットワークパラメータをチューニングする方法について説明します。

ネットワークのチューニング

表 2-1 ネットワークのチューニングのタスクマップ

タスク	説明	参照先
ネットワークルーティングデーモンを無効にします。	不審なネットワーク侵入者によるシステムへのアクセスを制限します。	24 ページの「ネットワークルーティングデーモンを無効にする方法」
ネットワークトポロジに関する情報の流布を回避します。	パケットのブロードキャストを回避します。	25 ページの「ブロードキャストパケット転送を無効にする方法」
	ブロードキャストエコー要求およびマルチキャストエコー要求への応答を回避します。	25 ページの「エコーリクエストへの応答を無効にする方法」
他のドメインへのゲートウェイであるシステム (ファイアウォールや VPN ノードなど) では、厳格な転送元および転送先のマルチホーミングをオンにします。	ヘッダーにゲートウェイのアドレスが指定されていないパケットがゲートウェイ外に移動することを回避します。	26 ページの「厳密なマルチホームを設定する方法」
不完全なシステム接続の数を制御することによって、DOS 攻撃を回避します。	TCP リスナーに対する不完全な TCP 接続の許容数を制限します。	27 ページの「不完全な TCP 接続の最大数を設定する方法」
許可される受信接続の数を制御することによって、DOS 攻撃を回避します。	TCP リスナーに対する中断中の TCP 接続のデフォルト最大数を指定します。	27 ページの「中断中の TCP 接続の最大数を設定する方法」
初期の TCP 接続に対して強固な乱数が生成されることを確認します。	RFC 6528 で規定されているシーケンス番号生成値に準拠します。	28 ページの「初期の TCP 接続に強固な乱数を指定する方法」
ICMP リダイレクトを妨げます。	ネットワークトポロジのインジケータを削除します。	28 ページの「ICMP リダイレクトを妨げる方法」

タスク	説明	参照先
ネットワークパラメータをセキュアなデフォルト値に戻します。	管理操作によって削減されたセキュリティーを強化します。	29 ページの「ネットワークパラメータをセキュアな値にリセットする方法」

▼ ネットワークルーティングデーモンを無効にする方法

この手順を使用して、デフォルトルーターを指定したインストール後にネットワークルーティングを回避します。それ以外の場合は、手動でルーティングを構成したあとに、この手順を実行します。

注記 - 多くのネットワーク構成の手順で、ルーティングデーモンを無効にする必要があります。したがって、より大規模な構成手順の一部として、このデーモンを無効にしておく場合があります。

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **ルーティングデーモンが動作していることを確認します。**

```
$ svcs -x svc:/network/routing/route:default
svc:/network/routing/route:default (in.routed network routing daemon)
 State: online since April 10, 2014 05:15:35 AM PDT
   See: in.routed(1M)
   See: /var/svc/log/network-routing-route:default.log
Impact: None.
```

サービスが実行中でない場合は、ここで停止できます。

2. **ルーティングデーモンを無効にします。**

```
# routeadm -d ipv4-forwarding -d ipv6-forwarding
# routeadm -d ipv4-routing -d ipv6-routing
# routeadm -u
```

3. **ルーティングデーモンが無効にされていることを確認します。**

```
$ svcs -x routing/route:default
svc:/network/routing/route:default (in.routed network routing daemon)
 State: disabled since April 11, 2014 10:10:10 AM PDT
Reason: Disabled by an administrator.
   See: http://support.oracle.com/msg/SMF-8000-05
   See: in.routed(1M)
Impact: This service is not running.
```

参照 routeadm(1M) のマニュアルページ

▼ ブロードキャストパケット転送を無効にする方法

デフォルトでは、Oracle Solaris はブロードキャストパケットを転送します。サイトのセキュリティーポリシーでブロードキャストフラッディングの可能性を減少させる必要がある場合は、この手順を使用してデフォルトを変更します。

注記 - `_forward_directed_broadcasts` ネットワークプロパティーを無効にすると、ブロードキャスト ping も無効になっています。

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IP パケットに対してブロードキャストパケット転送プロパティーを 0 に設定します。**

```
# ipadm set-prop -p _forward_directed_broadcasts=0 ip
```

2. **現在の値を検証します。**

```
# ipadm show-prop -p _forward_directed_broadcasts ip
PROTO  PROPERTY                    PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ip     _forward_directed_broadcasts  rw   0         --           0         0,1
```

参照 ipadm(1M) のマニュアルページ

▼ エコーリクエストへの応答を無効にする方法

この手順を使用して、ネットワークトポロジに関する情報の流布を回避します。

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IP パケットに対してブロードキャストエコー要求への応答プロパティーを 0 に設定して、現在の値を検証します。**

```
# ipadm set-prop -p _respond_to_echo_broadcast=0 ip

# ipadm show-prop -p _respond_to_echo_broadcast ip
PROTO  PROPERTY                 PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ip     _respond_to_echo_broadcast rw   0         --           1         0,1
```

2. **IP パケットに対してマルチキャストエコー要求への応答プロパティーを 0 に設定して、現在の値を検証します。**

```
# ipadm set-prop -p _respond_to_echo_multicast=0 ipv4
# ipadm set-prop -p _respond_to_echo_multicast=0 ipv6

# ipadm show-prop -p _respond_to_echo_multicast ipv4
PROTO  PROPERTY                 PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ipv4   _respond_to_echo_multicast rw   0         --           1         0,1
# ipadm show-prop -p _respond_to_echo_multicast ipv6
PROTO  PROPERTY                 PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ipv6   _respond_to_echo_multicast rw   0         --           1         0,1
```

参照 詳細は、『Oracle Solaris 11.2 カーネルのチューンアップ・リファレンスマニュアル 』の「_respond_to_echo_broadcast と _respond_to_echo_multicast (ipv4 または ipv6)」および `ipadm`(1M) のマニュアルページを参照してください。

▼ 厳密なマルチホームを設定する方法

他のドメインへのゲートウェイであるシステム (ファイアウォールや VPN ノードなど) では、この手順を使用して厳格なマルチホーミングをオンにします。`hostmodel` プロパティーは、マルチホームシステム上での IP パケットの送受信動作を制御します。

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IP パケットに対して `hostmodel` プロパティーを `strong` に設定します。**

```
# ipadm set-prop -p hostmodel=strong ipv4
# ipadm set-prop -p hostmodel=strong ipv6
```

2. **現在の値を確認し、可能な値に注目します。**

```
# ipadm show-prop -p hostmodel ip
PROTO  PROPERTY    PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ipv6   hostmodel   rw   strong    strong       weak      strong,src-priority,weak
ipv4   hostmodel   rw   strong    strong       weak      strong,src-priority,weak
```

参照 詳細は、『Oracle Solaris 11.2 カーネルのチューンアップ・リファレンスマニュアル 』の「hostmodel (ipv4 または ipv6)」および `ipadm`(1M) のマニュアルページを参照してください。

厳密なマルチホームの使用の詳細については、123 ページの「IPsec のトンネルモードで 2 つの LAN 間の接続を保護する方法」を参照してください。

▼ 不完全な TCP 接続の最大数を設定する方法

この手順を使用して、不完全な中断中の接続の数を制御することによってサービス拒否 (DOS) 攻撃を回避します。

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **受信接続の最大数を設定します。**

```
# ipadm set-prop -p _conn_req_max_q0=4096 tcp
```

2. **現在の値を検証します。**

```
# ipadm show-prop -p _conn_req_max_q0 tcp
PROTO  PROPERTY         PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
tcp    _conn_req_max_q0 rw   4096      --           128       1-4294967295
```

参照 詳細は、『Oracle Solaris 11.2 カーネルのチューンアップ・リファレンスマニュアル 』の「_conn_req_max_q0」および `ipadm`(1M) のマニュアルページを参照してください。

▼ 中断中の TCP 接続の最大数を設定する方法

この手順を使用して、許可される受信接続の数を制御することによって DOS 攻撃を回避します。

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **受信接続の最大数を設定します。**

```
# ipadm set-prop -p _conn_req_max_q=1024 tcp
```

2. **現在の値を検証します。**

```
# ipadm show-prop -p _conn_req_max_q tcp
PROTO  PROPERTY         PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
tcp    _conn_req_max_q  rw   1024      --           128       1-4294967295
```

参照 詳細は、『Oracle Solaris 11.2 カーネルのチューンアップ・リファレンスマニュアル 』の「_conn_req_max_q」および ipadm(1M) のマニュアルページを参照してください。

▼ 初期の TCP 接続に強固な乱数を指定する方法

この手順では、TCP の初期シーケンス番号生成パラメータが必ず RFC 6528 (http://www.ietf.org/rfc/rfc6528.txt) に準拠するようにします。

始める前に `solaris.admin.edit/etc.default/inetinit` 承認が割り当てられている管理者になる必要があります。デフォルトでは、`root` 役割がこの承認を持っています。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **`TCP_STRONG_ISS` 変数のデフォルト値が `2` であることを確認します。**

```
# grep TCP_STRONG /etc/default/inetinit
# TCP_STRONG_ISS sets the TCP initial sequence number generation parameters.
# Set TCP_STRONG_ISS to be:
TCP_STRONG_ISS=2
```

2. **`TCP_STRONG_ISS` の値が `2` 以外の場合は、`2` に変更します。**

```
# pfedit /etc/default/inetinit
TCP_STRONG_ISS=2
```

3. **システムをリブートします。**

```
# /usr/sbin/reboot
```

▼ ICMP リダイレクトを妨げる方法

ルーターは ICMP リダイレクトメッセージを使用して、ホストに宛先へのより直接的なルートを通知します。不正な ICMP リダイレクトメッセージは、中間者攻撃をまねく可能性があります。

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IP パケットに対して無視リダイレクトプロパティーを 1 に設定して、現在の値を検証します。**

   ICMP リダイレクトメッセージは、ホストのルートテーブルを変更し、認証されません。さらに、リダイレクトされたパケットの処理により、システムの CPU 要求が増加します。

```
# ipadm set-prop -p _ignore_redirect=1 ipv4
# ipadm set-prop -p _ignore_redirect=1 ipv6
# ipadm show-prop -p _ignore_redirect ipv4
PROTO  PROPERTY         PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ipv4  _ignore_redirect  rw   1         1            0         0,1
# ipadm show-prop -p _ignore_redirect ipv6
PROTO  PROPERTY         PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ipv6  _ignore_redirect  rw   1         1            0         0,1
```

2. **ICMP リダイレクトメッセージの送信を妨げます。**

   これらのメッセージには、ネットワークトポロジの一部を明らかにする可能性のあるルートテーブルの情報が含まれます。

```
# ipadm set-prop -p send_redirects=off ipv4
# ipadm set-prop -p send_redirects=off ipv6
# ipadm show-prop -p send_redirects ipv4
PROTO PROPERTY          PERM CURRENT  PERSISTENT   DEFAULT  POSSIBLE
ipv4  send_redirects    rw   off      off          on       on,off

# ipadm show-prop -p send_redirects ipv6
PROTO  PROPERTY        PERM CURRENT   PERSISTENT   DEFAULT  POSSIBLE
ipv6  send_redirects   rw   off       off          on       on,off
```

   詳細は、『Oracle Solaris 11.2 カーネルのチューンアップ・リファレンスマニュアル 』の「send_redirects (ipv4 or ipv6)」および `ipadm`(1M) のマニュアルページを参照してください。

▼ ネットワークパラメータをセキュアな値にリセットする方法

デフォルトでセキュアな多くのネットワークパラメータはチューニング可能で、デフォルトから変更されている可能性があります。サイトの条件が許す場合は、次のチューニング可能パラメータをデフォルト値に戻します。

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IP パケットに対してソースパケット転送プロパティーを 0 に設定して、現在の値を検証します。**

   デフォルト値で、なりすましパケットからの DOS 攻撃が回避されます。

```
# ipadm set-prop -p _forward_src_routed=0 ipv4
# ipadm set-prop -p _forward_src_routed=0 ipv6
# ipadm show-prop -p _forward_src_routed ipv4
PROTO  PROPERTY             PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ipv4  _forward_src_routed   rw   0         --           0         0,1
# ipadm show-prop -p _forward_src_routed ipv6
PROTO  PROPERTY             PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ipv6  _forward_src_routed   rw   0         --           0         0,1
```

   詳細は、『Oracle Solaris 11.2 カーネルのチューンアップ・リファレンスマニュアル 』の「forwarding (ipv4 または ipv6)」を参照してください。

2. **IP パケットに対してネットマスク応答プロパティーを 0 に設定して、現在の値を検証します。**

   デフォルト値で、ネットワークトポロジに関する情報の流布が回避されます。

```
# ipadm set-prop -p _respond_to_address_mask_broadcast=0 ip
# ipadm show-prop -p _respond_to_address_mask_broadcast ip
PROTO PROPERTY                          PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ip    _respond_to_address_mask_broadcast rw   0         --           0         0,1
```

3. **IP パケットに対してタイムスタンプ応答プロパティーを 0 に設定して、現在の値を検証します。**

   デフォルト値で、システムでの追加 CPU の要求が削除され、ネットワークに関する情報の流布が回避されます。

```
# ipadm set-prop -p _respond_to_timestamp=0 ip
# ipadm show-prop -p _respond_to_timestamp ip
PROTO  PROPERTY                       PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ip    _respond_to_timestamp           rw   0         --           0         0,1
```

4. **IP パケットに対してブロードキャストタイムスタンプ応答プロパティーを 0 に設定して、現在の値を検証します。**

   デフォルト値で、システムでの追加 CPU の要求が削除され、ネットワークに関する情報の流布が回避されます。

```
# ipadm set-prop -p _respond_to_timestamp_broadcast=0 ip
# ipadm show-prop -p _respond_to_timestamp_broadcast ip
PROTO  PROPERTY                       PERM CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ip    _respond_to_timestamp_broadcast  rw   0         --           0         0,1
```

5. **IP ソースルーティングを回避します。**

   デフォルト値により、パケットがネットワークセキュリティー対策をバイパスすることを回避します。ソースルーティングされたパケットにより、パケットの送信元は、ルーターに構成されているパスと異なるパスを提案できます。

   注記 - このパラメータは診断目的で `1` に設定できます。診断の終了後、値を `0` に戻します。

```
# ipadm set-prop -p _rev_src_routes=0 tcp
# ipadm show-prop -p _rev_src_routes tcp
PROTO PROPERTY          PERM CURRENT  PERSISTENT  DEFAULT  POSSIBLE
tcp   _rev_src_routes   rw   0        --          0        0,1
```

   詳細は、『Oracle Solaris 11.2 カーネルのチューンアップ・リファレンスマニュアル 』の「_rev_src_routes」を参照してください。

参照 `ipadm`(1M) のマニュアルページ

♦ ♦ ♦ 第 3 章

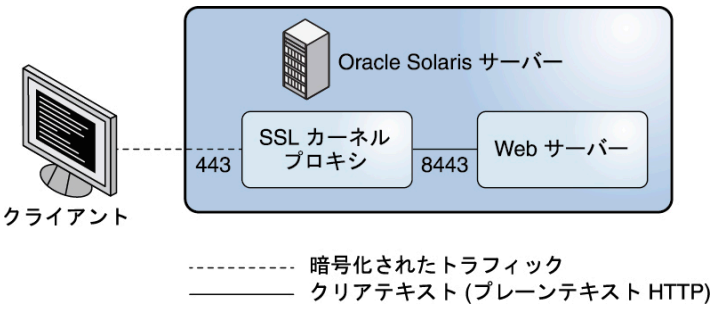
3 Web サーバーと Secure Sockets Layer プロトコル

この章では、Secure Sockets Layer (SSL) プロトコルを使用して、Oracle Solaris システムの Web サーバー通信を暗号化し、高速化する方法について説明します。

SSL カーネルプロキシ は Web サーバー通信を暗号化する

Oracle Solaris 上で実行するすべての Web サーバーはカーネルレベルでの SSL プロトコル、つまり *SSL カーネルプロキシ* を使用するように構成できます。そのような Web サーバーの例には、Apache 2.2 Web サーバーと Oracle iPlanet Web Server があります。SSL プロトコルを使えば、2 つのアプリケーションの間で、機密性、メッセージの完全性、およびエンドポイント認証を実現できます。SSL カーネルプロキシ が Web サーバー上で実行されていると、通信が高速化されます。次の図は、基本的な構成を示しています。

図 3-1 カーネル暗号化された Web サーバー通信



SSL カーネルプロキシ は、SSL プロトコルのサーバー側を実装します。プロキシにはいくつかの利点があります。

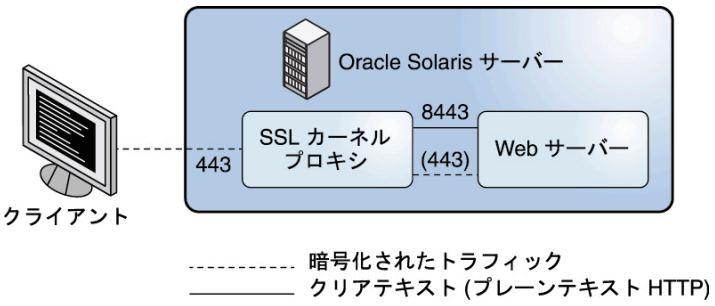
- このプロキシにより、Web サーバーのようなサーバーアプリケーションの SSL パフォーマンスが高速化するため、ユーザーレベルの SSL ライブラリに依存するアプリケーションよりも高いパフォーマンスを発揮します。アプリケーションのワークロードに応じて、パフォーマンスは 35% 以上向上する可能性があります。
- SSL カーネルプロキシ は透過的です。割り当てられた IP アドレスはありません。そのため、Webサーバーは、実際のクライアント IP アドレスと TCP ポートを参照します。
- SSL カーネルプロキシ と Web サーバーは連携するように設計されています。

  図3-1「カーネル暗号化された Web サーバー通信」に、SSL カーネルプロキシ を使用している Web サーバーの基本的なシナリオを示します。SSL カーネルプロキシ はポート 443 上に構成され、Web サーバーはポート 8443 上に構成されて、そこで暗号化されていない HTTP 通信を受信します。
- SSL カーネルプロキシ は、リクエストされた暗号化をサポートしていない場合に、ユーザーレベルの暗号化へのフォールバックを行うように構成できます。

  図3-2「ユーザーレベルフォールバックオプション付きのカーネル暗号化された Web サーバー通信」により複雑なシナリオを示します。Web サーバーと SSL カーネルプロキシ は、ユーザーレベルの Web サーバー SSL へのフォールバックを行うように構成されています。

  SSL カーネルプロキシ はポート 443 上に構成されています。Web サーバーは 2 つのポート上に構成されています。ポート 8443 は暗号化されていない HTTP 通信を受信し、ポート 443 はフォールバックポートです。フォールバックポートは、SSL カーネルプロキシ によってサポートされていない暗号化スイートで暗号化されている SSL トラフィックを受信します。

図 3-2 ユーザーレベルフォールバックオプション付きのカーネル暗号化された Web サーバー通信



SSL カーネルプロキシ はもっとも一般的な暗号化スイートに加えて、SSL 3.0 および TLS 1.0 プロトコルをサポートしています。完全な一覧については、ksslcfg(1M) のマニュアルページを参照してください。このプロキシは、サポートされないすべての暗号化スイートに対して、ユーザーレベルの SSL サーバーへのフォールバックを行うよう構成できます。

SSL カーネルプロキシ を使用して Web サーバーを保護する

次の手順に、SSL カーネルプロキシ を使用するように Web サーバーを構成する方法を示します。

- 35 ページの「SSL カーネルプロキシ を使用するように Apache 2.2 Web サーバーを構成する方法」
- 37 ページの「SSL カーネルプロキシ を使用するように Oracle iPlanet Web Server を構成する方法」
- 39 ページの「Apache 2.2 SSL にフォールバックするように SSL カーネルプロキシ を構成する方法」
- 42 ページの「ゾーンで SSL カーネルプロキシ を使用する方法」

▼ SSL カーネルプロキシ を使用するように Apache 2.2 Web サーバーを構成する方法

SSL カーネルプロキシ は Apache 2.2 Web サーバーでの SSL パケットの処理速度を向上できます。この手順では、図3-1「カーネル暗号化された Web サーバー通信」に示す簡単なシナリオを実装します。

始める前に Apache 2.2 Web サーバーを構成しています。この Web サーバーは Oracle Solaris に含まれています。

`root` 役割になる必要があります。

1. **Web サーバーを停止します。**

```
# svcadm disable svc:/network/http:apache22
```

2. **サーバー非公開鍵とサーバー証明書を 1 つのファイルに置きます。**

ssl.conf ファイルに SSLCertificateFile パラメータのみが指定されている場合、指定されたファイルは直接 SSL カーネルプロキシ で使用できます。

SSLCertificateKeyFile パラメータも指定されている場合は、証明書ファイルと非公開鍵ファイルを組み合わせる必要があります。ファイルを組み合わせるには、次のようなコマンドを実行します。

```
# cat cert.pem key.pem > cert-and-key.pem
```

3. **ksslcfg コマンドで使用するパラメータを決定します。**

すべてのオプションの一覧については、ksslcfg(1M) のマニュアルページを参照してください。指定する*必要がある*パラメータは次のとおりです。

- *key-format* – -f オプションと一緒に使用して、証明書と鍵の形式を定義します。SSL カーネルプロキシ の場合、サポートされる形式は pkcs11、pem、および pkcs12 です。
- *key-and-certificate-file* – -i オプションと一緒に使用して、サーバー鍵と pem および pkcs12 *key-format* オプションの証明書を格納するファイルの場所を設定します。
- *password-file* – -p オプションと一緒に使用して、pem または pkcs12 *key-format* オプションの非公開鍵を暗号化するために使用するパスワードを取得します。pkcs11 の場合、パスワードは PKCS #11 トークンに対して認証するために使用します。パスワードファイルは 0400 アクセス権で保護する必要があります。このファイルは無人リブート用に必要です。
- *token-label* – -T オプションと一緒に使用して、PKCS #11 トークンを指定します。
- *certificate-label* – -C オプションと一緒に使用して、PKCS #11 トークンの証明書オブジェクトのラベルを選択します。
- *proxy-port* – -x オプションと一緒に使用して、SSL プロキシポートを設定します。標準ポート 80 とは異なるポートを指定する必要があります。Web サーバーは暗号化されていないテキストトラフィックを SSL プロキシポートで待機します。一般に値は 8443 です。
- *ssl-port* – SSL カーネルプロキシ の待機するポートを指定します。一般に値は 443 です。

4. **SSL カーネルプロキシ のサービスインスタンスを作成します。**

次のいずれかの形式を使用して、SSL プロキシポートと関連パラメータを指定します。

- **鍵の形式として PEM または PKCS #12 を指定します。**

```
# ksslcfg create -f key-format -i key-and-certificate-file \
-p password-file -x proxy-port ssl-port
```

- **鍵の形式として PKCS #11 を指定します。**

```
# ksslcfg create -f pkcs11 -T PKCS11-token -C certificate-label \
-p password-file -x proxy-port ssl-port
```

5. **サービスインスタンスがオンラインであることを確認します。**

```
# svcs svc:/network/ssl/proxy
STATE          STIME    FMRI
online         02:22:22 svc:/network/ssl/proxy:default
```

次の出力は、サービスインスタンスが作成されなかったことを示します。

```
svcs: Pattern 'svc:/network/ssl/proxy' doesn't match any instances
STATE          STIME    FMRI
```

6. **SSL プロキシポート上で待機するように Web サーバーを構成します。**

`/etc/apache2/2.2/http.conf` ファイルを編集し、SSL プロキシポートを定義する行を追加します。サーバーの IP アドレスを使用した場合、Web サーバーはそのインタフェース上でのみ待機します。行は次のようになります。

```
Listen proxy-port
```

7. **Web サーバーの SMF 依存関係を設定します。**

Web サーバーサービスは、SSL カーネルプロキシ インスタンスの起動後にのみ起動できます。次のコマンドは、そうした依存関係を確立します。

```
# svccfg -s svc:/network/http:apache22
svc:/network/http:apache22> addpg kssl dependency
...apache22> setprop kssl/entities = fmri:svc:/network/ssl/proxy:kssl-INADDR_ANY-443
...apache22> setprop kssl/grouping = astring: require_all
...apache22> setprop kssl/restart_on = astring: refresh
...apache22> setprop kssl/type = astring: service
...apache22> end
```

8. **Web サーバーサービスを有効にします。**

```
# svcadm enable svc:/network/http:apache22
```

▼ SSL カーネルプロキシ を使用するように Oracle iPlanet Web Server を構成する方法

SSL カーネルプロキシ は Oracle iPlanet Web Server での SSL パケットの処理速度を向上できます。この手順では、図3-1「カーネル暗号化された Web サーバー通信」に示す簡単なシナリオを実装します。

始める前に Oracle iPlanet Web Server をインストールし、構成しています。サーバーは Oracle iPlanet Web Server (http://www.oracle.com/technetwork/middleware/iplanetwebserver-098726.html?ssSourceSiteId=ocomen) からダウンロードできます。手順については、Oracle iPLANET WEB SERVER 7.0.15 (http://docs.oracle.com/cd/E18958_01/index.htm) を参照してください。

Network Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **Web サーバーを停止します。**

   管理者の Web インタフェースを使ってサーバーを停止します。手順については、Oracle iPLANET WEB SERVER 7.0.15 (http://docs.oracle.com/cd/E18958_01/index.htm) を参照してください。

2. **`ksslcfg` コマンドで使用するパラメータを決定します。**

   すべてのオプションの一覧については、`ksslcfg`(1M) のマニュアルページを参照してください。指定する*必要がある*パラメータの一覧については、35 ページの「SSL カーネルプロキシを使用するように Apache 2.2 Web サーバーを構成する方法」の ステップ 3 を参照してください。

3. **SSL カーネルプロキシ のサービスインスタンスを作成します。**

   次のいずれかの形式を使用して、SSL プロキシポートと関連パラメータを指定します。

   - **鍵の形式として PEM または PKCS #12 を指定します。**

     ```
     # ksslcfg create -f key-format -i key-and-certificate-file \
     -p password-file -x proxy-port ssl-port
     ```

   - **鍵の形式として PKCS #11 を指定します。**

     ```
     # ksslcfg create -f pkcs11 -T PKCS11-token -C certificate-label \
     -p password-file -x proxy-port ssl-port
     ```

4. **インスタンスがオンラインであることを確認します。**

   ```
   # svcs svc:/network/ssl/proxy
   STATE          STIME    FMRI
   online         02:22:22 svc:/network/ssl/proxy:default
   ```

5. **SSL プロキシポート上で待機するように Web サーバーを構成します。**

手順については、Oracle iPLANET WEB SERVER 7.0.15 (http://docs.oracle.com/cd/E18958_01/index.htm) を参照してください。

6. **Web サーバーの SMF 依存関係を設定します。**

   Web サーバーサービスは、SSL カーネルプロキシ インスタンスの起動後にのみ起動できます。Web サーバーサービスの FMRI が `svc:/network/http:webserver7` であるとして、次のコマンドはその依存関係を確立します。

```
# svccfg -s svc:/network/http:webserver7
svc:/network/http:webserver7> addpg kssl dependency
...webserver7> setprop kssl/entities = fmri:svc:/network/ssl/proxy:kssl-INADDR_ANY-443
...webserver7> setprop kssl/grouping = astring: require_all
...webserver7> setprop kssl/restart_on = astring: refresh
...webserver7> setprop kssl/type = astring: service
...webserver7> end
```

7. **Web サーバーサービスを有効にします。**

```
# svcadm enable svc:/network/http:webserver7
```

▼ Apache 2.2 SSL にフォールバックするように SSL カーネルプロキシ を構成する方法

この手順では、最初から Apache 2.2 Web サーバーを構成し、プライマリ SSL セッション処理メカニズムとして、SSL カーネルプロキシ を構成します。クライアントが提供する一連の SSL 暗号化に SSL カーネルプロキシ が提供する暗号化が含まれない場合、Apache 2.2 Web サーバーはフォールバックメカニズムとして機能します。この手順は、図3-2「ユーザーレベルフォールバックオプション付きのカーネル暗号化された Web サーバー通信」に示す複雑なシナリオを実装します。

始める前に `root` 役割になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **Apache 2.2 Web サーバーで、サーバーの SSL カーネルプロキシ によって使用される鍵証明書を作成します。**

   a. **証明書署名要求 (CSR) を生成します。**

      次のコマンドは CSR とそれに関連付けられた SSL カーネルプロキシ の非公開鍵を作成します。

```
# cd /root
# openssl req \
> -x509 -new \
> -subj "/C=CZ/ST=Prague region/L=Prague/CN=`hostname`" \
> -newkey rsa:2048 -keyout webkey.pem \
> -out webcert.pem
Generating a 2048 bit RSA private key
.+++
........+++
writing new private key to 'webkey.pem'
Enter PEM pass phrase: JohnnyCashIsCool
Verifying - Enter PEM pass phrase:  JohnnyCashIsCool
#
# chmod 440 /root/webcert.pem ; chown root:webservd /root/webcert.pem
```

注記 - FIPS 140 に準拠する場合、RSA 鍵の最小の長さは 2048 です。詳細は、『Using a FIPS 140 Enabled System in Oracle Solaris 11.2 』を参照してください。

詳細は、openssl(5) のマニュアルページを参照してください。

b. **CSR を認証局 (CA) に送信します。**

c. **webcert.pem ファイルを CA からの署名付き証明書で置き換えます。**

2. **パスフレーズと公開/非公開鍵証明書で SSL カーネルプロキシ を構成します。**

a. **パスフレーズを作成、保存、保護します。**

```
# echo "RefrigeratorsAreCool" > /root/kssl.pass
# chmod 440 /root/kssl.pass; chown root:webservd /root/kssl.pass
```

注記 - パスフレーズに空白文字を含めることはできません。

b. **非公開鍵証明書と公開鍵証明書を 1 つのファイルに組み合わせます。**

```
# cat /root/webcert.pem /root/webkey.pem > /root/webcombo.pem
```

c. **公開/非公開鍵証明書とパスフレーズで SSL カーネルプロキシ を構成します。**

```
# ksslcfg create -f pem -i /root/webcombo.pem -x 8443 -p /root/kssl.pass 443
```

3. **暗号化されていない通信をポート 8443 で待機するように Web サーバーを構成します。**

/etc/apache2/2.2/httpd.conf ファイル内の Listen 行を編集します。

```
# pfedit /etc/apache2/2.2/httpd.conf
```

```
...
## Listen 80
Listen 8443
```

4. **SSL モジュールテンプレート `ssl.conf` を Apache 構成ディレクトリに追加します。**

```
# cp /etc/apache2/2.2/samples-conf.d/ssl.conf /etc/apache2/2.2/ssl.conf
```

このモジュールは、暗号化接続をポート 443 上で待機することを追加します。

5. **Web サーバーが `/root/kssl.pass` ファイル内のパスフレーズを復号化できるようにします。**

   a. **`kssl.pass` ファイルを読み取るシェルスクリプトを作成します。**

```
# pfedit /root/put-passphrase.sh
#!/usr/bin/ksh -p
## Reads SSL カーネルプロキシ passphrase
/usr/bin/cat /root/kssl.pass
```

   b. **スクリプトを実行可能ファイルにし、ファイルを保護します。**

```
# chmod 500 /root/put-passphrase.sh
# chown webservd:webservd /root/put-passphrase.sh
```

   c. **`ssl.conf` ファイルの `SSLPassPhraseDialog` パラメータをこのシェルスクリプトを呼び出すように変更します。**

```
# pfedit /etc/apache2/2.2/ssl.conf
...
## SSLPassPhraseDialog  builtin
SSLPassPhraseDialog exec:/root/put-passphrase.sh
```

6. **Web サーバーの公開鍵証明書と非公開鍵証明書を正しい場所に置きます。**

`ssl.conf` ファイル内の `SSLCertificateFile` および `SSLCertificateKeyFile` パラメータの値に、予想される配置と名前が格納されています。証明書を正しい場所にコピーまたはリンクできます。

```
# ln -s  /root/webcert.pem /etc/apache2/2.2/server.crt     SSLCertificateFile default location
# ln -s /root/webkey.pem /etc/apache2/2.2/server.key     SSLCertificateKeyFile default location
```

7. **Apache サービスを有効にします。**

```
# svcadm enable apache22
```

8. **(オプション) 2 つのポートが機能しているかどうかを確認します。**

`openssl s_client` コマンドと `kstat` コマンドを使用して、パケットを表示します。

a. **SSL カーネルプロキシ で使用可能な暗号化を使用します。**

```
# openssl s_client -cipher RC4-SHA -connect web-server:443
```

kstat カウンタ kssl_full_handshakes が 1 増加すると、SSL セッションが SSL カーネルプロキシ によって処理されたことを証明します。

```
# kstat -m kssl -s kssl_full_handshakes
```

b. **SSL カーネルプロキシ で使用できない暗号化を使用します。**

```
# openssl s_client -cipher CAMELLIA256-SHA -connect web-server:443
```

kstat カウンタ kssl_fallback_connections が 1 増加すると、パケットが到着したが、SSL セッションが Apache Web サーバーによって処理されたことを証明します。

```
# kstat -m kssl -s kssl_fallback_connections
```

例 3-1 SSL カーネルプロキシ を使用するように Apache 2.2 Web サーバーを構成する

次のコマンドは、pem 鍵形式を使う SSL カーネルプロキシ のサービスインスタンスを作成します。

```
# ksslcfg create -f pem -i cert-and-key.pem -p kssl.pass -x 8443 443
```

▼ ゾーンで SSL カーネルプロキシ を使用する方法

SSL カーネルプロキシ は次の制限付きで、ゾーン内で動作します。

- カーネル SSL の管理はすべて、大域ゾーンで行なう必要があります。大域ゾーンの管理者は、局所ゾーン内の証明書や鍵のファイルにアクセスできる必要があります。大域ゾーンで ksslcfg コマンドによってサービスインスタンスを構成すると、非大域ゾーンで Web サーバーを起動できるようになります。
- インスタンスを構成する際には、ksslcfg コマンドを実行して特定のホスト名または IP アドレスを指定する必要があります。特に、インスタンスは IP アドレスに INADDR_ANY を指定できません。

始める前に Web サーバーサービスは非大域ゾーンで構成され、有効にされます。

Network Security および Zone Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **非大域ゾーン内で Web サーバーを停止します。**

   たとえば、apache-zone ゾーン内で Apache Web サーバーを停止するには、次のコマンドを実行します。

   ```
   apache-zone # svcadm disable svc:/network/http:apache22
   ```

2. **大域ゾーンで、ゾーン内に SSL カーネルプロキシ のサービスインスタンスを作成します。**

   apache-zone のサービスインスタンスを作成するには、次のようなコマンドを実行します。

   ```
   # ksslcfg create -f pem -i /zone/apache-zone/root/keypair.pem \
   -p /zone/apache-zone/root/skppass -x 8443 apache-zone 443
   ```

3. **非大域ゾーンで、Web サービスインスタンスを有効にします。**

   たとえば、apache-zone で Web サービスを有効にします。

   ```
   apache-zone # svcadm enable svc:/network/http:apache22
   ```

♦ ♦ ♦ 第 4 章

4 Oracle Solaris の IP フィルタについて

この章では、Oracle Solaris の IP フィルタ機能の概要を説明します。IP フィルタのタスクについては、第5章「IP フィルタの構成」を参照してください。

この章では、次の内容について説明します。

IP フィルタとは

Oracle Solaris の IP フィルタ機能は、ステートフルパケットフィルタリングとネットワークアドレス変換 (NAT) を行います。IP フィルタには、ステートレスパケットフィルタリングと、アドレスプールの作成および管理を行う機能もあります。

パケットのフィルタリングは、ネットワークベースの攻撃に対する基本的な保護を提供します。IP フィルタは、IP アドレス、ポート、プロトコル、ネットワークインタフェース、およびトラフィックの転送方向によって、フィルタリングを行うことができます。また、発信元 IP アドレス、宛先 IP アドレス、IP アドレスの範囲、またはアドレスプールによってもフィルタリングを行うことができます。

IP フィルタは、オープンソースの IP フィルタソフトウェアから派生したものです。オープンソースの IP フィルタのライセンス契約の条件、作者、および著作権宣言文を参照するためのデフォルトパスは、`/usr/lib/ipf/IPFILTER.LICENCE` です。Oracle Solaris がデフォルト以外の場

所にインストールされている場合は、所定のパスを修正して、インストールした場所にあるファイルにアクセスします。

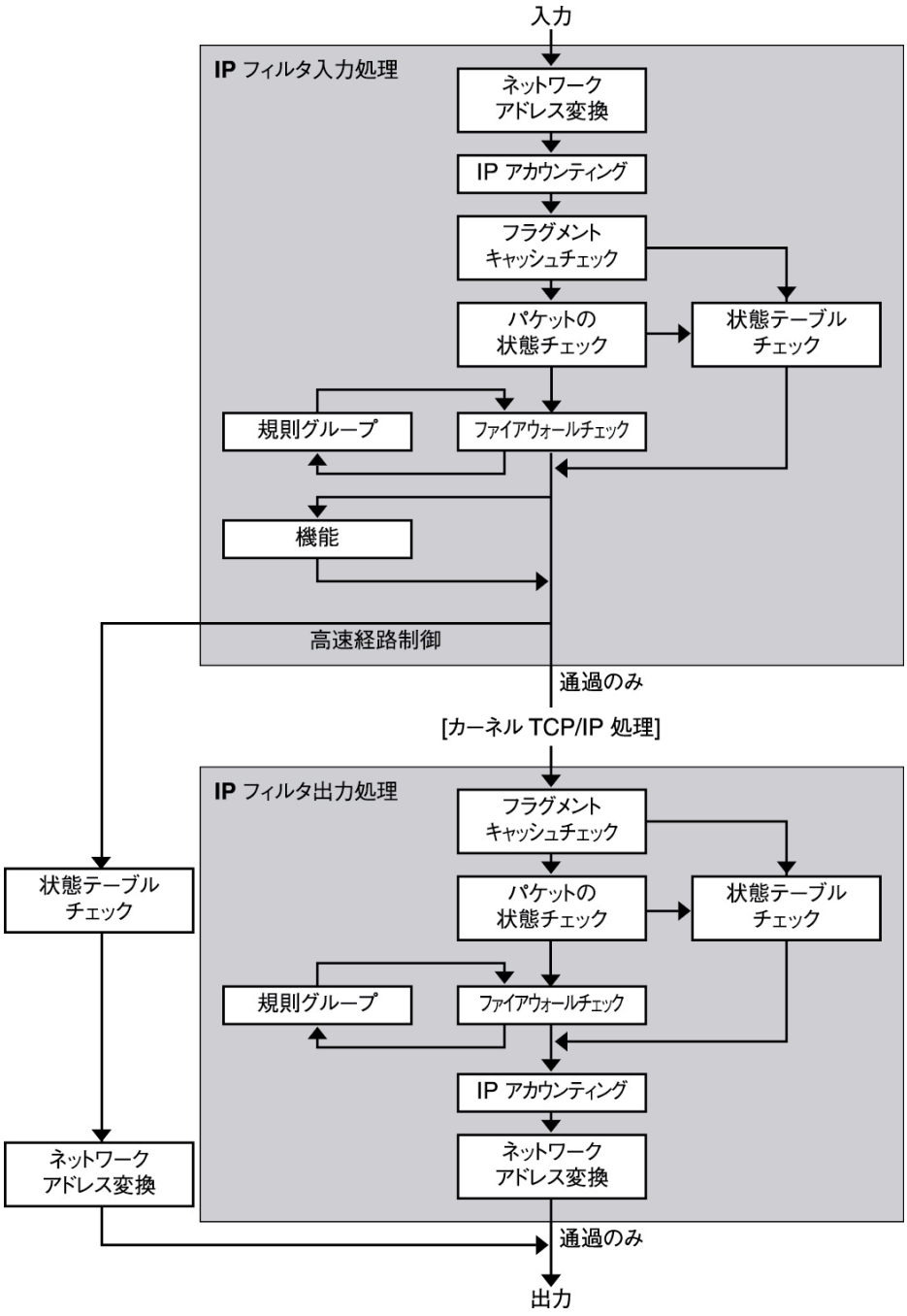
オープンソースの IP フィルタの情報ソース

Darren Reed によるオープンソースの IP フィルタソフトウェアのホームページは、http://coombs.anu.edu.au/~avalon/ip-filter.html にあります。このサイトには、チュートリアル「IP Filter Based Firewalls HOWTO」(Brendan Conoboy および Erik Fichtner 著、2002 年) へのリンクなど、オープンソースの IP フィルタに関する情報が含まれています。このチュートリアルは、BSD UNIX 環境でファイアウォールを作成する方法を手順ごとに説明しています。このチュートリアルは BSD UNIX 環境向けに書かれていますが、Oracle Solaris 上の IP フィルタの構成にも関連しています。

IP フィルタのパケット処理

IP フィルタは、パケットが処理されるときに一連の手順を実行します。次の図は、パケット処理の段階と、フィルタが TCP/IP プロトコルスタックとどのように統合されるかを示しています。

図 4-1 パケット処理の順序



パケット処理には次の手順が含まれます。

- **ネットワークアドレス変換 (NAT)**

  プライベート IP アドレスを異なる公開アドレスに変換するか、複数のプライベートアドレスの別名として単一の公開アドレスを使用します。NAT を使用すると、組織に既存のネットワークがあり、インターネットにアクセスする必要がある場合に、IP アドレスが枯渇する問題を解決できます。

- **IP アカウンティング**

  入力と出力の規則を個別に設定し、通過するバイト数を記録できます。規則に一致する数に達するたびに、パケットのバイト数を規則に追加し、段階的な統計を収集できます。

- **フラグメントキャッシュチェック**

  デフォルトで、断片化されたパケットはキャッシュされます。特定のパケットのすべてのフラグメントが到着すると、フィルタリング規則が適用され、フラグメントが許可されるか、ブロックされます。規則ファイルに `set defrag off` が表示されている場合、フラグメントはキャッシュされません。

- **パケットの状態チェック**

  規則に `keep state` が含まれている場合、指定されたセッション内のすべてのパケットは、規則で `pass` または `block` のどちらが指定されているかに応じて自動的に通されるかブロックされます。

- **ファイアウォールチェック**

  入力と出力の規則は個別に設定が可能で、パケットが IP フィルタを通過してカーネルの TCP/IP ルーチン内に受信したり、またはネットワーク上に送信されることを許可するかどうかを決定できます。

- **グループ**

  グループを使用すると、ツリー形式で規則セットを作成できます。

- **機能**

  機能とは、実行されるアクションです。`block`、 `pass`、`literal`、および `send ICMP response` などの機能を実行できます。

- **高速経路制御**

  高速ルートは、パケットをルーティングのための UNIX IP スタックに渡さないように IP Filter に指示し、TTL の減少を防ぎます。

- **IP 認証**

  認証されたパケットが、ファイアウォールループを 1 回だけ通過するようにして、重複した処理を防止します。

IP フィルタの使用ガイドライン

- IP フィルタは SMF サービス `svc:/network/ipfilter` によって管理されます。SMF の完全な概要については、『Oracle Solaris 11.2 でのシステムサービスの管理 』の第 1 章「サービス管理機能の概要」を参照してください。SMF に関連するステップごとの手順については、『Oracle Solaris 11.2 でのシステムサービスの管理 』の第 3 章「サービスの管理」を参照してください。
- IP フィルタでは、構成ファイルを直接編集する必要があります。
- IP フィルタは Oracle Solaris の一部としてインストールされます。システムが自動ネットワーク接続を使用するように構成されている場合、デフォルトで、IP フィルタサービスは有効にされています。自動ネットワークプロファイルは、`nwam`(5) および `netadm`(1M) のマニュアルページに説明するように、このファイアウォールを有効にします。自動的にネットワーク接続されるシステムでのカスタム構成の場合、IP フィルタサービスは有効にされません。このサービスを有効にするための関連タスクについては、59 ページの「IP フィルタサービスの構成」を参照してください。
- IP フィルタを管理するには、`root` 役割になるか、IP Filter Management 権利プロファイルを割り当てられている必要があります。IP Filter Management 権利プロファイルは、自分が作成したユーザーまたは役割に割り当てることができます。役割を作成してその役割をユーザーに割り当てるには、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「役割の作成」を参照してください。
- Oracle Solaris クラスタソフトウェアは、スケーラブルサービス用の IP フィルタによるフィルタリングはサポートしませんが、フェイルオーバーサービス用の IP フィルタはサポートします。IP フィルタをクラスタ内で構成するときのガイドラインおよび制限については、『*Oracle Solaris Cluster ソフトウェアのインストール*』の「Oracle Solaris OS の機能制限」を参照してください。
- ゾーン間のフィルタリングは、IP フィルタ規則が実装されているゾーンが、システム上のほかのゾーンの仮想ルーターとして機能する場合にかぎりサポートされます。

IP フィルタの構成ファイルの使用

IP フィルタを使用すると、ファイアウォールサービスまたはネットワークアドレス変換 (NAT) が利用できるようになります。ファイアウォールと NAT の規則はデフォルトで提供されません。カスタム構成ファイルを作成し、これらのファイルのパス名を IP フィルタサービスプロパティーの値として、設定する必要があります。サービスを有効にすると、システムのリブート時にこれら

のファイルが自動的にロードされます。サンプル構成ファイルについては、86 ページの「IP フィルタの構成ファイルの例」を参照してください。詳細については、svc.ipfd(1M) のマニュアルページを参照してください。

IP フィルタの規則セットの使用

ファイアウォールを管理するために、IP フィルタを使用して、ネットワークトラフィックをフィルタリングするための規則セットを指定します。次の種類の規則セットを作成できます。

- パケットフィルタリング規則セット
- ネットワークアドレス変換 (NAT) 規則セット

また、IP アドレスのグループを参照するためにアドレスプールを作成できます。作成したプールは、あとで規則セット内で使用できます。アドレスプールはルール処理を高速化できます。また、アドレスプールによって、大きなまとまりのアドレスをより簡単に管理できます。

IP フィルタのパケットのフィルタリング機能の使用

パケットフィルタリング規則セットを使用して、パケットのフィルタリングを設定します。ipf コマンドで、パケットフィルタリング規則セットを処理します。ipf コマンドの詳細については、ipf(1M) コマンドを参照してください。

パケットフィルタリング規則は、ipf コマンドによってコマンド行で作成することも、パケットフィルタリングの構成ファイル内で作成することもできます。構成ファイルをロードするには、ファイルを作成してから、そのパス名を IP フィルタサービスに指定する必要があります。

IP フィルタには、アクティブ規則セットと非アクティブ規則セットの 2 つのパケットフィルタリング規則セットを維持管理することができます。大部分の場合、作業ではアクティブ規則セットを使用します。ただし、ipf -I コマンドを使用すると、コマンドアクションをアクティブでない規則リストに適用できます。非アクティブ規則リストは、ユーザーが選択しない限り、IP フィルタによって使用されることはありません。非アクティブ規則リストによって、アクティブなパケットのフィルタリングに影響を与えずに、規則を保存できます。

IP フィルタは、構成された規則リストの最初から最後まで規則を処理してから、パケットの通過またはブロックを行います。IP フィルタは、パケットを通過させるかどうかを決めるフラグを維持

管理しています。フラグは、規則セット全体を調べ、最後に一致した規則を基にパケットを通過させるか、ブロックするかを決定します。

このプロセスには、2 つの例外があります。最初の例外は、パケットが quick キーワードを含む規則に一致した場合です。規則が quick キーワードを含む場合は、その規則に対する処理が実行され、それ以降の規則はチェックされません。2 番目の例外は、パケットが group キーワードを含む規則に一致した場合です。パケットがグループに一致すると、グループでタグ付けされた規則だけがチェックされます。

パケットのフィルタリング規則の構成

パケットのフィルタリング規則を作成するには、次の構文を使用します。

action `[in|out]` *option keyword, keyword...*

1. 各規則がアクションを開始します。IP フィルタは、パケットが規則に適合する場合、アクションを実行します。次の一覧に、パケットに対して実行される一般的なアクションを示します。

`block`	パケットはフィルタを通過できません。
`pass`	パケットはフィルタを通過します。
`log`	パケットをロギングしますが、パケットをブロックするか、通過させるかの決定は行いません。ログを参照するには、`ipmon` コマンドを使用します。
`count`	フィルタの統計にパケットを含めます。統計を参照するには、`ipfstat` コマンドを使用します。
`skip` *number*	フィルタは *number* フィルタリング規則をスキップします。
`auth`	パケット情報を確認するユーザープログラムが実行するパケット認証を要求します。このプログラムは、パケットを通過させるか、ブロックするかを決定します。

2. アクション後の出力は、`in` または `out` のはずです。ユーザーの選択により、パケットのフィルタリング規則が、受信パケットと発信パケットのどちらに適用されるのかが決定されます。
3. 次に、オプションの一覧からオプションを選択します。複数のオプションを使用する場合は、次の順序で使用してください。

`log`	規則が最後に一致した規則の場合、パケットをロギングします。ログを参照するには、`ipmon` コマンドを使用します。
`quick`	パケットが一致した場合、`quick` オプションを含む規則を実行します。これ以上の規則チェックは行われません。
`on` *interface-name*	パケットが指定したインタフェースを出入りする場合だけ、規則を適用します。
`dup-to` *interface-name*	パケットをコピーし、*interface-name* 上の複製をオプションで指定した IP アドレスに送信します。

注記 - ルール内の `dup-to` オプションによって、ネットワーク管理者は *network tap* を作成できます。このオプションはまだ Oracle Solaris でサポートされていますが、その重要性は大幅に減少しています。最新のスイッチを利用するとそのポートがネットワークタップを実行するように直接構成でき、この機能をルールで定義する必要がなくなります。ポートをネットワーク接続するように構成するには、スイッチのドキュメントを参照してください。

`to` *interface-name*	パケットを *interface-name* のアウトバウンドキューに移動します。

4. オプションの指定後、パケットが規則に一致するかどうかを決定するさまざまなキーワードを選択できます。次のキーワードは、次の順序で使用してください。

注記 - デフォルトでは、構成ファイルのいずれの規則にも一致しないパケットは、すべてフィルタを通過します。

`tos`	16 進数または 10 進数の整数で表されたサービスタイプの値を基に、パケットをフィルタリングします。
`ttl`	生存期間の値を基に、パケットの一致を取ります。パケットに保存されている生存期間の値は、破棄される前にパケットがネットワーク上に存在できる期間を示します。
`proto`	特定のプロトコルに対して一致を取ります。`/etc/protocols` ファイルに指定されている任意のプロトコル名を使用したり、そのプロト

コルを表す 10 進数の数を指定したりできます。キーワード `tcp/udp` は、TCP または UDP パケットとの一致を取るために使用できます。

`from/to/all/any` 発信元 IP アドレス、宛先 IP アドレス およびポート番号のいずれか、またはすべてに対して一致を取ります。`all` キーワードは、すべての発信元からのパケットおよびすべての宛先へのパケットを受諾するために使用します。

`with` パケットに関連する指定された属性に対して一致を取ります。オプションがない場合にパケットを一致させるには、キーワードの前に `not` または `no` と記述します。

`flags` 設定されている TCP フラグを基にフィルタリングを行う TCP で使用します。TCP フラグの詳細については、`ipf`(4) のマニュアルページを参照してください。

`icmp-type` ICMP のタイプによってフィルタリングを行います。このキーワードは `proto` オプションが `icmp` に設定されているときに使用され、`flags` オプションが指定されているときは使用されません。

`keep` *keep-options* 保存しておくパケットの情報を決定します。使用可能な *keep-options* には `state` オプションが含まれます。`state` オプションは、セッションに関する情報を、TCP、UDP、および ICMP パケットで保存できます。

`head` *number* 番号 *number* で指定されるフィルタリング規則に対して、新しいグループを作成します。

`group` *number* デフォルトグループではなく、グループ番号 *number* のグループに規則を追加します。ほかのグループを指定しない場合は、すべてのフィルタリング規則がグループ 0 に保存されます。

次の例は、規則を作成するためにパケットのフィルタリング規則構文をまとめる方法を示しています。IP アドレス `192.168.0.0/16` からの受信トラフィックをブロックするには、規則リストに次の規則を含めます。

```
block in quick from 192.168.0.0/16 to any
```

パケットフィルタリング規則を記述するときの詳細な文法および構文については、ipf(4) のマニュアルページを参照してください。パケットのフィルタリングに関連するタスクについては、67 ページの「IP フィルタのパケットフィルタリング規則セットの管理」を参照してください。例に示されている IP アドレススキーム (192.168.0.0/16) の説明については、『Oracle Solaris 11.2 でのネットワーク配備の計画 』の第 1 章「ネットワーク配備の計画」を参照してください。

IP フィルタの NAT 機能の使用

NAT は、発信元 IP アドレスと宛先 IP アドレスをほかのインターネットアドレスまたはイントラネットアドレスに変換するマッピング規則を設定します。これらの規則は、受信 IP パケットまたは発信 IP パケットの 発信元アドレスおよび宛先アドレスを変更し、パケットを送信します。また、NAT を使用して、あるポートから別のポートにトラフィックの方向を変更することもできます。NAT は、パケットに修正または方向の変更が行われても、パケットの完全性を維持します。

NAT 規則は、ipnat コマンドを使用してコマンド行で作成することも NAT 構成ファイルで作成することもできます。NAT 構成ファイルを作成し、そのパス名をサービスの config/ipnat_config_file プロパティーの値として設定する必要があります。デフォルト値は /etc/ipf/ipnat.conf です。詳細は、ipnat(1M) コマンドを参照してください。

NAT 規則は IPv4 と IPv6 アドレス両方に適用できます。ただし、アドレスの種類ごとに個別のルールを作成する必要があります。IPv6 アドレスを含む NAT 規則では、mapproxy および rdrproxy NAT コマンドを同時に使用することはできません。

NAT 規則の構成

次の構文で NAT 規則を作成します。

command interface-name parameters

1. 各規則の冒頭には、次のコマンドのいずれかが記述されています。

map	ある IP アドレスまたはネットワークを規制のないラウンドロビン方式で別の IP アドレスまたはネットワークにマッピングします。
rdr	ある IP アドレスとポートのペアから別の IP アドレスとポートのペアにパケットの方向を変更します。

`bimap` 外部 IP アドレスと内部 IP アドレス間で双方向の NAT を確立します。

`map-block` 静的 IP アドレスをベースにした変換を確立します。このコマンドは、アドレスを指定の範囲に変換するアルゴリズムに基づいています。

2. このコマンドに続く単語は、`bge0` などのインタフェース名です。
3. 次に、NAT 構成を決定するさまざまなパラメータを選択します。次に、この種のパラメータの例をいくつか挙げます。

`ipmask` ネットワークマスクを指定します。

`dstipmask` `ipmask` が変換されるアドレスを指定します。

`mapport` ポート番号の範囲と `tcp`、`udp` または `tcp/udp` プロトコルを指定します。

次の例は、NAT 規則を構築する方法を示しています。発信元アドレスが `192.168.1.0/24` のデバイス `net2` から発信されるパケットを書き換え、外部に対して発信元アドレスが `10.1.0.0/16` であることを示すには、NAT 規則セットに次の規則を含めます。

```
map net2 192.168.1.0/24 -> 10.1.0.0/16
```

次の規則は IPv6 アドレスに適用されます。

```
map net3 fec0:1::/64 -> 2000:1:2::/72 portmap tcp/udp 1025:65000
map-block net3 fe80:0:0:209::/64 -> 209:1:2::/72 ports auto
rdr net0 209::ffff:fe13:e43e port 80 -> fec0:1::e,fec0:1::f port 80 tcp round-robin
```

詳細な文法と構文については、ipnat(4) のマニュアルページを参照してください。

IP フィルタのアドレスプール機能の使用

アドレスプールは、アドレスとネットマスクのペアのグループに対して単一の参照を確立します。アドレスプールは、IP アドレスとルールを一致させるのに必要な時間を短縮します。また、アドレスプールによって、大きなまとまりのアドレスをより簡単に管理できます。

アドレスプール構成規則は、IP フィルタサービスによって読み込まれるファイル内に置くことができます。ファイルを作成して、そのパス名をサービスの `config/ippool_config_file` プロパティーの値として設定する必要があります。デフォルト値は `/etc/ipf/ippool.conf` です。

アドレスプールの構成

次の構文でアドレスプールを作成します。

`table role =` *role-name* `type =` *storage-format* `number =` *reference-number*

`table` 複数のアドレスへの参照を定義します。

`role` IP フィルタでのプールの役割を指定します。参照できる役割は `ipf` だけです。

`type` プールの保存形式を指定します。

`number` フィルタリング規則が使用する参照番号を指定します。

たとえば、アドレスが `10.1.1.1` および `10.1.1.2` でネットワークが `192.16.1.0` のグループをプール番号 `13` で参照する場合、アドレスプールの構成ファイルに次の規則を含めます。

```
table role = ipf type = tree number = 13
{ 10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24 };
```

次に、フィルタリング規則のプール番号 `13` を参照するには、次の例のような規則を構築します。

```
pass in from pool/13 to any
```

なお、プールへの参照を含む規則ファイルをロードする前に、プールファイルをロードする必要があります。プールファイルをロードしていない場合、次の出力のようにプールは未定義となります。

```
# ipfstat -io
empty list for ipfilter(out)
block in from pool/13(!) to any
```

プールをあとで追加しても、そのプールの追加によってカーネルの規則セットが更新されることはありません。そのプールを参照する規則ファイルも再ロードする必要があります。

詳細な文法と構文については、`ippool`(4) のマニュアルページを参照してください。

IP フィルタ用の IPv6

IPv6 パケットフィルタリングでは、発信元または宛先の IPv6 アドレス、IPv6 アドレスを含むプール、および IPv6 拡張ヘッダーに基づいて、パケットを取り出すことができます。

IPv6 は、多くの点で IPv4 に似ています。ただし、これら 2 つの IP バージョンは、ヘッダーとパケットサイズが異なっています。IP フィルタでは、これらは重要な要素です。IPv6 パケットには、「*ジャンボグラム*」と呼ばれる、65,535 バイトより大きなデータグラムが含まれています。IP フィルタでは、IPv6 ジャンボグラムはサポートされていません。

注記 - ジャンボグラムの詳細については、IPv6 Jumbograms, RFC 2675 (http://www.ietf.org/rfc/rfc2675.txt) を参照してください。

IPv6 に関連する IP フィルタのタスクは、IPv4 とほとんど変わりません。もっとも大きな違いは、特定のコマンドで `-6` オプションを使用することです。`ipf` コマンドと `ipfstat` コマンドには、IPv6 パケットフィルタリングで使用するための `-6` オプションが用意されています。IPv6 パケットフィルタリング規則をロードおよびフラッシュするときは、`ipf` コマンドで `-6` オプションを使用します。IPv6 統計を表示するときは、`ipfstat` コマンドに `-6` オプションを使用します。`ipmon` コマンドと `ippool` コマンドでも IPv6 がサポートされますが、IPv6 をサポートするためのオプションは指定しません。`ipmon` コマンドは、IPv6 パケットのロギングに対応するように拡張されています。`ippool` コマンドでは、IPv6 アドレスのプールをサポートしています。IPv4 アドレス用と IPv6 アドレス用の個別のプールを作成することも、IPv4 アドレスと IPv6 アドレスの両方を格納するプールを作成することもできます。

再利用可能な IPv6 パケットフィルタリング規則を作成するには、特定の IPv6 ファイルを作成する必要があります。次に、そのパス名を IP フィルタサービスの `config/ip6_config_file` プロパティーの値として設定します。デフォルト値は `/etc/ipf/ip6.conf` です。

IP フィルタに関連するタスクについては、第5章「IP フィルタの構成」を参照してください。

IP フィルタのマニュアルページ

次のマニュアルページで IP フィルタについて説明しています。

`ipf`(1M)	IP フィルタ規則を管理し、チューニング可能パラメータを表示し、他のタスクを実行します。
`ipf`(4)	IP フィルタパケットのフィルタリング規則を作成するための文法と構文を含む。
`ipfilter`(5)	IP フィルタソフトウェアについて説明します。
`ipfs`(1M)	リブート後も、NAT 情報と状態テーブル情報を保存し、復元します。

ipfstat(1M)	パケット処理の統計情報を取得して表示します。
ipmon(1M)	ログデバイスを開き、パケットフィルタリングと NAT の両方について記録されたパケットを表示します。
ipnat(1M)	NAT 規則を管理し、NAT 統計情報を表示します。
ipnat(4)	NAT 規則を作成するための文法と構文を含む
ippool(1M)	アドレスプールを作成し、管理します。
ippool(4)	IP フィルタアドレスプールを作成するための文法と構文を含む。
svc.ipfd(1M)	IP フィルタサービスの構成に関する情報を提供します。

♦ ♦ ♦ 第 5 章

5 IP フィルタの構成

この章では、IP フィルタのタスクの手順をステップごとに説明します。概要については、第4章「Oracle Solaris の IP フィルタについて」を参照してください。

この章で扱う内容は、次のとおりです。

IP フィルタサービスの構成

次のタスクマップは、IP フィルタ規則を作成し、サービスを有効または無効にする手順を一覧表示したものです。

表 5-1 IP フィルタサービスの構成のタスクマップ

タスク	参照先
IP フィルタが使用するファイルおよびサービスのステータスを表示します。	60 ページの「IP フィルタサービスのデフォルトを表示する方法」
ネットワークトラフィック、NAT 経由でのパケット、アドレスプールのパケットフィルタリング規則セットをカスタマイズします。	61 ページの「IP フィルタ構成ファイルの作成方法」
IP フィルタサービスを有効、リフレッシュ、または無効にします。	63 ページの「IP フィルタを有効にし、リフレッシュする方法」
フラグメントで到着するパケットのデフォルトの設定を変更します。	63 ページの「パケット再構築を無効にする方法」
システム上のゾーン間のトラフィックをフィルタリングします。	64 ページの「ループバックフィルタリングを有効にする方法」
IP フィルタの使用を停止します。	65 ページの「パケットフィルタリングを無効にする方法」

▼ IP フィルタサービスのデフォルトを表示する方法

始める前に `ipfstat` コマンドを実行するには、IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IP フィルタサービスの構成ファイル名と場所を表示します。**

```
$ svccfg -s ipfilter:default listprop   | grep file
config/ipf6_config_file                       astring     /etc/ipf/ipf6.conf
config/ipnat_config_file                      astring     /etc/ipf/ipnat.conf
config/ippool_config_file                     astring     /etc/ipf/ippool.conf
firewall_config_default/custom_policy_file    astring     none
```

最初の 3 つのファイルプロパティーには、デフォルトのファイルの場所があります。これらのファイルは作成するまで存在しません。構成ファイルの場所を変更する場合は、そのファイルのプロパティー値を変更する必要があります。手順については、61 ページの「IP フィルタ構成ファイルの作成方法」を参照してください。

独自のパケットフィルタリング規則をカスタマイズする場合は、4 番目のファイルのプロパティーを変更します。ステップ 1のステップ 2 と61 ページの「IP フィルタ構成ファイルの作成方法」を参照してください。

2. **IP フィルタサービスが有効になっているかどうかを判定します。**

   - 手動でネットワーク接続されたシステムでは、IP フィルタはデフォルトで有効にされません。

```
$ svcs -x ipfilter:default
svc:/network/ipfilter:default (IP Filter)
 State: disabled since Mon Sep 10 10:10:50 2012
Reason: Disabled by an administrator.
   See: http://oracle.com/msg/SMF-8000-05
   See: ipfilter(5)
Impact: This service is not running.
```

   - IPv4 ネットワーク上に自動的にネットワーク接続されたシステムでは、次のコマンドを実行して、IP フィルタポリシーを表示します。

```
# ipfstat -io
```

     - ポリシーを作成したファイルを表示するには、`/etc/nwam/loc/NoNet/ipf.conf` を参照します。このファイルは表示専用です。

- ポリシーを変更するには、61 ページの「IP フィルタ構成ファイルの作成方法」を参照してください。

注記 - IPv6 ネットワーク上の IP フィルタポリシーを表示するには、`ipfstat -6io` のように、`-6` オプションを追加します。詳細については、`ipfstat`(1M) のマニュアルページを参照してください。

▼ IP フィルタ構成ファイルの作成方法

自動的に構成されたネットワーク構成の IP フィルタポリシーを変更するか、または手動で構成されたネットワークで IP フィルタを使用するには、構成ファイルを作成し、これらのファイルについてサービスに通知し、サービスを有効にします。

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IP フィルタサービスのポリシーファイルの場所を指定します。**

   このファイルには、パケットフィルタリング規則セットが含まれます。

   a. **まず、ポリシーファイルを `custom` に設定します。**

   ```
   # svccfg -s ipfilter:default setprop firewall_config_default/policy = astring: "custom"
   ```

   b. **次に、場所を指定します。**

   たとえば、`/etc/ipf/myorg.ipf.conf` をパケットフィルタリングのルールセットの場所にします。

   ```
   # svccfg -s ipfilter:default \
   setprop firewall_config_default/custom_policy_file = astring: "/etc/ipf/myorg.ipf.conf"
   ```

2. **パケットフィルタリング規則セットを作成します。**

   パケットのフィルタリングについては、50 ページの「IP フィルタのパケットのフィルタリング機能の使用」を参照してください。構成ファイルの例については、86 ページの「IP フィルタの構成ファイルの例」および `/etc/nwam/loc/NoNet/ipf.conf` ファイルを参照してください。

注記 - 指定したポリシーファイルが空の場合、フィルタリングは行なわれません。空のパケットフィルタリングファイルは、次のような規則セットを含むことと同じです。

```
pass in all
pass out all
```

3. **(オプション) IP フィルタのネットワークアドレス変換 (NAT) 構成ファイルを作成します。**

   NAT 経由のパケットをフィルタリングするには、デフォルトのファイル名 `/etc/ipf/ipnat.conf` で NAT 規則のファイルを作成します。別の名前を使用する場合は、次のように `config/ipnat_config_file` サービスプロパティーの値を変更する必要があります。

   ```
   # svccfg -s ipfilter:default \
   setprop config/ipnat_config_file = astring: "/etc/ipf/myorg.ipnat.conf"
   ```

   NAT については、54 ページの「IP フィルタの NAT 機能の使用」を参照してください。

4. **(オプション) アドレスプール構成ファイルを作成します。**

   アドレスのグループを単一のアドレスプールとして参照するには、デフォルトのファイル名 `/etc/ipf/ippool.conf` でプールのファイルを作成します。別の名前を使用する場合は、次のように `config/ippool_config_file` サービスプロパティーの値を変更する必要があります。

   ```
   # svccfg -s ipfilter:default \
   setprop config/ippool_config_file = astring: "/etc/ipf/myorg.ippool.conf"
   ```

   アドレスプールには、IPv4 アドレスと IPv6 アドレスの任意の組み合わせを含めることができます。アドレスプールの詳細については、55 ページの「IP フィルタのアドレスプール機能の使用」を参照してください。

5. **(オプション) ループバックトラフィックのフィルタリングを有効にします。**

   システムに構成されているゾーン間のトラフィックのフィルタリングを行う場合は、ループバックフィルタリングを有効にする必要があります。64 ページの「ループバックフィルタリングを有効にする方法」を参照してください。ゾーンに適用する規則セットも定義する必要があります。

6. **(オプション) 断片化されたパケットの再構築を無効にします。**

   デフォルトで、フラグメントは、IP フィルタで再構築されます。デフォルトを変更するには、63 ページの「パケット再構築を無効にする方法」を参照してください。

▼ IP フィルタを有効にし、リフレッシュする方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

61 ページの「IP フィルタ構成ファイルの作成方法」を完了しています。

1. **IP フィルタを有効にします。**

   最初に IP フィルタを有効にするには、次のコマンドを入力します。

   ```
   # svcadm enable network/ipfilter
   ```

2. **サービスの実行中に、IP フィルタ構成ファイルを変更したら、サービスをリフレッシュします。**

   ```
   # svcadm refresh network/ipfilter
   ```

注記 - `refresh` コマンドは一時的にファイアウォールを無効にします。ファイアウォールを保持するには、規則を追加するか、新しい構成ファイルを追加します。例と手順については、66 ページの「IP フィルタ規則セットの操作」を参照してください。

▼ パケット再構築を無効にする方法

デフォルトで、フラグメントは、IP フィルタで再構築されます。この再構築を無効にするには、ポリシーファイルの先頭に規則を挿入します。

始める前に IP Filter Management 権利プロファイルと `solaris.admin.edit/`*path-to-IPFilter-policy-file* 承認が割り当てられている管理者になる必要があります。`root` 役割には、これらの権利がすべて含まれています。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IP フィルタを無効にします。**

   ```
   # svcadm disable network/ipfilter
   ```

2. **IP フィルタポリシーファイルの先頭に次の規則を追加します。**

   ```
   set defrag off;
   ```

   次のように、`pfedit` コマンドを使用します。

```
# pfedit /etc/ipf/myorg.ipf.conf
```

この規則はファイル内のすべての `block` および `pass` 規則より前に置く必要があります。ただし、この行の前にコメントを挿入することはできます。次に例を示します。

```
# Disable fragment reassembly
#
set defrag off;
# Define policy
#
block in all
block out all
other rules
```

3. **IP フィルタを有効にします。**

```
# svcadm enable network/ipfilter
```

4. **パケットが再構築中でないことを確認します。**

```
# ipf -T defrag
defrag  min 0   max 0x1 current 0
```

`current` の値が `0` の場合、フラグメントは再構築中ではありません。`current` が `1` の場合、フラグメントは再構築中です。

▼ ループバックフィルタリングを有効にする方法

始める前に IP Filter Management 権利プロファイルと `solaris.admin.edit/`*path-to-IPFilter-policy-file* 承認が割り当てられている管理者になる必要があります。`root` 役割には、これらの権利がすべて含まれています。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IP フィルタが実行中の場合は、停止させます。**

```
# svcadm disable network/ipfilter
```

2. **IP フィルタポリシーファイルの先頭に次の規則を追加します。**

```
set intercept_loopback true;
```

次のように、`pfedit` コマンドを使用します。

```
# pfedit /etc/ipf/myorg.ipf.conf
```

この行はファイルに定義されているすべての block および pass 規則より前に置く必要があります。ただし、この行の前にコメントを挿入することはできます。次に例を示します。

```
...
#set defrag off;
#
# Enable loopback filtering to filter between zones
#
set intercept_loopback true;
#
# Define policy
#
block in all
block out all
other rules
```

3. **IP フィルタを有効にします。**

```
# svcadm enable network/ipfilter
```

4. **ループバックフィルタリングのステータスを確認するには、次のコマンドを使用します。**

```
# ipf -T ipf_loopback
ipf_loopback    min 0   max 0x1 current 1
#
```

current の値が 0 の場合、ループバックフィルタリングは無効にされています。current が 1 の場合、ループバックフィルタリングは有効にされています。

▼ パケットフィルタリングを無効にする方法

この手順は、カーネルからすべての規則を削除し、サービスを無効にします。この手順を使用する場合、適切な構成ファイルで IP フィルタを有効にし、パケットフィルタリングと NAT を再起動する必要があります。詳細については、63 ページの「IP フィルタを有効にし、リフレッシュする方法」を参照してください。

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **サービスを無効にするには、svcadm コマンドを使用します。**

```
# svcadm disable network/ipfilter
```

サービスをテストまたはデバッグするために、サービスの実行中に規則セットを削除できます。詳細については、66 ページの「IP フィルタ規則セットの操作」を参照してください。

IP フィルタ規則セットの操作

次のような場合、パケットフィルタリングと NAT 規則を変更または非アクティブ化したほうがよいこともあります。

- テスト目的
- 問題の原因が IP フィルタにあると考えられる場合のシステムのトラブルシューティングを行う

次のタスクマップは、IP フィルタのルールセットに関連する手順を一覧表示したものです。

表 5-2 IP フィルタルールセットの操作のタスクマップ

タスク	参照先
アクティブなパケットフィルタリング規則セットを表示します。	67 ページの「アクティブなパケットフィルタリング規則セットを参照する方法」
アクティブでないパケットフィルタリング規則セットを参照する	67 ページの「アクティブでないパケットフィルタリング規則セットを参照する方法」
別のアクティブな規則セットをアクティブにする	68 ページの「別のパケットフィルタリング規則セット、または更新されたパケットフィルタリング規則セットをアクティブにする方法」
規則セットを削除する	69 ページの「パケットフィルタリング規則セットを削除する方法」
規則セットへ規則を追加する	70 ページの「アクティブなパケットフィルタリング規則セットに規則を追加する方法」 71 ページの「アクティブでないパケットフィルタリング規則セットに規則を追加する方法」
アクティブな規則セットとアクティブでない規則セット間を移動する	72 ページの「アクティブなパケットフィルタリング規則セットとアクティブでないパケットフィルタリング規則セットを切り替える方法」
アクティブでない規則セットをカーネルから削除する	73 ページの「カーネルからアクティブでないパケットフィルタリング規則セットを削除する方法」
アクティブな NAT 規則を参照する	74 ページの「IP フィルタでアクティブな NAT規則を表示する方法」
NAT 規則を削除します。	74 ページの「IP フィルタで NA T規則を非アクティブ化する方法」
アクティブな NAT 規則に規則を追加します。	75 ページの「NAT パケットフィルタリング規則に規則を追加する方法」
アクティブなアドレスプールを参照する	76 ページの「アクティブなアドレスプールを参照する方法」
アドレスプールを削除する	77 ページの「アドレスプールを削除する方法」

タスク	参照先
アドレスプールに規則を追加します。	77 ページの「規則をアドレスプールに追加する方法」

IP フィルタのパケットフィルタリング規則セットの管理

IP フィルタにより、アクティブなパケットフィルタリング規則セットとアクティブでないパケットフィルタリング規則セットの両方をカーネルに置くことができます。アクティブな規則セットによって、受信パケットと送信パケットに対して実行するフィルタリングが決まります。アクティブでない規則セットでも規則を格納します。アクティブでない規則セットは、アクティブな規則セットにしない限り、使用されることはありません。アクティブなパケットフィルタリング規則セットとアクティブでないパケットフィルタリング規則セットの両方を管理、参照、変更できます。

注記 - 次の手順に、IPv4 ネットワークの例を示します。IPv6 パケットの場合、`-IP フィルタサービス`のデフォルトを表示する方法のステップ 2 で説明するように、60 ページの「IP フィルタサービスのデフォルトを表示する方法」オプションを使用します。

▼ アクティブなパケットフィルタリング規則セットを参照する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **アクティブなパケットフィルタリング規則セットを表示します。**

  次の例は、カーネルにロードされたアクティブなパケットフィルタリング規則セットからの出力を示しています。

```
# ipfstat -io
empty list for ipfilter(out)
pass in quick on net1 from 192.168.1.0/24 to any
pass in all
block in on net1 from 192.168.1.10/32 to any
```

▼ アクティブでないパケットフィルタリング規則セットを参照する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **アクティブでないパケットフィルタリング規則セットを参照します。**
  次の例は、アクティブでないパケットフィルタリング規則セットからの出力を示しています。

```
# ipfstat -I -io
pass out quick on net1 all
pass in quick on net1 all
```

▼ 別のパケットフィルタリング規則セット、または更新されたパケットフィルタリング規則セットをアクティブにする方法

次のいずれかのタスクを実行する場合には、ここで示す手順を実行します。

- IP フィルタが現在使用しているパケットフィルタリング規則セット以外のパケットフィルタリング規則セットをアクティブにする
- 新規更新されたフィルタリング規則セットを再読み込みする

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **次の手順から 1 つを選択します。**
   - まったく異なる規則セットをアクティブにする場合は、別個のファイルに新規規則セットを作成します。
   - 構成ファイル内の現在の規則セットを更新します。
2. **現在の規則セットを削除し、新しい規則セットをロードします。**

```
# ipf -Fa -f filename
```

filename 内の規則によって、アクティブな規則セットが置き換えられます。

注記 - 更新した規則セットをロードするために `ipf -D` や `svcadm restart` などのコマンドを使わないでください。これらのコマンドは、新しい規則セットをロードする前にファイアウォールを無効にするため、ネットワークが危険にさらされます。

例 5-1 別のパケットフィルタリング規則セットのアクティブ化

次の例は、パケットフィルタリング規則セットを別の規則セットに置き換える方法を示しています。

```
# ipfstat -io
empty list for ipfilter(out)
```

```
pass in quick on net0 all
# ipf -Fa -f /etc/ipf/ipfnew.conf
# ipfstat -io
empty list for ipfilter(out)
block in log quick from 10.0.0.0/8 to any
```

例 5-2 更新したパケットフィルタリング規則セットの再読み込み

次の例は、現在アクティブでこれから更新するパケットフィルタリング規則セットを再読み込みする方法を示しています。

オプションとして、アクティブなルールセットを一覧表示します。

```
# ipfstat -io
empty list for ipfilter (out)
block in log quick from 10.0.0.0/8 to any
```

次に、`/etc/ipf/myorg.ipf.conf` 構成ファイルを編集してサービスをリフレッシュし、アクティブなルールセットを再度一覧表示します。

```
# svcadm refresh network/ipfilter
# ipfstat -io
empty list for ipfilter (out)
block in log quick from 10.0.0.0/8 to any
block in quick on net1 from 192.168.0.0/12 to any
```

▼ パケットフィルタリング規則セットを削除する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **規則セットを削除します。**

```
# ipf -F [a|i|o]
```

`-a`	すべてのフィルタリング規則を規則セットから削除します。
`-i`	受信パケットのフィルタリング規則を削除します。
`-o`	送信パケットのフィルタリング規則を削除します。

例 5-3 パケットフィルタリング規則セットの削除

次の例は、すべてのフィルタリング規則をアクティブなフィルタリング規則セットから削除する方法を示しています。

```
# ipfstat -io
block out log on net0 all
block in log quick from 10.0.0.0/8 to any
# ipf -Fa
# ipfstat -io
empty list for ipfilter(out)
empty list for ipfilter(in)
```

▼ アクティブなパケットフィルタリング規則セットに規則を追加する方法

既存のルールセットにルールを追加すると、テストまたはトラブルシューティングの際に役立つことがあります。規則を追加した場合も IP フィルタサービスは有効なままになります。ただし、サービスがリフレッシュされるか、再起動されるか、有効にされると、規則は IP フィルタサービスのプロパティーであるファイル内に存在しない限り、失われます。

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **次のいずれかの方法で規則をアクティブな規則セットに追加します。**

  - `ipf -f -` コマンドを使用して、コマンド行で、ルールセットにルールを追加します。

    ```
    # echo "block in on net1 proto tcp from 10.1.1.1/32 to any" | ipf -f -
    ```

    サービスのリフレッシュ、再起動、または有効化時に、これらの追加された規則は IP フィルタ構成に含まれません。

  - 次のコマンドを実行します。

    1. 適当なファイルに規則セットを作成します。
    2. 作成しておいた規則をアクティブな規則セットに追加します。

       ```
       # ipf -f filename
       ```

       filename の規則がアクティブな規則セットの最後に追加されます。IP フィルタは「最後に一致した規則を採用する」アルゴリズムを使用するため、`quick` キーワードを使用しないかぎり、追加した規則によってフィルタリングの優先順位が決まります。パケットが `quick` キーワードを含む規則に一致する場合は、その規則に対する処理が実行され、それ以降の規則はチェックされません。

filename が、IP フィルタ構成ファイルのいずれかのプロパティーの値である場合、サービスの有効化、再起動、またはリフレッシュ時に規則が再読み込みされます。そうでない場合は、追加された規則は一時規則セットを提供します。

例 5-4 アクティブなパケットフィルタリング規則セットへの規則の追加

次の例は、コマンド行から、アクティブなパケットフィルタリング規則セットに規則を追加する方法を示しています。

```
# ipfstat -io
empty list for ipfilter(out)
block in log quick from 10.0.0.0/8 to any
# echo "block in on net1 proto tcp from 10.1.1.1/32 to any" | ipf -f -
# ipfstat -io
empty list for ipfilter(out)
block in log quick from 10.0.0.0/8 to any
block in on net1 proto tcp from 10.1.1.1/32 to any
```

▼ アクティブでないパケットフィルタリング規則セットに規則を追加する方法

カーネル内でアクティブでないルールセットを作成すると、テストまたはトラブルシューティングの際に役立つことがあります。規則セットは、IP フィルタサービスを停止せずに、アクティブな規則セットと切り替えることができます。ただし、サービスをリフレッシュするか、再起動するか、有効にする場合、アクティブでない規則セットを追加する必要があります。

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **適当なファイルに規則セットを作成します。**

2. **作成しておいた規則をアクティブでない規則セットに追加します。**

   # ipf -I -f *filename*

   filename の規則がアクティブでない規則セットの最後に追加されます。IP フィルタは「最後に一致した規則を採用する」アルゴリズムを使用するため、`quick` キーワードを使用しないかぎり、追加した規則によってフィルタリングの優先順位が決まります。パケットが `quick` キーワードを含む規則に一致する場合は、その規則に対する処理が実行され、それ以降の規則はチェックされません。

例 5-5 アクティブでない規則セットへの規則の追加

次の例は、ファイルからアクティブでない規則セットに規則を追加する方法を示しています。

```
# ipfstat -I -io
pass out quick on net1 all
pass in quick on net1 all
# ipf -I -f /etc/ipf/ipftrial.conf
# ipfstat -I -io
pass out quick on net1 all
pass in quick on net1 all
block in log quick from 10.0.0.0/8 to any
```

▼ アクティブなパケットフィルタリング規則セットとアクティブでないパケットフィルタリング規則セットを切り替える方法

カーネル内で別のルールセットに切り替えると、テストまたはトラブルシューティングの際に役立つことがあります。規則セットは、IP フィルタサービスを停止せずに、アクティブにすることができます。

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **アクティブな規則セットとアクティブでない規則セットを切り替えます。**

```
# ipf -s
```

このコマンドを使用すると、カーネル内のアクティブな規則セットとアクティブでない規則セットを切り替えることができます。なお、アクティブでない規則セットが空の場合は、パケットフィルタリングは行われません。

注記 - IP フィルタサービスがリフレッシュされるか、再起動されるか、有効にされると、IP フィルタサービスのプロパティーであるファイル内にある規則は復元されます。アクティブでない規則セットは復元されません。

例 5-6 アクティブなパケットフィルタリング規則セットとアクティブでないパケットフィルタリング規則セットの切り替え

次の例は、`ipf -s` コマンドの使用によって、どのようにアクティブでない規則セットがアクティブな規則セットになり、アクティブな規則セットがアクティブでない規則セットになるのかを示しています。

- `ipf -s` コマンドを実行する前に、`ipfstat -I -io` コマンドからの出力でアクティブでない規則セット内の規則が示されます。`ipfstat -io` コマンドからの出力は、アクティブな規則セットの規則を示します。

```
# ipfstat -io
empty list for ipfilter(out)
block in log quick from 10.0.0.0/8 to any
block in on net1 proto tcp from 10.1.1.1/32 to any
# ipfstat -I -io
pass out quick on net1 all
pass in quick on net1 all
block in log quick from 10.0.0.0/8 to any
```

- `ipf -s` コマンドの実行後、`ipfstat -I -io` コマンドおよび `ipfstat -io` コマンドからの出力によって、2 つの規則セットの内容が切り替えられたことが示されます。

```
# ipf -s
Set 1 now inactive
# ipfstat -io
pass out quick on net1 all
pass in quick on net1 all
block in log quick from 10.0.0.0/8 to any
# ipfstat -I -io
empty list for inactive ipfilter(out)
block in log quick from 10.0.0.0/8 to any
block in on net1 proto tcp from 10.1.1.1/32 to any
```

▼ カーネルからアクティブでないパケットフィルタリング規則セットを削除する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **全削除コマンドで、アクティブでない規則セットを指定します。**

```
# ipf -I -Fa
```

注記 - 続けて `ipf -s` を実行すると、空のアクティブでない規則セットがアクティブな規則セットになります。アクティブな規則セットが空の場合は、フィルタリングが行われ*ません*。

例 5-7 カーネルからのアクティブでないパケットフィルタリング規則セットの削除

次の例は、すべての規則が削除されるように、アクティブでないパケットフィルタリング規則セットを消去する方法を示しています。

```
# ipfstat -I -io
empty list for inactive ipfilter(out)
block in log quick from 10.0.0.0/8 to any
block in on net1 proto tcp from 10.1.1.1/32 to any
# ipf -I -Fa
# ipfstat -I -io
empty list for inactive ipfilter(out)
empty list for inactive ipfilter(in)
```

IP フィルタ用 NAT 規則の管理

次の手順で IP フィルタの NAT 規則を管理、表示、および変更します。

▼ IP フィルタでアクティブな NAT規則を表示する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **アクティブな NAT 規則を参照します。**

  次の例は、アクティブな NAT 規則セットからの出力を示しています。

```
# ipnat -l
List of active MAP/Redirect filters:
map net0 192.168.1.0/24 -> 20.20.20.1/32

List of active sessions:
```

▼ IP フィルタで NA T規則を非アクティブ化する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **カーネルから NAT 規則を削除します。**

  ```
  # ipnat -FC
  ```

  -C オプションは、現在の NAT 規則リストのすべてのエントリを削除します。-F オプションは、現在アクティブな NAT マッピングを示す現在の NAT 変換テーブルのすべてのアクティブなエントリを削除します。

例 5-8 NAT 規則の削除

次の例は、現在の NAT 規則のエントリを削除する方法を示してます。

```
# ipnat -l
List of active MAP/Redirect filters:
map net0 192.168.1.0/24 -> 20.20.20.1/32

List of active sessions:
# ipnat -C
1 entries flushed from NAT list
# ipnat -l
List of active MAP/Redirect filters:

List of active sessions:
```

▼ NAT パケットフィルタリング規則に規則を追加する方法

既存のルールセットにルールを追加すると、テストまたはトラブルシューティングの際に役立つことがあります。規則を追加した場合も IP フィルタサービスは有効なままになります。ただし、サービスがリフレッシュされるか、再起動されるか、有効にされると、NAT 規則は IP フィルタサービスのプロパティーであるファイル内に存在しない限り、失われます。

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **次のいずれかの方法で規則をアクティブな規則セットに追加します。**

  - ipnat -f - コマンドを使用して、コマンド行で、NAT 規則セットに規則を追加します。

    ```
    # echo "map net0 192.168.1.0/24 -> 20.20.20.1/32" | ipnat -f -
    ```

    サービスのリフレッシュ、再起動、または有効化時に、これらの追加された規則は IP フィルタ構成に含まれません。
  - 次のコマンドを実行します。

1. 適当なファイルに追加の NAT 規則を作成します。
2. 作成しておいた規則をアクティブな NAT 規則に追加します。

   ```
   # ipnat -f filename
   ```

   filename の規則がアクティブな NAT 規則の最後に追加されます。

   filename が、IP フィルタ構成ファイルのいずれかのプロパティーの値である場合、サービスの有効化、再起動、またはリフレッシュ時に規則が再読み込みされます。そうでない場合は、追加された規則は一時規則セットを提供します。

例 5-9 NAT 規則セットへの規則の追加

次の例は、コマンド行から、NAT 規則セットに規則を追加する方法を示しています。

```
# ipnat -l
List of active MAP/Redirect filters:

List of active sessions:
# echo "map net0 192.168.1.0/24 -> 20.20.20.1/32" | ipnat -f -
# ipnat -l
List of active MAP/Redirect filters:
map net0 192.168.1.0/24 -> 20.20.20.1/32

List of active sessions:
```

IP フィルタのアドレスプールの管理

次の手順でアドレスプールを管理、表示、および変更します。

▼ アクティブなアドレスプールを参照する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **アクティブなアドレスプールを参照します。**

  次の例は、アクティブなアドレスプールの内容を参照する方法を示しています。

  ```
  # ippool -l
  table role = ipf type = tree number = 13
          { 10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24; };
  ```

▼ アドレスプールを削除する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **現在のアドレスプールのエントリを削除します。**

```
# ippool -F
```

例 5-10 アドレスプールの削除

次の例は、アドレスプールを削除する方法を示しています。

```
# ippool -l
table role = ipf type = tree number = 13
        { 10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24; };
# ippool -F
1 object flushed
# ippool -l
```

▼ 規則をアドレスプールに追加する方法

既存のルールセットにルールを追加すると、テストまたはトラブルシューティングの際に役立つことがあります。規則を追加した場合も IP フィルタサービスは有効なままになります。ただし、サービスがリフレッシュされるか、再起動されるか、有効にされると、アドレスプール規則は IP フィルタサービスのプロパティーであるファイル内に存在しない限り、失われます。

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **次のいずれかの方法で規則をアクティブな規則セットに追加します。**

   - `ippool -f -` コマンドを使用して、コマンド行でルールセットにルールを追加します。

     ```
     # echo "table role = ipf type = tree number = 13
     {10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24};" | ippool -f -
     ```

     サービスのリフレッシュ、再起動、または有効化時に、これらの追加された規則は IP フィルタ構成に含まれません。
   - 次のコマンドを実行します。

1. 適当なファイルに追加のアドレスプールを作成します。
2. 作成しておいた規則をアクティブなアドレスプールに追加します。

   ```
   # ippool -f filename
   ```

   filename の規則がアクティブなアドレスプールの最後に追加されます。

2. **規則に、元の規則セットにないプールが含まれる場合、次の手順を実行します。**

   a. **新しいパケットフィルタリング規則にプールを追加します。**

   b. **現在の規則セットに新しいパケットフィルタリング規則を追加します。**

   70 ページの「アクティブなパケットフィルタリング規則セットに規則を追加する方法」の指示に従います。

注記 - IP フィルタサービスをリフレッシュまたは再起動しないでください。追加したアドレスプール規則が失われます。

例 5-11 アドレスプールへの規則の追加

次の例は、コマンド行から、アドレスプール規則セットにアドレスプールを追加する方法を示しています。

```
# ippool -l
table role = ipf type = tree number = 13
        { 10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24; };
# echo "table role = ipf type = tree number = 100
 {10.0.0.0/32, 172.16.1.2/32, 192.168.1.0/24};" | ippool -f -
# ippool -l
table role = ipf type = tree number = 100
        { 10.0.0.0/32, 172.16.1.2/32, 192.168.1.0/24; };
table role = ipf type = tree number = 13
        { 10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24; };
```

IP フィルタの統計および情報の表示

表 5-3 IP フィルタの統計および情報の表示のタスクマップ

タスク	参照先
状態テーブルを表示します。	79 ページの「IP フィルタの状態テーブルを参照する方法」

タスク	参照先
パケットの状態に関する統計情報を表示します。	80 ページの「IP フィルタの状態統計を参照する方法」
IP フィルタのチューニング可能パラメータを一覧表示します。	81 ページの「IP フィルタのチューニング可能パラメータを表示する方法」
NAT 統計を表示します。	81 ページの「IP フィルタの NAT 統計を参照する方法」
アドレスプール統計の参照	82 ページの「IP フィルタのアドレスプール統計情報を表示する方法」

▼ IP フィルタの状態テーブルを参照する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **状態テーブルを参照します。**

```
# ipfstat
```

注記 - -t オプションを使用すると、状態テーブルを UNIX top ユーティリティー形式で表示できます。

例 5-12 IP フィルタの状態テーブルの参照

次の例に、状態テーブルの出力を示します。

```
# ipfstat
bad packets:            in 0    out 0
 IPv6 packets:          in 56286 out 63298
 input packets:         blocked 160 passed 11 nomatch 1 counted 0 short 0
output packets:         blocked 0 passed 13681 nomatch 6844 counted 0 short 0
 input packets logged:  blocked 0 passed 0
output packets logged:  blocked 0 passed 0
 packets logged:        input 0 output 0
 log failures:          input 0 output 0
fragment state(in):     kept 0  lost 0  not fragmented 0
fragment reassembly(in):bad v6 hdr 0     bad v6 ehdr 0  failed reassembly 0
fragment state(out):    kept 0  lost 0  not fragmented 0
packet state(in):       kept 0  lost 0
packet state(out):      kept 0  lost 0
ICMP replies:   0       TCP RSTs sent:  0
Invalid source(in):     0
Result cache hits(in):  152     (out):  6837
```

```
IN Pullups succeeded:   0       failed: 0
OUT Pullups succeeded:  0       failed: 0
Fastroute successes:    0       failures:       0
TCP cksum fails(in):    0       (out):  0
IPF Ticks:      14341469
Packet log flags set: (0)
        none
```

▼ IP フィルタの状態統計を参照する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **状態統計を参照します。**

```
# ipfstat -s
```

例 5-13 IP フィルタの状態統計の参照

次の例に、状態統計情報の出力を示します。

```
# ipfstat -s
IP states added:
        0 TCP
        0 UDP
        0 ICMP
        0 hits
        0 misses
        0 maximum
        0 no memory
        0 max bucket
        0 active
        0 expired
        0 closed
State logging enabled

State table bucket statistics:
        0 in use
        0.00% bucket usage
        0 minimal length
        0 maximal length
        0.000 average length
```

▼ IP フィルタのチューニング可能パラメータを表示する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **IP フィルタのカーネルチューニング可能パラメータを表示します。**

  次の出力は、切り詰められています。

```
# ipf -T list
fr_flags min 0 max 0xffffffff current 0
fr_active min 0 max 0 current 0
...
ipstate_logging min 0 max 0x1 current 1
...
fr_authq_ttl min 0x1 max 0x7fffffff current sz = 0
fr_enable_rcache min 0 max 0x1 current 0
```

▼ IP フィルタの NAT 統計を参照する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **NAT 統計を表示します。**

```
# ipnat -s
```

例 5-14 IP フィルタの NAT 統計の参照

次の例に、NAT 統計情報の例を示します。

```
# ipnat -s
mapped  in      0       out     0
added   0       expired 0
no memory       0       bad nat 0
inuse   0
rules   1
wilds   0
```

▼ IP フィルタのアドレスプール統計情報を表示する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **アドレスプール統計の参照**

```
# ippool -s
```

例 5-15 IP フィルタのアドレスプール統計の参照

次の例に、アドレスプール統計情報を示します。

```
# ippool -s
Pools:  3
Hash Tables:    0
Nodes:  0
```

IP フィルタ用ログファイルの操作

表 5-4 IP フィルタログファイルの操作のタスクマップ

タスク	参照先
IP フィルタログファイルを別個に作成する	82 ページの「IP フィルタのログファイルを設定する方法」
状態、NAT、および通常のログファイルを表示します。	84 ページの「IP フィルタのログファイルを参照する方法」
パケットログバッファーの消去	85 ページの「パケットログバッファーをフラッシュする方法」
あとで参照できるようにロギングされたパケットをファイルに保存する	85 ページの「ロギングされたパケットをファイルに保存する方法」

▼ IP フィルタのログファイルを設定する方法

デフォルトでは、IP フィルタのすべてのログ情報は `syslog` によって記録されます。`syslog` ログファイルに記録される可能性のあるほかのデータとは別に、IP フィルタのトラフィック情報を記録するログファイルを作成することをお勧めします。

始める前に root 役割になる必要があります。

1. **どの system-log サービスインスタンスが有効であるかを判定します。**

```
% svcs system-log
STATE          STIME    FMRI
disabled       13:11:55 svc:/system/system-log:rsyslog
online         13:13:27 svc:/system/system-log:default
```

注記 - rsyslog サービスインスタンスがオンラインである場合は、rsyslog.conf ファイルを変更します。

2. **/etc/syslog.conf を編集して、次の 2 行を追加します。**

```
# Save IP Filter log output to its own file
local0.debug             /var/log/log-name
```

注記 - 入力では、local0.debug を /var/log/*log-name* から区切るために、Space キーではなく Tab キーを使用します。詳細については、syslog.conf(4) および syslogd(1M) のマニュアルページを参照してください。

3. **新規ログファイルを作成します。**

```
# touch /var/log/log-name
```

4. **system-log サービスの構成情報をリフレッシュします。**

```
# svcadm refresh system-log:default
```

注記 - rsyslog サービスが有効になっている場合は、system-log:rsyslog サービスインスタンスをリフレッシュします。

例 5-16 IP フィルタログの作成

次の例は、ipmon.log を作成して IP フィルタ情報をアーカイブする方法を示しています。

syslog.conf を編集します。

```
pfedit /etc/syslog.conf
## Save IP Filter log output to its own file
local0.debug<Tab>/var/log/ipmon.log
```

次に、コマンド行でファイルを作成してサービスを再起動します。

```
# touch /var/log/ipmon.log
# svcadm restart system-log
```

▼ IP フィルタのログファイルを参照する方法

始める前に 82 ページの「IP フィルタのログファイルを設定する方法」を完了しています。

IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **状態、NAT、または通常のログファイルを参照します。**
  ログファイルを参照するには、適切なオプションと共に次のコマンドを入力してください。

  ```
  # ipmon -o [S|N|I] filename
  ```

  S　状態ログファイルを表示します。

  N　NAT ログファイルを表示します。

  I　通常の IP ログファイルを表示します。

  - **状態、NAT、および通常のログファイルをすべて表示するには、すべてのオプションを使用します。**

    ```
    # ipmon -o SNI filename
    ```

  - **ipmon デーモンの停止後、ipmon コマンドを使用して状態、NAT、および IP フィルタのログファイルを表示できます。**

    ```
    # pkill ipmon
    # ipmon -a filename
    ```

注記 - ipmon デーモンが実行中の場合は、ipmon -a 構文を使用しないでください。通常、このデーモンは、システムのブート時に自動的に起動されます。ipmon -a コマンドを実行して、ipmon の別のコピーを開きます。すると、両方のコピーが同じログ情報を読み取りますが、一方だけが特定のログメッセージを取得します。

ログファイルの表示の詳細については、ipmon(1M) のマニュアルページを参照してください。

例 5-17　IP フィルタのログファイルの参照

次の例は、/var/ipmon.log からの出力を示しています。

```
# ipmon -o SNI /var/ipmon.log
02/09/2012 15:27:20.606626 net0 @0:1 p 129.146.157.149 ->
129.146.157.145 PR icmp len 20 84 icmp echo/0 IN
```

または

```
# pkill ipmon
# ipmon -aD /var/ipmon.log
02/09/2012 15:27:20.606626 net0 @0:1 p 129.146.157.149 ->
129.146.157.145 PR icmp len 20 84 icmp echo/0 IN
```

▼ パケットログバッファーをフラッシュする方法

この手順では、バッファーをクリアし、画面に出力を表示します。

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **パケットログバッファーの消去**

  ```
  # ipmon -F
  ```

例 5-18 パケットログバッファーのフラッシュ

次の例は、ログファイルが削除されたときの出力を示しています。この例のように、ログファイルが空の場合でもシステムがレポートを出力します。

```
# ipmon -F
0 bytes flushed from log buffer
0 bytes flushed from log buffer
0 bytes flushed from log buffer
```

▼ ロギングされたパケットをファイルに保存する方法

トラブルシューティング時、または手動でトラフィックを監査する場合にパケットをファイルに保存できます。

始める前に `root` 役割になる必要があります。

- **ロギングされたパケットをファイルへ保存します。**

  ```
  # cat /dev/ipl > filename
  ```

  `Control-C` を入力して、コマンド行のプロンプトに戻って、このプロシージャーを中断するまで、パケットは *filename* ファイルに継続的にロギングされます。

例 5-19 ファイルへのロギングされたパケットの保存

次の例は、ロギングされたパケットがファイルに保存されたときの結果を表します。

```
# cat /dev/ipl > /tmp/logfile
^C#

# ipmon -f /tmp/logfile
02/09/2012 15:30:28.708294 net0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 52 -S IN
02/09/2012 15:30:28.708708 net0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 40 -A IN
02/09/2012 15:30:28.792611 net0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 70 -AP IN
02/09/2012 15:30:28.872000 net0 @0:1 p 129.146.157.149,33923 ->
 129.146.157.145,23 PR tcp len 20 40 -A IN
02/09/2012 15:30:28.872142 net0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 43 -AP IN
02/09/2012 15:30:28.872808 net0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 40 -A IN
02/09/2012 15:30:28.872951 net0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 47 -AP IN
02/09/2012 15:30:28.926792 net0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 40 -A IN
.
.
(output truncated)
```

IP フィルタの構成ファイルの例

次の例に、単一のホスト、サーバー、ルーターに適用するパケットフィルタリング規則を示します。構成ファイルは、次のような標準的な UNIX 構文規則に従っています。

- シャープ記号 (#) は、コメントを含む行を示します。
- 規則とコメントは、同一の行に共存できます。
- 規則を読みやすくするために、不要な空白を使用できます。
- 複数行に渡って規則を記述できます。行の最後のバックスラッシュ (¥) は、ルールが次の行に続いていることを示します。

構文についての詳細については、51 ページの「パケットのフィルタリング規則の構成」を参照してください。

例 5-20 IP フィルタのホスト構成

この例は、`net0` ネットワークインタフェースを備えるホストシステム上の構成を示しています。

```
# pass and log everything by default
pass in log on net0 all
pass out log on net0 all

# block, but don't log, incoming packets from other reserved addresses
block in quick on net0 from 10.0.0.0/8 to any
block in quick on net0 from 172.16.0.0/12 to any

# block and log untrusted internal IPs. 0/32 is notation that replaces
# address of the machine running IP Filter.
block in log quick from 192.168.1.15 to <thishost>
block in log quick from 192.168.1.43 to <thishost>

# block and log X11 (port 6000) and remote procedure call
# and portmapper (port 111) attempts
block in log quick on net0 proto tcp from any to net0/32 port = 6000 keep state
block in log quick on net0 proto tcp/udp from any to net0/32 port = 111 keep state
```

このルールセットは、すべてのデータが `net0` インタフェースを出入りできる 2 つの制限なしのルールで開始します。2 番目のルールセットは、プライベートアドレス空間 `10.0.0.0` および `172.16.0.0` からの受信パケットがファイアウォールの中に入るのをブロックします。次のルールセットは、ホストシステムからの特定の内部アドレスをブロックします。そして、最後の規則セットは、ポート 6000 およびポート 111 から受信されるパケットをブロックします。

例 5-21 IP フィルタのサーバー構成

この例は、Web サーバーとして機能するホストシステムの構成を示しています。このシステムは `net0` ネットワークインタフェースを備えています。

```
# web server with an net0 interface
# block and log everything by default;
# then allow specific services
# group 100 - inbound rules
# group 200 - outbound rules
# (0/32) resolves to our IP address)
*** FTP proxy ***

# block short packets which are packets
# fragmented too short to be real.
block in log quick all with short

# block and log inbound and outbound by default,
# group by destination
block in log on net0 from any to any head 100
block out log on net0 from any to any head 200


# web rules that get hit most often
pass in quick on net0 proto tcp from any \
to net0/32 port = http flags S keep state group 100
```

```
pass in quick on net0 proto tcp from any \
to net0/32 port = https flags S keep state group 100


# inbound traffic - ssh, auth
pass in quick on net0 proto tcp from any \
to net0/32 port = 22 flags S keep state group 100
pass in log quick on net0 proto tcp from any \
to net0/32 port = 113 flags S keep state group 100
pass in log quick on net0 proto tcp from any port = 113 \
to net0/32 flags S keep state group 100

# outbound traffic - DNS, auth, NTP, ssh, WWW, smtp
pass out quick on net0 proto tcp/udp from net0/32 \
to any port = domain flags S keep state group 200
pass in quick on net0 proto udp from any \
port = domain to net0/32 group 100

pass out quick on net0 proto tcp from net0/32 \
to any port = 113 flags S keep state group 200
pass out quick on net0 proto tcp from net0/32 port = 113 \
to any flags S keep state group 200

pass out quick on net0 proto udp from net0/32 to any \
port = ntp group 200
pass in quick on net0 proto udp from any \
port = ntp to net0/32 port = ntp group 100

pass out quick on net0 proto tcp from net0/32 \
to any port = ssh flags S keep state group 200

pass out quick on net0 proto tcp from net0/32 \
to any port = http flags S keep state group 200
pass out quick on net0 proto tcp from net0/32 \
to any port = https flags S keep state group 200

pass out quick on net0 proto tcp from net0/32 \
to any port = smtp flags S keep state group 200


# pass icmp packets in and out
pass in quick on net0 proto icmp from any to net0/32  keep state group 100
pass out quick on net0 proto icmp from net0/32 to any keep state group 200

# block and ignore NETBIOS packets
block in quick on net0 proto tcp from any \
to any port = 135 flags S keep state group 100

block in quick on net0 proto tcp from any port = 137 \
to any flags S keep state group 100
block in quick on net0 proto udp from any to any port = 137 group 100
block in quick on net0 proto udp from any port = 137 to any group 100

block in quick on net0 proto tcp from any port = 138 \
```

```
to any flags S keep state group 100
block in quick on net0 proto udp from any port = 138 to any group 100

block in quick on net0 proto tcp from any port = 139 to any flags S keep state
group 100
block in quick on net0 proto udp from any port = 139 to any group 100
```

例 5-22 IP フィルタのルーター構成

この例は内部インタフェース net0、および外部インタフェース net1 を備えるルーターの構成を示しています。

```
# internal interface is net0 at 192.168.1.1
# external interface is net1 IP obtained via DHCP
# block all packets and allow specific services
*** NAT ***
*** POOLS ***

# Short packets which are fragmented too short to be real.
block in log quick all with short

# By default, block and log everything.
block in log on net0 all
block in log on net1 all
block out log on net0 all
block out log on net1 all


# Packets going in/out of network interfaces that are not on the
# loopback interface should not exist.
block in log quick on net0 from 127.0.0.0/8 to any
block in log quick on net0 from any to 127.0.0.0/8
block in log quick on net1 from 127.0.0.0/8 to any
block in log quick on net1 from any to 127.0.0.0/8


# Deny reserved addresses.
block in quick on net1 from 10.0.0.0/8 to any
block in quick on net1 from 172.16.0.0/12 to any
block in log quick on net1 from 192.168.1.0/24 to any
block in quick on net1 from 192.168.0.0/16 to any


# Allow internal traffic
pass in quick on net0 from 192.168.1.0/24 to 192.168.1.0/24
pass out quick on net0 from 192.168.1.0/24 to 192.168.1.0/24


# Allow outgoing DNS requests from our servers on .1, .2, and .3
pass out quick on net1 proto tcp/udp from net1/32 to any port = domain keep state
pass in quick on net0 proto tcp/udp from 192.168.1.2 to any port = domain keep state
pass in quick on net0 proto tcp/udp from 192.168.1.3 to any port = domain keep state
```

```
# Allow NTP from any internal hosts to any external NTP server.
pass in quick on net0 proto udp from 192.168.1.0/24 to any port = 123 keep state
pass out quick on net1 proto udp from any to any port = 123 keep state


# Allow incoming mail
pass in quick on net1 proto tcp from any to net1/32 port = smtp keep state
pass in quick on net1 proto tcp from any to net1/32 port = smtp keep state
pass out quick on net1 proto tcp from 192.168.1.0/24 to any port = smtp keep state


# Allow outgoing connections: SSH, WWW, NNTP, mail, whois
pass in quick on net0 proto tcp from 192.168.1.0/24 to any port = 22 keep state
pass out quick on net1 proto tcp from 192.168.1.0/24 to any port = 22 keep state

pass in quick on net0 proto tcp from 192.168.1.0/24 to any port = 80 keep state
pass out quick on net1 proto tcp from 192.168.1.0/24 to any port = 80 keep state
pass in quick on net0 proto tcp from 192.168.1.0/24 to any port = 443 keep state
pass out quick on net1 proto tcp from 192.168.1.0/24 to any port = 443 keep state

pass in quick on net0 proto tcp from 192.168.1.0/24 to any port = nntp keep state
block in quick on net1 proto tcp from any to any port = nntp keep state
pass out quick on net1 proto tcp from 192.168.1.0/24 to any port = nntp keep state

pass in quick on net0 proto tcp from 192.168.1.0/24 to any port = smtp keep state

pass in quick on net0 proto tcp from 192.168.1.0/24 to any port = whois keep state
pass out quick on net1 proto tcp from any to any port = whois keep state


# Allow ssh from offsite
pass in quick on net1 proto tcp from any to net1/32 port = 22 keep state


# Allow ping out
pass in quick on net0 proto icmp all keep state
pass out quick on net1 proto icmp all keep state


# allow auth out
pass out quick on net1 proto tcp from net1/32 to any port = 113 keep state
pass out quick on net1 proto tcp from net1/32 port = 113 to any keep state


# return rst for incoming auth
block return-rst in quick on net1 proto tcp from any to any port = 113 flags S/SA


# log and return reset for any TCP packets with S/SA
block return-rst in log on net1 proto tcp from any to any flags S/SA

# return ICMP error packets for invalid UDP packets
block return-icmp(net-unr) in proto udp all
```

♦ ♦ ♦ 第 6 章

6 IP セキュリティーアーキテクチャーについて

IP セキュリティーアーキテクチャー (IPsec) は、IPv4 および IPv6 ネットワークで IP パケットを暗号化して保護します。

この章で扱う内容は、次のとおりです。

IPsec をネットワークに実装するには、第7章「IPsec の構成」を参照してください。参照情報については、第12章「IPsec および鍵管理のリファレンス」を参照してください。

IPsec の概要

IPsec は暗号化を使用することで IP パケットのコンテンツを保護し、パケットのコンテンツを認証することで整合性チェックを実行します。IPsec はネットワークレイヤーで実行されるため、ネットワークアプリケーションは IPsec を使用するようにそれ自体を構成することなく IPsec を利用できます。正しく使用すれば、IPsec は、ネットワークトラフィックの保護に有効なツールとなります。

IPsec では次の用語を使用します。

- **セキュリティープロトコル** – IP パケットに適用される保護。認証ヘッダー (AH) は、IP ヘッダーを含む完全なパケットのハッシュである整合性チェックベクトル (ICV) を追加することで IP パケットを保護します。受信者はパケットが変更されていないことを保証されます。暗号化による機密性の確保は行われません。

  カプセル化セキュリティーペイロード (ESP) は IP パケットのペイロードを保護します。パケットのペイロードは機密性を確保するために暗号化が可能で、ICV を使用してデータの整合性を保証できます。
- **セキュリティーアソシエーション (SA)** – 特定の SA と特定のトラフィックフローを一致させるのに使用される、暗号化パラメータ、鍵、IP セキュリティープロトコル、IP アドレス、IP プロトコル、ポート番号などのパラメータ。
- **セキュリティーアソシエーションデータベース (SADB)** – セキュリティーアソシエーションを格納するデータベース。SA は、セキュリティーパラメータインデックス (SPI)、セキュリティープロトコル、および宛先 IP アドレスから参照されます。これらの 3 つの要素は IPsec SA を一意に識別します。IPsec ヘッダー (ESP または AH) を持つ IP パケットをシステムが受け取ると、そのシステムは SADB で一致する SA を検索します。一致する SA が見つかると、IPsec はその SA を使用してパケットを復号化および検証できます。検証に失敗した場合または一致する SA が見つからない場合は、パケットが破棄されます。
- **鍵管理** – 暗号化アルゴリズムによって使用される鍵のセキュアな生成および配布と、それらの格納に使用される SA の生成。
- **セキュリティーポリシーデータベース (SPD)** – IP トラフィックに適用するセキュリティーポリシーを指定するデータベース。SPD は、トラフィックをフィルタリングしてパケットの処理方法を決定します。パケットは、破棄するか、検証済みとして通過させることができます。または、パケットを IPsec で保護する、つまりセキュリティーポリシーを適用することができます。

  アウトバウンドパケットの場合、IPsec ポリシーは IP パケットに IPsec を適用するべきかどうかを決定します。IPsec を適用すると、IP モジュールが SADB で一致する SA を検索し、この SA を使用してポリシーを適用します。

  インバウンドパケットの場合、IPsec ポリシーによって、受け取ったパケットの保護レベルが適切であることが確認されます。ポリシーが特定の IP アドレスからのパケットを IPsec で保護することを要求する場合、システムは保護されていないパケットをすべて破棄します。インバウンドパケットが IPsec によって保護されている場合、IP モジュールが SADB で一致する SA を検索し、その SA をパケットに適用します。

アプリケーションで IPsec を呼び出すと、ソケット単位レベルでも IP パケットにセキュリティーメカニズムが適用されます。ポートのソケットが接続され、そのあとで IPsec ポリシーがそのポートに適用された場合、そのソケットを使用するトラフィックは IPsec によって保護されません。当

然、IPsec ポリシーがポートに適用された*あとに*ポート上で開かれたソケットは、IPsec ポリシーによって保護されます。

IPsec パケットのフロー

図6-1「アウトバウンドパケットプロセスに適用された IPsec」は、アウトバウンドパケットで IPsec が呼び出された場合に、IP パケットがどのように処理されるかを示しています。フロー図は、認証ヘッダー (AH) とカプセル化されたセキュリティーペイロード (ESP) エンティティーがどこでパケットに適用されるかを示しています。以降のセクションでは、これらのエンティティーの適用方法とアルゴリズムの選択方法について説明します。

図6-2「IPsec をインバウンドパケットプロセスに適用」は、IPsec インバウンドプロセスを示しています。

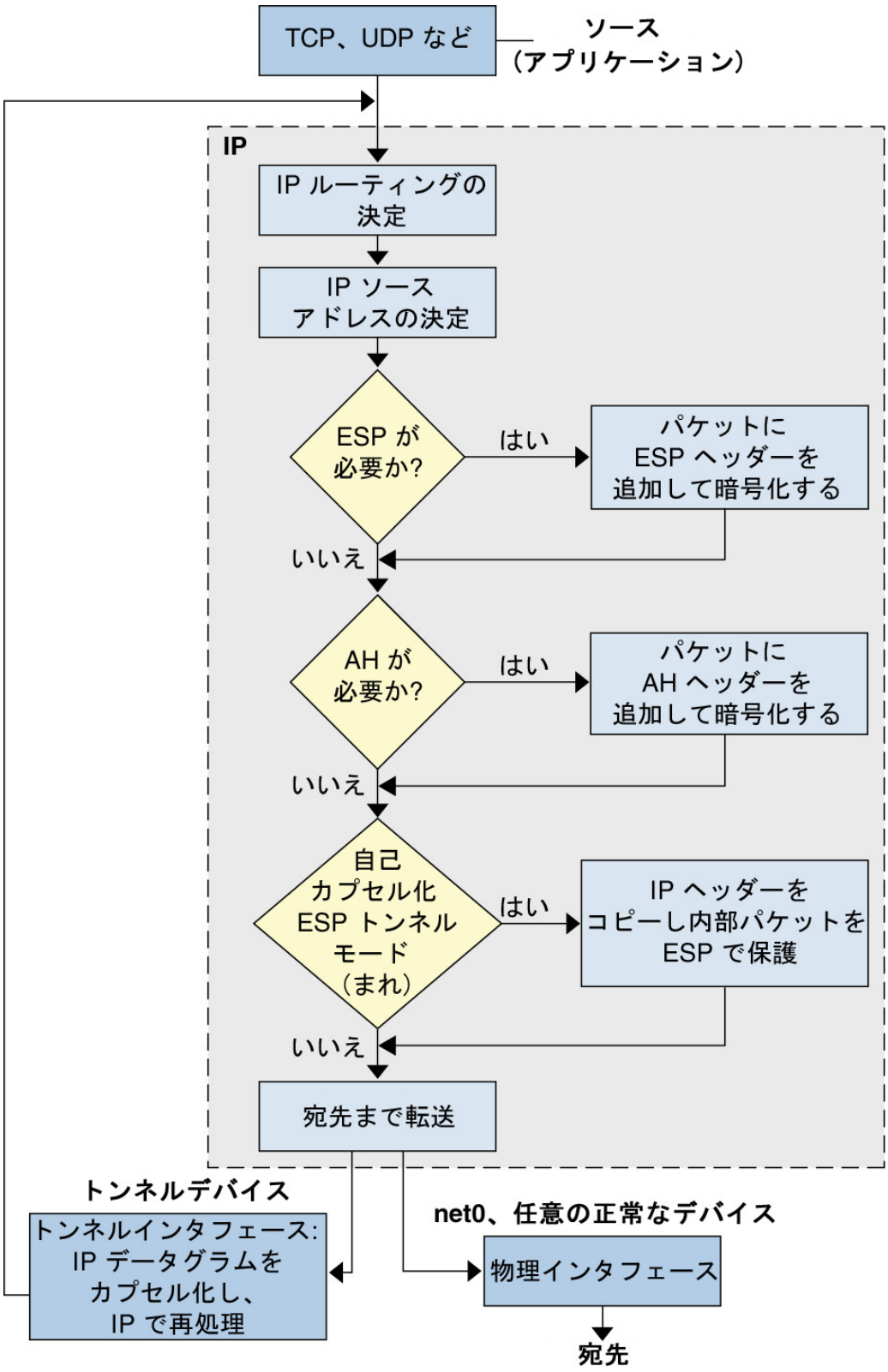
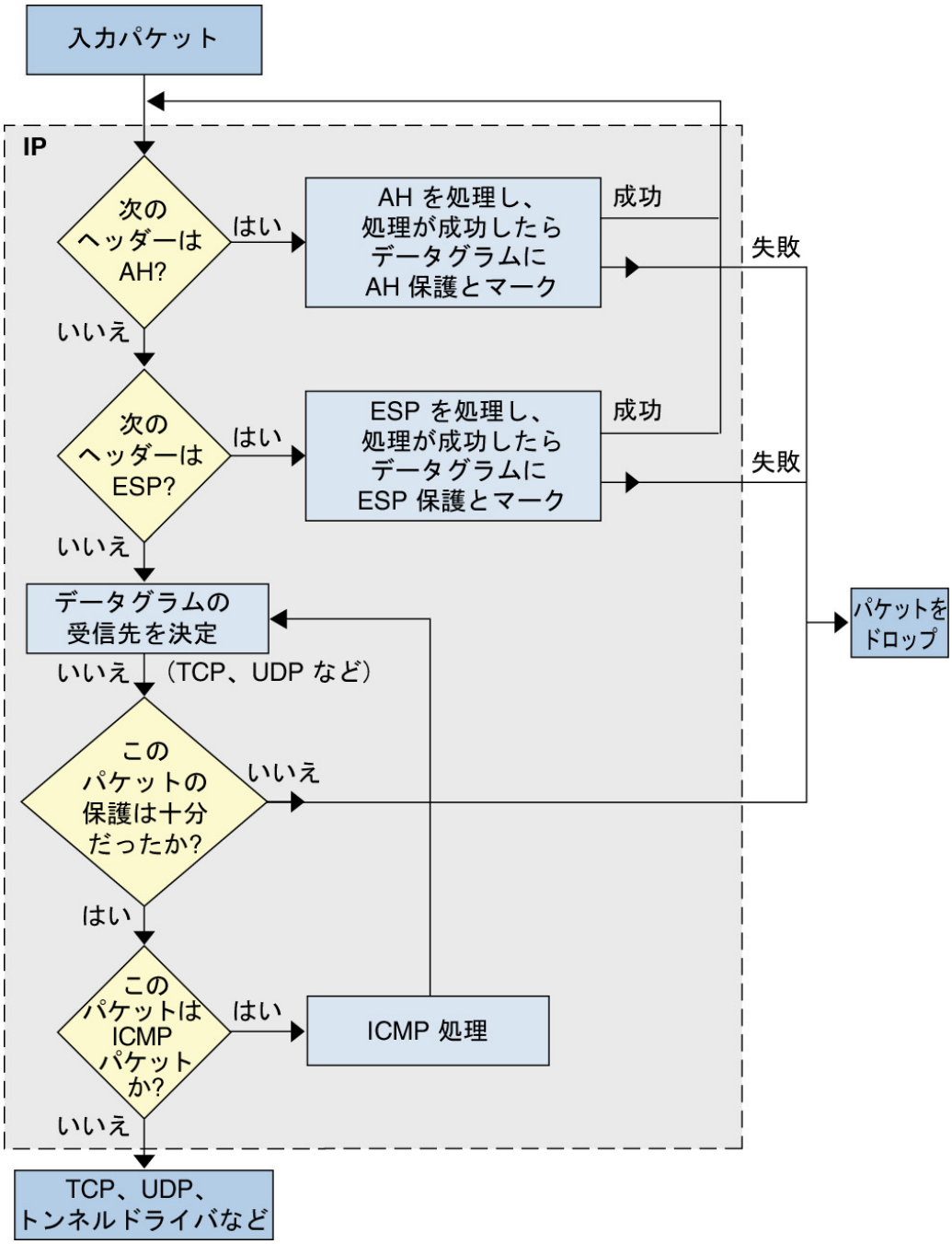

TCP、UDP など
ソース
（アプリケーション）
IP
IP ルーティングの
決定
IP ソース
アドレスの決定
ESP が
必要か?
はい
パケットに
ESP ヘッダーを
追加して暗号化する
いいえ
AH が
必要か?
はい
パケットに
AH ヘッダーを
追加して暗号化する
いいえ
自己
カプセル化
ESP トンネル
モード
（まれ）
はい
IP ヘッダーを
コピーし内部パケットを
ESP で保護
いいえ
宛先まで転送
トンネルデバイス
トンネルインタフェース:
IP データグラムを
カプセル化し、
IP で再処理
net0、任意の正常なデバイス
物理インタフェース
宛先

図 6-2 IPsec をインバウンドパケットプロセスに適用



IPsec セキュリティーアソシエーション

IPsec の *セキュリティーアソシエーション* (SA) は、同様に SA に格納されている IP パラメータに一致する IP パケットに適用するセキュリティープロパティーを定義します。各 SA は単方向です。ほとんどの通信は双方向のため、1 つの接続に 2 つの SA が必要です。

あわせて、次の 3 つの要素が IPsec SA を一意に識別します。

- セキュリティープロトコル (AH または ESP)
- 宛先 IP アドレス
- セキュリティーパラメータインデックス (SPI)

さらなる保護を提供する SA の SPI は、IPsec で保護されたパケットの AH または ESP ヘッダー内で渡されます。AH および ESP によって保護される範囲については、`ipsecah`(7P) および `ipsecesp`(7P) のマニュアルページを参照してください。完全性チェックサム値を使用して、パケットを認証します。認証が失敗すると、パケットがドロップされます。

セキュリティーアソシエーションは、*セキュリティーアソシエーションデータベース* (SADB) に格納されます。ソケットベースの管理インタフェース `PF_KEY` により、特権を持つアプリケーションでそのデータベースをプログラムによって管理できます。たとえば、IKE デーモンと `ipseckey` コマンドは `PF_KEY` ソケットインタフェースを使用します。

IPsec SADB のより完全な説明については、236 ページの「IPsec のセキュリティーアソシエーションデータベース」を参照してください。

SADB の管理方法の詳細については、`pf_key`(7P) および `ipseckey`(1M) のマニュアルページを参照してください。

IPsec セキュリティーアソシエーションの鍵管理

セキュリティーアソシエーション (SA) は、認証および暗号化で使用するキー作成素材を必要とします。この鍵情報の管理は*鍵管理*と呼ばれます。Oracle Solaris では、IKE と手動鍵管理という 2 つの方法で IPsec SA の鍵を管理できます。

IPsec SA を生成するための IKE

インターネット鍵交換 (IKE) プロトコルは、鍵管理を自動的に処理します。Oracle Solaris 11.2 は、IKE プロトコルの IKE バージョン 2 (IKEv2) および IKE バージョン 1 (IKEv1) をサポートしています。

IPsec SA の管理には IKE の使用をお勧めします。これらの鍵管理プロトコルには、次のメリットがあります。

- 単純な構成
- 強力なピア認証を提供
- 高品質でランダムな鍵ソースにより SA を自動的に生成
- 新しい SA の生成に管理者の介入を必要としない

詳細は、139 ページの「IKE の動作」を参照してください。

IKE を構成するには、第9章「IKEv2 の構成」を参照してください。IKEv2 プロトコルをサポートしていないシステムと通信している場合は、第10章「IKEv1 の構成」の指示に従ってください。

IPsec SA を生成するための手動鍵

手動鍵の使用は IKE よりも複雑で、リスクを伴う可能性があります。システムファイル `/etc/inet/secret/ipseckeys` には、暗号化鍵が含まれます。これらの鍵のセキュリティーが侵害された場合、これらを使用して、記録されているネットワークトラフィックが復号化される可能性があります。IKE では鍵が頻繁に変更されるため、そのような脅威にさらされる可能性ははるかに低くなります。`ipseckeys` ファイルまたはそのコマンドインタフェースである `ipseckey` は、IKE をサポートしていないシステムにのみ適しています。

`ipseckey` コマンドには少数の一般オプションしかありませんが、多くのコマンド言語をサポートしています。マニュアルキー操作に固有のプログラムインタフェースで要求を配信するように指定することもできます。詳細は、`ipseckey`(1M) および `pf_key`(7P) のマニュアルページを参照してください。

通常、手動での SA 生成は、何らかの理由で IKE を使用できない場合に使用します。ただし、SPI の値が一意であれば、手動での SA 生成と IKE を同時に使用することができます。

IPsec 保護プロトコル

IPsec には、データを保護するための 2 つのセキュリティープロトコルが用意されています。

- 認証ヘッダー (AH)
- カプセル化セキュリティーペイロード (ESP)

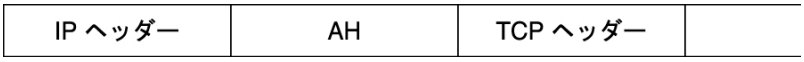
AH は、認証アルゴリズムを使用してデータの整合性を確保します。パケットの暗号化は行いません。

ESP は通常、暗号化アルゴリズムでパケットを保護し、認証アルゴリズムでデータの整合性を確保します。AES GCM のような一部の暗号化アルゴリズムは、暗号化と認証の両方を提供します。

AH プロトコルはネットワークアドレス変換 (NAT) では使用できません。

認証ヘッダー

認証ヘッダーは、IP パケットに対するデータ認証、強力な整合性、およびリプレー保護を提供します。AH では大部分の IP パケットを保護します。次の図に示されているように、AH は IP ヘッダーとトランスポートヘッダーの間に挿入されます。



トランスポートヘッダーは、TCP、UDP、SCTP、または ICMP のいずれかです。トンネルを使用している場合は、トランスポートヘッダーがこれ以外の IP ヘッダーである場合もあります。

カプセル化セキュリティーペイロード

カプセル化セキュリティーペイロード (ESP) プロトコルは、ESP がカプセル化する対象の機密性を守ります。また、AH が提供するサービスも提供します。ただし、ESP は外側の IP ヘッダーは保護しません。ESP は、保護されたパケットの整合性を保証するオプションの認証サービスを提供します。ESP は暗号化対応技術を使用するため、ESP を提供するシステムは輸出入管理法の対象となります。

ESP ヘッダーとトレーラが IP ペイロードをカプセル化します。暗号化と ESP を一緒に使用すると、次の図に示すように、暗号化は IP ペイロードデータにのみ適用されます。

暗号化部分

TCP パケットでは、ESP ヘッダーが認証され、そのヘッダーが TCP ヘッダーおよびそのデータをカプセル化します。パケットが IP 内 IP パケットの場合、ESP は内部 IP パケットを保護します。ソケット別ポリシーでは、「*自己カプセル化*」ができるため、ESP は必要に応じて IP オプションをカプセル化できます。

自己カプセル化は、`setsockopt()` を使用するプログラムを記述することで使用できます。自己カプセル化が設定されている場合は、IP 内 IP パケットを構築するために IP ヘッダーのコピーが作成されます。たとえば、TCP ソケットに自己カプセル化が設定されていない場合、パケットは次の形式で送信されます。

```
[ IP(a -> b) options + TCP + data ]
```

TCP ソケットに自己カプセル化が設定されている場合、パケットは次の形式で送信されます。

```
[ IP(a -> b) + ESP [ IP(a -> b) options + TCP + data ] ]
```

さらに詳しくは、101 ページの「IPsec のトランスポートモードとトンネルモード」を参照してください。

AH と ESP を使用する場合のセキュリティー上の考慮事項

次の表では、AH が提供する保護と ESP が提供する保護を比較します。

表 6-1 IPsec で AH と ESP が提供する保護

プロトコル	パケットの範囲	保護	対象となる攻撃
AH	パケットの IP ヘッダーからトランスポートデータの最後までを保護	強力な完全性およびデータ認証を提供します。 ■ 送信側が送ったものとまったく同じものを受信側が受け取ることを保証する ■ AH がリプレー保護を有効にしていない場合は、リプレー攻撃を受けやすい	リプレー、カットアンドペースト

プロトコル	パケットの範囲	保護	対象となる攻撃
ESP	パケットの ESP ヘッダーからトランスポートデータの最後までを保護	暗号化オプションで、IP ペイロードを暗号化します。機密性を確保します	盗聴
		認証オプションで、AH と同じペイロード保護を提供します	リプレー、カットアンドペースト
		両方のオプションで、強力な完全性、データ認証、および機密性を提供します	リプレー、カットアンドペースト、盗聴

IPsec の認証アルゴリズムと暗号化アルゴリズム

IPsec セキュリティーでは、認証と暗号化という 2 種類のアルゴリズムを使用します。AH プロトコルは認証アルゴリズムを使用します。ESP プロトコルは、暗号化アルゴリズムと認証アルゴリズムを使用できます。`ipsecalgs` コマンドを使用すると、システムのアルゴリズムとそれらのプロパティーの一覧を取得できます。詳細は、`ipsecalgs`(1M) のマニュアルページを参照してください。`getipsecalgbyname`(3NSL) のマニュアルページで説明されている機能を使用して、アルゴリズムのプロパティーを検索することもできます。

IPsec は暗号化フレームワークを使用して暗号化と認証を実行します。暗号化フレームワークによって、IPsec はハードウェアがサポートしているハードウェアアクセラレーションを利用できます。
詳細については、次を参照してください。

- 『Oracle Solaris 11.2 での暗号化と証明書の管理 』の第 1 章「暗号化フレームワーク」
- 『Oracle Solaris 11 セキュリティー開発者ガイド 』の第 8 章「Oracle Solaris 暗号化フレームワークの紹介」

IPsec の保護ポリシー

IPsec の保護ポリシーは、次のレベルで適用できます。

- システム規模レベル
- ソケット単位レベル

IPsec は、IPsec のポリシールールに一致するアウトバウンドパケットとインバウンドパケットにシステム規模のポリシーを適用します。ルールは特定のアルゴリズムを指定したり、いくつかの

アルゴリズムのうち 1 つを許可したりできます。システムで認識される追加のデータがあるため、アウトバウンドパケットにはその他のルールも適用できます。

インバウンドパケットは、受理または拒絶されます。インバウンドパケットを受理するか拒絶するかはいくつかの基準に基づいて決定されます。基準が重複または競合する場合は、最初に解析されたルールが使用されます。

ほとんどのパケットに適用される IPsec ポリシーに対して、例外を指定できます。つまり、IPsec ポリシーを*バイパス*できます。バイパスは、システム規模またはソケット単位で適用できます。

共有 IP アドレス上のゾーンを含むシステム内のトラフィックの場合、ポリシーは実施されますが、実際のセキュリティーメカニズムは適用されません。その代わりに、イントラシステム内パケットのアウトバウンドポリシーが、セキュリティー機能の適用されたインバウンドパケットになります。排他的 IP ゾーンの場合、ポリシーが実施され、実際のセキュリティーメカニズムが適用されます。

`ipsecinit.conf` ファイルと `ipsecconf` コマンドを使用して、IPsec ポリシーを構成します。詳細と例については、`ipsecconf`(1M) のマニュアルページと第7章「IPsec の構成」を参照してください。

IPsec のトランスポートモードとトンネルモード

IPsec 規格では、IPsec の動作モードとして*トランスポートモード*と*トンネルモード*という 2 つの異なるモードが定義されています。トランスポートモードとトンネルモードの主な違いは、ポリシーが適用される場所です。トンネルモードでは、元のパケットは別の IP ヘッダー内にカプセル化されます。ほかのヘッダー内のアドレスは異なる場合があります。

各モードで、パケットは AH または ESP、あるいはその両方によって保護できます。次のように、モードはポリシーアプリケーションによって異なります。

- トランスポートモードでは、外側のヘッダー内の IP アドレスを使用して、パケットに適用される IPsec ポリシーを決定します。
- トンネルモードでは、2 つの IP ヘッダーが送信されます。内側の IP パケットによって、その内容を保護する IPsec ポリシーが決まります。

  トンネルモードは、エンドシステムとセキュリティーゲートウェイのような中間システムとのあらゆる組み合わせに適用できます。

トランスポートモードでは、IP ヘッダー、次のヘッダー、および次のヘッダーでサポートされるすべてのポートを使用して、IPsec ポリシーを決定できます。実際、IPsec は 2 つの IP アドレスの

間で異なるトランスポートモードポリシーを適用でき、ポート単位まで細かく設定できます。たとえば、次のヘッダーが TCP であれば、ポートをサポートするので、外側の IP アドレスの TCP ポートに対して IPsec ポリシーを設定できます。

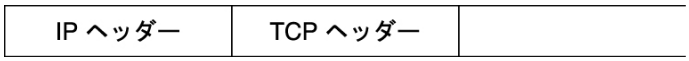
トンネルモードは、IP 内 IP パケットに対してのみ機能します。トンネルモードでは、IPsec ポリシーは内側の IP パケットの内容に適用されます。内側の IP アドレスごとに異なる IPsec ポリシーを適用できます。つまり、内側の IP ヘッダー、その次のヘッダー、および次のヘッダーでサポートされるポートを使用して、ポリシーを適用できます。トランスポートモードとは異なり、トンネルモードでは、外側の IP ヘッダーによって内側の IP パケットのポリシーが決まることはありません。

したがって、トンネルモードでは、ルーターの背後にある LAN のサブネットや、そのようなサブネットのポートに対して、IPsec ポリシーを指定することができます。これらのサブネット上の特定の IP アドレス (つまり、ホスト) に対しても、IPsec ポリシーを指定することができます。これらのホストのポートに対しても、固有の IPsec ポリシーを適用できます。ただし、トンネルを経由して動的ルーティングプロトコルが実行されている場合は、サブネットやアドレスは選択しないでください。ピアネットワークでのネットワークトポロジのビューが変化する可能性があるためです。そのような変化があると、静的な IPsec ポリシーが無効になります。静的ルートの構成を含むトンネリング手順の例については、119 ページの「IPsec による VPN の保護」を参照してください。

Oracle Solaris では、トンネルモードは IP トンネルネットワークインタフェースでのみ適用できます。トンネルインタフェースの詳細については、『Oracle Solaris 11.2 での TCP/IP ネットワーク、IPMP、および IP トンネルの管理 』の第 4 章「IP トンネルの管理について」を参照してください。IPsec ポリシーには、IP トンネルネットワークインタフェースを選択するための `tunnel` キーワードが用意されています。規則内に `tunnel` キーワードが含まれている場合は、その規則に指定されているすべてのセレクタが内側のパケットに適用されます。

次の図は、IP ヘッダーと保護されていない TCP パケットを示します。

図 6-3 TCP 情報を伝送する保護されていない IP パケット



トランスポートモードで、ESP は次の図のようにデータを保護します。網かけされた領域は、パケットの暗号化された部分を示します。

図 6-4　TCP 情報を伝送する保護された IP パケット

暗号化部分

トンネルモードでは、パケット全体が ESP ヘッダー*内*にあります。図6-3「TCP 情報を伝送する保護されていない IP パケット」のパケットは、トンネルモードでは外側の IPsec ヘッダー (この例では ESP) によって保護され、次の図のようになります。

図 6-5　トンネルモードで保護された IPsec パケット

暗号化部分

IPsec ポリシーには、トンネルモードおよびトランスポートモード用にキーワードが用意されています。詳細については、次をレビューしてください。

- ソケットごとのポリシーの詳細については、`ipsec`(7P)のマニュアルページを参照してください。
- ソケットごとのポリシーの例は、117 ページの「IPsec を使用してほかのサーバーとの Web サーバー通信を保護する方法」を参照してください。
- トンネルの詳細については、`ipsecconf`(1M) のマニュアルページを参照してください。
- トンネル構成の例は、123 ページの「IPsec のトンネルモードで 2 つの LAN 間の接続を保護する方法」を参照してください。

仮想プライベートネットワークと IPsec

仮想プライベートネットワーク (VPN) という用語は、多くの場合、インターネットなどのよりパブリックなネットワーク上に構築される、プライベートでセキュアなポイントツーポイントネットワーク

を説明するのに使用されます。このポイントツーポイントネットワーク、つまり VPN を使用して、プライベートネットワーク上のシステム同士、またはプライベートネットワーク上のシステムネットワーク同士を接続できます。

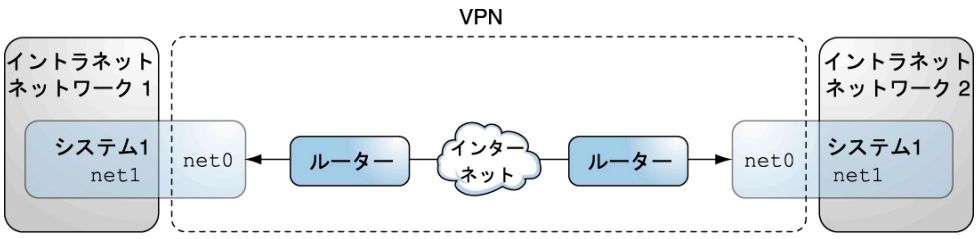
構成されたトンネルは、ポイントツーポイントインタフェースです。トンネルによって、IP パケットを別の IP パケット内にカプセル化できます。トンネルの構成には、トンネルソースとトンネル宛先が必要です。詳細は、『Oracle Solaris 11.2 での TCP/IP ネットワーク、IPMP、および IP トンネルの管理 』の「IP トンネルを作成および構成する方法」を参照してください。

トンネルは、IP への見かけ上の物理インタフェースを作成します。IP トンネルインタフェースを通過する IP トラフィックは IPsec で保護できます。

Oracle Solaris のトンネルインタフェースを使用すると、IP パケットを 1 つのシステムから別のシステムへカプセル化、つまり*トンネル*できます。トンネルされるパケットは、元の IP ヘッダーの前に IP ヘッダーを追加します。追加されたヘッダーは、パブリックネットワーク上でルーティング可能なアドレスを使用します。次の図では、これらのアドレスが `net0` インタフェースで表されています。

次の図は、2 つのサイトで IPsec を使用してサイト間の VPN を作成する方法を示しています。**イントラネット 1** と**イントラネット 2** の間のトラフィックは、ESP 内 IP のカプセル化を使用してインターネット経由でトンネルされます。この場合、外側の IP ヘッダーでは `net0` アドレスが使用され、内側の IP アドレスはイントラネットネットワークからトンネルされるパケットのアドレスになります。内側の IP アドレスには ESP が適用されているため、トラフィックがインターネットを通過する際、これらのアドレスは検査から保護されます。

図 6-6 仮想プライベートネットワーク



設定手順の詳細な例については、123 ページの「IPsec のトンネルモードで 2 つの LAN 間の接続を保護する方法」を参照してください。

IPsec と FIPS 140

FIPS 140 対応システムに対する FIPS 140 要件に準拠するように IPsec を簡単に構成できます。管理者は、FIPS 140 検証済みアルゴリズムのみを選択して鍵と証明書を作成する責任があります。このガイドの手順と例では、アルゴリズム any が指定されている場合を除き、FIPS 140 承認済みアルゴリズムを使用します。

注記 - FIPS 140-2 検証済みの暗号化の使用のみを認める厳密な要件がある場合は、Oracle Solaris 11.1 SRU 5.5 リリースまたは Oracle Solaris 11.1 SRU 3 リリースを実行している必要があります。Oracle は、これら 2 つのリリースの暗号化フレームワークに対する FIPS 140-2 の検証を完了しました。Oracle Solaris 11.2 は、この検証済みの基盤の上に構築され、パフォーマンス、機能、および信頼性に対処するソフトウェアの改良を含んでいます。これらの改良を活かすため、可能な場合はいつでも Oracle Solaris 11.2 を FIPS 140-2 モードで構成するようにしてください。

次のメカニズムは、IPsec で利用可能であり、FIPS 140 モードの Oracle Solaris で使用することが承認されています。

- 鍵の長さが 128-256 ビットの CBC、CCM、GCM、および GMAC モードの AES
- 3DES
- SHA1
- 鍵の長さが 256 および 512 ビットの SHA2

Oracle Solaris 用の FIPS 140 検証済みアルゴリズムの最終的なリストについては、http://www.oracle.com/technetwork/topics/security/140sp2061-2082028.pdfを参照してください。詳細は、『Using a FIPS 140 Enabled System in Oracle Solaris 11.2 』を参照してください。

IPsec と NAT 越え

IKE は、NAT ボックスを通して IPsec SA とネゴシエートできます。この機能により、システムが NAT デバイスの背後にある場合でも、リモートネットワークからシステムへのセキュアな接続が可能です。たとえば、自宅で働く社員や会議場からログオンする社員も IPsec で自分のトラフィックを保護できます。

NAT ボックスは、プライベートな内部アドレスを一意のインターネットアドレスに変換します。NAT は、ホテルなどのインターネットへの公共のアクセスポイントでは非常によく使用されています。

NAT ボックスが通信システム間にある場合に IKE を使用する機能は、「NAT traversal」、または NAT-T と呼ばれます。NAT-T には次の制限があります。

- AH プロトコルは不変の IP ヘッダーに依存するため、AH を NAT-T と連係させることはできません。NAT-T を使用する場合は、ESPプロトコルが使用します。
- NAT ボックスには特別な処理規則はありません。特別な IPsec 処理規則を持つ NAT ボックスは、NAT-T の実装の障害となる場合があります。
- NAT-T が機能するのは、IKE イニシエータが NAT ボックスの背後にあるシステムの場合だけです。ボックスが、ボックスの背後の適切なシステム各自に IKE パケットを転送するようにプログラムされていない場合は、IKE の応答者が NAT ボックスの背後にいることはできません。

次の RFC は、NAT 機能と NAT-T の制限事項について説明しています。RFC のコピーは、http://www.rfc-editor.org で入手できます。

- RFC 3022、『Traditional IP Network Address Translator (Traditional NAT)』、2001 年 1 月
- RFC 3715、『IPsec-Network Address Translation (NAT) Compatibility Requirements』、2004 年 3 月
- RFC 3947、『Negotiation of NAT-Traversal in the IKE』、2005 年 1 月
- RFC 3948、『UDP Encapsulation of IPsec Packets』、2005 年 1 月

IPsec と SCTP

Oracle Solaris は SCTP (Streams Control Transmission Protocol) をサポートしています。SCTP プロトコルと SCTP ポート番号を使用した IPsec ポリシーの指定はサポートされていますが、頑丈ではありません。RFC 3554 に指定されている SCTP の IPsec 拡張は、まだ実装されていません。これらの制限事項によって SCTP 向けの IPsec ポリシーの作成が複雑になる場合もあります。

SCTP は、単独の SCTP アソシエーションのコンテキストで、複数の発信元アドレスと宛先アドレスを利用できます。1 つの発信元アドレスまたは 1 つの宛先アドレスに IPsec ポリシーを適用すると、SCTP がそのアソシエーションの発信元アドレスまたは宛先アドレスを切り替えたときに、通信が失敗する恐れがあります。IPsec ポリシーは、元のアドレスしか認識しません。SCTP の詳細については、ストリーム制御伝送プロトコル (SCTP) に関する RFC をお読みください。

IPsec と Oracle Solaris ゾーン

IPsec はゾーン内でサポートされます。各ゾーンに独自の IPsec ポリシーと IKE 構成を設定できます。ゾーンは個別のホストと同様に扱えます。

例外として、共有 IP ゾーンは個別の IP スタックを持ちません。共有 IP ゾーンの場合、IPsec ポリシーと IKE 構成は大域ゾーンで実行されます。共有 IP ゾーンの IPsec ポリシールールは、そのゾーンに割り当てられている IP アドレスを使用します。

詳細は、『Oracle Solaris ゾーンの紹介 』の第 1 章「Oracle Solaris ゾーンの概要」を参照してください。

IPsec と仮想マシン

IPsec は仮想マシン (VM) で動作します。SPARC システムで VM を作成するには、Oracle VM サーバーを使用します。x86 システムでは、Oracle VM VirtualBox を使用できます。構成の詳細については、使用している Oracle VM のバージョンの管理ガイドを参照してください。

IPsec の構成コマンドとファイル

表6-2「IPsec の構成コマンドとファイルの例」は、IPsec を構成および管理するために使用するファイル、コマンド、およびサービス識別子について説明しています。完全性を期すために、鍵管理ファイルとコマンドも含めました。

サービス識別子の詳細については、『Oracle Solaris 11.2 でのシステムサービスの管理 』の第 1 章「サービス管理機能の概要」を参照してください。

IPsec をネットワークに実装する手順については、111 ページの「IPsec によるネットワークトラフィックの保護」を参照してください。

IPsec ユーティリティーとファイルの詳細については、第12章「IPsec および鍵管理のリファレンス」を参照してください。

表 6-2 IPsec の構成コマンドとファイルの例

IPsec のコマンド、ファイル、またはサービス	説明	マニュアルページ
svc:/network/ipsec/ipsecalgs	IPsec アルゴリズムを管理する SMF サービス。	ipsecalgs(1M)
svc:/network/ipsec/manual-key	手動で鍵が設定された IPsec SA を管理する SMF サービス。	ipseckey(1M)
svc:/network/ipsec/policy	IPsec ポリシーを管理する SMF サービス。	smf(5)、ipsecconf(1M)
svc:/network/ipsec/ike:ikev2、svc:/network/ipsec/ike:default	IKE を使用した IPsec SA 自動管理用の SMF サービスインスタンス。	smf(5)、in.ikev2d(1M)、in.iked(1M)
/etc/inet/ipsecinit.conf ファイル	IPsec ポリシーファイル。	ipsecconf(1M)
	SMF policy サービスによって、システムブート時に IPsec ポリシーを構成するために使用されます。	
ipsecconf コマンド	IPsec ポリシーコマンド。現在の IPsec ポリシーの表示および変更や、テストを行うときに役立ちます。	ipsecconf(1M)
	SMF policy サービスによって、システムブート時に IPsec ポリシーを構成するために使用されます。	
PF_KEY ソケットインタフェース	セキュリティーアソシエーションデータベース (SADB) のインタフェース。手動と自動の鍵管理を処理します。	pf_key(7P)
ipseckey コマンド	IPsec SA 鍵作成コマンド。ipseckey は、PF_KEY インタフェースに対するコマンド行フロントエンドです。ipseckey は、SA を作成、破棄、または修正できます。	ipseckey(1M)
/etc/inet/secret/ipseckeys ファイル	手動で鍵が設定された SA を含みます。	
	SMF manual-key サービスによって、システムブート時に SA を手動で構成するために使用されます。	
ipsecalgs コマンド	IPsec アルゴリズムコマンド。IPsec アルゴリズムとそのプロパティーの一覧を参照および変更するときに役立ちます。	ipsecalgs(1M)
	システムブート時に既知の IPsec アルゴリズムをカーネルと同期するために SMF ipsecalgs サービスで使用されます。	
/etc/inet/ipsecalgs ファイル	構成されている IPsec メカニズムとアルゴリズム定義を含みます。このファイルは、ipsecalgs コマンドによって管理されます。手動では絶対に編集しないでください。	
/etc/inet/ike/ikev2.config ファイル	IKEv2 の構成とポリシーファイル。鍵管理はこのファイルのルールとグローバルパラメータに基づきます。237 ページの「IKEv2 のユーティリティーおよびファイル」を参照してください。	ikev2.config(4)
/etc/inet/ike/config ファイル	IKEv1 の構成とポリシーファイル。デフォルトでは、このファイルはありません。鍵管理はこのファイルのルールとグローバルパラメータに基	ike.config(4)

IPsec のコマンド、ファイル、またはサービス	説明	マニュアルページ
	づきます。242 ページの「IKEv1 のユーティリティーおよびファイル」を参照してください。 このファイルが存在する場合、`svc:/network/ipsec/ike:default` サービスは IKEv1 デーモン `in.iked` を起動します。	

♦ ♦ ♦ 第 7 章

7 IPsec の構成

この章では、ネットワークに IPsec を実装する手順について説明します。手順は次の各セクションで説明します。

- 111 ページの「IPsec によるネットワークトラフィックの保護」
- 119 ページの「IPsec による VPN の保護」
- 128 ページの「IPsec のその他のタスク」

IPsec の概要については、第6章「IP セキュリティーアーキテクチャーについて」を参照してください。IPsec の参照情報については、第12章「IPsec および鍵管理のリファレンス」を参照してください。

注記 - これらのタスクは、システムに静的 IP アドレスがアサイン済みで、ネットワーク構成プロファイル `DefaultFixed` を実行していることを前提としています。`netadm list` コマンドが `Automatic` を返す場合は、`netcfg`(1M) のマニュアルページで詳細を確認してください。

IPsec によるネットワークトラフィックの保護

このセクションの手順では、2 つのシステム間のトラフィックおよび Web サーバーをセキュリティー保護できます。VPN を保護するには、119 ページの「IPsec による VPN の保護」を参照してください。IPsec を管理したり、IPsec や IKE で SMF コマンドを使用したりするための追加手順については、128 ページの「IPsec のその他のタスク」を参照してください。

次の情報は、すべての IPsec 構成タスクで使用されます。

- **IPsec とゾーン** – 各システムは大域ゾーンまたは排他的 IP ゾーンです。詳細は、107 ページの「IPsec と Oracle Solaris ゾーン」を参照してください。
- **IPsec と FIPS 140 モード** – IPsec 管理者は、Oracle Solaris 用の FIPS 140 検証済みアルゴリズムを選択する責任があります。この章の手順と例では、アルゴリズム `any` が指定されている場合を除き、FIPS 140 承認済みアルゴリズムを使用します。

- **IPsec と RBAC** – IPsec を管理する役割を使用するには、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の第 3 章「Oracle Solaris での権利の割り当て」を参照してください。例については、131 ページの「ネットワークセキュリティーの役割を構成する方法」を参照してください。
- **IPsec と SCTP** – IPsec を使用して Streams Control Transmission Protocol (SCTP) アソシエーションを保護できますが、注意が必要です。詳細は、106 ページの「IPsec と SCTP」を参照してください。
- **IPsec と Trusted Extensions のラベル** – Oracle Solaris の Trusted Extensions 機能が構成されたシステムでは、IPsec パケットにラベルを追加できます。詳細については、『Trusted Extensions 構成と管理 』の「ラベル付き IPsec の管理」を参照してください。
- **IPv4 および IPv6 アドレス** – このガイドの IPsec の例では、IPv4 アドレスを使用しています。Oracle Solaris は IPv6 アドレスもサポートします。IPv6 ネットワークの IPsec を構成するには、例で IPv6 アドレスに読み替えてください。トンネルを IPsec で保護するときに、内部アドレスや外部アドレスで IPv4 アドレスと IPv6 アドレスを混在させることができます。このタイプの構成では、たとえば IPv4 ネットワーク上で IPv6 トンネリングを行うことができます。

次のタスクマップに、1 台以上のシステム間で IPsec を設定する手順を示します。`ipsecconf`(1M)、`ipseckey`(1M)、および `ipadm`(1M) のマニュアルページでも、それぞれの例のセクションで役立つ手順を説明しています。

表 7-1 IPsec によるネットワークトラフィックの保護のタスクマップ

タスク	説明	参照先
システム間のトラフィックを保護します。	あるシステムから別のシステムへのパケットを保護します。	113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」
IPsec ポリシーによる Web サーバーを保護します。	Web 以外のトラフィックに IPsec の使用を求めます。Web クライアントは、IPsec チェックをバイパスする特定のポートによって識別されます。	117 ページの「IPsec を使用してほかのサーバーとの Web サーバー通信を保護する方法」
IKE を使用して IPsec SA 用のキーイング素材を自動作成します。	IPsec SA の作成に推奨される方法。	149 ページの「IKEv2 の構成」および 177 ページの「IKEv1 の構成」
セキュアな仮想プライベートネットワーク (VPN) を設定します。	2 つのシステム間でインターネット経由で IPsec を設定します。	119 ページの「IPsec による VPN の保護」
手動鍵管理を設定します。	IKE を使用せずに IPsec SA の生データを提供します。	128 ページの「IPsec の鍵を手動で作成する方法」

▼ IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法

この手順では、次の設定がすでになされているものとします。

- システムに静的 IP アドレスがアサイン済みで、ネットワーク構成プロファイル `DefaultFixed` を実行している。`netadm list` コマンドが `Automatic` を返す場合は、`netcfg`(1M) のマニュアルページで詳細を確認してください。
- 2 つのシステムが `enigma` および `partym` と名付けられている。
- 各システムが IP アドレスを持っています。これは、IPv4 アドレス、IPv6 アドレスのどちらでもかまいませんし、その両方でもかまいません。この手順では、IPv4 アドレスを使用します。
- 各システムは大域ゾーンまたは排他的 IP ゾーンである。詳細は、107 ページの「IPsec と Oracle Solaris ゾーン」を参照してください。
- 各システムはトラフィックを AES アルゴリズムで暗号化し、SHA-2 で認証する。

注記 - 一部のサイトでは SHA-2 アルゴリズムが要求される場合があります。

- 各システムは、共有セキュリティーアソシエーションを使用します。

  共有セキュリティーアソシエーションでは、2 つのシステムを保護するのに必要なのは 1 組だけの SA です。

注記 - Trusted Extensions システムのラベルと一緒に IPsec を使用するには、『Trusted Extensions 構成と管理 』の「マルチレベル Trusted Extensions ネットワークで IPsec 保護を適用する」にあるこの手順の拡張を参照してください。

始める前に 特定の権利を持つユーザーは、`root` でなくても次のコマンドを実行できます。

- 構成コマンドを実行するには、Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。
- この管理役割では、IPsec 関連のシステムファイルを編集し、`pfedit` コマンドを使用して鍵を作成できます。
- `hosts` ファイルを編集するには、`root` 役割になるか、そのファイルを編集するための明示的なアクセス権を持っている必要があります。例7-7「信頼できるユーザーに IPsec を構成および管理する権限を与える」を参照してください。

詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **各システムで、/etc/inet/hosts ファイルにホストエントリを追加します。**

   この手順により、ローカルネームサービスがネットワークに接続されたネームサービスに依存することなくシステム名を IP アドレスに解決できるようになります。

   a. **partym という名前のシステムでは、hosts ファイルに次のように入力します。**

   ```
   ## Secure communication with enigma
   192.168.116.16 enigma
   ```

   b. **enigma という名前のシステムでは、hosts ファイルに次のように入力します。**

   ```
   ## Secure communication with partym
   192.168.13.213 partym
   ```

2. **各システムで、IPsec ポリシーファイルを作成します。**

   ファイル名は /etc/inet/ipsecinit.conf です。例は、/etc/inet/ipsecinit.sample ファイルを参照してください。

   ```
   # pfedit /etc/inet/ipsecinit.conf
   ```

3. **IPsec ポリシーエントリを ipsecinit.conf ファイルに追加します。**

   IPsec ポリシーエントリの構文といくつかの例については、ipsecconf(1M) のマニュアルページを参照してください。

   a. **enigma システムで、次のポリシーを追加します。**

   ```
   {laddr enigma raddr partym} ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
   ```

   dir キーワードが使用されていないため、ポリシーはアウトバウンドとインバウンド両方のパケットに適用されます。

   b. **partym システムで、同じポリシーを追加します。**

   ```
   {laddr partym raddr enigma} ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
   ```

4. **各システムで、IPsec SA を管理するための IKE を構成します。**

   149 ページの「IKEv2 の構成」の構成手順のいずれかに従います。IKE 構成ファイルの構文については、`ikev2.config`(4) のマニュアルページを参照してください。IKEv1 プロトコルのみをサポートしているシステムと通信している場合は、177 ページの「IKEv1 の構成」および `ike.config`(4) のマニュアルページを参照してください。

   注記 - 鍵を手動で生成して維持する必要がある場合は、128 ページの「IPsec の鍵を手動で作成する方法」を参照してください。

5. **IPsec ポリシーファイルの構文を確認します。**

```
% pfbash
# /usr/sbin/ipsecconf -c /etc/inet/ipsecinit.conf
```

   エラーがあれば修正し、ファイルの構文を確認してから続行します。

6. **IPsec ポリシーをリフレッシュします。**

```
# svcadm refresh ipsec/policy:default
```

   IPsec ポリシーはデフォルトで有効になっているので、「リフレッシュ」を行います。IPsec ポリシーを無効にしてある場合は有効にしてください。

```
# svcadm enable ipsec/policy:default
```

7. **IPsec の鍵をアクティブ化します。**

   - **`ike` サービスが有効になっていない場合は有効にします。**

     注記 - IKEv1 プロトコルのみを実行できるシステムと通信している場合は、`ike:default` インスタンスを指定します。

```
# svcadm enable ipsec/ike:ikev2
```

   - **`ike` サービスが有効になっている場合は再起動します。**

```
# svcadm restart ike:ikev2
```

   ステップ 4 で鍵を手動で構成した場合は、128 ページの「IPsec の鍵を手動で作成する方法」の手順を実行して鍵をアクティブ化します。

8. **パケットが保護されていることを確認します。**

手順については、134 ページの「IPsec によってパケットが保護されていることを確認する方法」を参照してください。

例 7-1 ssh 接続を使用した IPsec ポリシーのリモート構成

この例では、root 役割の管理者が 2 番目のシステムにアクセスするための ssh コマンドを使用して、2 つのシステム上で IPsec ポリシーと鍵を構成します。管理者は両方のシステムで同じように定義されています。詳細は、ssh(1) のマニュアルページを参照してください。

1. 管理者は、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」の ステップ 1 から ステップ 5 までを実行して最初のシステムを構成します。
2. 別の端末ウィンドウで、管理者は同じように定義されたユーザー名と ID を使用して、ssh コマンドによってリモートでログインします。

```
local-system % ssh -l jdoe other-system
other-system # su - root
Enter password: xxxxxxxx
other-system #
```

3. ssh セッションの端末ウィンドウで、ステップ 1 から ステップ 7 までを実行して、2 番目のシステムの IPsec ポリシーと鍵を構成します。
4. 管理者は ssh セッションを終了します。

```
other-system # exit
local-system
# exit
```

5. 管理者は、ステップ 6 と ステップ 7 を実行して、最初のシステムの IPsec ポリシーを有効にします。

2 つのシステムが次に通信を行うとき、ssh 接続を使用した通信も含め、通信は IPsec で保護されます。

例 7-2 FIPS 140 モードで実行する IPsec ポリシーの構成

この例では、管理者が、鍵の長さが少なくとも 192 ビットの対称アルゴリズムを必要とするサイトのセキュリティーポリシーに従うように、FIPS 140 対応システムの IPsec ポリシーを構成します。

管理者は 2 つの使用可能な IPsec ポリシーを指定します。1 つ目のポリシーは、暗号化と認証に対して CCM モードの AES を指定し、2 つ目のポリシーは、暗号化に対して鍵の長さが 192 および 256 ビットの AES を指定し、認証に対して SHA384 を指定します。

```
{laddr machine1 raddr machine2} ipsec {encr_algs aes-ccm(192...) sa shared} or ipsec
{laddr machine1 raddr machine2} ipsec {encr_algs aes(192...) encr_auth_algs sha2(384) sa
shared}
```

▼ IPsec を使用してほかのサーバーとの Web サーバー通信を保護する方法

Web サーバーを実行するシステムで、IPsec を使用して Web クライアントのリクエスト以外のすべてのトラフィックを保護できます。保護されるネットワークトラフィックは、通常、Web サーバーとほかのバックエンドサーバーの間です。

Web クライアントが IPsec をバイパスできるだけでなく、この手順では IPsec ポリシーによってサーバーが DNS クライアントのリクエストを作成できるようになります。その他のすべてのトラフィックは IPsec で保護されます。

始める前に この手順では、2 つのサーバー上で IPsec を構成する 113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」の手順が実行済みで、次の条件が満たされていると想定します。

- 各システムは、固定アドレスを持つ大域ゾーンまたは排他的 IP ゾーンである。詳細は、107 ページの「IPsec と Oracle Solaris ゾーン」を参照してください。
- Web サーバーとの通信は IPsec によってすでに保護されている。
- IKE によってキーイング素材が生成されています。
- パケットが保護されていることを確認してあります。

特定の権利を持つユーザーは、`root` でなくてもこれらのコマンドを実行できます。

- 構成コマンドを実行するには、Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。
- IPsec 関連のシステムファイルを編集して鍵を作成するには、`pfedit` コマンドを使用します。
- `hosts` ファイルを編集するには、`root` 役割になるか、そのファイルを編集するための明示的なアクセス権を持っている必要があります。

詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **IPsec ポリシー検査をバイパスするサービスを指定します。**

   Web サーバーの場合、TCP ポート 80 (HTTP) と 443 (保護 HTTP) が該当します。Web サーバーが DNS 名検査をするときは、TCP と UDP の両方にポート 53 も組み込む必要がある場合もあります。

2. **Web サーバーポリシーを IPsec ポリシーファイルに追加します。**

   `ipsecinit.conf` ファイルに次の行を追加します。

   ```
   # pfedit /etc/inet/ipsecinit.conf
   ...
   # Web traffic that web server should bypass.
   {lport  80 ulp tcp dir both} bypass {}
   {lport 443 ulp tcp dir both} bypass {}

   # Outbound DNS lookups should also be bypassed.
   {rport 53 dir both} bypass {}

   # Require all other traffic to use ESP with AES and SHA-2.
   # Use a unique SA for outbound traffic from the port
   {} ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
   ```

   これで、保護トラフィックだけがシステムへのアクセスを許可されます。ただし、ステップ 1 で説明した、検査を省略するトラフィックは例外です。

3. **IPsec ポリシーファイルの構文を確認します。**

   ```
   # ipsecconf -c /etc/inet/ipsecinit.conf
   ```

4. **IPsec ポリシーをリフレッシュします。**

   ```
   # svcadm refresh ipsec/policy
   ```

5. **IPsec 用の鍵をリフレッシュします。**

   `ike` サービスを再起動します。

   ```
   # svcadm restart ike:ikev2
   ```

注記 - IKEv1 プロトコルのみを実行できるシステムと通信している場合は、`ike:default` インスタンスを指定します。

鍵を手動で構成した場合は、128 ページの「IPsec の鍵を手動で作成する方法」の手順に従います。

これで設定が完了しました。

6. **(オプション) Web 以外のトラフィックのために Web サーバーと通信する場合は、リモートシステムを有効にします。**

   リモートシステムの `/etc/inet/ipsecinit.conf` ファイルに次の行を追加します。

```
## Communicate with web server about nonweb stuff
##
{raddr webserver} ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
```

   構文を検証したあと、IPsec ポリシーをリフレッシュしてアクティブ化します。

```
remote-system # ipsecconf -c /etc/inet/ipsecinit.conf
remote-system # svcadm refresh ipsec/policy
```

   IPsec ポリシーが一致した場合にかぎり、リモートシステムは、非 Web トラフィックを持つ Web サーバーと安全に通信できます。

7. **(オプション) トンネルごとのエントリも含め、IPsec ポリシーエントリを一致した順序で表示します。**

```
# ipsecconf -L -n
```

IPsec による VPN の保護

IPsec を使用して VPN を保護できます。背景については、101 ページの「IPsec のトランスポートモードとトンネルモード」を参照してください。このセクションの例や手順では IPv4 アドレスを使用しますが、それらの例や手順は IPv6 VPN にも適用されます。簡単な説明については、111 ページの「IPsec によるネットワークトラフィックの保護」を参照してください。

トンネルモード用の IPsec ポリシーの例については、120 ページの「トンネルモードを使用して VPN を IPsec で保護する例」を参照してください。

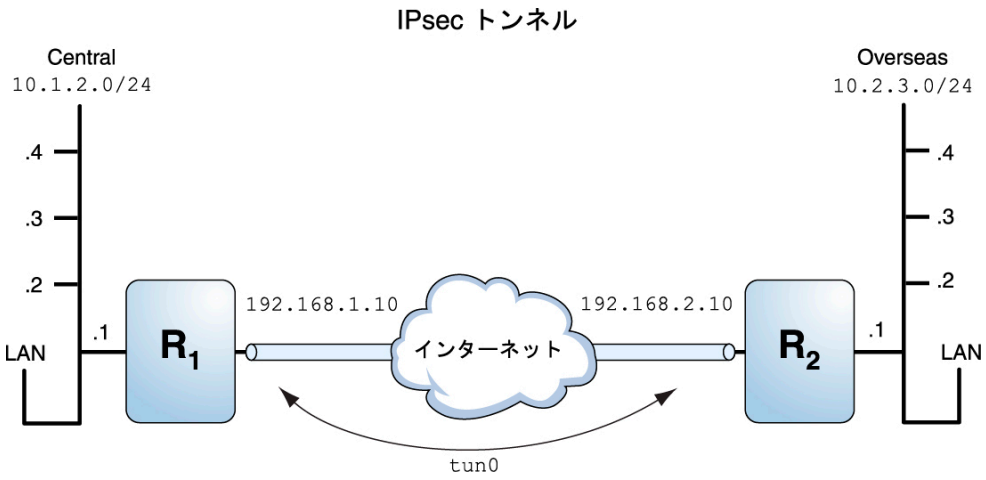
トンネルモードを使用して VPN を IPsec で保護する例

次の図のトンネルは、次のように LAN のすべてのサブネットに対して構成されています。

```
## Tunnel configuration for ##
# Tunnel name is tun0
# Intranet point for the source is 10.1.2.1
# Intranet point for the destination is 10.2.3.1
# Tunnel source is 192.168.1.10
# Tunnel destination is 192.168.2.10

# Tunnel name address object is tun0/to-central
# Tunnel name address object is tun0/to-overseas
```

図 7-1 IPsec で保護されたトンネル



次の例はこの図に基づいています。

例 7-3 すべてのサブネットで使用できるトンネルの作成

この例では、図7-1「IPsec で保護されたトンネル」の Central LAN のローカル LAN からのすべてのトラフィックが、ルーター 1 からルーター 2 にトンネリングされたあとに、Overseas LAN のすべてのローカル LAN に配信されます。トラフィックは AES で暗号化されます。

```
## IPsec policy ##
{tunnel tun0 negotiate tunnel}
```

```
 ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
```

例 7-4 2 つのサブネットだけを接続するトンネルの作成

この例では、Central LAN のサブネット `10.1.2.0/24` と Overseas LAN のサブネット `10.2.3.0/24` の間のトラフィックだけがトンネリングされ、暗号化されます。Central に対するほかの IPsec ポリシーがない場合、Central LAN がこのトンネル経由でほかの LAN にトラフィックを配信しようとすると、トラフィックはルーター 1 でドロップされます。

```
## IPsec policy ##
{tunnel tun0 negotiate tunnel laddr 10.1.2.0/24 raddr 10.2.3.0/24}
 ipsec {encr_algs aes encr_auth_algs sha512 shared}
```

IPsec で VPN を保護するタスクのためのネットワークトポロジの説明

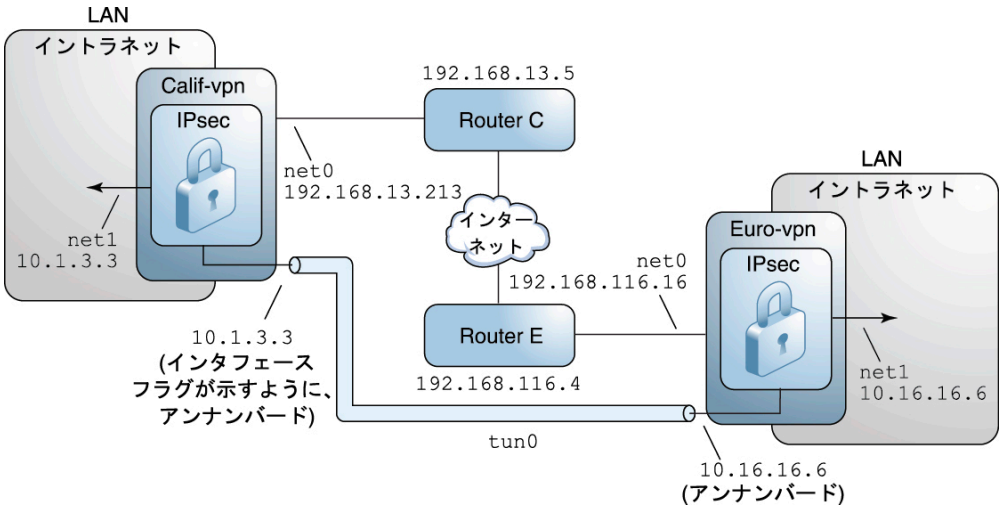
このセクションの手順では、次の設定を前提としています。図7-2「インターネット経由で接続されたオフィス間の VPN の例」はこのネットワークを表しています。

- 各システムは IPv4 アドレス空間を使用します。

  これらの手順は IPv6 アドレス、または IPv4 アドレスと IPv6 アドレスの組み合わせでも機能します。
- 各システムには 2 つのインタフェースがあります。`net0` インタフェースはインターネットに接続しています。この例では、インターネット IP アドレスは `192.168` で始まります。`net1` インタフェースは社内の LAN、すなわちイントラネットに接続します。この例では、イントラネット IP アドレスは `10` で始まります。
- 各システムには、AES アルゴリズムを使用した ESP 暗号化が必要です。AES アルゴリズムは 128 ビットまたは 256 ビットの鍵を使用します。
- 各システムには、SHA-2 アルゴリズムを使用した ESP 認証が必要です。この例では、SHA-2 アルゴリズムは 512 ビットの鍵を使用します。
- 各システムは、インターネットに直接アクセスするルーターに接続できます。
- 各システムは、共有セキュリティーアソシエーションを使用します。

次の図は、手順内で使用される構成パラメータを示しています。

図 7-2 インターネット経由で接続されたオフィス間の VPN の例



構成パラメータを次の表に示します。

パラメータ	ヨーロッパ	カリフォルニア
システム名	euro-vpn	calif-vpn
システムイントラネットインタフェース	net1	net1
システムイントラネットアドレス、相手ネットワークへのデフォルトのルート	10.16.16.6	10.1.3.3
システムイントラネットアドレスオブジェクト	net1/inside	net1/inside
システムインターネットインタフェース	net0	net0
システムインターネットアドレス	192.168.116.16	192.168.13.213
インターネットルーターの名前	router-E	router-C
インターネットルーターのアドレス	192.168.116.4	192.168.13.5
トンネル名	tun0	tun0
トンネル名アドレスオブジェクト	tun0/v4tunaddr	tun0/v4tunaddr

トンネル名の詳細については、『Oracle Solaris 11.2 での TCP/IP ネットワーク、IPMP、および IP トンネルの管理 』の「IP トンネルの管理」を参照してください。アドレスオブジェクトの詳細については、『Oracle Solaris 11.2 でのネットワークコンポーネントの構成と管理 』の

「IPv4 インタフェースを構成する方法」および ipadm(1M) のマニュアルページを参照してください。

▼ IPsec のトンネルモードで 2 つの LAN 間の接続を保護する方法

トンネルモードでは、内側の IP パケットによって、その内容を保護する IPsec ポリシーが決まります。

この手順は、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」の手順の応用です。設定については、121 ページの「IPsec で VPN を保護するタスクのためのネットワークトポロジの説明」を参照してください。

特定のコマンドを実行する理由の詳細については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」の対応する手順を参照してください。

注記 - 両方のシステムでこの手順を実行してください。

この手順では、2 つのシステムを接続するだけでなく、これら 2 つのシステムに接続している 2 つのイントラネットを接続します。この手順における 2 つのシステムはゲートウェイとして機能します。

注記 - トンネルモードの IPsec を Trusted Extensions システムのラベルと組み合わせて使用する場合は、『Trusted Extensions 構成と管理 』の「信頼できないネットワーク上でトンネルを構成する」のこの手順の拡張を参照してください。

始める前に 各システムは大域ゾーンまたは排他的 IP ゾーンである。詳細は、107 ページの「IPsec と Oracle Solaris ゾーン」を参照してください。

特定の権利を持つユーザーは、root でなくてもこれらのコマンドを実行できます。

- 構成コマンドを実行するには、Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。
- IPsec 関連のシステムファイルを編集して鍵を作成するには、pfedit コマンドを使用します。
- hosts ファイルを編集するには、root 役割になるか、そのファイルを編集するための明示的なアクセス権を持っている必要があります。

詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **IPsec を構成する前に、パケットフローを制御します。**

   a. **IP 転送と IP 動的ルーティングを無効にします。**

   ```
   # routeadm -d ipv4-routing
   # ipadm set-prop -p forwarding=off ipv4
   # routeadm -u
   ```

   IP 転送をオフにすると、このシステムを介したあるネットワークから別のネットワークへのパケット転送ができなくなります。routeadm コマンドの説明については、routeadm(1M) のマニュアルページを参照してください。

   b. **IP 厳密宛先マルチホームを有効にします。**

   ```
   # ipadm set-prop -p hostmodel=strong ipv4
   ```

   IP 厳密宛先マルチホームを有効にするには、システムの宛先アドレスのうちの 1 つに宛てたパケットが正しい宛先アドレスに到着する必要があります。

   hostmodel パラメータを strong に設定したときは、ある特定のインタフェースに到着するパケットには、そのインタフェースのローカル IP アドレスの 1 つが指定されている必要があります。その他のパケットは、システムのほかのローカルアドレスが指定されているものも含めてすべて捨てられます。

   c. **ほとんどのネットワークサービスが無効になっていることを確認します。**

   ssh サービスが実行中であることを確認します。

   ```
   % svcs | grep network
   …
   online         Aug_09   svc:/network/ssh:default
   ```

2. **/etc/inet/ipsecinit.conf ファイルに VPN の IPsec ポリシーを追加します。**

   その他の例については、120 ページの「トンネルモードを使用して VPN を IPsec で保護する例」を参照してください。

このポリシーでは、ローカル LAN 上のシステムとゲートウェイの内部 IP アドレスの間に IPsec 保護は必要でないため、bypass 文を追加します。

a. **euro-vpn システムで、ipsecinit.conf ファイルに次のエントリを追加します。**

```
# LAN traffic to and from this host can bypass IPsec.
{laddr 10.16.16.6 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-2.
{tunnel tun0 negotiate tunnel}
 ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
```

b. **calif-vpn システムで、ipsecinit.conf ファイルに次のエントリを追加します。**

```
# LAN traffic to and from this host can bypass IPsec.
{laddr 10.1.3.3 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-2.
{tunnel tun0 negotiate tunnel}
 ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
```

3. **各システムで、2 つのシステム間の IPsec SA のペアを追加するために IKE を構成します。**

149 ページの「IKEv2 の構成」の構成手順のいずれかに従って、IKE を構成します。IKE 構成ファイルの構文については、ikev2.config(4) のマニュアルページを参照してください。IKEv1 プロトコルのみをサポートしているシステムと通信している場合は、177 ページの「IKEv1 の構成」および ike.config(4) のマニュアルページを参照してください。

注記 - 鍵を手動で生成して維持する必要がある場合は、128 ページの「IPsec の鍵を手動で作成する方法」を参照してください。

---

4. **IPsec ポリシーファイルの構文を確認します。**

```
# ipsecconf -c /etc/inet/ipsecinit.conf
```

エラーがあれば修正し、ファイルの構文を確認してから続行します。

5. **IPsec ポリシーをリフレッシュします。**

```
# svcadm refresh ipsec/policy
```

IPsec ポリシーはデフォルトで有効になっているので、「リフレッシュ」を行います。IPsec ポリシーを無効にしてある場合は有効にしてください。

```
# svcadm enable ipsec/policy
```

6. **トンネル `tun0` を作成および構成します。**

次のコマンドは、内部および外部インタフェースを構成し、`tun0` トンネルを作成し、そのトンネルに IP アドレスを割り当てます。

a. **`calif-vpn` システムで、トンネルを作成して構成します。**

```
# ipadm create-ip net1
# ipadm create-addr -T static -a local=10.1.3.3 net1/inside
# dladm create-iptun -T ipv4 -a local=192.168.13.213,remote=192.168.116.16 tun0
# ipadm create-ip tun0
# ipadm create-addr -T static \
-a local=10.1.3.3,remote=10.16.16.6 tun0/v4tunaddr
```

最初のコマンドは、IP インタフェース `net1` を作成します。2 番目のコマンドは、`net1` にアドレスを追加します。3 番目のコマンドは、IP インタフェース `tun0` を作成します。4 番目のコマンドは、トンネルリンク内でカプセル化されている IP アドレスを追加します。詳細は、`dladm`(1M) および `ipadm`(1M) のマニュアルページを参照してください。

b. **`euro-vpn` システムで、トンネルを作成して構成します。**

```
# ipadm create-ip net1
# ipadm create-addr -T static -a local=10.16.16.6 net1/inside
# dladm create-iptun -T ipv4 -a local=192.168.116.16,remote=192.168.13.213 tun0
# ipadm create-ip tun0
# ipadm create-addr -T static \
-a local=10.16.16.6,remote=10.1.3.3 tun0/v4tunaddr
```

注記 - `ipadm` コマンドの `-T` オプションは、作成するアドレスのタイプを指定します。`dladm` コマンドの `-T` オプションは、トンネルを指定します。

これらのコマンドの詳細については、`dladm`(1M) と `ipadm`(1M) のマニュアルページ、および『Oracle Solaris 11.2 でのネットワークコンポーネントの構成と管理 』の「IPv4 インタフェースを構成する方法」を参照してください。カスタマイズ名については、『Oracle Solaris 11.2 でのネットワークコンポーネントの構成と管理 』の「Oracle Solaris のネットワークデバイスとデータリンク名」を参照してください。

7. **各システムで、転送を構成します。**

```
# ipadm set-ifprop -m ipv4 -p forwarding=on net1
# ipadm set-ifprop -m ipv4 -p forwarding=on tun0
# ipadm set-ifprop -m ipv4 -p forwarding=off net0
```

IP 転送とは、別のインタフェースから到着したパケットを転送できることを意味します。IP 転送はまた、送信するパケットがもともとは別のインタフェースから発信されたパケットである可能性

も意味します。パケットを正しく転送するには、受信インタフェースと送信インタフェースの IP 転送を有効にしておきます。

net1 インタフェースはイントラネットの「*内部*」にあるため、net1 の IP 転送は有効にしておきます。tun0 はインターネットを通してこれら 2 つのシステムを接続するため、tun0 の IP 転送は有効にしておく必要があります。net0 インタフェースの IP 転送は無効になっているため、インターネット上の「*外部*」の脅威によってパケットが保護イントラネットに侵入するのを防ぐことができます。

8. **各システムで、プライベートインタフェースの広告を禁止します。**

```
# ipadm set-addrprop -p private=on net0
```

net0 の IP 転送が無効になっていても、ルーティングプロトコルの実装によっては、このインタフェースを通知することがあります。たとえば、in.routed プロトコルは、イントラネット内のピアにパケットが転送される際に net0 を有効なインタフェースとして通知する場合があります。インタフェースの「private」フラグを設定して、このような通知が行われないようにします。

9. **ネットワークサービスを再起動します。**

```
# svcadm restart svc:/network/initial:default
```

10. **net0 インタフェース経由のデフォルトルートを手動で追加します。**

デフォルトルートは、インターネットに直接アクセスできるルーターでなければなりません。

a. **calif-vpn システムで次のルートを追加します。**

```
# route -p add net default 192.168.13.5
```

b. **euro-vpn システムで、次のルートを追加します。**

```
# route -p add net default  192.168.116.4
```

net0 インタフェースはイントラネットの一部ではありませんが、インターネットを介してそのピアシステムにアクセスする必要があります。net0 は、自身のピアを見つけるために、インターネット経路制御情報を必要とします。インターネットの残りの要素にとって、VPN システムは、ルーターというよりもホストのような存在です。したがって、デフォルトルーターを使用するか、ルーター発見プロトコルを実行すれば、ピアシステムを見つけることができます。詳細は、route(1M) および in.routed(1M) のマニュアルページを参照してください。

IPsec のその他のタスク

次のタスクマップは、IPsec を管理するときに使用する可能性のあるタスクを示したものです。

表 7-2 IPsec のその他のタスクのタスクマップ

タスク	説明	参照先
手動による IPsec SA の作成または置き換えを行います。	IPsec SA の生データを提供します。	128 ページの「IPsec の鍵を手動で作成する方法」
ネットワークセキュリティーの役割を作成します。	セキュリティーネットワークを設定できるが、root 役割より権限が少ない役割を作成します。	131 ページの「ネットワークセキュリティーの役割を構成する方法」
すべてのネットワーク管理タスクを処理できる権利プロファイルを作成します。	ネットワーク管理を実行できるが、root 役割より権限が少ない役割を作成します。	例7-7「信頼できるユーザーに IPsec を構成および管理する権限を与える」
IPsec がパケットを保護しているかどうかを検査します。	snoop の出力を調べ、IP パケットがどのように保護されているかを示すヘッダーをチェックします。	134 ページの「IPsec によってパケットが保護されていることを確認する方法」
IPsec と鍵情報を一連の SMF サービスとして管理します。	サービスの有効化、無効化、リフレッシュ、および再起動を行います。サービスのプロパティー値も変更します。	220 ページの「IPsec および手動鍵サービスプロパティーの表示」

▼ IPsec の鍵を手動で作成する方法

次の手順は、鍵の管理に使用しているのが IKE だけではない場合に IPsec 鍵を提供します。

ipseckey コマンドを使用して追加された IPsec SA には永続性がなく、システムのリブート時に失われます。IPsec SA に永続性を持たせるために、/etc/inet/secret/ipseckeys ファイルにエントリを追加します。

注意 - 手動で鍵処理をする必要がある場合は、生成する鍵が確実にセキュアであるように細心の注意を払う必要があります。これらの鍵は、データのセキュリティー保護に実際に使用されます。

始める前に 共有 IP ゾーンの鍵情報の手動管理は、大域ゾーンで行う必要があります。排他的 IP ゾーンの場合は、鍵情報をその排他的 IP ゾーンで構成します。

root 役割になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IPsec SA の鍵を生成します。**

鍵は、ipsecinit.conf ファイル内の特定のポリシーをサポートしている必要があります。たとえば、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」のポリシーを使用することもできます。

```
{laddr enigma raddr partym} ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
```

このポリシーは AES および SHA-2 アルゴリズムを使用します。

a. **必要な鍵を決定します。**

SA 用に、aes、sha512、および セキュリティーパラメータインデックス (SPI) の鍵を生成する必要があります。

- SPI の値として、2 つの 16 進数の乱数。1 つはアウトバウンドトラフィック用です。もう 1 つはインバウンドトラフィック用です。それぞれの乱数の最大桁数は 8 桁です。
- SHA-2 認証アルゴリズム用の 2 つの 16 進数の乱数。各数字の長さは 512 文字でなければいけません。1 つは dst enigma 用です。もう 1 つは dst partym 用です。
- AES 暗号化アルゴリズム用の 2 つの 16 進数の乱数。各数字の長さは 128 文字でなければいけません。1 つは dst enigma 用です。もう 1 つは dst partym 用です。

注記 - ipsecalgs -l コマンドは、アルゴリズムの鍵のサイズを表示します。手動鍵を使用する場合はこの手順に従い、SHA512 および AES アルゴリズムを使用してください。手動鍵には、弱いアルゴリズム、複合モードのアルゴリズム、または GMAC アルゴリズムは使用しないでください。

---

b. **必要な鍵を生成します。**

- 乱数発生関数がすでにある場合は、それを使用してください。
- 『Oracle Solaris 11.2 での暗号化と証明書の管理 』の「pktool コマンドを使用して対称鍵を生成する方法」とそのセクションの IPsec の例に従って、pktool コマンドを使用します。

2. **IPsec の手動鍵ファイルに鍵を追加します。**

a. **enigma システムの /etc/inet/secret/ipseckeys ファイルを次のように編集します。**

```
## ipseckeys - This file takes the file format documented in
##  ipseckey(1m).
#   Note that naming services might not be available when this file
#   loads, just like ipsecinit.conf.
#
#   Backslashes indicate command continuation.
#
```

```
# for outbound packets on enigma
add esp spi 0x8bcd1407 \
   src 192.168.116.16 dst 192.168.13.213  \
   encr_alg aes \
   auth_alg sha512  \
   encrkey  d41fb74470271826a8e7a80d343cc5aa... \
   authkey  e896f8df7f78d6cab36c94ccf293f031...
#
# for inbound packets
add esp spi 0x122a43e4 \
   src 192.168.13.213 dst 192.168.116.16 \
   encr_alg aes \
   auth_alg sha512  \
   encrkey dd325c5c137fb4739a55c9b3a1747baa... \
   authkey ad9ced7ad5f255c9a8605fba5eb4d2fd...
```

b. **読み取り専用ファイルを保護します。**

```
# chmod 400 /etc/inet/secret/ipseckeys
```

`pfedit -s` コマンドを使用して `ipseckeys` ファイルを作成した場合は、権限が正しく設定されています。詳細は、`pfedit`(1M) のマニュアルページを参照してください。

c. **ファイルの構文を確認します。**

```
# ipseckey -c /etc/inet/secret/ipseckeys
```

注記 - 2 つのシステムの鍵は同じ*でなければなりません*。

3. **IPsec の鍵をアクティブにします。**

■ **`manual-key` サービスが有効になっていない場合は有効にします。**

```
% svcs manual-key
STATE          STIME    FMRI
disabled       Apr_10   svc:/network/ipsec/manual-key:default
# svcadm enable ipsec/manual-key
```

■ **`manual-key` サービスが有効になっている場合はリフレッシュします。**

```
# svcadm refresh ipsec/manual-key
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその

他の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

▼ ネットワークセキュリティーの役割を構成する方法

Oracle Solaris の権利機能を使用してシステムを管理している場合は、ネットワーク管理の役割またはネットワークセキュリティーの役割を提供するためにこの手順を使用します。

始める前に 役割を作成して割り当てるには、root 役割になる必要があります。標準ユーザーは、使用可能な権利プロファイルの内容を一覧表示できます。

1. **使用可能なネットワーク関連の権利プロファイルを一覧表示します。**

```
% getent prof_attr | grep Network | more
...
Network Management:RO::Manage the host and network configuration...
Network Security:RO::Manage network and host security...:profiles=Network Wifi
Security,Network Link Security,Network IPsec Management...
Network Wifi Management:RO::Manage wifi network configuration...
Network Wifi Security:RO::Manage wifi network security...
Network Link Security:RO::Manage network link security...
Network IPsec Management:RO::Manage IPsec and IKE...
System Administrator:RO::Can perform most non-security administrative tasks:
profiles=...Network Management...
Information Security:RO::Maintains MAC and DAC security policies:
profiles=...Network Security...
```

Network Management プロファイルは、System Administrator プロファイルを補完するプロファイルです。System Administrator 権利プロファイルを役割に含めると、その役割は Network Management プロファイルでコマンドを実行できます。

2. **Network Management 権利プロファイル内のコマンドを一覧表示します。**

```
% profiles -p "Network Management" info
...
cmd=/usr/sbin/dladm
cmd=/usr/sbin/dlstat
...
cmd=/usr/sbin/svcadm
cmd=/usr/sbin/svccfg
cmd=/usr/sbin/dumpcap
```

3. **サイトでのネットワークセキュリティーの役割の範囲を決定します。**

決定には、ステップ 1 の権利プロファイルの定義を参考にしてください。

- すべてのネットワークセキュリティーを扱う役割を作成する場合は、Network Security 権利プロファイルを使用します。
- IPsec と IKE だけを扱う役割を作成するには、Network IPsec Management 権利プロファイルを使用します。
- ネットワーク管理とセキュリティーを処理する役割を作成するには、Network Management プロファイルに加えて、Network Security または Network IPsec Management 権利プロファイルを使用します。

4. **役割を作成して、その役割を 1 人または複数のユーザーに割り当てます。**

   手順については、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「役割の作成」および例7-7「信頼できるユーザーに IPsec を構成および管理する権限を与える」を参照してください。

例 7-5 ネットワーク管理およびセキュリティーの役割の作成と割り当て

この例では、管理者が Network Management と Network Security という 2 つの権利プロファイルを 1 つの役割に割り当てます。次に、管理者は信頼できるユーザーにこの役割を割り当てます。

```
# roleadd -c "Network Mgt and Security" \
-S ldap -K profiles="Network Management Plus" netmgtsec
# passwd netmgtsec
New Password: xxxxxxxx
Confirm password: xxxxxxxx
# usermod -R netmgtsec jdoe
```

ユーザー `jdoe` が `netmgtsec` 役割になると、`jdoe` はこれらのプロファイルの権利を利用できるようになります。

```
% su - netsecmgt
Password: xxxxxxxx
#
```

例 7-6 ネットワークセキュリティーの責任を役割に振り分ける

この例では、管理者がネットワークセキュリティーの責任を 2 つの役割に振り分けます。一方の役割は Wifi とリンクのセキュリティーを管理し、もう一方の役割は IPsec と IKE を管理します。各役割には、シフトごとに 1 人、合計 3 人を割り当てます。

管理者によって次のように役割が作成されます。

1. 最初の役割には LinkWifi という名前を付けます。

2. この役割には Network Wifi、Network Link Security、および Network Management 権利プロファイルを割り当てます。
3. 管理者は該当するユーザーにこの LinkWifi 役割を割り当てます。
4. 2 番目の役割には IPsec Administrator という名前を付けます。
5. この役割には Network IPsec Management および Network Management 権利プロファイルを割り当てます。
6. 管理者は該当するユーザーにこの IPsec Administrator 役割を割り当てます。

例 7-7　信頼できるユーザーに IPsec を構成および管理する権限を与える

この例では、管理者が 1 人のユーザーに IPsec を構成および管理する役割を与えます。

Network Management と IPsec Network Management の権利プロファイルに加えて、管理者はユーザーに `hosts` ファイルを編集する権限とそのログを読み取る権限を与えます。

1. 管理者は、ファイル編集用とログ読み取り用の 2 つの権利プロファイルを作成します。

```
# profiles -p -S LDAP "Hosts Configuration"
profiles:Network Configuration> set desc="Edits root-owned network files"
...Configuration> add auth=solaris.admin.edit/etc/hosts
...Configuration> commit
...Configuration> end
...Configuration> exit

# profiles -p -S LDAP "Read Network Logs"
profiles:Read Network Logs> set desc="Reads root-owned network log files"
...Logs> add cmd=/usr/bin/more
...Logs:more>set privs={file_dac_read}:/var/user/ikeuser/*
...Logs:more>set privs={file_dac_read}:/var/log/ikev2/*
...Logs:more> set privs={file_dac_read}:/etc/inet/ike/*
...Logs:more> set privs={file_dac_read}:/etc/inet/secret/*
...Logs:more>end
...Logs> add cmd=/usr/bin/tail
...Logs:tail>set privs={file_dac_read}:/var/user/ikeuser/*
...Logs:tail>set privs={file_dac_read}:/var/log/ikev2/*
...Logs:tail>set privs={file_dac_read}:/etc/inet/ike/*
...Logs:tail> set privs={file_dac_read}:/etc/inet/secret/*
...Logs:tail>end
```

```
...Logs> add cmd=/usr/bin/page
...Logs:page>set privs={file_dac_read}:/var/user/ikeuser/*
...Logs:page>set privs={file_dac_read}:/var/log/ikev2/*
...Logs:page>set privs={file_dac_read}:/etc/inet/ike/*
...Logs:page> set privs={file_dac_read}:/etc/inet/secret/*
...Logs:page>end
...Logs> exit
```

この権利プロファイルによって、ユーザーは more、tail、および page コマンドを使用してログを読み取れるようになります。cat コマンドと head コマンドは使用できません。

2. 管理者は、ユーザーが IPsec とその鍵サービスのすべての構成および管理タスクを実行できる権利プロファイルを作成します。

```
# profiles -p "Site Network Management"
profiles:Site Network Management> set desc="Handles all network files and logs"
...Management> add profiles="Network Management"
...Management> add profiles="Network IPsec Management"
...Management> add profiles="Hosts Configuraton"
...Management> add profiles="Read Network Logs"
...Management> commit; end; exit
```

3. 管理者はそのプロファイルの役割を作成し、役割にパスワードを割り当て、ネットワーキングとセキュリティーを理解している信頼できるユーザーにその役割を割り当てます。

```
# roleadd -S LDAP -c "Network Management Guru" \
-m -K profiles="Site Network Management" netadm
# passwd netadm
Password: xxxxxxxx
Confirm password: xxxxxxxx
# usermod -S LDAP -R +netadm jdoe
```

4. 帯域外で、管理者は jdoe に役割のパスワードを渡します。

▼ IPsec によってパケットが保護されていることを確認する方法

パケットが保護されていることを確認するには、snoop コマンドで接続をテストします。snoop 出力に表示される接頭辞は、次のとおりです。

- AH: 接頭辞は、AH がヘッダーを保護していることを示します。この接頭辞が表示されるのは、auth_alg を使ってトラフィックを保護している場合です。
- ESP: 接頭辞は、暗号化されたデータが送信されていることを示します。この接頭辞が表示されるのは、encr_auth_alg か encr_alg を使ってトラフィックを保護している場合です。

始める前に さらに、接続をテストするためには、両方のシステムにアクセスできなければなりません。

snoop の出力を作成するには、root 役割になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **partym など、1 つのシステム上で root 役割になります。**

```
% su -
Password: xxxxxxxx
#
```

2. **(オプション) SA の詳細を表示します。**

```
# ipseckey dump
```

この出力は、使用されている SA に一致する SPI 値、使用されていたアルゴリズム、鍵などを示します。

3. **このシステムで、リモートシステムからパケットをスヌープする準備をします。**

partym の端末ウィンドウで、enigma システムからパケットをスヌープします。

```
# snoop -d net0 -o /tmp/snoop_capture enigma
Using device /dev/e1000g (promiscuous mode)
```

4. **リモートシステムからパケットを送信します。**

別の端末ウィンドウで、enigma システムにリモートからログインします。パスワードを入力します。次に、root 役割になり、パケットを enigma システムから partym システムに送信します。パケットは、snoop -v enigma コマンドで取り込む必要があります。

```
partym% ssh enigma
Password: xxxxxxxx
enigma% su -
Password: xxxxxxxx
enigma# ping partym
```

5. **snoop の出力を調べます。**

```
partym# snoop -i /tmp.snoop_capture -v
```

snoop の出力を Wireshark アプリケーションにロードすることもできます。詳細は、212 ページの「IPsec および IKE システムのトラブルシューティングを準備する方法」および 235 ページの「snoop コマンドと IPsec」を参照してください。

このファイルで、冒頭の IP ヘッダー情報のあとに AH と ESP の情報が含まれている出力を確認します。次のような AH と ESP の情報は、パケットが保護されていることを示します。

```
IP:   Time to live = 64 seconds/hops
IP:   Protocol = 51 (AH)
IP:   Header checksum = 4e0e
IP:   Source address = 192.168.116.16, enigma
IP:   Destination address = 192.168.13.213, partym
IP:   No options
IP:
AH:  ----- Authentication Header -----
AH:
AH:  Next header = 50 (ESP)
AH:  AH length = 4 (24 bytes)
AH:  <Reserved field = 0x0>
AH:  SPI = 0xb3a8d714
AH:  Replay = 52
AH:  ICV = c653901433ef5a7d77c76eaa
AH:
ESP:  ----- Encapsulating Security Payload -----
ESP:
ESP:  SPI = 0xd4f40a61
ESP:  Replay = 52
ESP:     ....ENCRYPTED DATA....

ETHER:  ----- Ether Header -----
...
```

♦ ♦ ♦ 第 8 章

8 インターネット鍵交換について

インターネットキー交換 (Internet Key Exchange、IKE) は、Ipsec の鍵管理を自動化します。IKE について説明するこの章の内容は次のとおりです。

IKE プロトコルの最新バージョンを実装する手順については、第9章「IKEv2 の構成」を参照してください。IKEv1 の使用を続行するには、第10章「IKEv1 の構成」を参照してください。参照情報については、第12章「IPsec および鍵管理のリファレンス」を参照してください。IPsec については、第6章「IP セキュリティーアーキテクチャーについて」を参照してください。

IKE の概要

IPsec セキュリティーアソシエーション (SA) の鍵情報を管理することを「*鍵管理*」といいます。自動鍵管理には、鍵の作成、認証、および交換にセキュアな通信チャネルが必要です。Oracle Solaris では、インターネット鍵交換 (IKE) を使用して鍵管理を自動化します。IKE は、秘密鍵の手動配布に伴う管理オーバーヘッドとセキュリティーのリスクを排除します。

IKE では、ハードウェア暗号化アクセラレーションと鍵ストレージを利用できます。ハードウェア暗号化アクセラレータによって、CPU に負荷のかかる鍵操作をシステム外で処理できます。ハードウェア上での鍵の格納によって、保護機能が強化されます。

Oracle Solaris は、IKE プロトコルの 2 つのバージョンをサポートしています。

- インターネット鍵交換プロトコルバージョン 2 (IKEv2)、RFC 5996 に基づく IKE バージョン 2 (IKEv2)
- インターネット鍵交換 (IKE)、RFC 2409 に基づく IKE バージョン 1 (IKEv1)

IKE の概念および用語

次の概念と用語は IKE の両方のバージョンに共通です。2 つのバージョンでは、異なる形で実装されている可能性があります。

- **鍵のネゴシエーションおよび交換** – セキュアな方法で行われる、鍵情報の交換およびピアの識別情報の認証。このプロセスでは非対称暗号化アルゴリズムを使用します。主な 2 つの方法は、RSA と Diffie-Hellman プロトコルです。

  IKE は IKE デーモンを実行するシステム間の IPsec SA を作成および管理します。IKE は鍵情報の伝送を保護するセキュアなチャネルをネゴシエートします。デーモンは、`/dev/random` デバイスを使用して乱数発生関数から鍵を作成します。デーモンは、鍵を一定の割合 (構成可能) で変更します。この鍵情報は、IPsec ポリシーの構成ファイル `ipsecinit.conf` に指定されているアルゴリズムによって使用されます。

- **Diffie-Hellman (DH) アルゴリズム** – セキュアでないチャネルで 2 つのシステムが共有シークレットをセキュアに生成できる鍵交換アルゴリズム。

- **RSA アルゴリズム** – ピアシステムの識別情報の認証 (通常は X.509 証明書の所有権の提供による) に使用される非対称鍵アルゴリズム。アルゴリズム名は、作成者の Rivest、Shamir、Adleman の三氏に因んでいます。

  または、この目的に DSA または ECDSA アルゴリズムが使用されることもあります。

- **Perfect forward secrecy (PFS)** – PFS では、データ伝送を保護するために使用される鍵が、追加の鍵を導き出すために使用されることはありません。さらに、データ伝送を保護するために使用される鍵のソースが、追加の鍵を導き出すために使用されることはありません。したがって、PFS は以前に記録されたトラフィックの復号化を回避できます。

- **Oakley グループ** – PFS のネゴシエーションに使用されます。インターネット鍵交換 (IKE) RFC のセクション 6 を参照してください。

- **IKE ポリシー** – IKE デーモンがピアシステムとのセキュアな鍵交換チャネルの設定を試行する際に使用可能なパラメータを定義する IKE ルールのセット。これは、IKEv2 では IKE SA、IKEv1 ではフェーズ 1 と呼ばれます。

  パラメータには、アルゴリズム、鍵のサイズ、Oakley グループ、および認証方式が含まれます。Oracle Solaris IKE デーモンは認証方式として事前共有鍵および証明書をサポートしています。

IKE の動作

IKE デーモンを実行しているシステムは、このシステムと IKE デーモンを実行している別のシステムの間にセキュリティーアソシエーション (SA) を作成するのに必要なパラメータをネゴシエートできます。この SA および後続の IPsec SA のネゴシエーションに使用されるプロトコルは IKE と呼ばれます。Oracle Solaris のこのバージョンは、IKE プロトコルのバージョン 1 (IKEv1) およびバージョン 2 (IKEv2) をサポートしています。

IKE セキュリティーアソシエーション (IKEv1 では ISAKMP またはフェーズ 1 SA とも呼ばれる) は、これら 2 つの IKE システム間のさらなるプロトコル交換をセキュリティー保護します。これらの交換では、暗号化アルゴリズム、IPsec ポリシー、および IPsec SA の作成に必要なその他のパラメータをネゴシエートします。

IKE デーモンを実行しているシステムは、その他のシステムの代理として IPsec SA をネゴシエートするように構成することもできます。この方法で構成されたシステムは、*セキュリティーゲートウェイ*と呼ばれます。IKE ネゴシエーションが成功すると、IPsec SA を使用してネットワークパケットを保護できます。

注記 - Oracle Solaris 11.2 では、IKEv2 は FIPS 140-2 レベル 1 で検証された暗号化フレームワークの暗号化アルゴリズムを使用しますが、IKEv1 は使用しません。デフォルトでは、FIPS 140 は有効になっていません。2 つのバージョンの機能比較については、143 ページの「IKEv2 と IKEv1 の比較」を参照してください。FIPS 140-2 モードを有効にするには、『Oracle Solaris 11.2 での暗号化と証明書の管理 』の「FIPS 140 が有効になったブート環境を作成する方法」を参照してください。

FIPS 140-2 検証済みの暗号化の使用のみを認める厳密な要件がある場合は、Oracle Solaris 11.1 SRU 5.5 リリースまたは Oracle Solaris 11.1 SRU 3 リリースを実行している必要があります。Oracle は、これら 2 つのリリースの暗号化フレームワークに対する FIPS 140-2 の検証を完了しました。Oracle Solaris 11.2 は、この検証済みの基盤の上に構築され、パフォーマンス、機能、および信頼性に対処するソフトウェアの改良を含んでいます。これらの改良を活かすため、可能な場合はいつでも Oracle Solaris 11.2 を FIPS 140-2 モードで構成するようにしてください。

IKE SA を作成するためにネゴシエートされるパラメータには、IKE 交換および一部の認証情報を保護する暗号化アルゴリズムが含まれます。認証情報は、IKE プロトコル交換を含むパケットが信頼できるかどうかを判断するために使用されます。信頼とは、そのパケットが信頼できるシステムから来たもので、そのシステムになりすましているシステムから来たものではないことを意味します。

Oracle Solaris は、事前共有鍵と公開鍵証明書という 2 種類の IKE 認証情報をサポートしています。

IKE と事前共有鍵認証

事前共有鍵は、2 つの IKE システムのみが認識する 16 進数または ASCII 文字の文字列です。この鍵は、IKE 交換の前に両方のエンドポイントが鍵の値を認識している必要があることから、*事前共有鍵*と呼ばれます。この鍵は両方のシステムで IKE 構成に含まれている必要があります。事前共有鍵は IKE ペイロードの生成に使用され、IKE プロトコルを実装するパケットを作成します。これらの IKE ペイロードを処理するシステムは、受け取るペイロードの認証に同じ鍵を使用します。

事前共有鍵は、IKE エンドポイント間で IKE プロトコルを使用して交換されることはありません。通常、鍵は電話などの異なる媒体でピアシステムと共有されます。

この認証方法を使用するピアの事前共有鍵は同一である必要があります。鍵は各システム上のファイルに格納されます。

IKE と公開鍵証明書

公開鍵証明書およびそれらのトラストチェーンが、秘密の情報を手動で交換することなくシステムをデジタルに識別するメカニズムを提供します。そのため、公開鍵証明書は事前共有鍵よりもセキュアです。

公開鍵証明書は、公開鍵の値、名前や署名者など証明書の生成に関する情報、証明書のハッシュまたはチェックサム、およびハッシュのデジタル署名をエンコードするブロブデータです。これらの値が一体となって証明書を形成します。デジタル署名は証明書が変更されていないことを保証します。

公開鍵は、*非公開鍵*と呼ばれる別の値から数学的に生成される値です。非公開鍵から公開鍵を生成する数学的アルゴリズムでは、公開鍵から非公開鍵を取得することはできません。そのため、公開鍵証明書は自由に共有できます。公開鍵の生成に使用されるアルゴリズムの例として、RSA および楕円曲線があります。

デジタル署名とは、RSA、DSA、ECDSA などのデジタル署名アルゴリズムによって証明書の内容を渡した結果です。これらのアルゴリズムは、証明書に含まれない非公開署名鍵を使用してデジタル署名を作成します。署名は証明書に追加されます。この場合も、証明書の内容およ

び署名から署名鍵を計算することはできません。簡単に言うと、署名鍵から生成された公開値を使用して、証明書の署名、および証明書の内容を簡単に検証できます。

証明書は、自己署名することも (この場合、証明書の署名はその証明書の公開鍵で検証できる)、第三者が署名することもできます。第三者が証明書に署名する場合、証明書の検証に使用される公開鍵の値も公開鍵証明書として配布されます。この 2 つ目の証明書は、信頼される認証局 (CA)、または仲介者によって署名されます。この仲介者は最終的に署名機関、つまりルート CA またはトラストアンカーに信頼されます。

これら公開鍵証明書のコンポーネントとそれらを実装する手順および構造を合わせて、多くの場合、公開鍵インフラストラクチャー (PKI) と呼びます。組織の PKI の範囲は異なる場合があります。単純な PKI は、ローカル使用の少数の証明書に署名する CA で構成されていることがあります。より大きな PKI になると、グローバルに認知されているトラストアンカーを正式な CA として使用します。

IKE での公開鍵証明書の使用

このセクションでは、IKE での公開鍵証明書の作成および使用の全体的な手順を説明します。特定の手順については、150 ページの「事前共有鍵による IKEv2 の構成」および 178 ページの「事前共有鍵による IKEv1 の構成」を参照してください。

1. 自己署名付き証明書または認証局 (CA) による証明書を使用するには、まず公開鍵/非公開鍵ペアを生成します。
   - 自己署名付き証明書の場合は、次に IKE ピアがこれらの証明書を交換し、証明書が本物であることを帯域外で検証したあと、ピアの証明書をローカルのキーストアにインポートします。これにより、キーストアにオリジナルの自己署名付き証明書とインポートされた証明書が格納されます。
   - CA からの証明書の場合、さらにいくつかの手順を実行します。公開鍵/非公開鍵ペアを生成する際に、証明書署名要求 (CSR) も生成します。CSR には公開鍵および*識別子*が含まれています。一般的な識別子は、識別名 (DN)です。例:

     ```
     DN=”O=Example\, Inc, OU=qa, L=Silicon Valley, ST=CA, CN=enigma”
     ```

ヒント - 別の証明書の識別子と一致する可能性を低くするために、できるだけ固有の DN またはその他の識別子を作成します。

---

2. 署名のために CA に CSR を送信します。

一般的なプロセスでは、Web フォームに CSR を貼り付けて、そのフォームを CA に送信します。CA から複数の署名付き証明書が送信される場合があります。

3. CA から署名付き証明書を取得したあと、それらを IKEv2 キーストアまたは IKEv1 データベースにインポートします。

   CA から送信されるすべての証明書をインポートする必要があります。これらの証明書は、トラストアンカー (ルート CA) から個々に識別される署名付き証明書への「トラストチェーン」で構成されます。

4. IKE ピアでこの手順を繰り返します。
5. IKE ルール内の証明書を使用します。

   DN などの識別子で証明書を指定します。CA が署名する証明書の場合、特定の CA によって署名されている証明書を受け入れるように IKE を構成できます。

失効した証明書の処理

署名付き証明書は、署名機関がその有効性を保証するため、有効として信頼されます。証明書が損なわれた場合や無効と判断された場合は、CA によって取り消されます。

CA は、多くの場合証明書失効リスト (CRL) と呼ばれる失効した証明書のリストを保持しています。オンライン証明書ステータスプロトコル (OCSP) を使用して、証明書のステータスを動的に確認できます。一部の公開鍵証明書には URI が埋め込まれています。それらは CRL を確認できる Web の場所、または OCSP サーバーの Web の場所を識別します。

詳細は、RFC 2459: 証明書と CRL プロファイルおよび RFC 2560: オンライン証明書ステータスプロトコル - OCSP を参照してください。

公開証明書を使用するシステムでの時間の調整

公開鍵証明書には、発行日時および有効な残り時間が含まれています。そのため、証明書を生成および使用するシステム上のクロックが正確である必要があります。Network Time Protocol (NTP) ソフトウェアを使用して、システムのクロックを同期できます。Oracle Solaris のリリースにはデラウェア大学の NTP パブリックドメインソフトウェアが含まれています。ドキュメントは NTP ドキュメントの Web サイトで入手できます。`service/network/ptp` パッケージをインストールして高精度時間プロトコル (PTP) サービスを構成することもできます。ネットワーク接続された測定および制御システムの高精度時刻同期プロトコルのための IEEE 1588 規格を参照してください。

IKEv2 と IKEv1 の比較

次の表は、Oracle Solaris システムでの IKEv2 と IKEv1 のバージョンの実装を比較しています。

表 8-1 Oracle Solaris での IKEv2 および IKEv1 の実装

機能	IKEv2	IKEv1
証明書のトラストチェーン	キーストア内のオブジェクトに基づいて暗黙的	`ike/config` ファイルの `cert_trust` パラメータ
証明書の作成	`ikev2cert` コマンド	`ikecert certlocal` コマンド
証明書のインポート	`ikev2cert import` コマンドは PKCS #11 キーストアに証明書および鍵をインポートできます	`ikecert certdb` コマンドは IKE キーストアにスタンドアロンの証明書をインポートできます
証明書の所有者	`ikeuser`	`root`
証明書のポリシーファイル	`kmf-policy.xml`	`ike/config` ファイル内の一部のポリシー
証明書ストレージ	PKCS #11 ソフトトークンライブラリ	ローカルの IKEv1 データベース
構成ファイルのディレクトリ	`/etc/inet/ike/`	`/etc/inet/ike/` および `/etc/inet/secret/`
構成の所有者	`ikeuser` アカウント	`root` アカウント
デーモン	`in.ikev2d`	`in.iked`
デーモン間のトラフィック用の FIPS 140 アルゴリズム †	IKE SA は暗号化フレームワークを使用します	すべての交換で暗号化フレームワークが使用されるわけではありません
IPsec トラフィック用の FIPS 140 アルゴリズム †	暗号化フレームワークを使用します	暗号化フレームワークを使用します
IKE ポリシーファイル	`ike/ikev2.config`	`ike/config`
IKE 事前共有鍵	`ike/ikev2.preshared`	`secret/ike.preshared`
NAT ポート	UDP ポート `4500`	UDP ポート `4500`
ポート	UDP ポート `500`	UDP ポート `500`
権利プロファイル	Network IPsec Management	Network IPsec Management
サービス名 (FMRI)	`svc:/ipsec/ike:ikev2`	`svc:/ipsec/ike:default`

† Oracle Solaris 11.1 SRU 5.5 および SRU 3 の暗号化フレームワーク機能は、FIPS 140-2 レベル 1 で検証されています。FIPS 140 モードが有効であり、暗号化フレームワークが使用されている場合は、FIPS 140 検証済みアルゴリズムが使用されます。デフォルトでは、FIPS 140 モードは有効になっていません。

IKEv2 プロトコル

このセクションでは、IKEv2 の実装について説明します。IKEv1 については、145 ページの「IKEv1 プロトコル」を参照してください。比較については、143 ページの「IKEv2 と IKEv1 の比較」を参照してください。両方のプロトコルに適用される情報については、137 ページの「IKE の概要」を参照してください。Oracle Solaris は、IKE プロトコルの両方のバージョンを同時にサポートします。

IKEv2 デーモン `in.ikev2d` は、IPsec SA の鍵情報をネゴシエートして認証します。`in.ikev2d`(1M) のマニュアルページを参照してください。

IKEv2 構成の選択

`/etc/inet/ike/ikev2.config` 構成ファイルには、`in.ikev2d` デーモンの構成が含まれています。構成はいくつかのルールで構成されています。各エントリには、このシステムが同様に構成された IKEv2 ピアに対して使用できるアルゴリズムや認証データなどのパラメータが含まれます。

`in.ikev2d` デーモンは、事前共有鍵 (PSK) および識別情報の公開鍵証明書をサポートしています。

`ikev2.config`(4) のマニュアルページにサンプルルールが提供されています。各ルールには一意のラベルが必要です。次は、マニュアルページのサンプルルールの説明ラベルのリストです。

- `IP 識別情報および PSK 認証`
- `IP アドレス接頭辞および PSK 認証`
- `IPv6 アドレス接頭辞および PSK 認証`
- `DN 識別情報による証明書認証`
- `多くのピア ID タイプによる証明書認証`
- `ワイルドカードピア ID による証明書認証`
- `オーバーライド変換`
- `混合した認証タイプ`
- `ワイルドカードと要求される署名者`

注記 - 事前共有鍵は、IP アドレス、DN、FQDN、E メールアドレスなど多くのピア ID タイプのいずれでも使用できます。

公開証明書の IKEv2 ポリシー

`kmf-policy.xml` ファイルには IKEv2 の証明書検証ポリシーが含まれています。`kmfcfg dbfile=/etc/inet/ike/kmf-policy.xml policy=default` コマンドは、証明書検証ポリシーを変更するときに使用します。一般的な変更には、OCSP および CRL の使用、証明書検証中のネットワークタイムアウトの時間などがあります。また、管理者はポリシーによって有効日付の適用や鍵の使用要件など、証明書検証のさまざまな要素を変更できます。証明書検証のデフォルト要件をゆるめることはお勧めしません。

IKEv1 プロトコル

次のセクションでは、IKEv1 の概要について説明します。IKEv1 は、より高速でセキュリティー保護された鍵管理を提供する IKEv2 に置き換わっています。IKEv2 については、144 ページの「IKEv2 プロトコル」を参照してください。比較については、143 ページの「IKEv2 と IKEv1 の比較」を参照してください。両方のプロトコルに共通する情報については、137 ページの「IKE の概要」を参照してください。IKEv1 と IKEv2 は同時に実行可能で、ほかのシステム上のピアプロトコルとネゴシエートできます。

IKEv1 の鍵ネゴシエーション

IKEv1 デーモン `in.iked` は、セキュアな方法で鍵をネゴシエートして IPsec SA を認証します。IKEv1 は Perfect Forward Secrecy (PFS) を提供します。PFS では、データ伝送を保護する鍵を使用しないで追加鍵を取得します。また、データ伝送の鍵の作成に使用するシードを再利用しません。`in.iked`(1M) のマニュアルページを参照してください。

IKEv1 フェーズ 1 交換

IKEv1 プロトコルには 2 つのフェーズがあります。Oracle Solaris は、*メインモード*フェーズ 1 交換をサポートしています。メインモード交換は、2 つのピア間で ISAKMP セキュリティーアソシエーション (SA) を作成するのに使用できるパラメータをネゴシエートします。この ISAKMP SA は非対称の暗号化を使用して鍵情報を交換し、事前共有鍵または公開鍵証明書を使用してピアを認証します。IPsec SA とは異なり、ISAKMP SA は双方向であるため、1 つの SA だけ必要です。

フェーズ 1 交換での IKEv1 による ISAKAMP SA のネゴシエート方法は構成可能です。IKEv1 は、/etc/inet/ike/config ファイルから構成情報を読み取ります。次の構成情報があります。

- グローバルパラメータ (公開鍵証明書の名前など)
- Perfect Forward Secrecy (PFS) が必要かどうか
- このシステムの IKE ピア
- フェーズ 1 交換を保護するアルゴリズム
- 認証方式

  認証方式には、事前共有鍵と公開鍵証明書の 2 つがあります。公開鍵証明書は、自己署名することも認証局 (CA) に発行してもらうこともできます。

詳細は、ike.config(4) のマニュアルページを参照してください。

IKEv1 フェーズ 2 交換

フェーズ 2 交換は*クイックモード*と呼ばれます。クイックモード交換は、IPsec SA を作成するのに必要な IPsec アルゴリズムと鍵情報をネゴシエートします。この交換はフェーズ 1 でネゴシエートされる ISAKMP SA によって保護 (暗号化) されています。

クイックモード交換のアルゴリズムおよびセキュリティープロトコルは、IPsec ポリシーファイル /etc/inet/ipsecinit.conf から取得されます。

IPsec SA は期限が切れると鍵が再生成されます。SA のライフタイムは、IPsec SA の作成時に in.iked デーモンによって設定されます。この値は構成可能です。

詳細は、ipsecconf(1M) および in.iked(1M) のマニュアルページを参照してください。

IKEv1 構成の選択

/etc/inet/ike/config 構成ファイルには、in.iked デーモンの構成が含まれています。構成はいくつかのルールで構成されています。各エントリには、このシステムが同様に構成された IKEv1 ピアに対して使用できるアルゴリズムや認証データなどのパラメータが含まれます。in.iked デーモンは、事前共有鍵および識別情報の公開鍵証明書をサポートしています。

エントリ auth_method preshared は、事前共有鍵が使用されることを示します。auth_method の値が preshared 以外の場合には、公開鍵証明書が使用されることを示します。

IKEv1 では、事前共有鍵は特定の IP アドレスまたはアドレス範囲に関連付けられています。鍵は、各システムの /etc/inet/secret/ike.preshared ファイルに保存されます。

詳細は、139 ページの「IKE の動作」および ike.config(4) と ike.preshared(4) のマニュアルページを参照してください。

♦ ♦ ♦ 第 9 章

9 IKEv2 の構成

この章では、使用するシステムに合わせてインターネット鍵交換バージョン 2 (IKEv2) を構成する方法について説明します。IKEv2 を構成して有効にすると、指定された IPsec エンドポイントの鍵情報が自動的に生成されます。この章では、次の内容について説明します。

- 149 ページの「IKEv2 の構成」
- 150 ページの「事前共有鍵による IKEv2 の構成」
- 156 ページの「IKEv2 の公開鍵証明書を格納するためのキーストアの初期化」
- 150 ページの「事前共有鍵による IKEv2 の構成」

IKE の概要については、第8章「インターネット鍵交換について」を参照してください。IKE の参照情報については、第12章「IPsec および鍵管理のリファレンス」を参照してください。詳細な手順については、`ikeadm`(1M)、`pktool`(1)、`ikev2cert`(1M)、`ikev2.config`(4)、`in.ikev2d`(1M)、および `kmfcfg`(1) のマニュアルページの例を参照してください。

IKEv2 の構成

IKE の認証には、事前共有鍵、自己署名付き証明書、および認証局 (CA) の証明書を使用できます。ルールによって、特定の認証方法と保護されているエンドポイントが関連付けられます。したがって、1 つのシステムに 1 つまたはすべての認証方法を使用できます。IKEv2 システムで IKEv1 を実行することもできます。通常は、IKEv2 をサポートしていないシステムで通信を保護するために IKEv1 を実行します。IKEv2 は、鍵および証明書のストレージに PKCS #11 ハードウェアトークンを使用することもできます。

注記 - これらのタスクは、システムに静的 IP アドレスがアサイン済みで、ネットワーク構成プロファイル `DefaultFixed` を実行していることを前提としています。`netadm list` コマンドが `Automatic` を返す場合は、`netcfg`(1M) のマニュアルページで詳細を確認してください。

IKEv2 を構成したら、第7章「IPsec の構成」で、これらの IKEv2 ルールを使用して鍵を管理する IPsec 手順を実行してください。次のセクションでは、具体的な IKEv2 構成に注目します。

事前共有鍵による IKEv2 の構成

IKEv2 を使用するようにピアシステムまたはサブネットを構成し、さらに、あなたがこれらのサブネットの管理者である場合、事前共有鍵の使用が適しています。事前共有鍵は、テストの際にも使用される場合があります。詳細は、140 ページの「IKE と事前共有鍵認証」を参照してください。

▼ 事前共有鍵で IKEv2 を構成する方法

この手順では、名前 `enigma` と `partym` を使用しているシステムの名前に置き換えてください。両方の IKE エンドポイントを構成します。

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「`ssh` 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **各システムで `/etc/inet/ike/ikev2.config` ファイルを編集します。**

```
# pfedit /etc/inet/ike/ikev2.config
```

2. **ファイル内に、事前共有鍵を使用するルールを作成します。**

注記 - ステップ 4 で鍵を作成します。

このファイルのルールおよびグローバルパラメータは、システムの `ipsecinit.conf` ファイル内の IPsec ポリシーの鍵を管理する必要があります。次の IKEv2 構成の例で

は、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」の `ipsecinit.conf` の例の鍵を管理します。

a. **たとえば、`enigma` システムの `ikev2.config` ファイルを変更します。**

注記 - この例では、グローバルパラメータセクションの 2 つの変換を示します。ピアはこれらの変換のいずれかで構成できます。特定の変換を要求するには、その変換をルールに含めます。

```
### ikev2.config file on enigma, 192.168.116.16

## Global parameters
# This default value will apply to all transforms that follow
#
ikesa_lifetime_secs 3600
#
# Global transform definitions.  The algorithm choices are
# based on RFC 4921.
#
## Two transforms are acceptable to this system, Group 20 and Group 19.
## A peer can be configured with 19 or 20.
## To ensure that a particular peer uses a specific transform,
## include the transform in the rule.
##
# Group 20 is 384-bit ECP - Elliptic Curve over Prime
ikesa_xform { encr_alg aes(256..256) auth_alg sha384 dh_group 20 }
# Group 19 is 256-bit ECP
ikesa_xform { encr_alg aes(128..128) auth_alg sha256 dh_group 19 }
#
## The rule to communicate with partym
##  Label must be unique
{ label "enigma-partym"
  auth_method preshared
  local_addr  192.168.116.16
  remote_addr 192.168.13.213
}
```

b. **`partym` システムの `ikev2.config` ファイルを変更します。**

```
## ikev2.config file on partym, 192.168.13.213
## Global Parameters
#
...
ikesa_xform { encr_alg aes(256..256) auth_alg sha384 dh_group 20 }
ikesa_xform { encr_alg aes(128..128) auth_alg sha256 dh_group 19 }
...
## The rule to communicate with enigma
##  Label must be unique
{ label "partym-enigma"
  auth_method preshared
  local_addr  192.168.13.213
```

```
    remote_addr 192.168.116.16
}
```

3. **各システムで、ファイルの構文を検証します。**

```
# /usr/lib/inet/in.ikev2d -c
```

4. **各システムの `/etc/inet/ike/ikev2.preshared` ファイルに事前共有鍵を追加します。**

注意 - このファイルは特殊な権限を持ち、`ikeuser` によって所有されています。このファイルは決して削除したり置き換えたりしないでください。代わりに、ファイルが元のプロパティーを保持できるように `pfedit` コマンドを使用して内容を編集してください。

a. **たとえば、`enigma` システムの `ikev2.preshared` ファイルは次のようになります。**

```
# pfedit -s /etc/inet/ike/ikev2.preshared
## ikev2.preshared on enigma, 192.168.116.16
#…
## label must match the rule that uses this key
{ label "enigma-partym"
## The preshared key can also be represented in hex
## as in 0xf47cb0f432e14480951095f82b
   key "This is an ASCII Cqret phrAz, use str0ng p@ssword tekniques"
}
```

`pfedit` コマンドのオプションについては、`pfedit`(1M) のマニュアルページを参照してください。

b. **`partym` システムでは、`ikev2.preshared` ファイルは固有のラベル以外はほぼ同じです。**

```
## ikev2.preshared on partym, 192.168.13.213
#…
## label must match the label of the rule that uses this key
{ label "partym-enigma"
## The preshared key can also be represented in hex
## as in 0xf47cb0f432e14480951095f82b
 key "This is an ASCII Cqret phrAz, use str0ng p@ssword tekniques"
 }
```

5. **IKEv2 サービスインスタンスを有効にします。**

```
# svcadm enable ipsec/ike:ikev2
```

事前共有鍵を置き換えるときは、ピアシステムで事前共有鍵ファイルを編集して `ikev2` サービスを再起動します。

```
# svcadm restart ikev2
```

例 9-1　ローカルとリモートで異なる IKEv2 共有鍵を使用する

この例では、IKEv2 の管理者がシステムごとに事前共有鍵を作成してそれらを交換し、各鍵を事前共有鍵ファイルに追加します。事前共有鍵エントリのラベルは `ikev2.config` ファイル内のルールのラベルと一致します。その後、彼らは `in.ikev2d` デーモンを再起動します。

相手システムの事前共有鍵を受け取ったあと、管理者は `ikev2.preshared` ファイルを編集します。`partym` のファイルは次のとおりです。

```
# pfedit -s /etc/inet/ike/ikev2.preshared
#…
{ label "partym-enigma"
## local and remote preshared keys
local_key  "P-LongISH key Th@t m^st Be Ch*angEd \'reguLarLy)"
remote_key "E-CHaNge lEyeGhtB+lBs et KeeS b4 2LoOoOoOoOng"
}
```

したがって、`enigma` の `ikev2.preshared` 鍵ファイルは次のようになります。

```
#…
{ label "enigma-partym"
## local and remote preshared keys
local_key  "E-CHaNge lEyeGhtB+lBs et KeeS b4 2LoOoOoOoOng"
remote_key "P-LongISH key Th@t m^st Be Ch*angEd \'reguLarLy)"
}
```

管理者は各システムで IKEv2 サービスインスタンスを再起動します。

```
# svcadm restart ikev2
```

次の手順　IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

その他の例については、`ikev2.config`(4) および `ikev2.preshared`(4) のマニュアルページを参照してください。

▼ IKEv2 で事前共有鍵を使用する場合に新しいピアを追加する方法

同じピア間で動作中の構成に対して IPsec ポリシーエントリを追加した場合、IPsec ポリシーサービスをリフレッシュする必要があります。IKE の再構成または再起動は不要です。

IPsec ポリシーに新しいピアを追加した場合、IPsec の変更に加えて IKEv2 構成を変更する必要があります。

始める前に `ipsecinit.conf` ファイルをリフレッシュし、ピアシステムの IPsec ポリシーをリフレッシュしました。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「`ssh` 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **IPsec を使用する新規システム用の鍵を管理するための IKEv2 のルールを作成します。**

   a. **たとえば、`enigma` システムで、次の規則を `/etc/inet/ike/ikev2.config` ファイルに追加します。**

   ```
   # pfedit ikev2.config
   ## ikev2.config file on enigma, 192.168.116.16
   ...
   ## The rule to communicate with ada
   ##  Label must be unique
   {label "enigma-ada"
    auth_method preshared
    local_addr  192.168.116.16
    remote_addr 192.168.15.7
   }
   ```

   `pfedit` コマンドのオプションについては、`pfedit`(1M) のマニュアルページを参照してください。

   b. **`ada` システムで、次の規則を追加します。**

```
## ikev2.config file on ada, 192.168.15.7
...
## The rule to communicate with enigma
{label "ada-enigma"
 auth_method preshared
 local_addr  192.168.15.7
 remote_addr 192.168.116.16
}
```

2. **(オプション) 各システムで、ファイルの構文を検証します。**

```
# /usr/lib/inet/in.ikev2d -c -f /etc/inet/ike/ikev2.config
```

3. **ピアシステム用の IKEv2 事前共有鍵を作成します。**

   a. **enigma システムで、次の情報を /etc/inet/ike/ikev2.preshared ファイルに追加します。**

```
# pfedit -s /etc/inet/ike/ikev2.preshared
## ikev2.preshared on enigma for the ada interface
...
## The rule to communicate with ada
##  Label must match the label of the rule
{ label "enigma-ada"
  # enigma and ada's shared key
  key "Twas brillig and the slivey toves did *s0mEtHiNg* be CareFULL hEEEr"
}
```

   pfedit コマンドのオプションについては、pfedit(1M) のマニュアルページを参照してください。

   b. **ada システムで、次の情報を ikev2.preshared ファイルに追加します。**

```
# ikev2.preshared on ada for the enigma interface
#
{ label "ada-enigma"
  # ada and enigma's shared key
  key "Twas brillig and the slivey toves did *s0mEtHiNg* be CareFULL hEEEr"
}
```

4. **各システムで、変更をカーネルに読み込みます。**

   ■ **サービスが有効になっている場合はリフレッシュします。**

```
# svcadm refresh ikev2
```

   ■ **サービスが有効になっていない場合は有効にします。**

```
# svcadm enable ikev2
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

IKEv2 の公開鍵証明書を格納するためのキーストアの初期化

IKEv2 で公開証明書を使用するには、PKCS #11 キーストアを作成する必要があります。もっとも一般的に使用されるキーストアは、Oracle Solaris の暗号化フレームワーク機能が提供する `pkcs11_softtoken` を使用します。

IKEv2 の `pkcs11_softtoken` キーストアは、特殊なユーザー `ikeuser` が所有するディレクトリにあります。デフォルトのディレクトリは `/var/user/ikeuser` です。ユーザー ID `ikeuser` はシステムに用意されていますが、キーストアは自分で作成する必要があります。キーストアを作成する場合は、キーストアの PIN を作成します。IKEv2 サービスがキーストアにログインするにはこの PIN が必要です。

`pkcs11_softtoken` キーストアは、IKEv2 で使用される非公開鍵、公開鍵、および公開証明書を保持しています。これらの鍵および証明書は、`pktool` コマンドのラッパーである `ikev2cert` コマンドで管理されます。このラッパーは、`ikeuser` が所有する `pkcs11_softtoken` キーストアにすべての鍵および証明書の操作が適用されるようにします。

PIN を `ikev2` サービスのプロパティー値として追加していない場合は、`/var/log/ikev2/in.ikev2d.log` ファイルに次のメッセージが表示されます。

```
date: (n)  No PKCS#11 token "pin" property defined
for the smf(5) service: ike:ikev2
```

公開鍵証明書を使用していない場合は、このメッセージを無視してかまいません。

▼ IKEv2 公開鍵証明書用キーストアを作成および使用する方法

IKEv2 で公開証明書を使用する予定がある場合は、キーストアを作成する必要があります。キーストアを使用するには、ログインする必要があります。in.ikev2d デーモンが起動したら、ユーザーまたは自動プロセスがデーモンに PIN を提供します。サイトのセキュリティーで自動ログインを許可している場合は、それを構成する必要があります。デフォルトはキーストアを使用するための対話型ログインです。

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **IKEv2 キーストアに PIN を設定します。**

   ikev2cert setpin コマンドを使用して IKEv2 キーストアを作成します。このコマンドは、PKCS #11 キーストアの所有者を ikeuser に設定します。

   PIN に空白を使用しないでください。たとえば、WhatShouldIWrite は有効ですが、「What Should」は有効ではありません。

```
%  pfbash
# /usr/sbin/ikev2cert setpin
Enter token passphrase: changeme
Create new passphrase:      Type strong passphrase
Re-enter new passphrase: xxxxxxxx
Passphrase changed.
```

注意 - このパスフレーズを安全な場所に保管してください。キーストアを使用するにはそれが必要です。

2. **自動的に、または対話形式でキーストアにログインします。**

   自動ログインが推奨されます。サイトのセキュリティーポリシーが自動ログインを許可していない場合は、in.ikev2d デーモンを再起動する際に対話形式でキーストアにログインする必要があります。

   ■ **自動ログインを有効にするようにキーストアを構成します。**

   a. **PIN を pkcs11_softtoken/pin サービスプロパティーの値として追加します。**

```
# svccfg -s ike:ikev2 editprop
```

一時編集ウィンドウが開きます。

b. **setprop pkcs11_token/pin = 行のコメントを解除します。**

```
# setprop pkcs11_token/pin = astring: ()       Original entry
setprop pkcs11_token/pin = astring: ()  Uncommented entry
```

c. **括弧を ステップ 1 の PIN に置き換えます。**

```
setprop pkcs11_token/pin = astring: PIN-from-Step-1
```

コロンと PIN の間にスペースを残します。

d. **ファイル最下部の `refresh` 行のコメントを解除して、変更を保存します。**

```
# refresh
refresh
```

e. **(オプション) `pkcs11_token/pin` プロパティーの値を検証します。**

`pkcs11_token/pin` プロパティーは、`ikeuser` 所有のキーストアにアクセスするときに確認される値を保持します。

```
# svccfg -s ike:ikev2 listprop pkcs11_token/pin
pkcs11_token/pin     astring       PIN
```

■ **自動キーストアログインが構成されていない場合は、キーストアに手動でログインします。**

`in.ikev2d` デーモンが起動するたびにこのコマンドを実行します。

```
# pfbash
# ikeadm -v2 token login "Sun Metaslot"
Enter PIN for PKCS#11 token 'Sun Metaslot':      Type the PIN from Step 1
ikeadm: PKCS#11 operation successful
```

3. **(オプション) キーストアに PIN が設定されていることを確認します。**

```
# ikev2cert tokens
Flags: L=Login required  I=Initialized  X=User PIN expired  S=SO PIN expired
Slot ID     Slot Name                   Token Name                      Flags
-------     ---------                   ----------                      -----
1           Sun Crypto Softtoken        Sun Software PKCS#11 softtoken    LI
```

`Flags` 列の `LI` は、PIN が設定されていることを示します。

4. **pkcs11_softtoken を手動でログアウトするには、ikeadm コマンドを使用します。**

```
# ikeadm -v2 token logout "Sun Metaslot"
ikeadm: PKCS#11 operation successful
```

2 つのサイト間の通信を一定の時間に制限するためにログアウトする場合があります。ログアウトによって秘密鍵が使用できなくなるため、新しい IKEv2 セッションを開始できません。既存の IKEv2 セッションは、`ikeadm delete ikesa` コマンドでセッション鍵を削除しないかぎり続きます。事前共有鍵ルールは引き続き機能します。`ikeadm`(1M) のマニュアルページを参照してください。

公開鍵証明書による IKEv2 の構成

公開証明書は大規模な配備に適しています。詳細については、140 ページの「IKE と公開鍵証明書」を参照してください。

公開鍵証明書は、暗号化フレームワークによってソフトトークンキーストアに格納されます。ハードウェアが接続されているシステムでは、証明書をハードウェア上で生成および格納することもできます。手順については、173 ページの「ハードウェア上で IKEv2 の公開鍵証明書を生成および格納する方法」を参照してください。

背景情報については、139 ページの「IKE の動作」を参照してください。

次のタスクマップは、IKEv2 の公開鍵証明書の作成手順を一覧表示したものです。この手順には、システムに Sun Crypto Accelerator 6000 ボードが接続されている場合にハードウェアキーストアに証明書を格納する方法が含まれています。

表 9-1 公開鍵証明書による IKEv2 の構成のタスクマップ

タスク	説明	参照先
証明書のキーストアを作成します。	IKEv2 の証明書を格納する PKCS #11 キーストアを初期化します。	156 ページの「IKEv2 の公開鍵証明書を格納するためのキーストアの初期化」
自己署名付き公開鍵証明書で IKEv2 を構成します。	自分で署名した公開鍵証明書を作成します。証明書をピアにエクスポートし、ピアの証明書をインポートします。	160 ページの「自己署名付き公開鍵証明書により IKEv2 を構成する方法」
CA の証明書で IKEv2 を構成します。	CSR を作成して、返されたすべての証明書をキーストアにインポートする必要があります。その後、IKE ピアの証明書を検証してインポートします。	166 ページの「CA の署名付き証明書で IKEv2 を構成する方法」

タスク	説明	参照先
失効した証明書の処理方法を構成します。	CRL が使用されているか、および OCSP サーバーがポーリングされているか (ネットワーク遅延の処理方法を含む) を確認します。	169 ページの「IKEv2 で証明書検証ポリシーを設定する方法」
接続されたハードウェアのキーストアに証明書のストレージを構成します。	Sun Crypto Accelerator 6000 ボードを取り付けて、それを使用するように IKEv2 を構成します。	173 ページの「ハードウェア上で IKEv2 の公開鍵証明書を生成および格納する方法」

▼ 自己署名付き公開鍵証明書により IKEv2 を構成する方法

この手順では、公開鍵証明書を作成して署名します。非公開鍵および証明書は、IKEv2 の PKCS #11 ソフトトークンキーストアに格納されます。公開鍵証明書を IKE ピアに送信し、次に相手の公開証明書を送信してもらいます。

自己署名付き証明書を使用するすべての IKE システムでこの手順を実行します。

始める前に 証明書を使用するには、157 ページの「IKEv2 公開鍵証明書用キーストアを作成および使用する方法」を完了している必要があります。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルを使用している必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **キーストアに自己署名付き証明書を作成します。**

   `ikev2cert gencert` コマンドの引数の説明については、`pktool`(1) のマニュアルページで `pktool gencert keystore=pkcs11` サブコマンドを確認してください。

   `subject` 引数の形式については、141 ページの「IKE での公開鍵証明書の使用」を参照してください。

   注記 - 証明書にラベルを付けます。ラベルはローカルキーストア内で証明書およびそれに対応する鍵を識別します。

a. **たとえば、partym システムでは、コマンドは次のようになります。**

```
# pfbash
 # ikev2cert gencert \
 label="ITpartym" \
 subject="O=exampleco, OU=IT, C=US, CN=partym" \
 serial=0x87654321
    keytype=rsa
    keylen=2048
 Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
```

次のエラーメッセージは、PIN の入力ミスまたはキーストアが初期化されていないことを示しています。

```
Error creating certificate and keypair:
 keystore error: CKR_PIN_INCORRECT
 libkmf error: KMF_ERR_AUTH_FAILED

 Error creating certificate and keypair:
 keystore error: CKR_PIN_EXPIRED: PIN expired and must be changed
 libkmf error: KMF_ERR_BAD_PARAMETER: invalid parameter
```

ヒント - pktool コマンド構文の表示は、証明書エントリの一部が誤って入力されていることを示します。許可されていないアルゴリズムの使用、二重引用符と等号の抜け、およびその他の入力ミスがないかコマンドを確認してください。無効な引数を見つける 1 つの方法として、コマンドを取得したあと、引数を 1 回につき 1 つずつ削除します。

---

b. **enigma システムでは、コマンドは次のようになります。**

```
# ikev2cert gencert \
 label=ITenigma \
 subject="O=exampleco, OU=IT, C=US, CN=enigma" \
 serial=0x86428642
    keytype=rsa
    keylen=2048
 Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
```

2. **(オプション) 鍵および証明書を一覧表示します。**

```
enigma # /usr/sbin/ikev2cert list objtype=both
 Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
 No.     Key Type        Key Len.        Key Label
 -----------------------------------------------------
 Asymmetric private keys:
 1)      RSA                             ITenigma
 Asymmetric public keys:
 1)      RSA                             ITenigma
 Certificates:
 1) X.509 certificate
```

```
Label: ITenigma
Subject: C=US, O=exampleco, OU=IT, CN=enigma
Issuer: C=US, O=exampleco, OU=IT, CN=enigma
Not Before: April 10 21:49:00 2014 GMT
Not After: April 10 21:49:00 2015 GMT
Serial: 0x86426420
Signature Algorithm: sha1WithRSAEncryption
X509v3 Subject Key Identifier:
 34:7a:3b:36:c7:7d:4f:60:ed:ec:4a:96:33:67:f2:ac:87:ce:35:cc
SHA1 Certificate Fingerprint:
 68:07:48:65:a2:4a:bf:18:f5:5b:95:a5:01:42:c0:26:e3:3b:a5:30
```

ヒント - デフォルトのハッシュアルゴリズムは SHA1 です。強力な署名アルゴリズムで証明書を作成するには、keytype オプションと、keytype=rsa hash=sha384 などの別のハッシュアルゴリズムを使用します。オプションについては、pktool(1) のマニュアルページを参照してください。

3. **相手システムに証明書を送信します。**

   a. **各システムで、証明書だけをファイルにエクスポートします。**

   outformat=pem オプションを使用すると、公開証明書が直接インポートに適した形式のファイルに書き出されます。ラベルはキーストア内で証明書を識別します。

```
# cd /tmp
 # ikev2cert export objtype=cert outformat=pem outfile=filename label=label
 Enter PIN for Sun Software PKCS#11 softtoken:xxxxxxxx
```

   b. **証明書を電子メール、sftp、または ssh でほかのシステムに送信します。**

   たとえば、両方のシステムを管理している場合は sftp コマンドを使用して相手システムの証明書を取得します。

```
enigma # sftp jdoe@partym:/tmp/ITpartym.pem /tmp/ITpartym.pem.cert
 partym # sftp jdoe@enigma:/tmp/ITenigma.pem /tmp/ITenigma.pem.cert
```

   パスワードを入力するよう求められます。この例では、jdoe にパスワードを指定してください。

4. **証明書が同一であることを確認します。**

   キーストアに証明書をロードする*前に*、正しい証明書を受け取っていることを確認する必要があります。

   a. **各システムにエクスポートされたファイルのダイジェストを作成します。**

たとえば、partym の管理者が、partym の証明書を含むファイルのダイジェストを相手の管理者に電子メールで送信します。enigma の管理者は、enigma の証明書ファイルのダイジェストを電子メールで送信します。

```
partym # digest -a sha1 /tmp/ITpartym.pem
 c6dbef4136c0141ae62110246f288e5546a59d86

 enigma # digest -a sha1 ITenigma.pem
 6b288a6a6129d53a45057065bd02b35d7d299b3a
```

b. **相手のシステムで、最初のシステムの証明書を含むファイルに digest コマンドを実行します。**

```
enigma # digest -a sha1 /tmp/ITpartym.pem.cert
 c6dbef4136c0141ae62110246f288e5546a59d86

 partym # digest -a sha1 /tmp/ITenigma.pem.cert
 6b288a6a6129d53a45057065bd02b35d7d299b3a
```

ダイジェストは一致する必要があります。一致しない場合は、ファイルをキーストアにインポートしないでください。証明書の妥当性を検証する別の方法については、例9-3「公開鍵証明書をそのフィンガープリントで検証する」を参照してください。

5. **検証のあと、相手システムの証明書を自身のキーストアにインポートします。**

証明書をキーストアにインポートする場合、システム上で証明書を一意に識別するラベルを証明書に割り当てる必要があります。ラベルは、公開鍵とその公開鍵証明書を関連付けます。

```
enigma# ikev2cert import label=ITpartym1 infile=/tmp/ITpartym.pem.cert

partym# ikev2cert import label=ITenigma1 infile=/tmp/ITenigma.pem.cert
```

6. **(オプション) キーストア内のオブジェクトを一覧表示します。**

そのリストと ステップ 2 のリストを比較します。たとえば、enigma キーストアに partym の証明書が追加されています。

```
enigma # /usr/sbin/ikev2cert list objtype=both
 Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
 No.     Key Type        Key Len.        Key Label
 -----------------------------------------------------
 Asymmetric private keys:
 1)      RSA                             ITenigma
 Asymmetric public keys:
 1)      RSA                             ITenigma
 Certificates:
 1) X.509 certificate
  Label: ITenigma
```

```
 Subject: C=US, O=exampleco, OU=IT, CN=enigma
 Issuer: C=US, O=exampleco, OU=IT, CN=enigma
 Not Before: April 10 21:49:00 2014 GMT
 Not After: April 10 21:49:00 2015 GMT
 Serial: 0x86426420
 Signature Algorithm: sha1WithRSAEncryption
 X509v3 Subject Key Identifier:
  34:7a:3b:36:c7:7d:4f:60:ed:ec:4a:96:33:67:f2:ac:87:ce:35:cc
 SHA1 Certificate Fingerprint:
  68:07:48:65:a2:4a:bf:18:f5:5b:95:a5:01:42:c0:26:e3:3b:a5:30

2) X.509 certificate
 Label: ITpartym1
 Subject: C=US, O=exampleco, OU=IT, CN=partym
 Issuer: C=US, O=exampleco, OU=IT, CN=partym
 Not Before: April 10 21:40:00 2014 GMT
 Not After: April 10 21:40:00 2015 GMT
 Serial: 0x87654321
 Signature Algorithm: sha1WithRSAEncryption
 X509v3 Subject Key Identifier:
  ae:d9:c8:a4:19:68:fe:2d:6c:c2:9a:b6:06:55:b5:b5:d9:d9:45:c6
 SHA1 Certificate Fingerprint:
  83:26:44:29:b4:1f:af:4a:69:0d:87:c2:dc:f4:a5:1b:4f:0d:36:3b
```

7. **各システムで、証明書を IKEv2 ルールで使用します。**

   pfedit コマンドを使用して /etc/inet/ike/ikev2.config ファイルを編集します。

   a. **たとえば、partym システムでは、ikev2.config ファイルのルールは次のように表示されます。**

   ```
   ##  ... Global transform that applies to any rule without a declared transform
    ikesa_xform { dh_group 21 auth_alg sha512 encr_alg aes }
    ##  ... Any self-signed
    ## end-entity certificates must be present in the keystore or
    ## they will not be trusted.
    {
     label "partym-enigma"
     auth_method cert
     local_id DN = "O=exampleco, OU=IT, C=US, CN=partym"
     remote_id DN = "O=exampleco, OU=IT, C=US, CN=enigma"
    }
    ...
   ```

   b. **enigma システムでは、ikev2.config ファイルの local_id の値に enigma の証明書の DN を使用します。**

      リモートパラメータには、partym の証明書の DN をその値として使用します。label キーワードの値がローカルシステム上で一意であることを確認します。

      …

```
ikesa_xform { dh_group 21 auth_alg sha512 encr_alg aes }
…
{
 label "enigma-partym"
 auth_method cert
 local_id DN = "O=exampleco, OU=IT, C=US, CN=enigma"
 remote_id DN = "O=exampleco, OU=IT, C=US, CN=partym"
}
...
```

8. **(オプション) 各システムで、`ikev2.config` ファイルの妥当性を確認します。**

```
# /usr/lib/inet/inikev2.d -c
```

続行する前に、入力ミスや不正確な記述を修正してください。

9. **各システムで、IKEv2 サービスインスタンスの状態を確認します。**

```
# svcs ikev2
 STATE          STIME    FMRI
 disabled       Sep_07   svc:/network/ipsec/ike:ikev2
```

10. **各システムで、IKEv2 サービスインスタンスを有効にします。**

```
partym # svcadm enable ipsec/ike:ikev2

 enigma # svcadm enable ipsec/ike:ikev2
```

例 9-2 ライフタイムの制限を付けた自己署名付き証明書を作成する

この例では、証明書が 2 年間有効であることを管理者が指定します。

```
# ikev2cert gencert \
  > label=DBAuditV \
  > serial=0x12893467235412 \
  > subject="O=exampleco, OU=DB, C=US, CN=AuditVault" \
  > altname=EMAIL=auditV@example.com \
  > keytype=ec curve=secp521r1 hash=sha512 \
     > lifetime=2-year
```

例 9-3 公開鍵証明書をそのフィンガープリントで検証する

この例では、管理者が証明書のフィンガープリントを使用して証明書を検証します。この方法のデメリットは、管理者がフィンガープリントを表示する前に、ピアの証明書をキーストアにインポートする必要があることです。

管理者は証明書をインポートして、ikev2cert list objtype=cert コマンドでそれを一覧表示したあと、その出力から証明書のフィンガープリントをコピーして相手のシステム管理者に送信します。

```
SHA1 Certificate Fingerprint:
            83:26:44:29:b4:1f:af:4a:69:0d:87:c2:dc:f4:a5:1b:4f:0d:36:3b
```

検証が失敗した場合は、証明書をインポートした管理者が証明書とその公開鍵をキーストアから削除する必要があります。

```
# ikev2cert delete label=label-name
       Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
       1 public key(s) found, do you want to delete them (y/N) ? y
       1 certificate(s) found, do you want to delete them (y/N) ? y
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

▼ CA の署名付き証明書で IKEv2 を構成する方法

多くの通信システムを保護する組織は、通常、認証局 (CA) による公開証明書を使用します。背景情報については、140 ページの「IKE と公開鍵証明書」を参照してください。

CA の証明書を使用するすべての IKE システムでこの手順を実行します。

始める前に 証明書を使用するには、157 ページの「IKEv2 公開鍵証明書用キーストアを作成および使用する方法」を完了している必要があります。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **書き込み可能なディレクトリに変更します。**

   次のエラーメッセージは、CSR ファイルをディスクに書き込めないことを示している場合があります。

   ```
   Warning: error accessing "CSR-file"
   ```

   たとえば、/tmp ディレクトリを使用します。

   ```
   # cd /tmp
   ```

2. **証明書署名要求を作成します。**

   ikev2cert gencsr コマンドを使用して、証明書署名要求 (CSR) を作成します。このコマンドの引数の説明については、pktool(1) のマニュアルページで pktool gencsr keystore=pkcs11 サブコマンドを確認してください。

   たとえば、次のコマンドは partym システム上に CSR を含むファイルを作成します。

   ```
   # pfbash
   # /usr/sbin/ikev2cert gencsr \
   keytype=rsa
   keylen=2048
   label=Partym1 \
   outcsr=/tmp/Partymcsr1 \
   subject="C=US, O=PartyCompany\, Inc., OU=US-Partym, CN=Partym"
   Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
   ```

3. **(オプション) CA の Web フォームに貼り付けるために、CSR の内容をコピーします。**

   ```
   # cat /tmp/Partymcsr1
   -----BEGIN CERTIFICATE REQUEST-----
   MIICkDCCAXoCAQAwTzELMAkGA1UEBhMCVVMxGzAZBgNVBAoTElBhcnR5Q29tcGFu
   eSwgSW5jLjESMBAGA1UECxMJVVMtUGFydHltMQ8wDQYDVQQDEwZQYXJ0eW0wggEi
   MA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCMbINmgZ4XWUv2q1fshZUN/SLb
   WNLXZxdKwt5e71o0owjyby69eL7HE0QBUij73nTkXE3n4gxojBZE+hvJ6GOCbREA
   jgSquP2US7Bn9XEcXRrsOc7MCFPrsA+hVIcNHpKNseUOU/rg+wzoo5hA1ixtWuXH
   bYDeEWQi5tlZgDZoCWGrdHEjwVyHfvz+a0WBjyZBYOueBhXaa68QqSOSnRVDX56Q
   3p4H/AR4h0dcSja72XmMKPU5p3RVb8n/hrfKjiDjiGYXD4D+WZxQ65xxCcnALvVH
   nZHUlAtP7QHX4RXlQVNNwEsY6C95RX9297rNWLsYvp/86xWrQkTlNqVAeUKhAgMB
   AAEwCwYJKoZIhvcNAQEFA4IBAQB3R6rmZdqcgN8Tomyjp2CFTdyAWixkIATXpLM1
   GL5ghrnDvadD61M+vS1yhFlIcSNM8fLRrCHIKtAmB8ITnggJ//rzbHq3jdla/iQt
   kgGoTXfz8j6B57Ud6l+MBLiBSBy0QK4GIg8Ojlb9Kk5HsZ48mIoI/Qb7FFW4p9dB
   JEUn0eYhkaGtwJ21YNNvKgOeOcnSZy+xP9Wa9WpfdsBO4TicLDw0Yq7koNnfL0IB
   Fj2bt/wI7iZ1Dcpwphsiwn W9K9YynAJZzHd1ULVpn5Kd7vSRz9youLLzSb+9ilgO
   E43DW0hRk6P/Uq0N4e1Zca4otezNxyEqlPZI7pJ5uOo0sbiw
   -----END CERTIFICATE REQUEST-----
   ```

4. **CSR を認証局 (CA) に送信します。**

CA から CSR の送信方法を指示されることがあります。ほとんどの機関は、Web サイトに送信フォームを掲載しています。フォームの記入に当たっては、その送信が正当なものであることを証明する必要があります。通常は、CSR をフォームに貼り付けます。

ヒント - 証明書を貼り付けるための拡張ボタンがある Web フォームもあります。CSR は PKCS#10 形式で生成されます。したがって、Web フォームの PKCS#10 と記載された部分を探します。

---

5. **CA から受け取る各証明書をキーストアにインポートします。**

`ikev2cert import` は証明書をキーストアにインポートします。

a. **公開鍵と CA から受信した証明書をインポートします。**

```
# ikev2cert import objtype=cert label=Partym1 infile=/tmp/Partym1Cert
```

ヒント - 管理を容易にするために、インポートした証明書には元の CSR と同じラベルを割り当てます。

---

b. **CA からのルート証明書をインポートします。**

```
# ikev2cert import objtype=cert infile=/tmp/Partym1CAcert
```

c. **中間 CA 証明書をキーストアにインポートします。**

ヒント - 管理を容易にするために、インポートした中間証明書には元の CSR と同じラベルを割り当てます。

CA が各中間証明書を個別のファイルで送信してきた場合は、前述の証明書をインポートしたようにそれらをインポートします。ただし、CA が証明書チェーンを PKCS#7 ファイルとして送信している場合は、個々の証明書をそのファイルから抽出したあと、前述の証明書をインポートしたように各証明書をインポートする必要があります。

注記 - `openssl` コマンドを実行するには、`root` 役割になる必要があります。`openssl(5)` のマニュアルページを参照してください。

```
# openssl pkcs7 -in pkcs7-file -print_certs
# ikev2cert import objtype=cert label=Partym1 infile=individual-cert
```

6. **証明書検証ポリシーを設定します。**

証明書に CRL または OCSP のセクションが含まれている場合、サイト要件に従って証明書検証ポリシーを構成する必要があります。手順については、169 ページの「IKEv2 で証明書検証ポリシーを設定する方法」を参照してください。

7. **証明書を使用するすべての IKE システムで手順を完了したら、すべてのシステムで `ikev2` サービスを有効にします。**

   ピアシステムには、トラストアンカー証明書および構成済みの `ikev2.config` ファイルが必要です。

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

▼ IKEv2 で証明書検証ポリシーを設定する方法

IKEv2 システムでの証明書の処理方法に関するいくつかの要素を構成できます。

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「`ssh` 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **デフォルトの証明書検証ポリシーを表示します。**

   証明書ポリシーはインストール時に `/etc/inet/ike/kmf-policy.xml` ファイルに設定されます。このファイルは `ikeuser` によって所有され、`kmfcfg` コマンドを使用して変更します。デフォルトの証明書検証ポリシーは、CRL を `/var/user/ikeuser/crls` ディレクトリにダウンロードします。OCSP の使用もデフォルトで有効になっています。サイトがインターネットへの接続にプロキシを要求する場合は、プロキシを構成する必要があります。171 ページの「IKEv2 で失効した証明書を処理する方法」を参照してください。

```
# pfbash
# kmfcfg list dbfile=/etc/inet/ike/kmf-policy.xml policy=default
Policy Name: default
Ignore Certificate Validity Dates: false      Unknown purposes or applications for the certificate
Ignore Unknown EKUs: false
Ignore Trust Anchor in Certificate Validation: false
Trust Intermediate CAs as trust anchors: false
Maximum Certificate Path Length: 32
Certificate Validity Period Adjusted Time leeway: [not set]
Trust Anchor Certificate: Search by Issuer
Key Usage Bits: 0     Identifies critical parts of certificate
Extended Key Usage Values: [not set]     Purposes or applications for the certificate
HTTP Proxy (Global Scope): [not set]
Validation Policy Information:
    Maximum Certificate Revocation Responder Timeout: 10
    Ignore Certificate Revocation Responder Timeout: true
    OCSP:
        Responder URI: [not set]
        OCSP specific proxy override: [not set]
        Use ResponderURI from Certificate: true
        Response lifetime: [not set]
        Ignore Response signature: false
        Responder Certificate: [not set]
    CRL:
        Base filename: [not set]
        Directory: /var/user/ikeuser/crls
        Download and cache CRL: true
        CRL specific proxy override: [not set]
        Ignore CRL signature: false
        Ignore CRL validity date: false
IPsec policy bypass on outgoing connections: true
Certificate to name mapper name: [not set]
Certificate to name mapper pathname: [not set]
Certificate to name mapper directory: [not set]
Certificate to name mapper options: [not set]
```

2. **証明書で、変更する検証オプションを示す機能を確認します。**

   たとえば、CRL または OCSP URI を含む証明書は、使用する URI を指定して証明書の失効ステータスを確認する検証ポリシーを使用できます。タイムアウトを構成することもできます。

3. **kmfcfg(1) のマニュアルページで構成オプションを確認してください。**

4. **証明書検証ポリシーを構成します。**

   サンプルポリシーについては、171 ページの「IKEv2 で失効した証明書を処理する方法」を参照してください。

▼ IKEv2 で失効した証明書を処理する方法

失効した証明書とは、なんらかの理由で効力を失った証明書です。失効した証明書を使用していると、セキュリティーのリスクとなります。証明書の失効を確認する際のオプションがあります。静的なリストを使用するか、HTTP プロトコルを介して失効を動的に確認できます。

始める前に CA から証明書を受け取ってインストールしていること。

失効した証明書を確認する CRL および OSCP メソッドに精通していること。詳細およびポインタについては、140 ページの「IKE と公開鍵証明書」を参照してください。

Network IPsec Management 権利プロファイルが割り当てられている管理者になり、プロファイルシェルを使用する必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **CA から受け取った証明書内の CRL および OCSP セクションを探します。**

   CSR のラベルから証明書を識別できます。

```
# pfbash
# ikev2cert list objtype=cert | grep Label:
Enter PIN for Sun Software PKCS#11 softtoken:
        Label: Partym1
```

   たとえば、次の出力の抜粋は、証明書内の CRL および OCSP URI の部分を表示しています。

```
# ikev2cert list objtype=cert label=Partym1
X509v3 extensions:
    ...
    X509v3 CRL Distribution Points:
         Full Name:
       URI:http://onsitecrl.PKI.example.com/OCCIPsec/LatestCRL.crl
    X509v3 Authority Key Identifier:
         ...
    Authority Information Access:
        OCSP - URI:http://ocsp.PKI.example.com/revokes/
    X509v3 Certificate Policies:
         Policy: 2.16.840.1.113733.1.7.23.2
```

   `CRL Distribution Points` エントリで、URI 値はこの組織の CRL が Web 上のファイルで利用可能であることを示しています。OCSP エントリは、個々の証明書のステータスをサーバーから動的に判断できることを示しています。

2. **プロキシを指定して CRL または OCSP サーバーの使用を有効にします。**

```
# kmfcfg modify \
dbfile=/etc/inet/ike/kmf-policy.xml \
policy=default \
http-proxy=www-proxy.ja.example.com:80
```

プロキシがオプションのサイトでは、プロキシを指定する必要はありません。

3. **証明書検証ポリシーが更新されていることを確認します。**

   たとえば、OCSP が更新されたことを確認します。

```
# kmfcfg list \
dbfile=/etc/inet/ike/kmf-policy.xml \
policy=default
...
    OCSP:
        Responder URI: [not set]
        Proxy: www-proxy.ja.example.com:80
        Use ResponderURI from Certificate: true
        Response lifetime: [not set]
        Ignore Response signature: false
        Responder Certificate: [not set]
```

4. **IKEv2 サービスを再起動します。**

```
# svcadm restart ikev2
```

5. **(オプション) CRL または OCSP の使用を停止します。**

   - **CRL の使用を停止するには、次を入力します。**

```
# pfexec kmfcfg modify \
dbfile=/etc/inet/ike/kmf-policy.xml policy=default \
crl-none=true
```

     `crl-none=true` 引数は、ローカルキャッシュからダウンロードした CRL を使用するようにシステムに強制します。

   - **OCSP の使用を停止するには、次を入力します。**

```
# pfexec kmfcfg modify \
dbfile=/etc/inet/ike/kmf-policy.xml policy=default \
ocsp-none=true
```

例 9-4 システムが IKEv2 証明書検証を待機する時間を変更する

この例では、管理者が証明書の検証の待機を 20 秒に制限します。

```
# kmfcfg modify dbfile=/etc/inet/ike/kmf-policy.xml policy=default \
```

```
    cert-revoke-responder-timeout=20
```

デフォルトでは、応答がタイムアウトすると、ピアの認証が成功します。ここでは、認証に失敗すると接続が拒否されるポリシーを管理者が構成します。この構成では、OCSP または CRL サーバーが応答しなくなると証明書検証が失敗します。

```
# kmfcfg modify dbfile=/etc/inet/ike/kmf-policy.xml policy=default \
    ignore-cert-revoke-responder-timeout=false
```

ポリシーをアクティブ化するには、管理者が IKEv2 サービスを再起動します。

```
# svcadm restart ikev2
```

▼ ハードウェア上で IKEv2 の公開鍵証明書を生成および格納する方法

公開鍵証明書はまた、接続されたハードウェアに格納できます。Sun Crypto Accelerator 6000 ボードによってストレージが提供され、公開鍵の操作をシステムからこのボードにオフロードできます。

ハードウェア上で公開鍵証明書を生成および格納することは、システム上で公開鍵証明書を生成および格納することと似ています。ハードウェアでは、ハードウェアキーストアの識別に `ikev2cert gencert token=`*hw-keystore* コマンドが使用されます。

始める前に この手順では、Sun Crypto Accelerator 6000 ボードがシステムに接続されていると仮定します。また、この手順ではボード用のソフトウェアがインストールされ、ハードウェアキーストアが構成されているものと仮定します。手順については、Sun Crypto Accelerator 6000 ボード製品のライブラリドキュメント (http://docs.oracle.com/cd/E19321-01/index.html)を参照してください。これらの手順にはキーストアの設定が含まれています。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **接続されている Sun Crypto Accelerator 6000 ボードのトークン ID を持っていることを確認します。**

```
# pfbash
# ikev2cert tokens

Flags: L=Login required I=Initialized X=User PIN expired S=SO PIN expired
Slot ID  Slot Name                        Token Name                       Flags
-------  ---------                        ----------                       -----
1        sca6000                          sca6000                          LI
2        n2cp/0 Crypto Accel Bulk 1.0     n2cp/0 Crypto Accel Bulk 1.0
3        ncp/0 Crypto Accel Asym 1.0      ncp/0 Crypto Accel Asym 1.0
4        n2rng/0 SUNW_N2_Random_Number_Ge n2rng/0 SUNW_N2_RNG
5        Sun Crypto Softtoken             Sun Software PKCS#11 softtoken   LI
```

2. **自己署名付き証明書または CSR を生成して、トークン ID を指定します。**

注記 - Sun Crypto Accelerator 6000 ボードは、RSA で最大 2048 ビットの鍵をサポートします。DSA の場合、このボードは最大 1024 ビットの鍵をサポートします。

次のオプションのいずれかを選択します。

■ **自己署名付き証明書の場合は、この構文を使用します。**

```
# ikev2cert gencert token=sca6000 keytype=rsa \
hash=sha256 keylen=2048 \
subject="CN=FortKnox, C=US" serial=0x6278281232 label=goldrepo
Enter PIN for sca6000:      See Step 3
```

■ **証明書署名要求の場合は、この構文を使用します。**

```
# ikev2cert gencsr token=sca6000 -i
> keytype=
> hash=
> keylen=
> subject=
> serial=
> label=
> outcsr=
Enter PIN for sca6000 token:      See Step 3
```

`ikev2cert` コマンドの引数の説明については、`pktool(1)` のマニュアルページを参照してください。

3. **PIN のプロンプトに、Sun Crypto Accelerator 6000 ユーザー名、コロン、およびユーザーのパスワードを入力します。**

注記 - キーストアのユーザー名とパスワードを知っている必要があります。

Sun Crypto Accelerator 6000 ボードがユーザー admin、パスワード inThe%4ov で構成されている場合は、次を入力します。

```
Enter PIN for sca6000 token: admin:inThe%4ov
-----BEGIN X509 CERTIFICATE-----
MIIBuDCCASECAQAwSTELMAkGA1UEBhMCVVMxFTATBgNVBAoTDFBhcnR5Q29tcGFu
…
oKUDBbZ9O/pLWYGr
-----END X509 CERTIFICATE-----
```

4. **通信先に証明書を送信します。**

   次のオプションのいずれかを選択します。

   ■ **リモートシステムに自己署名付き証明書を送信します。**

   証明書は、電子メールのメッセージに貼り付けることもできます。

   ■ **CA に証明書署名要求を送信します。**

   CA の指示に従って CSR を送信します。詳細な説明は、141 ページの「IKE での公開鍵証明書の使用」を参照してください。

5. **証明書をハードウェアキーストアにインポートします。**

   CA から受け取った証明書をインポートして、ステップ 3 のユーザーおよび PIN を指定します。

```
# ikev2cert import token=sca6000 infile=/tmp/DCA.ACCEL.CERT1
Enter PIN for sca6000 token:      Type user:password
# ikev2cert import token=sca6000 infile=/tmp/DCA.ACCEL.CA.CERT
Enter PIN for sca6000 token:      Type user:password
```

6. **自動的に、または対話形式で使用するハードウェアキーストアを有効にします。**

   自動ログインが推奨されます。サイトのセキュリティーポリシーが自動ログインを許可していない場合は、in.ikev2d デーモンを再起動する際に対話形式でキーストアにログインする必要があります。

   ■ **キーストアへの自動ログインを構成します。**

   a. **PIN を pkcs11_token/uri サービスプロパティーの値として追加します。**

   このプロパティーの説明については、238 ページの「IKEv2 サービス」を参照してください。

```
# svccfg -s ike:ikev2 editprop
```

一時編集ウィンドウが開きます。

b. **setprop pkcs11_token/uri = 行のコメントを解除して、括弧を次の形式でトークン名に置き換えます。**

```
# setprop pkcs11_token/uri = ()       Original entry
setprop pkcs11_token/uri = pkcs11:token=sca6000
```

c. **setprop pkcs11_token/uri = 行のコメントを解除して、括弧を** ステップ **3** **の** *username:PIN* **に置き換えます。**

```
# setprop pkcs11_token/uri = ()       Original entry
setprop pkcs11_token/uri = admin:PIN-from-Step-3
```

d. **ファイル最下部の refresh 行のコメントを解除して、変更を保存します。**

```
# refresh
refresh
```

e. **(オプション) pkcs11_token プロパティーの値を確認します。**

```
# svccfg -s ikev2 listprop pkcs11_token
pkcs11_token/pin     astring      username:PIN
pkcs11_token/uri     astring  pkcs11:token=sca6000
```

■ **自動ログインが構成されていない場合は、ハードウェアキーストアに手動でログインします。**

in.ikev2d デーモンが起動するたびにこのコマンドを実行します。

```
# pfexec ikeadm -v2 token login sca6000
Enter PIN for sca6000 token: admin:PIN-from-Step-3
ikeadm: sca6000 operation successful
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

♦ ♦ ♦ 第 10 章

10 IKEv1 の構成

この章では、使用するシステムに合わせてインターネット鍵交換バージョン 1 (IKEv1) を構成する方法について説明します。IKEv1 を構成すると、ネットワークでの IPsec の鍵情報が自動的に生成されます。この章では、次の内容について説明します。

- 178 ページの「事前共有鍵による IKEv1 の構成」
- 183 ページの「公開鍵証明書による IKEv1 の構成」
- 201 ページの「移動体システム用の IKEv1 の構成」
- 209 ページの「接続したハードウェアを検出するように IKEv1 を構成する」

注記 - IKEv2 だけを実装する場合は、第9章「IKEv2 の構成」に進みます。

IKE の概要については、第8章「インターネット鍵交換について」を参照してください。IKE の参照情報については、第12章「IPsec および鍵管理のリファレンス」を参照してください。詳細な手順については、`ikeadm`(1M)、`ikecert`(1M)、および `ike.config`(4) のマニュアルページで使用例のセクションを参照してください。

注記 - これらのタスクは、システムに静的 IP アドレスがアサイン済みで、ネットワーク構成プロファイル `DefaultFixed` を実行していることを前提としています。`netadm list` コマンドが `Automatic` を返す場合は、`netcfg`(1M) のマニュアルページで詳細を確認してください。

IKEv1 の構成

IKE の認証には、事前共有鍵、自己署名付き証明書、および認証局 (CA) の証明書を使用できます。`ike/config` ファイル内のルールが、特定の IKEv1 認証方法と IKEv1 ピアを関連付けます。したがって、1 つのシステムに 1 つまたはすべての IKE 認証方法を使用できます。PKCS #11 ライブラリへのポインタによって、IKEv1 は、接続されたハードウェアアクセラレータを使用できます。

IKEv1 を構成したら、第7章「IPsec の構成」で、IKEv1 構成を使用する IPsec タスクを実行してください。

事前共有鍵による IKEv1 の構成

IKEv1 を使用するようにピアシステムまたはサブネットを構成し、さらに、あなたがこれらのサブネットの管理者である場合、事前共有鍵の使用が適しています。事前共有鍵は、テストの際にも使用される場合があります。詳細は、140 ページの「IKE と事前共有鍵認証」を参照してください。

▼ 事前共有鍵で IKEv1 を構成する方法

IKE 実装では、鍵の長さが異なるさまざまなアルゴリズムが提供されます。鍵の長さは、サイトのセキュリティーに応じて選択します。一般的に、鍵の長さが長いほど、セキュリティーが高くなります。

この手順では、ASCII 形式の鍵を生成します。

これらの手順には、システム名 `enigma` および `partym` を使用します。`enigma` と `partym` を各自使用しているシステムの名前に置き換えてください。

注記 - Trusted Extensions システムのラベルと一緒に IPsec を使用するには、『Trusted Extensions 構成と管理 』の「マルチレベル Trusted Extensions ネットワークで IPsec 保護を適用する」にあるこの手順の拡張を参照してください。

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「`ssh` 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **システムごとに、`/etc/inet/ike/config` ファイルを作成します。**

   `/etc/inet/ike/config.sample` をテンプレートとして使用できます。

2. **システムごとに、規則とグローバルパラメータを `ike/config` ファイルに入力します。**

   これらの規則やグローバルパラメータは、システムの `ipsecinit.conf` ファイルに設定されている IPsec ポリシーが正しく動作するものでなければなりません。次の IKEv1 構成の例は、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」の `ipsecinit.conf` の例で機能します。

   a. **たとえば、`enigma` システムの `/etc/inet/ike/config` ファイルを次のように変更します。**

```
### ike/config file on enigma, 192.168.116.16

## Global parameters
#
## Defaults that individual rules can override.
p1_xform
  { auth_method preshared oakley_group 5 auth_alg sha encr_alg 3des }
p2_pfs 2
#
## The rule to communicate with partym
#  Label must be unique
{ label "enigma-partym"
  local_addr 192.168.116.16
  remote_addr 192.168.13.213
  p1_xform
   { auth_method preshared oakley_group 5 auth_alg sha256 encr_alg aes }
  p2_pfs 5
}
```

   b. **`partym` システムの `/etc/inet/ike/config` ファイルを次のように変更します。**

```
### ike/config file on partym, 192.168.13.213
## Global Parameters
#
p1_xform
  { auth_method preshared oakley_group 5 auth_alg sha encr_alg 3des }
p2_pfs 2

## The rule to communicate with enigma
#  Label must be unique
{ label "partym-enigma"
  local_addr 192.168.13.213
  remote_addr 192.168.116.16
p1_xform
 { auth_method preshared oakley_group 5 auth_alg sha256 encr_alg aes }
p2_pfs 5
}
```

3. **システムごとに、ファイルの構文を確認します。**

```
# /usr/lib/inet/in.iked -c -f /etc/inet/ike/config
```

4. **事前共有鍵を各システムの /etc/inet/secret/ike.preshared ファイルに追加します。**

   a. **たとえば、enigma システムの ike.preshared ファイルは次のようになります。**

```
## ike.preshared on enigma, 192.168.116.16
#…
{ localidtype IP
 localid 192.168.116.16
 remoteidtype IP
 remoteid 192.168.13.213
 # The preshared key can also be represented in hex
# as in 0xf47cb0f432e14480951095f82b
# key "This is an ASCII Cqret phrAz, use str0ng p@ssword tekniques"
}
```

   b. **partym システムの ike.preshared ファイルは次のようになります。**

```
## ike.preshared on partym, 192.168.13.213
#…
{ localidtype IP
 localid 192.168.13.213
 remoteidtype IP
 remoteid 192.168.116.16
 # The preshared key can also be represented in hex
# as in 0xf47cb0f432e14480951095f82b
 key "This is an ASCII Cqret phrAz, use str0ng p@ssword tekniques"
 }
```

5. **IKEv1 サービスを有効にします。**

```
# svcadm enable ipsec/ike:default
```

例 10-1 IKEv1 事前共有鍵をリフレッシュする

IKEv1 管理者が事前共有鍵をリフレッシュするときは、ピアシステム上のファイルを編集し、in.iked デーモンを再起動します。

最初に、事前共有鍵を使用する 2 つのサブネット内のすべてのシステムで、管理者が事前共有鍵エントリを変更します。

```
# pfedit -s /etc/inet/secret/ike.preshared
…
{ localidtype IP
 localid 192.168.116.0/24
 remoteidtype IP
 remoteid 192.168.13.0/24
 # The two subnet's shared passphrase for keying material
key "LOooong key Th@t m^st Be Ch*angEd \'reguLarLy)"
 }
```

次に、管理者は各システムの IKEv1 サービスを再起動します。

`pfedit` コマンドのオプションについては、`pfedit`(1M) のマニュアルページを参照してください。

```
# svcadm enable ipsec/ike:default
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

▼ 新規ピアシステムのために IKEv1 を更新する方法

同じピア間で動作中の構成に対して IPsec ポリシーエントリを追加した場合、IPsec ポリシーサービスをリフレッシュする必要があります。IKEv1 の再構成または再起動は不要です。

IPsec ポリシーに新しいピアを追加した場合、IPsec の変更に加えて IKEv1 構成を変更する必要があります。

始める前に `ipsecinit.conf` ファイルをリフレッシュし、ピアシステムの IPsec ポリシーをリフレッシュしました。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「`ssh` 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **IPsec を使用する新規システム用の鍵を管理するための IKEv1 のルールを作成します。**

   a. **たとえば、`enigma` システムで、次の規則を `/etc/inet/ike/config` ファイルに追加します。**

   ```
   ### ike/config file on enigma, 192.168.116.16

   ## The rule to communicate with ada

   {label "enigma-to-ada"
   ```

```
 local_addr 192.168.116.16
 remote_addr 192.168.15.7
 p1_xform
 {auth_method preshared oakley_group 5 auth_alg sha256 encr_alg aes}
 p2_pfs 5
 }
```

b. **ada システムで、次の規則を追加します。**

```
### ike/config file on ada, 192.168.15.7

## The rule to communicate with enigma

{label "ada-to-enigma"
 local_addr 192.168.15.7
 remote_addr 192.168.116.16
 p1_xform
 {auth_method preshared oakley_group 5 auth_alg sha256 encr_alg aes}
 p2_pfs 5
}
```

2. **ピアシステム用の IKEv1 事前共有鍵を作成します。**

a. **enigma システムで、/etc/inet/secret/ike.preshared ファイルに次の情報を追加します。**

```
## ike.preshared on enigma for the ada interface
##
{ localidtype IP
  localid 192.168.116.16
  remoteidtype IP
  remoteid 192.168.15.7
  # enigma and ada's shared key
  key "Twas brillig and the slivey toves did *s0mEtHiNg* be CareFULL hEEEr"
}
```

b. **ada システムで、次の情報を ike.preshared ファイルに追加します。**

```
## ike.preshared on ada for the enigma interface
##
{ localidtype IP
  localid 192.168.15.7
  remoteidtype IP
  remoteid 192.168.116.16
  # ada and enigma's shared key
  key "Twas brillig and the slivey toves did *s0mEtHiNg* be CareFULL hEEEr"
}
```

3. **各システムで、ike サービスをリフレッシュします。**

```
# svcadm refresh ike:default
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

公開鍵証明書による IKEv1 の構成

公開鍵証明書を使用すると、通信するシステムが秘密鍵情報を帯域外で共有する必要がなくなります。認証局 (CA) からの公開鍵証明書では、通常、外部機関とのネゴシエーションが必要となります。この証明書は非常に簡単に拡大できるため、通信するシステムを数多く保護できます。

公開鍵証明書はまた、接続されたハードウェア内で生成して格納できます。手順については、209 ページの「接続したハードウェアを検出するように IKEv1 を構成する」を参照してください。

すべての証明書には、X.509 識別名 (DN) 形式の一意の名前があります。また、証明書には、電子メールアドレス、DNS 名、IP アドレスなどのサブジェクトの別名が 1 つまたは複数ある場合があります。IKEv1 構成内の証明書は、完全な DN またはサブジェクトの別名のいずれかによって識別できます。これらの別名の形式は *tag=value* で、値の形式はそのタグのタイプに対応します。たとえば、`email` タグの形式は *name@ domain.suffix* です。

次のタスクマップは、IKEv1 の公開鍵証明書の作成手順を一覧表示したものです。これらの手順には、接続されたハードウェア上で証明書を高速化および格納する方法が含まれます。

表 10-1 公開鍵証明書による IKEv1 の構成のタスクマップ

タスク	説明	参照先
自己署名付き公開鍵証明書で IKEv1 を構成します。	システムごとに鍵および 2 つの証明書を作成および格納します。 ■ 自己署名付き証明書およびその鍵 ■ ピアシステムからの公開鍵証明書	184 ページの「自己署名付き公開鍵証明書により IKEv1 を構成する方法」
認証局で IKEv1 を構成します。	証明書署名要求を作成して、CA からの証明書を各システムに配置します。141 ページの「IKE での公開鍵証明書の使用」を参照してください。	189 ページの「CA の署名付き証明書で IKEv1 を構成する方法」

タスク	説明	参照先
ローカルハードウェアで公開鍵証明書を構成します。	次のいずれかが含まれます。 ■ ローカルハードウェアで自己署名付き証明書を生成してから、リモートシステムからの公開鍵をハードウェアに追加する。 ■ ローカルハードウェアで証明書署名要求を生成してから、CA からの公開鍵証明書をハードウェアに追加します。	195 ページの「ハードウェア上で IKEv1 の公開鍵証明書を生成および格納する方法」
CA からの証明書失効リスト (CRL) を更新します。	中央の配布ポイントから CRL にアクセスします。	199 ページの「IKEv1 で失効した証明書を処理する方法」

注記 - システム上でパケットおよび IKE ネゴシエーションにラベルを付けるには、『Trusted Extensions 構成と管理 』の「ラベル付き IPsec の構成」の手順に従ってください。

公開鍵証明書は Trusted Extensions システムの大域ゾーン内で管理されます。Trusted Extensions は証明書を管理および格納する方法を変更しません。

▼ 自己署名付き公開鍵証明書により IKEv1 を構成する方法

この手順では、公開鍵/非公開鍵および証明書、すなわち証明書ペアを作成します。非公開鍵はディスク上のローカル証明書データベースに格納され、`ikecert certlocal` コマンドを使用して参照できます。公開鍵および証明書は、公開証明書データベースに格納されます。これは `ikecert certdb` コマンドを使用して参照できます。公開証明書をピアシステムと交換します。この 2 つの証明書が IKEv1 転送の認証に使用されます。

自己署名付き証明書は、CA からの公開鍵証明書よりもオーバーヘッドが少ないのですが、あまり簡単には拡大できません。CA から発行される証明書とは異なり、自己署名付き証明書は証明書を交換した 2 人の管理者によって確認される必要があります。

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「`ssh` 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **各 IKEv1 システムで、自己署名付き証明書を `ike.privatekeys` データベース内に作成します。**

   `ikecert certlocal` コマンドの引数については、`ikecert`(1M) のマニュアルページを参照してください。

   a. **たとえば、`partym` システムでは、コマンドは次のようになります。**

```
# ikecert certlocal -ks -m 2048 -t rsa-sha512 \
-D "O=exampleco, OU=IT, C=US, CN=partym" \
-A IP=192.168.13.213
Creating private key.
Certificate added to database.
-----BEGIN X509 CERTIFICATE-----
MIIC1TCCAb2gAwIBAgIEfdZgKjANBgkqhkiG9w0BAQUFADAaMRgwFgYDVQQDEw9T
a...+
zBGi4QkNdI3f
-----END X509 CERTIFICATE-----
```

   ここでは

   `-ks` 自己署名付き証明書を作成します。

   `-m` *keysize* 鍵のサイズを指定します。

   `-t` *keytype* 使用するアルゴリズムのタイプを指定します。

   `-D` *dname* 証明書サブジェクトの X.509 識別名 (DN) を指定します。例は、141 ページの「IKE での公開鍵証明書の使用」を参照してください。

   `-A` *altname* 証明書の別名またはニックネームを指定します。*altname* の形式は *tag=value* です。有効なタグは `IP`、`DNS`、`email`、および `DN` です。

注記 - `-D` および `-A` オプションの値は、`192.168.13.213` のように、システムではなく、証明書だけを識別する名前です。実際、これらは証明書ニックネームのため、正しい証明書がピアシステムにインストールされていることを帯域外で検証する必要があります。

---

   b. **`enigma` システムでは、コマンドは次のようになります。**

```
# ikecert certlocal -ks -m 2048 -t rsa-sha512 \
-D "O=exampleco, OU=IT, C=US, CN=enigma" \
-A IP=192.168.116.16
Creating private key.
Certificate added to database.
-----BEGIN X509 CERTIFICATE-----
MIIC1TCCAb2gAwIBAgIEBl5JnjANBgkqhkiG9w0BAQUFADAaMRgwFgYDVQQDEw9T
```

```
...
y85m6LHJYtC6
-----END X509 CERTIFICATE-----
```

2. **証明書を保存し、リモートシステムに送信します。**

   出力は、証明書の公開部分のエンコード済みバージョンです。この証明書は、電子メールのメッセージに貼り付けても安全です。ステップ 4 で示すように、受け取り側は正しい証明書をインストールしていることを帯域外で検証する必要があります。

   a. **たとえば、`partym` 証明書の公開部分を `enigma` 管理者に送信します。**

   ```
   To: admin@enigma.ja.example.com
   From: admin@party.us.example.com
   Message: -----BEGIN X509 CERTIFICATE-----
   MIIC1TCCAb2gAwIBAgIEfdZgKjANBgkqhkiG9w0BAQUFADAaMRgwFgYDVQQDEw9T
   a...+
   zBGi4QkNdI3f
   -----END X509 CERTIFICATE------
   ```

   b. **`enigma` 管理者から、`enigma` 証明書の公開部分が送信されます。**

   ```
   To: admin@party.us.example.com
   From: admin@enigma.ja.example.com
   Message: ----BEGIN X509 CERTIFICATE-----
   MIIC1TCCAb2gAwIBAgIEBl5JnjANBgkqhkiG9w0BAQUFADAaMRgwFgYDVQQDEw9T
   ...
   y85m6LHJYtC6
   -----END X509 CERTIFICATE-----
   ```

3. **各システムで、受け取った証明書を公開鍵データベースに追加します。**

   a. **`root` が読み取るファイルに管理者の電子メールを保存します。**

   b. **ファイルを `ikecert` コマンドにリダイレクトします。**

   ```
   # ikecert certdb -a < /tmp/certificate.eml
   ```

   このコマンドは、`BEGIN` タグと `END` タグの間にあるテキストをインポートします。

4. **その証明書がその管理者からのものであることを相手の管理者に確認します。**

   たとえば、ほかの管理者に電話して、自分が持つ公開証明書のハッシュが、ほかの管理者のみが持つ非公開証明書のハッシュに一致することを検証できます。

   a. **`partym` に格納されている証明書を一覧表示します。**

次の例で、Note 1 はスロット 0 の証明書の識別名 (DN) を示します。スロット 0 の非公開証明書は同じハッシュを持つ (注 3 を参照) ため、これらの証明書は同一の証明書ペアです。公開証明書が機能するには、一致するペアを持つ必要があります。`certdb` サブコマンドは公開部分を示し、`certlocal` サブコマンドは非公開部分を示します。

```
partym # ikecert certdb -l

Certificate Slot Name: 0   Key Type: rsa
 (Private key in certlocal slot 0)
 Subject Name: <O=exampleco, OU=IT, C=US, CN=partym>     Note 1
 Key Size: 2048
 Public key hash: 80829EC52FC5BA910F4764076C20FDCF

Certificate Slot Name: 1   Key Type: rsa
 (Private key in certlocal slot 1)
 Subject Name: <O=exampleco, OU=IT, C=US, CN=Ada>
 Key Size: 2048
 Public key hash: FEA65C5387BBF3B2C8F16C019FEBC388
partym # ikecert certlocal -l
Local ID Slot Name: 0   Key Type: rsa
 Key Size: 2048
 Public key hash: 80829EC52FC5BA910F4764076C20FDCF     Note 3

Local ID Slot Name: 1   Key Type: rsa-sha512
        Key Size: 2048
        Public key hash: FEA65C5387BBF3B2C8F16C019FEBC388

Local ID Slot Name: 2   Key Type: rsa
 Key Size: 2048
 Public key hash: 2239A6A127F88EE0CB40F7C24A65B818
```

このチェックでは、`partym` システムが有効な証明書ペアを持つことが検証されました。

b. **`enigma` システムが `partym` の公開証明書を持つことを確認します。**

公開鍵ハッシュは電話で伝えることができます。

前の手順の `partym` における Note 3 のハッシュと、`enigma` における Note 4 を比較します。

```
enigma # ikecert certdb -l

Certificate Slot Name: 0   Key Type: rsa
        (Private key in certlocal slot 0)
        Subject Name: <O=exampleco, OU=IT, C=US, CN=Ada>
        Key Size: 2048
        Public key hash: 2239A6A127F88EE0CB40F7C24A65B818

Certificate Slot Name: 1   Key Type: rsa
        (Private key in certlocal slot 1)
```

```
        Subject Name: <O=exampleco, OU=IT, C=US, CN=enigma>
        Key Size: 2048
        Public key hash: FEA65C5387BBF3B2C8F16C019FEBC388

Certificate Slot Name: 2   Key Type: rsa
        (Private key in certlocal slot 2)
        Subject Name: <O=exampleco, OU=IT, C=US, CN=partym>
        Key Size: 2048
        Public key hash: 80829EC52FC5BA910F4764076C20FDCF     Note 4
```

enigma の公開証明書データベースに格納されている最後の証明書の公開鍵ハッシュとサブジェクト名が、前の手順の partym の非公開証明書と一致します。

5. **システムごとに、両方の証明書を信頼します。**

   /etc/inet/ike/config ファイルを編集して、証明書を認識します。

   パラメータ cert_trust、remote_addr、および remote_id の値は、リモートシステムの管理者が提供します。

   a. **たとえば、partym システム上の ike/config ファイルは次のようになります。**

```
# Explicitly trust the self-signed certs
# that we verified out of band. The local certificate
# is implicitly trusted because we have access to the private key.

cert_trust "O=exampleco, OU=IT, C=US, CN=enigma"
# We could also use the Alternate name of the certificate,
# if it was created with one.  In this example, the Alternate Name
# is in the format of an IP address:
# cert_trust "192.168.116.16"

## Parameters that may also show up in rules.

p1_xform
  { auth_method preshared oakley_group 5 auth_alg sha256 encr_alg 3des }
p2_pfs 5

{
 label "US-partym to JA-enigma"
 local_id_type dn
 local_id "O=exampleco, OU=IT, C=US, CN=partym"
 remote_id "O=exampleco, OU=IT, C=US, CN=enigma"

 local_addr  192.168.13.213
# We could explicitly enter the peer's IP address here, but we don't need
# to do this with certificates, so use a wildcard address. The wildcard
# allows the remote device to be mobile or behind a NAT box
 remote_addr 0.0.0.0/0
```

```
 p1_xform
  {auth_method rsa_sig oakley_group 2 auth_alg sha256 encr_alg aes}
}
```

b. **enigma システムで、ike/config ファイルにローカルパラメータの enigma 値を追加します。**

リモートパラメータには、partym 値を使用します。label キーワードの値がローカルシステム上で一意であることを確認します。

```
…
{
 label "JA-enigma to US-partym"
 local_id_type dn
 local_id "O=exampleco, OU=IT, C=US, CN=enigma"
 remote_id "O=exampleco, OU=IT, C=US, CN=partym"

 local_addr  192.168.116.16
 remote_addr 0.0.0.0/0

 p1_xform
  {auth_method rsa_sig oakley_group 2 auth_alg sha256 encr_alg aes}
}
```

6. **ピアシステムで、IKEv1 を有効にします。**

```
partym # svcadm enable ipsec/ike:default
enigma # svcadm enable ipsec/ike
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

▼ CA の署名付き証明書で IKEv1 を構成する方法

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **`ikecert certlocal -kc` コマンドを使用して証明書署名要求 (CSR) を作成します。**

   コマンドの引数については、184 ページの「自己署名付き公開鍵証明書により IKEv1 を構成する方法」の ステップ 1 を参照してください。

   ```
   # ikecert certlocal -kc -m keysize -t keytype \
   -D dname -A altname
   ```

   a. **たとえば、次のコマンドは `partym` システムに CSR を作成します。**

   ```
   # ikecert certlocal -kc -m 2048 -t rsa-sha384 \
   > -D "C=US, O=PartyCompany\, Inc., OU=US-Partym, CN=Partym" \
   > -A "DN=C=US, O=PartyCompany\, Inc., OU=US-Partym"
   Creating software private keys.
     Writing private key to file /etc/inet/secret/ike.privatekeys/2.
   Enabling external key providers - done.
   Certificate Request:
     Proceeding with the signing operation.
     Certificate request generated successfully (…/publickeys/0)
   Finished successfully.
   -----BEGIN CERTIFICATE REQUEST-----
   MIIByjCCATMCAQAwUzELMAkGA1UEBhMCVVMxHTAbBgNVBAoTFEV4YW1wbGVDb21w
   …
   lcM+tw0ThRrfuJX9t/Qa1R/KxRlMA3zckO80mO9X
   -----END CERTIFICATE REQUEST-----
   ```

   b. **次のコマンドは `enigma` システムに CSR を作成します。**

   ```
   # ikecert certlocal -kc -m 2048 -t rsa-sha384 \
   > -D "C=JA, O=EnigmaCo\, Inc., OU=JA-Enigmax, CN=Enigmax" \
   > -A "DN=C=JA, O=EnigmaCo\, Inc., OU=JA-Enigmax"
   Creating software private keys.
   …
   Finished successfully.
   -----BEGIN CERTIFICATE REQUEST-----
   MIIBuDCCASECAQAwSTELMAkGA1UEBhMCVVMxFTATBgNVBAoTDFBhcnR5Q29tcGFu
   …
   8qlqdjaStLGfhDOO
   -----END CERTIFICATE REQUEST-----
   ```

2. **CA に CSR を送信します。**

   CA から CSR の送信方法を指示されることがあります。ほとんどの機関は、Web サイトに送信フォームを掲載しています。フォームの記入に当たっては、その送信が正当なものであることを証明する必要があります。通常は、CSR をフォームに貼り付けます。要求が組織によって確認されている場合は、組織から署名付き証明書が発行されます。詳細は、141 ページの「IKE での公開鍵証明書の使用」を参照してください。

3. **各証明書をシステムに追加します。**

`ikecert certdb -a` コマンドの `-a` オプションは、張り付けられたオブジェクトをシステムの適切な証明書データベースに追加します。詳細については、140 ページの「IKE と公開鍵証明書」を参照してください。

a. **管理者になります。**

詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。リモートで管理する場合は、例 7-1「`ssh` 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

b. **公開鍵と CA から受け取った証明書を追加します。**

```
# ikecert certdb -a < /tmp/PKIcert.eml
```

c. **CA の公開証明書を追加します。**

中間証明書の追加が必要な場合もあります。

```
# ikecert certdb -a < /tmp/PKIca.eml
```

d. **CA から証明書失効リストが送られている場合は、その CRL を `certrldb` データベースに追加します。**

```
# ikecert certrldb -a
Press the Return key
Paste the CRL
-----BEGIN CRL-----
…
-----END CRL----
Press the Return key
Press Control-D
```

4. **`/etc/inet/ike/config` ファイル内の `cert_root` キーワードを使用して、証明書を発行した CA を識別します。**

CA の証明書の識別名 (DN) を使用します。

a. **たとえば、`partym` システムの `ike/config` ファイルは次のようになります。**

```
# Trusted root cert
# This certificate is from Example CA
# This is the X.509 distinguished name for the CA's cert

cert_root "C=US, O=ExampleCA\, Inc., OU=CA-Example, CN=Example CA"
```

```
## Parameters that may also show up in rules.

p1_xform
 { auth_method rsa_sig oakley_group 1 auth_alg sha384 encr_alg aes}
p2_pfs 2

{
 label "US-partym to JA-enigma - Example CA"
 local_id_type dn
 local_id  "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"
 remote_id "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax"

 local_addr  192.168.13.213
 remote_addr 192.168.116.16

 p1_xform
  {auth_method rsa_sig oakley_group 2 auth_alg sha256 encr_alg aes}
}
```

注記 - auth_method パラメータのすべての引数は同じ行になければなりません。

b. **enigma システム上で、同様なファイルを作成します。**

特に、enigma ike/config ファイルは次を行う必要があります。

- cert_root には同じ値を使用する。
- ローカルパラメータには enigma 値を使用する。
- リモートパラメータには partym 値を使用する。
- labelキーワードには一意の値を作成する。この値は、リモートシステムの label 値とは異なる値でなくてはなりません。

```
…
cert_root "C=US, O=ExampleCA\, Inc., OU=CA-Example, CN=Example CA"
…
{
 label "JA-enigma to US-partym - Example CA"
 local_id_type dn
 local_id   "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax"
 remote_id  "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"

 local_addr  192.168.116.16
 remote_addr 192.168.13.213
…
```

5. **失効した証明書の処理について、IKEv1 ポリシーを設定します。**

適切なオプションを選択します。

■ OCSP 利用不可

公開鍵証明書には OCSP サーバーに到達するための URI が記述されているものの、使用しているシステムがインターネットに接続できない場合は、キーワード `ignore_ocsp` を `ike/config` ファイルに追加します。

```
# Trusted root cert
…
cert_root "C=US, O=ExampleCA\, Inc., OU=CA-Example,…
ignore_ocsp
…
```

`ignore_ocsp` キーワードは、証明書が有効であると仮定するように IKEv1 に伝えます。

■ CRL 利用不可

CA が CRL の信頼できるソースを提供しない場合、または使用しているシステムがインターネットに接続して CRL を取得できない場合は、キーワード `ignore_crls` を `ike/config` ファイルに追加します。

```
# Trusted root cert
…
cert_root "C=US, O=ExampleCA\, Inc., OU=CA-Example,…
ignore_crls
…
```

■ CRL または OCSP の URI を利用可能

CA が失効した証明書の中央配布ポイントを提供している場合は、`ike/config` ファイルを変更して URI を使用できます。

例は、199 ページの「IKEv1 で失効した証明書を処理する方法」を参照してください。

例 10-2 IKEv1 の構成時に `rsa_encrypt` を使用する

`ike/config` ファイルで `auth_method rsa_encrypt` を使用する場合は、ピアの証明書を `publickeys` データベースに追加する必要があります。

1. その証明書をリモートシステムの管理者に送信します。

   証明書は、電子メールのメッセージに貼り付けることもできます。

   たとえば、`partym` の管理者が次のメッセージを送信します。

```
To: admin@enigma.ja.example.com
From: admin@party.us.example.com
```

```
Message: -----BEGIN X509 CERTIFICATE-----
MII…
----END X509 CERTIFICATE-----
```

enigma の管理者が次のメッセージを送信します。

```
To: admin@party.us.example.com
From: admin@enigma.ja.example.com
Message: -----BEGIN X509 CERTIFICATE-----
MII
…
-----END X509 CERTIFICATE-----
```

2. システムごとに、電子メールで送信された証明書をローカルの publickeys データベースに追加します。

```
# ikecert certdb -a < /tmp/saved.cert.eml
```

RSA 暗号化の認証方法は、IKE 内の識別子を盗聴者から隠します。rsa_encrypt メソッドはピアの識別子を隠すため、IKEv1 はピアの証明書を取得できません。結果として、rsa_encrypt メソッドでは、IKEv1 ピアが互いの公開鍵を知っておく必要があります。

そのため、/etc/inet/ike/config ファイルの auth_method に rsa_encrypt を指定する場合には、ピアの証明書を publickeys データベースに追加する必要があります。この結果、publickeys データベースには、通信するシステムペアごとに 3 つ以上の証明書が存在することになります。

- ユーザーの公開鍵証明書
- CA の証明書チェーン
- ピアの公開鍵証明書

トラブルシューティング – IKEv1 ペイロードには 3 つ以上の証明書が含まれており、rsa_encrypt で暗号化するには大きくなりすぎることがあります。「authorization failed (承認に失敗しました)」や「malformed payload (ペイロードが不正です)」などのエラーは、rsa_encrypt メソッドがペイロード全体を暗号化できないことを示します。証明書を 2 つしか必要としない rsa_sig などのメソッドを使用して、ペイロードのサイズを減らします。

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他

の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

▼ ハードウェア上で IKEv1 の公開鍵証明書を生成および格納する方法

ハードウェア上で公開鍵証明書を生成および格納することは、システム上で公開鍵証明書を生成および格納することと似ています。ハードウェア上では、`ikecert certlocal` および `ikecert certdb` コマンドがハードウェアを識別しなければなりません。トークン ID に `-T` オプションを指定すると、コマンドがハードウェアを識別するようになります。

始める前に

- ハードウェアの構成が完了していること。
- `/etc/inet/ike/config` ファイルの `pkcs11_path` キーワードが別のライブラリを指している場合を除き、ハードウェアは `/usr/lib/libpkcs11.so` ライブラリを使用します。ライブラリが、RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki) 規格、つまり PKCS #11 ライブラリに準拠して実装されている必要があります。

  設定の手順については、209 ページの「Sun Crypto Accelerator 6000 ボードを検出するように IKEv1 を構成する方法」を参照してください。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **自己署名付き証明書または CSR を生成して、トークン ID を指定します。**

注記 - Sun Crypto Accelerator 6000 ボードは、RSA で最大 2048 ビットの鍵をサポートします。DSA の場合、このボードは最大 1024 ビットの鍵をサポートします。

次のオプションのいずれかを選択します。

- **自己署名付き証明書の場合は、この構文を使用します。**

```
# ikecert certlocal -ks -m 2048 -t rsa-sha512 \
> -D "C=US, O=PartyCompany, OU=US-Partym, CN=Partym" \
```

```
> -a -T dca0-accel-stor IP=192.168.116.16
Creating hardware private keys.
Enter PIN for PKCS#11 token:      Type user:password
```

-T オプションの引数は、接続された Sun Crypto Accelerator 6000 ボードのトークン ID です。

■ **CSR の場合は、この構文を使用します。**

```
# ikecert certlocal -kc -m 2048 -t rsa-sha512 \
> -D "C=US, O=PartyCompany, OU=US-Partym, CN=Partym" \
> -a -T dca0-accel-stor IP=192.168.116.16
Creating hardware private keys.
Enter PIN for PKCS#11 token:      Type user:password
```

ikecert コマンドの引数の詳細については、ikecert(1M) のマニュアルページを参照してください。

2. **PIN のプロンプトに、Sun Crypto Accelerator 6000 ユーザー名、コロン、およびユーザーのパスワードを入力します。**

Sun Crypto Accelerator 6000 ボードのユーザー ikemgr のパスワードが rgm4tigt の場合、次のように入力します。

```
Enter PIN for PKCS#11 token: ikemgr:rgm4tigt
```

注記 - ikecert コマンドに -p オプションを付けて入力すると、PKCS #11 トークンがディスクに*クリアテキストとして*格納され、root 権限によって保護されます。PIN をディスクに保存しない場合は、in.iked コマンドの実行後に ikeadm コマンドを使用してトークンをロック解除する必要があります。

パスワードの入力後、証明書が次を出力します。

```
Enter PIN for PKCS#11 token: ikemgr:rgm4tigt
-----BEGIN X509 CERTIFICATE-----
MIIBuDCCASECAQAwSTELMAkGA1UEBhMCVVMxFTATBgNVBAoTDFBhcnR5Q29tcGFu
…
oKUDBbZ9O/pLWYGr
-----END X509 CERTIFICATE-----
```

3. **相手に証明書を送信します。**

次のオプションのいずれかを選択します。

■ **リモートシステムに自己署名付き証明書を送信します。**

証明書は、電子メールのメッセージに貼り付けることもできます。

■ **CSR を認証局 (CA) に送信します。**

CA の指示に従って証明書要求を送信します。詳細な説明は、189 ページの「CA の署名付き証明書で IKEv1 を構成する方法」の ステップ 2 を参照してください。

4. **システム上で、証明書を認識するように `/etc/inet/ike/config` ファイルを編集します。**

次のオプションのどちらか 1 つを選択します。

■ **自己署名付き証明書**

`cert_trust`、`remote_id`、および `remote_addr` パラメータには、リモートシステムの管理者から提供される値を使用します。たとえば、`enigma` システムの `ike/config` ファイルは次のようになります。

```
# Explicitly trust the following self-signed certs
# Use the Subject Alternate Name to identify the cert

cert_trust "192.168.116.16"       Local system's certificate Subject Alt Name
cert_trust "192.168.13.213"       Remote system's certificate Subject Alt name

…
{
 label "JA-enigma to US-partym"
 local_id_type dn
 local_id "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax"
 remote_id "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"

 local_addr  192.168.116.16
 remote_addr 192.168.13.213

 p1_xform
  {auth_method rsa_sig oakley_group 2 auth_alg sha256 encr_alg aes}
}
```

■ **証明書要求**

CA が `cert_root` キーワードの値として提供する名前を入力します。たとえば、`enigma` システムの `ike/config` ファイルは次のようになります。

```
# Trusted root cert
# This certificate is from Example CA
# This is the X.509 distinguished name for the CA that it issues.

cert_root "C=US, O=ExampleCA\, Inc., OU=CA-Example, CN=Example CA"

…
{
 label "JA-enigma to US-partym - Example CA"
```

```
 local_id_type dn
 local_id "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax"
 remote_id  "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"

 local_addr  192.168.116.16
 remote_addr 192.168.13.213

 p1_xform
  {auth_method rsa_sig oakley_group 2 auth_alg sha256 encr_alg aes}
}
```

5. **通信先から受け取った証明書をハードウェアに格納します。**

   ステップ 2 で応答したように、PIN 要求に応答します。

   注記 - 公開鍵証明書は、公開鍵を生成したハードウェアに追加する*必要があります*。

   ■ **自己署名付き証明書。**

   リモートシステムの自己署名付き証明書を追加します。この例では、証明書は DCA.ACCEL.STOR.CERT ファイルに格納されています。

```
# ikecert certdb -a -T dca0-accel-stor < DCA.ACCEL.STOR.CERT
Enter PIN for PKCS#11 token:      Type user:password
```

   自己署名付き証明書が rsa_encrypt を auth_method パラメータの値として使用していた場合、ピアの証明書をハードウェア格納場所に追加します。

   ■ **CA からの証明書。**

   CA があなたの証明書要求と組織の証明書から生成した証明書を追加します。

   中間証明書の追加が必要な場合もあります。

```
# ikecert certdb -a -T dca0-accel-stor < DCA.ACCEL.STOR.CERT
Enter PIN for PKCS#11 token:      Type user:password

# ikecert certdb -a -T dca0-accel-stor < DCA.ACCEL.STOR.CA.CERT
Enter PIN for PKCS#11 token:      Type user:password
```

   CA からの証明書失効リスト (CRL) を追加するには、199 ページの「IKEv1 で失効した証明書を処理する方法」を参照してください。

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他

の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

▼ IKEv1 で失効した証明書を処理する方法

失効した証明書とは、なんらかの理由で効力を失った証明書です。失効した証明書を使用していると、セキュリティーのリスクとなります。証明書の失効を確認する際のオプションがあります。静的なリストを使用するか、HTTP プロトコルを介して失効を動的に確認できます。失効した証明書を処理する 4 つの方法があります。

- 証明書に CRL または OCSP の Uniform Resource Indicator (URI) が組み込まれている場合に、CRL または OCSP を無視するように IKEv1 に指示できます。このオプションは ステップ 5 に示しています。
- CA からの公開鍵証明書に URI のアドレスが組み込まれている場合に、その URI から CRL または OCSP にアクセスするように IKEv1 に指示できます。
- CA からの公開鍵証明書に LDAP サーバーの DN (ディレクトリ名) エントリが組み込まれている場合に、その LDAP サーバーから CRL にアクセスするように IKEv1 に指示できます。
- `ikecert certrldb` コマンドの引数として CRL を指定できます。例については、例 10-3「CRL を IKEv1 のローカルの `certrldb` データベースに貼り付ける」を参照してください。

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **CA から受け取った証明書を表示します。**

   `ikecert certdb` コマンドの引数については、`ikecert`(1M) のマニュアルページを参照してください。

   たとえば、次の証明書は企業の PKI から発行されました。詳細は変更されています。

```
# ikecert certdb -lv cert-protect.example.com
Certificate Slot Name: 0   Type: dsa-sha256
   (Private key in certlocal slot )
 Subject Name: <O=Example, CN=cert-protect.example.com>
 Issuer Name: <CN=ExampleCo CO (Cl B), O=Example>
 SerialNumber: 14000D93
   Validity:
```

```
   Not Valid Before: 2013 Sep 19th, 21:11:11 GMT
   Not Valid After:  2017 Sep 18th, 21:11:11 GMT
Public Key Info:
   Public Modulus  (n) (2048 bits): C575A…A5
   Public Exponent (e) (  24 bits): 010001
Extensions:
   Subject Alternative Names:
           DNS = cert-protect.example.com
   Key Usage: DigitalSignature KeyEncipherment
   [CRITICAL]
CRL Distribution Points:
   Full Name:
      URI = #Ihttp://www.example.com/pki/pkismica.crl#i
      DN = <CN=ExampleCo CO (Cl B), O=Example>
   CRL Issuer:
   Authority Key ID:
   Key ID:              4F … 6B
   SubjectKeyID:        A5 … FD
   Certificate Policies
   Authority Information Access
```

`CRL Distribution Points` エントリに注目してください。

- URI エントリは、この機関の CRL が Web 上にあることを示しています。
- DN エントリは、CRL が LDAP サーバー上にあることを示しています。一度、IKE がアクセスすると、CRL は将来に備えてキャッシュに格納されます。

CRL にアクセスするには、配布ポイントまで到達する必要があります。

2. **中央の配布ポイントから CRL にアクセスするには、次のメソッドのうちの 1 つを選択します。**

   ■ **URI を使用します。**

   キーワード `use_http` をホストの `/etc/inet/ike/config` ファイルに追加します。たとえば、`ike/config` ファイルは次のようになります。

   ```
   # Use CRL or OCSP from organization's URI
   use_http
   …
   ```

   ■ **Web プロキシを使用します。**

   キーワード `proxy` を `ike/config` ファイルに追加します。キーワード `proxy` は、次のように引数として URL を取ります。

   ```
   # Use web proxy to reach CRLs or OCSP
   proxy "http://proxy1:8080"
   ```

   ■ **LDAP サーバーを使用します。**

ホストの /etc/inet/ike/config ファイルの ldap-list キーワードに引数として LDAP サーバーの名前を指定します。LDAP サーバーの名前は、使用する機関にたずねてください。ike/config ファイルのエントリは次のようになります。

```
# Use CRL from organization's LDAP
ldap-list "ldap1.example.com:389,ldap2.example.com"
…
```

IKE は CRL を取り出し、証明書の期限が切れるまで CRL を保持します。

例 10-3 CRL を IKEv1 のローカルの certrldb データベースに貼り付ける

CA の CRL を中央配布ポイントから利用できない場合は、ローカルの certrldb データベースに CRL を手動で追加できます。CA の指示に従って CRL をファイルに抽出し、その CRL を ikecert certrldb -a コマンドでデータベースに追加します。

```
# ikecert certrldb -a < ExampleCo.Cert.CRL
```

移動体システム用の IKEv1 の構成

ソースと宛先を識別するために、IPsec と IKE は一意の ID を必要とします。一意の IP アドレスを持たない遠隔地のシステムまたは移動体システムの場合、別の種類の ID を使用する必要があります。システムを一意に識別するために、DNS、DN、または email などの ID の種類を使用できます。

一意の IP アドレスを持つ遠隔地のシステムまたは移動体システムで、別の種類の ID で構成するようにします。たとえば、システムが NAT 越しに中央システムに接続しようとした場合、そのシステムの一意なアドレスは使用されません。NAT ボックスが任意の IP アドレスを割り当てるため、中央システムは認識できません。

事前共有鍵は固定 IP アドレスを必要とするため、事前共有鍵も移動体システム用の認証メカニズムとしては機能しません。自己署名付き証明書、または CA からの証明書を使用すると、移動体システムは中央サイトと通信できます。

次のタスクマップは、リモートから中央サイトにログインするシステムを処理するために IKEv1 を構成する手順を一覧表示したものです。

表 10-2　移動体システム用の IKEv1 の構成のタスクマップ

タスク	説明	参照先
オフサイトから中央サイトへ通信します。	遠隔地のシステムが中央サイトと通信できるようにします。遠隔地のシステムは移動体システムの可能性もあります。	202 ページの「遠隔地のシステム用に IKEv1 を構成する方法」
移動体システムからのトラフィックを受信する中央システムで CA の公開証明書と IKEv1 を使用します。	固定 IP アドレスを持たないシステムからの IPsec トラフィックを受信するゲートウェイシステムを構成します。	例10-4「移動体システムから保護されたトラフィックを受信するために IKEv1 を使用する中央コンピュータを構成する」
固定 IP アドレスを持たないシステムで CA の公開証明書と IKEv1 を使用します。	中央サイト (会社の本社など) とのトラフィックを保護するように、移動体システムを構成します。	例10-5「NAT 越しのシステムを IPsec と IKEv1 で構成する」
移動体システムからのトラフィックを受信する中央システムで自己署名付き証明書と IKEv1 を使用します。	移動体システムから IPsec トラフィックを受信するように、ゲートウェイシステムを自己署名付き証明書で構成します。	例10-6「移動体システムからの自己署名付き証明書の受け入れ」
固定 IP アドレスを持たないシステムで自己署名付き証明書と IKEv1 を使用します。	中央サイトとのトラフィックを保護するように、移動体システムを自己署名付き証明書で構成します。	例10-7「自己署名付き証明書による中央システムとの接続」

▼ 遠隔地のシステム用に IKEv1 を構成する方法

始める前に　`root` 役割になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。リモートで管理する場合は、例7-1「`ssh` 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **移動体システムを認識するように、中央システムを構成します。**

   a. **`ipsecinit.conf` ファイルを構成します。**

      中央システムには、IP アドレスの広い範囲を許可するポリシーを必要とします。そのあと、IKE ポリシーの証明書で接続システムが合法であることを確認します。

      ```
      # /etc/inet/ipsecinit.conf on central
      # Keep everyone out unless they use this IPsec policy:
      {} ipsec {encr_algs aes encr_auth_algs sha256 sa shared}
      ```

   b. **IKEv1 構成ファイルを構成します。**

      DNS は中央システムを識別します。証明書を使用して、システムを認証します。

      ```
      ## /etc/inet/ike/ike.config on central
      ```

```
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
#
# Use web proxy
proxy "http://somecache.domain:port/"
#
# Use LDAP server
ldap_server   "ldap-server1.domain.org,ldap2.domain.org:port"
#
# List CA-signed certificates
cert_root    "C=US, O=Domain Org, CN=Domain STATE"
#
# List self-signed certificates - trust server and enumerated others
#cert_trust    "DNS=central.domain.org"
#cert_trust    "DNS=mobile.domain.org"
#cert_trust    "DN=CN=Domain Org STATE (CLASS), O=Domain Org
#cert_trust    "email=root@central.domain.org"
#cert_trust    "email=user1@mobile.domain.org"
#

# Rule for mobile systems with certificate
{
  label "Mobile systems with certificate"
  local_id_type DNS
# CA's public certificate ensures trust,
# so allow any remote_id and any remote IP address.
  remote_id ""
  remote_addr 0.0.0.0/0

p2_pfs 5

p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg aes auth_alg sha256 }
}
```

2. **各移動体システムにログインして、中央システムを見つけるように構成します。**

   a. **/etc/hosts ファイルを構成します。**

   /etc/hosts ファイルは、移動体システムのアドレスを必要としませんが、提供は可能です。このファイルは、中央システムの公開 IP アドレスである *central* を含んでいる必要があります。

   ```
   # /etc/hosts on mobile
   central 192.xxx.xxx.x
   ```

   b. **ipsecinit.conf ファイルを構成します。**

移動体システムは、公開 IP アドレスで中央システムを見つける必要があります。システムは同じ IPsec ポリシーで構成する必要があります。

```
# /etc/inet/ipsecinit.conf on mobile
# Find central
{raddr 192.xxx.xxx.x} ipsec {encr_algs aes encr_auth_algs sha256 sa shared}
```

c. IKEv1 構成ファイルを構成します。

識別子は IP アドレスであってはなりません。移動体システムに有効な識別子は次のとおりです。

- DN=*ldap-directory-name*
- DNS=*domain-name-server-address*
- email=*email-address*

証明書は、移動体システムである *mobile* の認証に使用されます。

```
## /etc/inet/ike/ike.config on mobile
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
#
# Use web proxy
proxy "http://somecache.domain:port/"
#
# Use LDAP server
ldap_server   "ldap-server1.domain.org,ldap2.domain.org:port"
#
# List CA-signed certificates
cert_root    "C=US, O=Domain Org, CN=Domain STATE"
#
# Self-signed certificates - trust me and enumerated others
#cert_trust    "DNS=mobile.domain.org"
#cert_trust    "DNS=central.domain.org"
#cert_trust    "DN=CN=Domain Org STATE (CLASS), O=Domain Org
#cert_trust    "email=user1@domain.org"
#cert_trust    "email=root@central.domain.org"
#
# Rule for off-site systems with root certificate
{
 label "Off-site mobile with certificate"
 local_id_type DNS

# NAT-T can translate local_addr into any public IP address
# central knows me by my DNS
```

```
 local_id "mobile.domain.org"
 local_addr 0.0.0.0/0

# Find central and trust the root certificate
 remote_id "central.domain.org"
 remote_addr 192.xxx.xxx.x

p2_pfs 5

p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg aes auth_alg sha256 }
}
```

3. **ike:default サービスを有効にします。**

```
# svcadm enable svc:/network/ipsec/ike:default
```

例 10-4 移動体システムから保護されたトラフィックを受信するために IKEv1 を使用する中央コンピュータを構成する

IKE は、NAT ボックス越しのネゴシエーションを開始できます。しかし、IKE の理想的な設定は NAT ボックスをはさまないことです。次の例では、CA の公開証明書は移動体システムと中央システムに格納されています。中央システムは NAT 越しのシステムからの IPsec ネゴシエーションを受け入れます。main1 は、遠隔地のシステムからの接続を受け入れることができる会社のシステムです。遠隔地のシステムを設定する方法については、例10-5「NAT 越しのシステムを IPsec と IKEv1 で構成する」を参照してください。

```
## /etc/hosts on main1
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on main1
# Keep everyone out unless they use this IPsec policy:
{} ipsec {encr_algs aes encr_auth_algs sha256 sa shared}

## /etc/inet/ike/ike.config on main1
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
#
# Use web proxy
proxy "http://cache1.domain.org:8080/"
#
# Use LDAP server
ldap_server   "ldap1.domain.org,ldap2.domain.org:389"
#
# List CA-signed certificate
cert_root "C=US, O=ExampleCA Inc, OU=CA-Example, CN=Example CA"
#
```

```
# Rule for off-site systems with root certificate
{
  label "Off-site system with root certificate"
  local_id_type DNS
  local_id "main1.domain.org"
  local_addr 192.168.0.100

# CA's public certificate ensures trust,
# so allow any remote_id and any remote IP address.
  remote_id ""
  remote_addr 0.0.0.0/0

p2_pfs 5

p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg aes auth_alg sha256}
p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg aes auth_alg sha256}
p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg aes auth_alg sha256}
p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg aes auth_alg sha256}
}
```

例 10-5 NAT 越しのシステムを IPsec と IKEv1 で構成する

次の例では、CA の公開証明書は移動体システムと中央システムに格納されています。`mobile1` は、家から会社の本社に接続しています。インターネットサービスプロバイダ (ISP) ネットワークは NAT ボックスを使用しているため、ISP は`mobile1` に非公開アドレスを割り当てることができます。NAT ボックスは、非公開アドレスを公開 IP アドレスに変換します。この公開アドレスは、ほかの ISP ネットワークノードと共有されます。企業の本社は NAT を越えません。企業の本社のコンピュータを設定する方法については、例10-4「移動体システムから保護されたトラフィックを受信するために IKEv1 を使用する中央コンピュータを構成する」を参照してください。

```
## /etc/hosts on mobile1
mobile1 10.1.3.3
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on mobile1
# Find main1
{raddr 192.168.0.100} ipsec {encr_algs aes encr_auth_algs sha256 sa shared}

## /etc/inet/ike/ike.config on mobile1
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
#
# Use web proxy
```

```
proxy "http://cache1.domain.org:8080/"
#
# Use LDAP server
ldap_server   "ldap1.domain.org,ldap2.domain.org:389"
#
# List CA-signed certificate
cert_root "C=US, O=ExampleCA Inc, OU=CA-Example, CN=Example CA"
#
# Rule for off-site systems with root certificate
{
  label "Off-site mobile1 with root certificate"
  local_id_type DNS
  local_id "mobile1.domain.org"
  local_addr 0.0.0.0/0

# Find main1 and trust the root certificate
  remote_id "main1.domain.org"
  remote_addr 192.168.0.100

p2_pfs 5

p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg aes auth_alg sha256 }
}
```

例 10-6 移動体システムからの自己署名付き証明書の受け入れ

次の例では、自己署名付き証明書が発行されており、移動体システムと中央システムに格納されています。main1 は、遠隔地のシステムからの接続を受け入れることができる会社のシステムです。遠隔地のシステムを設定する方法については、例10-7「自己署名付き証明書による中央システムとの接続」を参照してください。

```
## /etc/hosts on main1
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on main1
# Keep everyone out unless they use this IPsec policy:
{} ipsec {encr_algs aes encr_auth_algs sha256 sa shared}

## /etc/inet/ike/ike.config on main1
# Global parameters
#
# Self-signed certificates - trust me and enumerated others
cert_trust    "DNS=main1.domain.org"
cert_trust    "jdoe@domain.org"
cert_trust    "user2@domain.org"
cert_trust    "user3@domain.org"
#
# Rule for off-site systems with trusted certificate
{
  label "Off-site systems with trusted certificates"
  local_id_type DNS
```

```
  local_id "main1.domain.org"
  local_addr 192.168.0.100

# Trust the self-signed certificates
# so allow any remote_id and any remote IP address.
  remote_id ""
  remote_addr 0.0.0.0/0

p2_pfs 5

p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg aes auth_alg sha256 }
}
```

例 10-7 自己署名付き証明書による中央システムとの接続

次の例では、mobile1 は家から会社の本社に接続しています。自己署名付き証明書が発行されており、移動体システムと中央システムに格納されています。ISP ネットワークは NAT ボックスを使用しているため、ISP は mobile1 に非公開アドレスを割り当てることができます。NAT ボックスは、非公開アドレスを公開 IP アドレスに変換します。この公開アドレスは、ほかの ISP ネットワークノードと共有されます。企業の本社は NAT を越えません。企業の本社のコンピュータを設定する方法については、例10-6「移動体システムからの自己署名付き証明書の受け入れ」を参照してください。

```
## /etc/hosts on mobile1
mobile1 10.1.3.3
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on mobile1
# Find main1
{raddr 192.168.0.100} ipsec {encr_algs aes encr_auth_algs sha256 sa shared}

## /etc/inet/ike/ike.config on mobile1
# Global parameters

# Self-signed certificates - trust me and the central system
cert_trust    "jdoe@domain.org"
cert_trust    "DNS=main1.domain.org"
#
# Rule for off-site systems with trusted certificate
{
  label "Off-site mobile1 with trusted certificate"
  local_id_type email
  local_id "jdoe@domain.org"
  local_addr 0.0.0.0/0

# Find main1 and trust the certificate
  remote_id "main1.domain.org"
  remote_addr 192.168.0.100

p2_pfs 5
```

```
p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg aes auth_alg sha256 }
}
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

接続したハードウェアを検出するように IKEv1 を構成する

公開鍵証明書はまた、接続されたハードウェアに格納できます。Sun Crypto Accelerator 6000 ボードによってストレージが提供され、公開鍵の操作をシステムからこのボードにオフロードできます。

▼ Sun Crypto Accelerator 6000 ボードを検出するように IKEv1 を構成する方法

始める前に 次の手順では、Sun Crypto Accelerator 6000 ボードがシステムに接続されていると仮定します。さらに、ボードに必要なソフトウェアがすでにインストールされ、構成されているものとします。手順については、Sun Crypto Accelerator 6000 ボード製品のライブラリドキュメント (http://docs.oracle.com/cd/E19321-01/index.html)を参照してください。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、例7-1「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell を使用して ZFS をリモートで管理する方法」でセキュアなリモートログイン手順を確認してください。

1. **PKCS #11 ライブラリがリンクされていることを確認します。**

   IKEv1 はライブラリのルーチンを使用して、Sun Crypto Accelerator 6000 ボード上で鍵の生成および鍵の格納を処理します。

```
$ ikeadm get stats
…
PKCS#11 library linked in from /usr/lib/libpkcs11.so
$
```

2. **接続された Sun Crypto Accelerator 6000 ボードのトークン ID を見つけます。**

```
$ ikecert tokens
Available tokens with library "/usr/lib/libpkcs11.so":

"Sun Metaslot                    "
```

ライブラリは、32 文字のトークン ID (キーストア名 とも呼ぶ) を戻します。この例では、`ikecert` コマンドに `Sun Metaslot` トークンを使用すると、IKEv1 鍵を格納および高速化できます。

トークンを使用する手順については、195 ページの「ハードウェア上で IKEv1 の公開鍵証明書を生成および格納する方法」を参照してください。

`ikecert` コマンドにより、後続スペースが自動的に付加されます。

例 10-8 メタスロットトークンの検索と使用

トークンは、ディスク、接続されたボード、または暗号化フレームワークが提供するソフトトークンキーストアに格納できます。次に、ソフトトークンキーストアのトークン ID の例を示します。

```
$ ikecert tokens
Available tokens with library "/usr/lib/libpkcs11.so":

"Sun Metaslot                    "
```

ソフトトークンキーストアのパスフレーズを作成する方法については、`pktool(1)` のマニュアルページを参照してください。

次に、ソフトトークンキーストアに証明書を追加するコマンドの例を示します。`Sun.Metaslot.cert` は、CA 証明書を格納しているファイルです。

```
# ikecert certdb -a -T "Sun Metaslot" < Sun.Metaslot.cert
Enter PIN for PKCS#11 token:      Type user:passphrase
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、119 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他の例については、113 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

♦ ♦ ♦ 第 11 章

11 IPsec およびその鍵管理サービスのトラブルシューティング

この章では、IPsec とその鍵のトラブルシューティング方法、構成情報の表示方法、アクティブな IPsec、IKE、および手動鍵サービスに関する情報の表示方法について説明します。

この章では、次の内容について説明します。

IPsec およびその鍵管理構成のトラブルシューティング

トラブルシューティングが必要な問題の発生前または発生中に、システムのトラブルシューティングを設定できます。

トラブルシューティングの際は、Network IPsec Management 権利プロファイルを持つ管理者として、プロファイルシェルで多くのコマンドを実行できます。ただし、ログを読み取るには `root` 役割になる必要があります。

トラブルシューティングセクションのプロンプトは、コマンドの実行に権利が必要かどうかを示します。

- `#` プロンプト – 適切な管理権利またはそれらの権利がある役割を持つユーザーがコマンドを実行できます。
- `%` プロンプト – 通常のユーザーがコマンドを実行できます。

▼ IPsec および IKE システムのトラブルシューティングを準備する方法

IPsec およびその鍵管理サービスを有効にする前に、システムでトラブルシューティングに役立つログとツールを設定できます。

1. **IPsec および IKEv2 サービスのログの場所を指定します。**

   -L オプションはログのフルパスを指定します。これらのログには情報メッセージとエラーメッセージが含まれます。

   ```
   % svcs -L policy
   /var/svc/log/network-ipsec-policy:default.log

   % svcs -L ikev2
   /var/svc/log/network-ipsec-ike:ikev2.log
   ```

2. **IKEv2 のデバッグログファイルを構成します。**

   root 役割はこれらのログを読み取ることができます。

   ```
   % svccfg -s ikev2 listprop | grep debug
   config/debug_level             astring     op
   config/debug_logfile           astring     /var/log/ikev2/in.ikev2d.log
   ```

   デバッギングのレベルについては、ikeadm(1M) のマニュアルページに記載されています。値 verbose および all は、トラブルシューティングの際に便利です。

3. **(オプション) デバッグレベルを構成します。**

   次のコマンドはデバッグレベルを永続的に設定します。デバッグレベルを一時的に設定するには、例11-3「実行中の IKE デーモンの新しいデバッグレベルの設定」を参照してください。

   ```
   # svccfg -s ikev2 setprop config/debug_level = all
   ```

   ikev2 サービスが有効になっている場合、新しいデバッグレベルを使用するにはサービスをリフレッシュする必要があります。

   ```
   # svcadm refresh ikev2
   ```

4. **(オプション) wireshark パッケージをインストールします。**

   Wireshark アプリケーションは snoop 出力を読み取ることができます。

   ```
   % pkg info -r wireshark
            Name: diagnostic/wireshark
         Summary: Graphical network protocol analyzer
   ```

```
      Category: Applications/Internet
         State: Not installed
     Publisher: solaris
...
          FMRI: pkg://solaris/diagnostic/wireshark@version
# pkg install diagnostic/wireshark
```

▼ IPsec と IKE を実行する前にシステムをトラブルシューティングする方法

サービス実行前に、IPsec 構成ファイルの構文、IPsec 鍵ファイル、およびキーストア内の証明書の有効性を確認できます。

1. **IPsec 構成ファイルの構文を検証します。**

```
# ipsecconf -c /etc/inet/ipsecinit.conf
ipsecconf: Invalid pattern on line 5: ukp
ipsecconf: form_ipsec_conf error
ipsecconf: Malformed command (fatal):
{ ukp 58 type 133-137 dir out} pass {}

ipsecconf: 1 policy rule(s) contained errors.
ipsecconf: Fatal error - exiting.
```

出力でエラーが表示される場合は、検証が成功するまでエラーを修正してコマンドを実行します。

2. **ipseckeys ファイルの構文を検証します。**

```
# ipseckey -c /etc/inet/secret/ipseckeys
Config file /etc/inet/secret/ipseckeys has insecure permissions,
will be rejected in permanent config.
```

出力にエラーが表示されている場合は、エラーを修正してからサービスをリフレッシュします。

```
# svcadm refresh ipsec/policy
```

注記 - IKE 構成ファイルおよび IKE 事前共有鍵ファイルは、実行中の IKE デーモンによって検証されます。

3. **証明書の妥当性を検証します。**

- IKEv2 の自己署名付き証明書の妥当性を検証するには、160 ページの「自己署名付き公開鍵証明書により IKEv2 を構成する方法」の ステップ 4 を実行してください。

- IKEv2 で公開鍵証明書が失効していないことを確認するには、169 ページの「IKEv2 で証明書検証ポリシーを設定する方法」の手順に従ってください。
- IKEv1 の自己署名付き証明書の妥当性を検証するには、184 ページの「自己署名付き公開鍵証明書により IKEv1 を構成する方法」の ステップ 4 を実行してください。
- IKEv1 で公開鍵証明書が失効していないことを確認するには、199 ページの「IKEv1 で失効した証明書を処理する方法」の手順に従ってください。

次の手順 IPsec およびその鍵サービスを有効にしても構成が機能しない場合は、サービスの実行中にトラブルシューティングする必要があります。

▼ IPsec 実行中にシステムをトラブルシューティングする方法

IKE を使用してパケットを交換している、または交換を試みている実行中のシステム上で、`ikeadm` コマンドを使用して統計、ルール、事前共有鍵などの項目を表示できます。ログファイルおよび Wireshark アプリケーションなどの一部のツールを使用することもできます。

1. **次の項目を調査します。**

   - **ポリシーと適切な鍵管理サービスが有効になっていることを確認します。**

     次のテストシステムでは、鍵管理に `manual-key` サービスが使用されています。

     ```
     % svcs -a  | grep ipsec
     online         Feb_04   svc:/network/ipsec/manual-key:default
     online         Feb_04   svc:/network/ipsec/ipsecalgs:default
     online         Feb_04   svc:/network/ipsec/policy:default
     disabled       Feb_28   svc:/network/ipsec/ike:ikev2
     disabled       Feb_28   svc:/network/ipsec/ike:default
     ```

     このサービスが無効になっている場合は有効にします。

     両方の IKE サービスを同時に使用できます。手動鍵と IKE を同時に使用することもできますが、この構成はトラブルシューティングが困難な不具合を引き起こす場合があります。

   - **IKEv2 サービスのログファイルの終わりを表示します。**

     ```
     # svcs -xL ikev2
     svc:/network/ipsec/ike:ikev2 (IKEv2 daemon)
      State: disabled since October  10, 2013 10:10:40 PM PDT
     Reason: Disabled by an administrator.
     ```

```
      See: http://support.oracle.com/msg/SMF-8000-05
      See: in.ikev2d(1M)
      See: /var/svc/log/network-ipsec-ike:ikev2.log
Impact: This service is not running.
      Log:
Oct 01 13:20:20: (1)  Property "debug_level" set to: "op"
Oct 01 13:20:20: (1)  Errors and debug messages will be written to:
                        /var/log/ikev2/in.ikev2d.log
[ Oct 10 10:10:10 Method "start" exited with status 0. ]
[ Oct 10 10:10:40 Stopping because service disabled. ]
[ Oct 10 10:10:40 Executing stop method (:kill). ]

      Use: 'svcs -Lv svc:/network/ipsec/ike:ikev2' to view the complete log.
```

■ **(オプション) 実行中のデーモンのデバッグレベルの一時的な値を設定できます。**

```
# ikeadm set debug verbose /var/log/ikev2/in.ikev2d.log
Successfully changed debug level from 0x80000000 to 0x6204
Debug categories enabled:
        Operational / Errors
        Config file processing
        Interaction with Audit
        Verbose Operational
```

2. **ipsecconf コマンドの出力がポリシーファイルのコンテンツと一致していることを確認します。**

```
# ipsecconf
#INDEX 14
...
{  laddr 10.133.66.222 raddr 10.133.64.77 }
 ipsec   { encr_algs aes(256) encr_auth_algs sha512 sa shared }
...
{  laddr 10.134.66.122 raddr 10.132.55.55 }
 ipsec   { encr_algs aes(256) encr_auth_algs sha512 sa shared }

# cat /etc/inet/ipsecinit.conf
...
{  laddr 10.133.66.222 raddr 10.133.64.77 }
 ipsec   { encr_algs aes(256) encr_auth_algs sha512 sa shared }

{  laddr 10.134.66.122 raddr 10.132.55.55 }
 ipsec   { encr_algs aes(256) encr_auth_algs sha512 sa shared }
```

注記 - ワイルドカードアドレスによって一致しているかどうかがわかりにくい場合があるため、ipsecinit.conf ファイル内の特定のアドレスが ipsecconf の出力のワイルドカードアドレスの範囲内に含まれていることを確認してください。

ipsecconf コマンドの出力が表示されない場合は、ポリシーサービスが有効になっていることを確認してサービスをリフレッシュしてください。

```
% svcs policy
```

```
STATE          STIME    FMRI
online         Apr_10   svc:/network/ipsec/policy:default
```

出力にエラーが表示される場合は、ipsecinit.conf ファイルを編集してエラーを修正してからサービスをリフレッシュしてください。

3. **IKEv2 構成を確認します。**

修正を必要とする可能性のある構成の出力については、例11-1「無効な IKEv2 構成の修正」および例11-2「一致しないルールメッセージの修正」を参照してください。次の例の出力は、構成が有効であることを示しています。

```
# /usr/lib/inet/in.ikev2d -c
Feb 04 12:08:25: (1)    Reading service properties from smf(5) repository.
Feb 04 12:08:25: (1)    Property "config_file" set to: "/etc/inet/ike/ikev2.config"
Feb 04 12:08:25: (1)    Property "debug_level" set to: "all"
Feb 04 12:08:25: (1)    Warning: debug output being written to stdout.
Feb 04 12:08:25: (1)    Checking IKE rule #1: "Test 104 to 113"
Feb 04 12:08:25: (1)    Configuration file /etc/inet/ike/ikev2.config is valid.
Feb 04 12:08:25: (1)    Pre-shared key file /etc/inet/ike/ikev2.preshared is valid.
```

注記 - デバッグ出力に関する警告は、デバッグログファイルを指定したあとでも変更されません。debug_logfile サービスプロパティーの値を指定する場合、この警告はデバッグ出力がそのファイルに渡されていることを意味します。それ以外の場合、デバッグ出力はコンソールに渡されます。

- Checking IKE rule 行で、IKE ルールに適切な IP アドレスが接続していることを確認します。たとえば、次のエントリは一致しています。ipsecinit.conf ファイルの laddr の値が ikev2.config ファイルの local_addr の値と一致し、リモートアドレス同士が一致しています。

```
{  laddr 10.134.64.104 raddr 10.134.66.113 }      /** ipsecinit.conf **/
                ipsec {encr_algs aes encr_auth_algs sha512 sa shared}


 local_addr   10.134.64.104                          /** ikev2.config **/
 remote_addr  10.134.66.113                          /** ikev2.config **/
```

エントリが対応していない場合は、構成を修正して正しい IP アドレスを識別します。

注記 - ルールには、アドレスの範囲を示す 10.134.0.0/16 のようなワイルドカードアドレスが含まれる場合があります。特定のアドレスに対して範囲を確認します。

- `Pre-shared key file` 行にファイルが有効でないことが示されている場合は、ファイルを修正します。

  入力ミスを確認します。また、IKEv2 で、`ikev2.config` 内のルールのラベル値が `ikev2.preshared` ファイル内のラベル値と一致することを確認します。2 つの鍵を使用している場合は、次に、1 つのシステム上のローカル事前共有鍵がピアのリモート事前共有鍵と一致し、リモート鍵がピアのローカル鍵と一致することを確認します。

  構成がまだ機能しない場合は、218 ページの「IPsec および IKE の意味上の誤りのトラブルシューティング」を参照してください。

例 11-1 無効な IKEv2 構成の修正

次の出力では、IKE SA のライフタイムが短すぎます。

```
# /usr/lib/inet/in.ikev2d -c
...
May 08 08:52:49: (1) WARNING: Problem in rule "Test 104 to 113"
May 08 08:52:49: (1)  HARD lifetime too small (60 < 100)
May 08 08:52:49: (1)   -> Using 100 seconds (minimum)
May 08 08:52:49: (1) Checking IKE rule #1: "config 10.134.13.113 to 10.134.13.104"
...
```

この値は `ikev2.config` ファイルに明示的に設定されています。警告を解除するには、ライフタイムの値を少なくとも `100` に変更してサービスをリフレッシュします。

```
# pfedit /etc/inet/ike/ikev2.config
...
## childsa_lifetime_secs   60
childsa_lifetime_secs   100
...
# /usr/lib/inet/in.ikev2d -c
...
# svcadm refresh ikev2
```

例 11-2 一致しないルールメッセージの修正

次の出力では、事前共有鍵は定義されていますがルールに使用されていません。

```
# /usr/lib/inet/in.ikev2d -c
Feb 4 12:58:31: (1)  Reading service properties from smf(5) repository.
Feb 4 12:58:31: (1)  Property "config_file" set to: "/etc/inet/ike/ikev2.config"
Feb 4 12:58:31: (1)  Property "debug_level" set to: "op"
Feb 4 12:58:31: (1)  Warning: debug output being written to stdout.
Feb 4 12:58:31: (1)  Checking IKE rule #1: "Test 104 to 113"
Feb 4 12:58:31: (1)  Configuration file /etc/inet/ike/ikev2.config is valid.
Feb 4 12:58:31: (1)  No matching IKEv2 rule for pre-shared key ending on line 12
Feb 4 12:58:31: (1)  Pre-shared key file /etc/inet/ike/ikev2.preshared is valid.
```

この出力は、ルールが 1 つだけ存在することを示しています。

- ルールが事前共有鍵を必要とする場合、事前共有鍵のラベルがルールのラベルと一致しません。ikev2.config ルールのラベルと ikev2.preshared 鍵のラベルが一致するように修正します。
- ルールで証明書が使用されている場合は、ikev2.preshared ファイルの 12 行目で終わる事前共有鍵を削除するかコメントを解除すると No matching メッセージを回避できます。

例 11-3 実行中の IKE デーモンの新しいデバッグレベルの設定

次の出力では、ikev2 サービスのデバッグ出力が all に設定されています。

```
# /usr/lib/inet/in.ikev2d -c
Feb 4 12:58:31: (1)  Reading service properties from smf(5) repository.
...
Feb 4 12:58:31: (1)  Property "debug_level" set to: "all"
...
```

213 ページの「IPsec と IKE を実行する前にシステムをトラブルシューティングする方法」の ステップ 2 を実行してもデバッグ出力が all ではなく op の場合は、ikeadm コマンドを使用して、実行中の IKE デーモンのデバッグレベルを設定してください。

```
# ikeadm set debug_level all
```

IPsec および IKE の意味上の誤りのトラブルシューティング

214 ページの「IPsec 実行中にシステムをトラブルシューティングする方法」の調査で問題に対処できない場合は、ファイルの構文やサービス構成ではなく、構成の意味に問題があると考えられます。

- ike:default と ike:ikev2 両方のサービスインスタンスが有効になっている場合は、IKEv2 と IKEv1 のルールが重複しないようにしてください。同じネットワークエンドポイントに適用されるルールによって IPsec SA が重複することがあり、これによって特定の状況で接続が失われる可能性があります。

  IKE ルールを変更する場合は、ルールをカーネルに読み込みます。

  ```
  # ikeadm -v[1|2] read rule
  ```

- IKEv1 を実行中の場合は、ルール内のアルゴリズムのメカニズムが接続先の IKEv1 システムで利用できることを確認してください。利用可能なアルゴリズムを表示するに

は、IKEv2 をサポートしていないシステム上で `ikeadm dump` *algorithms* コマンドを実行します。

```
# ikeadm dump groups      Available Diffie-Hellman groups
# ikeadm dump encralgs    All IKE encryption algorithms
# ikeadm dump authalgs    All IKE authentication algorithms
```

両方のシステムで利用可能なアルゴリズムを使用するように、IPsec と IKEv1 両方のポリシーファイルを修正します。その後、IKEv1 サービスを再起動して IPsec サービスをリフレッシュします。

```
# svcadm restart ike:default; svcadm refresh ipsec/policy
```

- IKEv1 で事前共有鍵を使用中にリモートの IKEv1 システムをリブートする場合は、ローカルシステム上で `ipseckey flush` コマンドを実行します。
- 自己署名付き証明書を使用している場合は、同じ DN を持つ証明書が再作成されていないことと、証明書のハッシュ値が一致していることを相手の管理者に確認します。確認の手順については、160 ページの「自己署名付き公開鍵証明書により IKEv2 を構成する方法」の ステップ 4 を参照してください。

  証明書を更新する場合は、新しい証明書をインポートしたあと、IKEv2 サービスをリフレッシュして再起動してください。
- 現在の IKEv2 構成を表示するには `ikeadm -v2 dump | get` コマンドを使用します。使用方法のサマリーについては、221 ページの「IKE 情報の表示」を参照してください。
- IPsec 関連の統計を表示するには `kstat` コマンドを使用します。詳細は、`kstat`(1M) のマニュアルページを参照してください。

```
# kstat -m ipsecesp
# kstat -m ipsecah
# kstat -m ip
```

次の例の `kstat` 出力は、`ipsecesp` モジュールに問題がないことを示しています。

```
# kstat -m ipsecesp
module: ipsecesp                        instance: 0
name:   esp_stat                        class:    net
        acquire_requests                18
        bad_auth                        0
        bad_decrypt                     0
        bad_padding                     0
```

```
bytes_expired                   0
crtime                          4.87974774
crypto_async                    0
crypto_failures                 0
crypto_sync                     172
good_auth                       86
keysock_in                      135
num_aalgs                       9
num_ealgs                       13
out_discards                    0
out_requests                    86
replay_early_failures           0
replay_failures                 0
sa_port_renumbers               0
snaptime                        5946769.7947628
```

- 保護されていないトラフィックを表示するには snoop コマンドを使用します。Wireshark アプリケーションは snoop 出力を読み取ることができます。snoop 出力の例は、134 ページの「IPsec によってパケットが保護されていることを確認する方法」を参照してください。

IPsec およびその鍵サービスに関する情報の表示

注記 - ほとんどのコマンドでは、Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

IPsec および手動鍵サービスプロパティーの表示

IPsec ポリシーファイルと手動鍵を保持しているファイルの名前を表示できます。

- IPsec 構成ファイルの名前を表示するには:

```
% svccfg -s policy listprop config/config_file
config/config_file         astring     /etc/inet/ipsecinit.conf
```

- IPsec の手動鍵を保持しているファイルの名前を表示するには:

```
% svccfg -s manual-key listprop config/config_file
config/config_file         astring     /etc/inet/secret/ipseckeys
```

IKE 情報の表示

IKE サービスのプロパティー、IKE 状態および IKE デーモンオブジェクトの要素、および証明書検証ポリシーを表示できます。両方の IKE サービスを実行している場合は、サービスごとの情報または両方のサービスの情報を表示できます。これらのコマンドは、テスト、トラブルシューティング、およびモニター中に役立ちます。

- IKE サービスインスタンスのプロパティーの表示 – 出力には、構成ファイルの名前を含む IKEv2 サービスの構成可能なプロパティーが表示されます。

注記 - IPsec、IKEv2、または IKEv1 サービスの `config` グループ内のプロパティーを変更できるか、またはプロパティーを変更すべきかどうかを確認するには、`ipsecconf`(1M)、`in.ikev2d`(1M)、および `in.iked`(1M) のマニュアルページを確認してください。たとえば、IKEv2 構成ファイルは特殊な権限で作成され、`ikeuser` によって所有されます。権限とファイルの所有者は決して変更しないでください。

```
% svccfg -s ipsec/ike:ikev2 listprop config
config                      application
config/allow_keydump        boolean     false
config/config_file          astring     /etc/inet/ike/ikev2.config
config/ignore_errors        boolean     false
config/kmf_policy           astring     /etc/inet/ike/kmf-policy.xml
config/max_child_sas        integer     0
config/max_threads          integer     0
config/min_threads          integer     0
config/preshared_file       astring     /etc/inet/ike/ikev2.preshared
config/response_wait_time   integer     30
config/value_authorization  astring     solaris.smf.value.ipsec
config/debug_logfile        astring
config/debug_level          astring     op
```

次の例の出力には、IKEv1 サービスの構成可能なプロパティーが表示されています。:default サービスインスタンスは指定しないでください。

```
% svccfg -s ipsec/ike listprop config
config                       application
config/admin_privilege     astring     base
config/config_file         astring     /etc/inet/ike/config
config/debug_level         astring     op
config/debug_logfile       astring     /var/log/in.iked.log
config/ignore_errors       boolean     false
config/value_authorization astring     solaris.smf.value.ipsec
```

- IKE デーモンの現在の状態の表示 – 次の例の出力には、ikeadm コマンドの引数が表示されています。これらの引数はデーモンの現在の状態を表示します。

注記 - ikeadm コマンドを使用するには、IKE デーモンを実行している必要があります。

```
% ikeadm help
...
        get   debug|priv|stats|p1|ikesa|rule|preshared|defaults [identifier]
        dump  p1|ikesa|rule|preshared|certcache|groups|encralgs|authalgs
        read  rule|preshared [filename]
        help  [get|set|add|del|dump|flush|read|write|token|help]
```

- ikeadm コマンドに対する特定の引数の構文の表示 – コマンド引数の構文を表示するには、help サブコマンドを使用します。例:

```
% ikeadm help read
This command reads a new configuration file into
in.iked, discarding the old configuration info.

Sets of data that may be read include:
        rule            all phase 1/ikesa rules
        preshared       all preshared keys

A filename may be provided to specify a source file
other than the default.
```

- 事前共有鍵の表示 – IKEv1 および IKEv2 の事前共有鍵を表示できます。

注記 - 1 つの IKE バージョンのみを実行している場合は、-v オプションを省略できます。

IKEv2 の場合:

```
# ikeadm -v2 dump preshared
```

IKEv1 の場合:

```
# ikeadm set priv keymat
# ikeadm -v1 dump preshared

PSKEY: Rule label: "Test PSK 197 to 56"
PSKEY: Local pre-shared key (80 bytes): 74206272696c6c696720...3/584
PSKEY: Remote pre-shared key (80 bytes): 74206272696c6c696720...3/584

Completed dump of preshared keys
```

- IKE SA の表示 – 出力には、SA、変換、ローカルシステムとリモートシステム、およびその他の詳細に関する情報が含まれます。通信が要求されていない場合は、SA が存在しないため、表示する情報がありません。

```
# ikeadm -v2 dump ikesa
IKESA: SPIs: Local 0xd3db95689459cca4  Remote 0xb5878717f5cfa877
...
XFORM: Encryption alg: aes-cbc(256..256); Authentication alg: hmac-sha512
...
LOCIP: AF_INET: port 500, 10.1.2.3 (example-3).
...
REMIP: AF_INET: port 500, 10.1.4.5 (ex-2).
...
LIFTM: SA expires in 11459 seconds (3.18 hours)
...
STATS: 0 IKE SA rekeys since initial AUTH.
LOCID: Initiator identity, type FQDN
...
CHILD: ESP Inbound SPI: 0x94841ca3, Outbound SPI 0x074ae1e5
```

```
...
Completed dump of IKE SA info
```

- アクティブな IKE ルールの表示 – リストされる IKE ルールは使用中でない場合がありますが、使用は可能です。

```
# ikeadm -v2 dump rule

GLOBL: Label 'Test Rule1 for PSK', key manager cookie 1
GLOBL: Local auth method=pre-shared key
GLOBL: Remote auth method=pre-shared key

GLOBL: childsa_pfs=false
GLOBL: authentication_lifetime=86400 seconds (1.00 day)
GLOBL: childsa_lifetime=120 seconds (2.00 minutes)
GLOBL: childsa_softlife=108 seconds (1.80 minute)
GLOBL: childsa_idletime=60 seconds
GLOBL: childsa_lifetime_kb=122880 kilobytes (120.00 MB)
GLOBL: childsa_softlife_kb=110592 kilobytes (108.00 MB)
LOCIP: IP address range(s):
LOCIP: 10.142.245.197
REMIP: IP address range(s):
REMIP: 10.134.64.56
LOCID: Identity descriptors:
LOCID: Includes:
LOCID:       fqdn="gloria@ms.mag"
REMID: Identity descriptors:
REMID: Includes:
REMID:       fqdn="gloria@ms.mag"
XFRMS: Available Transforms:

XF  0: Encryption alg: aes-cbc(128..256); Authentication alg: hmac-sha512
XF  0: PRF: hmac-sha512 ; Diffie-Hellman Group: 2048-bit MODP (group 14)
XF  0: IKE SA lifetime before rekey: 14400 seconds (4.00 hours)

Completed dump of policy rules
```

- IKEv2 の証明書検証ポリシーの表示 – `dbfile` および `policy` の値を指定する必要があります。

- 動的にダウンロードされる CRL は、応答者タイムアウトの調整に管理者の介入が必要になる場合があります。

  次の例の出力では、証明書に組み込まれた URI から CRL がダウンロードされたあと、リストがキャッシュされます。キャッシュに期限切れの CRL が含まれている場合は、新しい CRL がダウンロードされて古い CRL を上書きします。

```
# kmfcfg list dbfile=/etc/inet/ike/kmf-policy.xml policy=default
…
Validation Policy Information:
    Maximum Certificate Revocation Responder Timeout: 10
    Ignore Certificate Revocation Responder Timeout: true
…
    CRL:
        Base filename: [not set]
        Directory: /var/user/ikeuser/crls
        Download and cache CRL: true
        CRL specific proxy override: www-proxy.cagate.example.com:80
        Ignore CRL signature: false
        Ignore CRL validity date: false
IPsec policy bypass on outgoing connections: true
…
```

- 静的にダウンロードされる CRL は、管理者が頻繁に確認する必要があります。

  管理者が次の値に CRL エントリを設定すると、その管理者は CRL の手動ダウンロード、ディレクトリの入力、および現在の CRL の維持に責任を負います。

```
…
        Directory: /var/user/ikeuser/crls
        Download and cache CRL: false
        Proxy: [not set]
…
```

IPsec およびその鍵サービスの管理

IPsec ポリシーはデフォルトで有効になっていますが、構成情報は含まれていません。

鍵の管理はデフォルトでは有効ではありません。IKE または手動鍵管理、あるいはその両方を構成できます。各 IKE ルールは、どの鍵管理サービスが使用されているかを示しています。ikeadm コマンドは実行中の IKE デーモンを変更できます。

IPsec およびその鍵サービスの構成および管理

- IPsec の構成およびリフレッシュのあとにポリシーを表示:

```
# pfedit /etc/inet/ipsecinit.conf
# ipsecconf -c /etc/inet/ipsecinit.conf
# svcadm refresh ipsec/policy
# ipsecconf -Ln
```

- IPsec の手動鍵の構成および有効化:

```
# pfedit -s /etc/inet/secret/ipseckeys
# svcadm enable ipsec/manual-key
```

- IKEv2 の構成および有効化:

```
# pfedit /etc/inet/ike/ikev2.config
# /usr/lib/inet/in.ikev2d -c
# svcadm enable ipsec/ike:ikev2
```

- IKEv1 の構成および有効化:

```
# pfedit /etc/inet/ike/config
# /usr/lib/inet/in.iked -c
# svcadm enable ipsec/ike:default
```

- IPsec および IKE が有効になっているシステム上でそれらのサービスが構成されていることを確認:

```
# ipsecconf -Ln
# ikeadm -v2 dump rule
# ikeadm set priv keymat
# ikeadm -v1 dump rule
```

- 鍵管理の変更:

  IKEv2 の場合:

```
# pfedit /etc/inet/ike/ikev2.config
# /usr/lib/inet/in.ikev2d -c
```

```
# svcadm restart ipsec/ike:ikev2
```

IKEv1 の場合:

```
# pfedit /etc/inet/ike/config
# /usr/lib/inet/in.iked -c
# svcadm restart ipsec/ike:default
```

手動鍵管理の場合:

```
# pfedit -s /etc/inet/secret/ipseckeys
# ipseckey -c /etc/inet/secret/ipseckeys
# svcadm refresh ipsec/manual-key
```

- IPsec および IKE の構成可能なプロパティーの変更:

  IPsec サービス:

  ```
  # svccfg -s ipsec/policy setprop config/property = value
  # svcadm refresh ipsec/policy; svcadm restart ipsec/policy
  ```

  IKEv2 サービス:

  ```
  # svccfg -s ike:ikev2 editprop
  # svcadm refresh ipsec/ike:ikev2; svcadm restart ipsec/ike:ikev2
  ```

  IKEv1 サービス:

  ```
  # svccfg -s ipsec/ike setprop config/property = value
  # svcadm refresh ipsec/ike:ikev2; svcadm restart ipsec/ike:ikev2
  ```

  手動鍵サービス:

  ```
  # svccfg -s ipsec/manual-key setprop config/property = value
  # svcadm refresh ipsec/manual-key; svcadm restart ipsec/manual-key
  ```

- IKEv2 の事前共有鍵の構成。

  ```
  # pfedit -s /etc/inet/ike/ikev2.preshared
  # /usr/lib/inet/in.ikev2d -c
  # svcadm restart ikev2
  ```

- IKEv1 の事前共有鍵の構成:

  ```
  # pfedit -s /etc/inet/secret/ike.preshared
  ```

```
# svcadm restart ike
```

実行中の IKE デーモンの管理

詳細は、ikeadm(1M) のマニュアルページを確認してください。このセクションのコマンドは、IKEv2 または IKEv1 デーモンの実行中のみ利用できます。

- 実行中の IKE デーモンの変更:

  次の出力は、現在のデーモンの状態を変更できる ikeadm コマンドの引数を示しています。一部の引数は IKEv2 または IKEv1 デーモンに固有です。

```
% ikeadm help
...
        set   priv level
        set   debug level [filename]
        add   rule|preshared {definition}|filename
        del   p1|ikesa|rule|preshared identifier
        flush p1|ikesa|certcache
        write rule|preshared filename
        token login|logout PKCS#11-Token-Object
```

- ikeadm コマンド固有の引数の構文を表示:

```
% ikeadm help add
This command adds items to in.iked's tables.

Objects that may be set include:
        rule            a phase 1 or IKE SA policy rule
        preshared       a preshared key

Objects may be entered on the command-line, as a
series of keywords and tokens contained in curly
braces ('{', '}'); or the name of a file containing
the object definition may be provided.

For security purposes, preshared keys may only be
entered on the command-line if ikeadm is running in
```

```
interactive mode.
```

- `ikeadm` コマンドで IKEv2 デーモンを変更:

```
# ikeadm add rule | preshared {definition} | filename
# ikeadm flush ikesa
# ikeadm del ikesa | rule | preshared identifier
# ikeadm set debug level
# ikeadm token login | logout PKCS#11-Token-Object
# ikeadm write rule | preshared filename
```

- `ikeadm` コマンドで IKEv1 デーモンを変更:

```
# ikeadm set debug level
# ikeadm set privlevel
# ikeadm add rule | preshared {definition} | filename
# ikeadm del p1 | rule | preshared identifier
# ikeadm flush p1 | certcache
# ikeadm del rule | preshared id
# ikeadm write rule | preshared filename
```

♦ ♦ ♦ 第 12 章

12 IPsec および鍵管理のリファレンス

この章には、IPsec、IKEv2、および IKEv1 に関する参照情報が記載されています。

使用しているネットワークに IPsec を実装する方法については、第7章「IPsec の構成」を参照してください。IPsec の概要については、第6章「IP セキュリティーアーキテクチャーについて」を参照してください。

IKE の実装方法については、第9章「IKEv2 の構成」を参照してください。概要については、第8章「インターネット鍵交換について」を参照してください。

IPsec リファレンス

IPsec のサービス、ファイル、およびコマンド

このセクションには、IPsec サービス、一部の IPsec RFC、および IPsec に関連するファイルとコマンドがリストされます。

IPsec サービス

サービス管理機能 (SMF) は、次の IPsec サービスを提供します。

- `svc:/network/ipsec/policy` サービス – IPsec ポリシーを管理します。デフォルトでは、このサービスは有効になっています。`config_file` プロパティーの値によって `ipsecinit.conf` ファイルの場所が決まります。DefaultFixed ネットワーク構成プロファイ

ルを実行しているシステムの初期値は /etc/inet/ipsecinit.conf です。このプロファイルを実行していないシステムでは、プロパティー値が空です。

- svc:/network/ipsec/ipsecalgs サービス – IPsec で使用できるアルゴリズムを管理します。デフォルトでは、このサービスは有効になっています。
- svc:/network/ipsec/manual-key サービス – 手動鍵管理をアクティブにします。デフォルトでは、このサービスは無効になっています。config_file プロパティーの値によって ipseckeys 構成ファイルの場所が決まります。初期値は /etc/inet/secret/ipseckeys です。
- svc:/network/ipsec/ike サービス – IKE を管理します。デフォルトでは、このサービスは無効になっています。構成可能なプロパティーについては、238 ページの「IKEv2 サービス」および 243 ページの「IKEv1 サービス」を参照してください。

SMF については、『Oracle Solaris 11.2 でのシステムサービスの管理 』の第 1 章「サービス管理機能の概要」を参照してください。smf(5)、svcadm(1M)、および svccfg(1M) のマニュアルページも参照してください。

ipsecconf コマンド

ホストの IPsec ポリシーを構成するには、ipsecconf コマンドを使用します。このコマンドを実行してポリシーを構成すると、IPsec ポリシーのエントリがカーネル内に作成されます。システムは、これらのエントリを使用して、インバウンドおよびアウトバウンドの IP パケットすべてがポリシーに沿っているかどうかを検査します。トンネリングされずに転送されるパケットは、このコマンドを使用して追加されるポリシー検査の対象になりません。また、ipsecconf コマンドはセキュリティーポリシーデータベース (SPD) 内の IPsec エントリも管理します。IPsec ポリシーオプションについては、ipsecconf(1M) のマニュアルページを参照してください。

ipsecconf コマンドを呼び出すには、root 役割になる必要があります。このコマンドは、両方向のトラフィックを保護するエントリを構成できます。このコマンドは、片方向だけのトラフィックを保護するエントリも構成できます。

ローカルアドレスとリモートアドレスというパターンのポリシーエントリは、1 つのポリシーエントリで両方向のトラフィックを保護します。たとえば、指定されたホストに対して方向が指定されていない場合、laddr host1 と raddr host2 というパターンを含むエントリは、両方向のトラフィックを保護します。そのため、各ホストにポリシーエントリを 1 つだけ設定すれば済みます。

ipsecconf コマンドで追加されたポリシーエントリには持続性がなく、システムのリブート時に失われます。システムのブート時に IPsec ポリシーが確実にアクティブになるようにするには、/etc/inet/ipsecinit.conf ファイルにポリシーエントリを追加したあと、policy サービスをリフレッシュするか有効化します。例については、111 ページの「IPsec によるネットワークトラフィックの保護」を参照してください。

ipsecinit.conf 構成ファイル

Oracle Solaris を起動したときに IPsec セキュリティーポリシーを有効化するには、特定の IPsec ポリシーエントリを使用して構成ファイルを作成し IPsec を初期化します。このファイルのデフォルトの名前は /etc/inet/ipsecinit.conf です。ポリシーエントリとその形式の詳細については、ipsecconf(1M) のマニュアルページを参照してください。ポリシーの構成が完了したら、svcadm refresh ipsec/policy コマンドでポリシーをリフレッシュできます。

サンプルの ipsecinit.conf ファイル

Oracle Solaris ソフトウェアには、サンプルの IPsec ポリシーファイル ipsecinit.sample が含まれます。このファイルをテンプレートとして独自の ipsecinit.conf ファイルを作成できます。ipsecinit.sample ファイルには、次のエントリが含まれています。

```
...
# In the following simple example, outbound network traffic between the local
# host and a remote host will be encrypted. Inbound network traffic between
# these addresses is required to be encrypted as well.
#
# This example assumes that 10.0.0.1 is the IPv4 address of this host (laddr)
# and 10.0.0.2 is the IPv4 address of the remote host (raddr).
#

{laddr 10.0.0.1 raddr 10.0.0.2} ipsec
 {encr_algs aes encr_auth_algs sha256 sa shared}

# The policy syntax supports IPv4 and IPv6 addresses as well as symbolic names.
# Refer to the ipsecconf(1M) man page for warnings on using symbolic names and
# many more examples, configuration options and supported algorithms.
#
# This example assumes that 10.0.0.1 is the IPv4 address of this host (laddr)
# and 10.0.0.2 is the IPv4 address of the remote host (raddr).
#
# The remote host will also need an IPsec (and IKE) configuration that mirrors
# this one.
#
# The following line will allow ssh(1) traffic to pass without IPsec protection:
```

```
{lport 22 dir both} bypass {}

#
# {laddr 10.0.0.1 dir in} drop {}
#
# Uncommenting the above line will drop all network traffic to this host unless
# it matches the rules above. Leaving this rule commented out will allow
# network packets that do not match the above rules to pass up the IP
# network stack. ,,,
```

ipsecinit.conf と ipsecconf のセキュリティーについて

確立された接続の IPsec ポリシーを変更することはできません。ポリシーの変更ができないソケットを、「*ラッチされたソケット*」と呼びます。新しいポリシーエントリは、すでにラッチされたソケットを保護しません。詳細は、connect(3SOCKET) および accept(3SOCKET) のマニュアルページを参照してください。自信がない場合は、接続を再起動してください。詳細は、ipsecconf(1M) のマニュアルページのセキュリティーのセクションを参照してください。

ipsecalgs コマンド

暗号化フレームワークは、認証と暗号化のアルゴリズムを IPsec に提供します。ipsecalgs コマンドを使用すると、各 IPsec プロトコルでサポートされているアルゴリズムを一覧表示できます。ipsecalgs の構成は /etc/inet/ipsecalgs ファイルに保存されます。通常、このファイルを変更する必要はなく、直接編集してはいけません。ただし、ファイルを変更する必要がある場合は、ipsecalgs コマンドを使用します。サポートされるアルゴリズムは、システムのブート時に svc:/network/ipsec/ipsecalgs:default サービスによってカーネルと同期されます。

有効な IPsec プロトコルおよびアルゴリズムは、RFC 2407 に記載されている ISAKMP 解釈ドメイン (DOI) によって記述されます。特に、ISAKMP DOI は、有効な IPsec アルゴリズム PROTO_IPSEC_AH とそのプロトコルの PROTO_IPSEC_ESP の命名規約や番号付け規約を定義します。1 つのアルゴリズムは 1 つのプロトコルだけに関連します。このような ISAKMP DOI 定義は、/etc/inet/ipsecalgs ファイルにあります。アルゴリズム番号とプロトコル番号は、Internet Assigned Numbers Authority (IANA) によって定義されます。ipsecalgs コマンドは、IPsec アルゴリズムのリストを拡張します。

アルゴリズムの詳細については、ipsecalgs(1M) のマニュアルページを参照してください。暗号化フレームワークの詳細については、『Oracle Solaris 11.2 での暗号化と証明書の管理 』の第 1 章「暗号化フレームワーク」を参照してください。

ipseckey コマンド

ipseckey コマンドは、各種オプションとともに IPsec の鍵を手動で管理します。ipseckey コマンドの説明については、ipseckey(1M) のマニュアルページを参照してください。

ipseckey におけるセキュリティーについて

ipseckey コマンドを使用すると、Network Security または Network IPsec Management 権利プロファイルを持つ役割は、暗号鍵に関する機密情報を入力できます。場合によっては、不正にこの情報にアクセスして IPsec トラフィックのセキュリティーを損なうことも可能です。

注記 - 可能な場合は、手動の鍵処理ではなく IKE を使用します。

詳細は、ipseckey(1M) のマニュアルページのセキュリティーのセクションを参照してください。

kstat コマンド

kstat コマンドは、ESP、AH、およびその他の IPsec データの統計を表示できます。IPsec 関連のオプションは、218 ページの「IPsec および IKE の意味上の誤りのトラブルシューティング」にリストされています。kstat(1M) のマニュアルページも参照してください。

snoop コマンドと IPsec

snoop コマンドは、AH ヘッダーと ESP ヘッダーを構文解析できます。ESP はそのデータを暗号化するため、ESP で暗号化および保護されたヘッダーは snoop コマンドでは読み取ることができません。AH はデータを暗号化しないため、AH で保護されたトラフィックは snoop コマンドで検査できます。このコマンドに -V オプションを指定すると、いつ AH がパケットに使用されているかを表示できます。詳細は、snoop(1M) のマニュアルページを参照してください。

保護されたパケットに snoop コマンドを実行した場合の詳細な出力については、134 ページの「IPsec によってパケットが保護されていることを確認する方法」を参照してください。

このリリースにバンドルされている無料のオープンソースソフトウェア Wireshark (http://www.wireshark.org/about.html) のように、サードパーティーのネットワークアナライザも利用できます。

IPsec RFC

インターネットエンジニアリングタスクフォース (IETF) は、IP 層のセキュリティーアーキテクチャーを説明するいくつかの RFC (Requests for Comments) を公表しています。RFC へのリンクについては、http://www.ietf.org/を参照してください。次の RFC リストは、比較的一般的な IP セキュリティーの参考文献です。

- RFC 2411、『IP Security Document Roadmap』、1998 年 11 月
- RFC 2401、『Security Architecture for the Internet Protocol』、1998 年 11 月
- RFC 2402、『IP Authentication Header』、1998 年 11 月
- RFC 2406、『IP Encapsulating Security Payload (ESP)』、1998 年 11 月
- RFC 2408、『Internet Security Association and Key Management Protocol (ISAKMP)』、1998 年 11 月
- RFC 2407、『The Internet IP Security Domain of Interpretation for ISAKMP』、1998 年 11 月
- RFC 2409、『The Internet Key Exchange (IKEv1)』、1998 年 11 月
- RFC 5996、『Internet Key Exchange Protocol Version 2 (IKEv2)』、2010 年 9 月
- RFC 3554、『On the Use of Stream Control Transmission Protocol (SCTP) with IPsec』、2003 年 7 月

IPsec のセキュリティーアソシエーションデータベース

IPsec セキュリティーサービスの鍵情報は、セキュリティーアソシエーションデータベース (SADB) に保存されます。セキュリティーアソシエーション (SA) は、インバウンドパケットとアウトバウンドパケットを保護します。

`in.iked` デーモンと `ipseckey` コマンドは `PF_KEY` ソケットインタフェースを使用して SADB を保守します。SADB が要求やメッセージを処理する方法の詳細については、`pf_key`(7P) のマニュアルページを参照してください。

IPsec での鍵管理

インターネット鍵交換 (IKE) プロトコルにより、IPsec の鍵管理が自動的に行われます。IPsec SA は `ipseckey` コマンドによって手動で管理することもできますが、IKE が推奨されます。詳細は、96 ページの「IPsec セキュリティーアソシエーションの鍵管理」を参照してください。

Oracle Solaris のサービス管理機能 (SMF) 機能は、次の IPsec 用鍵管理サービスを提供します。

- `svc:/network/ipsec/ike` サービス – 自動鍵管理のための SMF サービスです。`ike` サービスには 2 つのインスタンスがあります。`ike:ikev2` サービスインスタンスは `in.ikev2d` デーモン (IKEv2) を実行して自動鍵管理を提供します。`ike:default` サービスは `in.iked` デーモン (IKEv1) を実行します。IKE については、第8章「インターネット鍵交換について」を参照してください。デーモンの詳細については、`in.ikev2d`(1M) および `in.iked`(1M) のマニュアルページを参照してください。
- `svc:/network/ipsec/manual-key:default` サービス – 手動鍵管理のための SMF サービスです。`manual-key` サービスは `ipseckey` コマンドを各種オプションで実行して、鍵を手動で管理します。`ipseckey` コマンドの説明については、`ipseckey`(1M) のマニュアルページを参照してください。

IKEv2 リファレンス

IKEv2 は IKEv1 の後継です。比較については、143 ページの「IKEv2 と IKEv1 の比較」を参照してください。

IKEv2 のユーティリティーおよびファイル

次の表は、IKEv2 ポリシーの構成ファイル、IKEv2 鍵の保管場所、および IKEv2 を実装する各種コマンドとサービスについてまとめたものです。サービスの詳細については、『Oracle Solaris 11.2 でのシステムサービスの管理 』の第 1 章「サービス管理機能の概要」を参照してください。

表 12-1 IKEv2 のサービス名、コマンド、構成と鍵の保管場所、およびハードウェアデバイス

ファイル、場所、コマンド、またはサービス	説明	マニュアルページ
`svc:/network/ipsec/ike:ikev2`	IKEv2 を管理する SMF サービス。	`smf`(5)

ファイル、場所、コマンド、またはサービス	説明	マニュアルページ
/usr/lib/inet/in.ikev2d	インターネット鍵交換 (IKE) のデーモン。ike:ikev2 サービスが使用可能なときに自動鍵管理をアクティブ化します。	in.ikev2d(1M)
/usr/sbin/ikeadm [-v 2]	IKEv2 ポリシーの表示および一時的な変更用の IKE 管理コマンド。使用可能な Diffie-Hellman グループなどの IKEv2 管理オブジェクトを表示できます。	ikeadm(1M)
/usr/sbin/ikev2cert	構成の所有者 ikeuser として公開鍵証明書を作成および格納するための証明書データベース管理コマンド。pktool コマンドを呼び出します。	ikev2cert(1M) pktool(1)
/etc/inet/ike/ikev2.config	IKEv2 ポリシーのデフォルト構成ファイル。インバウンド IKEv2 要求のマッチングとアウトバウンド IKEv2 要求の準備に関するサイトのルールが含まれています。 このファイルが存在する場合、ike:ikev2 サービスが有効になると in.ikev2d デーモンが起動します。このファイルの場所は svccfg コマンドを使用して変更できます。	ikev2.config(4)
/etc/inet/ike/ikev2.preshared	証明書ベースの認証を使用していない 2 つの IKEv2 インスタンスが互いを認証するために使用できる秘密鍵が含まれています。	ikev2.preshared(4)
ソフトトークンキーストア	ikeuser が所有する IKEv2 の非公開鍵および公開鍵証明書が含まれています。	pkcs11_softtoken (5)

IKEv2 サービス

サービス管理機能 (SMF) は、IKEv2 を管理するための svc:/network/ipsec/ike:ikev2 サービスインスタンスを提供します。デフォルトでは、このサービスは無効になっています。このサービスを有効にする前に、/etc/inet/ike/ikev2.config ファイル内に有効な IKEv2 構成を作成する必要があります。

次の ike:ikev2 サービスプロパティーは構成可能です。

- config_file プロパティー – IKEv2 構成ファイルの場所を指定します。初期値は /etc/inet/ike/ikev2.config です。このファイルは特殊な権限を持ち、ikeuser に所有されている必要があります。別のファイルを使用しないでください。
- debug_level プロパティー – in.ikev2d デーモンのデバッグレベルを設定します。初期値は op (動作)です。指定可能な値については、ikeadm(1M) のマニュアルページで、オブジェクトタイプの下にあるデバッグレベルの表を参照してください。
- debug_logfile プロパティー – IKEv2 のデバッグのログファイルの場所を指定します。初期値は /var/log/ikev2/in.ikev2d.log です。

- **kmf_policy プロパティー** – 証明書ポリシーのログファイルの場所を設定します。デフォルト値は /etc/inet/ike/kmf-policy.xml です。このファイルは特殊な権限を持ち、ikeuser に所有されている必要があります。別のファイルを使用しないでください。
- **pkcs11_token/pin プロパティー** – IKEv2 デーモンの起動時にキーストアへのログインに使用する PIN を設定します。この値は、ikev2cert setpin コマンドでトークンに設定した値と一致する必要があります。
- **pkcs11_token/uri プロパティー** – キーストアへの PKCS #11 URI を設定します。暗号化アクセラレータカード上のハードウェアストレージを使用するには、この値を指定する必要があります。

SMF については、『Oracle Solaris 11.2 でのシステムサービスの管理 』の第 1 章「サービス管理機能の概要」を参照してください。smf(5)、svcadm(1M)、および svccfg(1M) のマニュアルページも参照してください。

IKEv2 デーモン

in.ikev2d デーモンは、Oracle Solaris システム上で IPsec の暗号化鍵の管理を自動化します。また、同じプロトコルを実行するリモートシステムとのネゴシエーションを行い、認証された鍵情報が、保護された方法でセキュリティーアソシエーション (SA) に提供されます。このデーモンは、IKEv2 プロトコルを使用して通信を保護するために IPsec の使用を予定しているすべてのシステム上で実行されている必要があります。

デフォルトでは、svc:/network/ipsec/ike:ikev2 サービスは有効になっていません。/etc/inet/ike/ikev2.config ファイルを構成して ike:ikev2 サービスインスタンスを有効にすると、システムブート時に SMF が in.ikev2d デーモンを起動します。

IKEv2 デーモンを実行すると、システムはそのピア IKEv2 エンティティーに対して自分自身を認証し、セッション鍵を確立します。構成ファイルで指定した間隔で、IKE 鍵が自動的に置き換えられます。in.ikev2d デーモンは、ネットワークからの着信 IKE 要求と PF_KEY ソケット経由のアウトバウンドトラフィックの要求を待機します。詳細は、pf_key(7P) のマニュアルページを参照してください。

2 つのコマンドが IKEv2 デーモンをサポートします。ikeadm コマンドを使用して IKE ポリシーを表示できます。詳細は、240 ページの「IKEv2 の ikeadm コマンド」を参照してください。ikev2cert コマンドでは、公開鍵証明書と非公開鍵証明書を表示および管理できます。詳細は、241 ページの「IKEv2 ikev2cert コマンド」を参照してください。

IKEv2 構成ファイル

IKEv2 構成ファイル `/etc/inet/ike/ikev2.config` は、IPsec ポリシーファイル `/etc/inet/ipsecinit.conf` で保護されている、指定されたネットワークエンドポイントの鍵をネゴシエートするのに使用されるルールを管理します。

IKE での鍵管理には、ルールとグローバルパラメータが関係します。IKE ルールは、その鍵情報で保護するシステムやネットワークを識別します。さらに、ルールは認証方式も指定します。グローバルパラメータには、IKEv2 SA の鍵が再生成されるまでのデフォルトの時間 `ikesa_lifetime_secs` などの項目が含まれます。IKEv2 構成ファイルの例については、150 ページの「事前共有鍵による IKEv2 の構成」を参照してください。IKEv2 ポリシーエントリの例と説明については、`ikev2.config`(4) のマニュアルページを参照してください。

IKEv2 がサポートする IPsec SA は、IPsec 構成ファイル `/etc/inet/ipsecinit.conf` のポリシーに従って IP パケットを保護します。

`ike/ikev2.config` ファイルのセキュリティーに関する注意点は、`ipsecinit.conf` ファイルの注意点と同様です。詳細は、234 ページの「`ipsecinit.conf` と `ipsecconf` のセキュリティーについて」を参照してください。

IKEv2 の `ikeadm` コマンド

`in.ikev2d` デーモンの実行中は、`ikeadm [-v2]` コマンドを使用して次を実行できます。

- IKEv2 状態の要素の表示。
- ポリシー規則、事前共有鍵、使用可能な Diffie-Hellman グループ、暗号化アルゴリズムと認証アルゴリズム、および既存のアクティブな IKEv2 SA などの IKEv2 デーモンオブジェクトの表示。

このコマンドのオプションの例と詳しい説明については、`ikeadm`(1M) のマニュアルページを参照してください。

`ikeadm` コマンドのセキュリティーについては、`ipseckey` コマンドのセキュリティーと同様です。詳細は、235 ページの「`ipseckey` におけるセキュリティーについて」を参照してください。

IKEv2 事前共有鍵ファイル

/etc/inet/ike/ikev2.preshared には、IKEv2 サービスによって使用される事前共有鍵が含まれます。このファイルは ikeuser によって所有され、0600 で保護されます。

ike/ikev2.config ファイル内で事前共有鍵を要求するルールを構成する場合は、デフォルトの ikev2.preshared ファイルをカスタマイズする必要があります。IKEv2 はこれらの事前共有鍵を使用して IKEv2 ピアを認証するため、このファイルは in.ikev2d デーモンが事前共有鍵を要求するルールを読み込む前に有効になっている必要があります。

IKEv2 ikev2cert コマンド

ikev2cert コマンドは、公開および非公開の鍵および証明書を生成、格納、および管理するのに使用されます。このコマンドは、ike/ikev2.config ファイルが公開鍵証明書を要求するときに使用します。IKEv2 はこれらの証明書を使用して IKEv2 ピアを認証するため、証明書は in.ikev2d デーモンがその証明書を要求するルールを読み込む前に適用されている必要があります。

ikev2cert コマンドは pktool コマンドを ikeuser として呼び出します。

次の ikev2cert コマンドは IKEv2 の証明書を管理します。このコマンドは ikeuser アカウントによって実行される必要があります。結果は PKCS #11 ソフトトークンキーストアに格納されます。

- ikev2cert setpin – ikeuser ユーザーの PIN を生成します。この PIN は証明書を使用する際に必要です。
- ikev2cert gencert – 自己署名付き証明書を生成します。
- ikev2cert gencsr – 証明書署名要求 (CSR) を生成します。
- ikev2cert list – キーストア内の証明書を一覧表示します。
- ikev2cert export – エクスポート用ファイルに証明書をエクスポートします。
- ikev2cert import – 証明書または CRL をインポートします。

ikev2cert サブコマンドの構文については、pktool(1) のマニュアルページを参照してください。例については、ikev2cert(1M) のマニュアルページを参照してください。ソフトトークンキーストアについては、cryptoadm(1M) のマニュアルページを参照してください。

IKEv1 リファレンス

次のセクションでは、IKEv1 の参照情報を示します。IKEv1 は、より高速な自動鍵管理を提供する IKEv2 に置き換わっています。IKEv2 の詳細については、237 ページの「IKEv2 リファレンス」を参照してください。比較については、143 ページの「IKEv2 と IKEv1 の比較」を参照してください。

IKEv1 のユーティリティーおよびファイル

次の表は、IKEv1 ポリシーの構成ファイル、IKEv1 鍵の保管場所、および IKEv1 を実装する各種コマンドとサービスについてまとめたものです。サービスの詳細については、『Oracle Solaris 11.2 でのシステムサービスの管理 』の第 1 章「サービス管理機能の概要」を参照してください。

表 12-2 IKEv1 のサービス名、コマンド、構成と鍵の保管場所、およびハードウェアデバイス

サービス、コマンド、ファイル、またはデバイス	説明	マニュアルページ
`svc:/network/ipsec/ike:default`	IKEv1 を管理する SMF サービス。	`smf`(5)
`/usr/lib/inet/in.iked`	インターネット鍵交換 (IKEv1) デーモン。`ike` サービスが使用可能なときに自動鍵管理をアクティブ化します。	`in.iked`(1M)
`/usr/sbin/ikeadm [-v1]`	IKE ポリシーの表示および一時的な変更用の IKE 管理コマンド。フェーズ 1 アルゴリズムや使用可能な Diffie-Hellman グループなどの IKE 管理オブジェクトを表示できます。	`ikeadm`(1M)
`/usr/sbin/ikecert`	公開鍵証明書が格納されているローカルデータベースを操作する証明書データベース管理コマンド。データベースは、接続されたハードウェアにも格納できます。	`ikecert`(1M)
`/etc/inet/ike/config`	デフォルトの IKEv1 ポリシー構成ファイル。インバウンド IKEv1 要求のマッチングとアウトバウンド IKEv1 要求の準備に関するサイトのルールが含まれています。 このファイルが存在する場合、`ike` サービスが有効になると `in.iked` デーモンが起動します。このファイルの場所は `svccfg` コマンドを使用して変更できます。	`ike.config`(4)
`ike.preshared`	`/etc/inet/secret` ディレクトリにある事前共有鍵ファイル。フェーズ 1 交換での認証の秘密鍵が含まれます。事前共有鍵で IKEv1 を構成するときに使用されます。	`ike.preshared`(4)
`ike.privatekeys`	`/etc/inet/secret` ディレクトリにある非公開鍵ディレクトリ。公開鍵と非公開鍵のペアの非公開部分が含まれています。	`ikecert`(1M)

サービス、コマンド、ファイル、またはデバイス	説明	マニュアルページ
publickeys ディレクトリ	公開鍵および証明書ファイルが格納されている /etc/inet/ike ディレクトリ内のディレクトリ。公開鍵と非公開鍵のペアの公開部分が含まれています。	ikecert(1M)
crls ディレクトリ	公開鍵および証明書ファイルの失効リストが格納されている /etc/inet/ike ディレクトリ内のディレクトリ。	ikecert(1M)
Sun Crypto Accelerator 6000 ボード	オペレーティングシステムの処理をオフロードすることで公開鍵の処理を高速化するハードウェア。公開鍵、非公開鍵、および公開鍵証明書も格納します。Sun Crypto Accelerator 6000 ボードはレベル 3 の FIPS 140-2 認定デバイスです。	ikecert(1M)

IKEv1 サービス

サービス管理機能 (SMF) は、IKEv1 を管理するための svc:/network/ipsec/ike:default サービスを提供します。デフォルトでは、このサービスは無効になっています。このサービスを有効にする前に、IKEv1 構成ファイル /etc/inet/ike/config を作成する必要があります。

次の ike サービスのプロパティーは構成可能です。

- config_file プロパティー – IKEv1 構成ファイルの場所を設定します。初期値は /etc/inet/ike/config です。
- debug_level プロパティー – in.iked デーモンのデバッグレベルを設定します。初期値は op (動作)です。指定可能な値については、ikeadm(1M) のマニュアルページで、オブジェクトタイプの下にあるデバッグレベルの表を参照してください。
- admin_privilege プロパティー – in.iked デーモンの特権レベルを設定します。初期値は base です。ほかの値は modkeys と keymat です。詳細は、245 ページの「IKEv1 ikeadm コマンド」を参照してください。

SMF については、『Oracle Solaris 11.2 でのシステムサービスの管理 』の第 1 章「サービス管理機能の概要」を参照してください。smf(5)、svcadm(1M)、および svccfg(1M) のマニュアルページも参照してください。

IKEv1 デーモン

in.iked デーモンは IPsec SA の管理を自動化し、これには IPsec を使用するパケットを保護する暗号化鍵が含まれます。このデーモンは、IKEv1 プロトコルを実行しているピアシステムと、ISAKMP SA および IPsec SA をセキュアにネゴシエートします。

デフォルトでは、svc:/network/ipsec/ike:default サービスは有効になっていません。/etc/inet/ike/config ファイルを構成して ike:default サービスを有効にすると、システムブート時に SMF が in.iked デーモンを起動します。/etc/inet/ike/config ファイルに加え、その他の構成がほかのファイルおよびデータベースに、または SMF プロパティーとして格納されています。詳細は、242 ページの「IKEv1 のユーティリティーおよびファイル」、および ike.preshared(4)、ikecert(1M)、in.iked(1M) のマニュアルページを参照してください。

ike:default サービスを有効にすると、in.iked デーモンが構成ファイルを読み込んで、SA に対する IKE ピアからの外部要求および IPsec からの内部要求を待機します。

IKEv1 ピアからの外部要求の場合、ike:default サービスの構成によってデーモンの応答方法が決まります。内部要求は PF_KEY インタフェースを介してルーティングされます。このインタフェースは、IPsec SA を格納してパケットの暗号化および復号化を実行する IPsec のカーネル部分と、ユーザーランドで実行する鍵管理デーモン in.iked との間の通信を処理します。カーネルは、パケットの保護に SA を必要とする場合、PF_KEY インタフェースを介して in.iked デーモンにメッセージを送信します。詳細は、pf_key(7P) のマニュアルページを参照してください。

2 つのコマンドが IKEv1 デーモンをサポートします。ikeadm コマンドは、実行中のデーモンにコマンド行インタフェースを提供します。ikecert コマンドは、ディスクおよびハードウェア上の証明書データベース ike.privatekeys および publickeys を管理します。

これらのコマンドの詳細については、in.iked(1M)、ikeadm(1M)、および ikecert(1M) のマニュアルページを参照してください。

IKEv1 構成ファイル

IKEv1 構成ファイル /etc/inet/ike/config は、IPsec 構成ファイル /etc/inet/ipsecinit.conf 内のポリシーに従って、IPsec の保護を必要とするネットワークパケットのための SA を管理します。

IKE での鍵管理には、ルールとグローバルパラメータが関係します。IKEv1 ルールは、別の IKEv1 デーモンを実行中のシステムを識別します。さらに、ルールは認証方式も指定します。グローバルパラメータには、接続されたハードウェアアクセラレータへのパスなどがあります。IKEv1 ポリシーファイルの例については、150 ページの「事前共有鍵による IKEv2 の構成」を参照してください。IKEv1 ポリシーエントリの例と説明については、ike.config(4) のマニュアルページを参照してください。

/etc/inet/ike/config ファイルには、RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki) 規格に従って実装されているライブラリへのパスが含まれることがあります。IKEv1 はこの PKCS #11 ライブラリを使用して、鍵アクセラレーションおよび鍵の格納のためにハードウェアにアクセスします。

ike/config ファイルのセキュリティーに関する注意点は、ipsecinit.conf ファイルの注意点と同様です。詳細は、234 ページの「ipsecinit.conf と ipsecconf のセキュリティーについて」を参照してください。

IKEv1 ikeadm コマンド

ikeadm コマンドを実行すると、次のことができます。

- IKE 状態の要素の表示
- IKE デーモンのプロパティーの変更
- フェーズ 1 交換時の SA 作成に関する統計の表示
- IKE プロトコル交換のデバッグ
- すべてのフェーズ 1 SA、ポリシー規則、事前共有鍵、使用可能な Diffie-Hellman グループ、フェーズ 1 暗号化および認証アルゴリズム、証明書キャッシュなどの IKE デーモンオブジェクトの表示

このコマンドのオプションの例と詳しい説明については、ikeadm(1M) のマニュアルページを参照してください。

実行する IKE デーモンの特権レベルにより、表示および変更可能な IKE デーモンの要素が決まります。使用可能な特権レベルは 3 つあります。

base レベル	鍵を表示したり変更したりすることはできません。base レベルはデフォルトの特権レベルです。
keymat レベル	ikeadm コマンドで実際の鍵を表示できます。

`modkeys` レベル　　事前共有鍵の削除、変更、追加ができます。

特権の一時的な変更には、`ikeadm` コマンドを使用できます。永続的に変更する場合は、`ike` サービスの `admin_privilege` プロパティーを変更します。特権の一時的な変更については、228 ページの「実行中の IKE デーモンの管理」を参照してください。

`ikeadm` コマンドのセキュリティーについては、`ipseckey` コマンドのセキュリティーと同様です。235 ページの「`ipseckey` におけるセキュリティーについて」を参照してください。`ikeadm` コマンドに固有の詳細については、`ikeadm`(1M) のマニュアルページを参照してください。

IKEv1 事前共有鍵ファイル

事前共有鍵を手動で作成すると、鍵は、`/etc/inet/secret` ディレクトリのファイルに格納されます。`ike/config` 内に事前共有鍵を使用するルールを構成すると、`ike.preshared` ファイルにフェーズ 1 交換用の事前共有鍵が含まれます。`ipseckeys` ファイルには、IP パケットの保護に使用される事前共有鍵が含まれます。これらのファイルは `0600` で保護されます。`secret` ディレクトリは `0700` で保護されます。

フェーズ 1 交換の認証に事前共有鍵を使用するため、このファイルを `in.iked` デーモンの開始前に有効にする必要があります。

手動による IPsec 鍵の管理の例については、128 ページの「IPsec の鍵を手動で作成する方法」を参照してください。

IKEv1 公開鍵のデータベースおよびコマンド

`ikecert` コマンドは、ローカルシステムの公開鍵/非公開鍵、公開証明書、および静的 CRL のデータベースを管理します。このコマンドは、IKEv1 構成ファイルが公開鍵証明書を要求するときに使用します。IKEv1 ではこれらのデータベースを使用してフェーズ 1 交換を認証するため、`in.iked` デーモンを起動する前に、それらのデータベースに必要な情報が含まれていなければなりません。3 つのサブコマンド `certlocal`、`certdb`、および `certrldb` は、3 つの各データベースを処理します。

システムに Sun Crypto Accelerator 6000 ボードが接続されている場合、`ikecert` コマンドは PKCS #11 ライブラリを使用してハードウェア鍵と証明書のストレージにアクセスします。

詳細は、ikecert(1M) のマニュアルページを参照してください。メタスロットとソフトトークンキーストアについては、cryptoadm(1M) のマニュアルページを参照してください。

IKEv1 ikecert tokens コマンド

tokens 引数は、使用可能なトークン ID を一覧表示します。トークン ID により、ikecert certlocal コマンドと ikecert certdb コマンドは、公開鍵証明書と CSR を生成できます。鍵および証明書も接続された Sun Crypto Accelerator 6000 ボードに格納できます。ikecert コマンドは、PKCS #11 ライブラリを使用してハードウェアキーストアにアクセスします。

IKEv1 ikecert certlocal コマンド

certlocal サブコマンドは非公開鍵データベースを管理します。このサブコマンドを選択すると、非公開鍵の追加、表示、および削除を行うことができます。このサブコマンドは、自己署名付き証明書または CSR も作成します。-ks オプションを選択すると、自己署名付き証明書が作成されます。-kc オプションは CSR を作成します。鍵はシステムの /etc/inet/secret/ike.privatekeys ディレクトリに格納されるか、-T オプションを指定した場合は、接続されたハードウェアに格納されます。

非公開鍵を作成する場合は、ikecert certlocal コマンドのオプションに関連するエントリが ike/config ファイルに存在しなければなりません。ikecert オプションと ike/config エントリの対応を次の表に示します。

表 12-3 IKEv1 の ikecert オプションと ike/config エントリの対応

ikecert オプション	ike/config エントリ	説明
-A *subject-alternate-name*	cert_trust *subject-alternate-name*	証明書を一意に識別するニックネーム。指定可能な値は IP アドレス、電子メールアドレス、およびドメイン名です。
-D *X.509-distinguished-name*	*X.509-distinguished-name*	国 (C)、組織名 (ON)、組織単位 (OU)、共通名 (CN) を含む認証局のフルネーム。
-t dsa-sha1 | dsa-sha256	auth_method dsa_sig	RSAよりもわずかに遅い認証方式。
-t rsa-md5 および -t rsa-sha1 | rsa-sha256 | rsa-sha384 | rsa-sha512	auth_method rsa_sig	DSAよりもわずかに速い認証方式。 RSA 公開鍵は、最大ペイロードを暗号化するのに十分な長さが必要。通常、X.509 識別名などの ID ペイロードが最大ペイロードになります。

ikecert オプション	ike/config エントリ	説明
-t rsa-md5 および -t rsa-sha1 | rsa-sha256 | rsa-sha384 | rsa-sha512	auth_method rsa_encrypt	RSA 暗号化により、IKE にある ID が不正侵入者から保護されますが、IKE ピアには互いの公開鍵の認識が要求されます。

ikecert certlocal -kc コマンドで CSR を発行する場合、コマンドの出力を認証局 (CA) に送信します。会社が独自の公開鍵インフラストラクチャー (PKI) を運用している場合は、PKI の管理者に出力を送信します。その後、CA または PKI の管理者が証明書を作成します。返された証明書は certdb サブコマンドに渡されます。CA から返された証明書失効リスト (CRL) は、certrldb サブコマンドに渡されます。

IKEv1 ikecert certdb コマンド

certdb サブコマンドは、公開鍵データベースを管理します。そのサブコマンドを選択すると、公開鍵と証明書を追加、表示、および削除できます。このコマンドは、リモートシステムで ikecert certlocal -ks コマンドによって生成された証明書を入力として受け入れます。手順については、184 ページの「自己署名付き公開鍵証明書により IKEv1 を構成する方法」を参照してください。このコマンドは、CA から受信する証明書も入力として受け入れます。手順については、189 ページの「CA の署名付き証明書で IKEv1 を構成する方法」を参照してください。

証明書と公開鍵は、システムの /etc/inet/ike/publickeys ディレクトリに格納されます。-T オプションを指定した場合、証明書、非公開鍵、公開鍵は、システムに接続されたハードウェアに格納されます。

IKEv1 ikecert certrldb コマンド

certrldb サブコマンドは、証明書失効リスト (CRL) データベース /etc/inet/ike/crls を管理します。CRL データベースには、公開鍵の失効リストが保存されています。よって、このリストには、すでに有効でない証明書が明記されます。CA から CRL が提供される場合、その CRL を ikecert certrldb コマンドで CRL データベースにインストールできます。手順については、199 ページの「IKEv1 で失効した証明書を処理する方法」を参照してください。

IKEv1 /etc/inet/ike/publickeys ディレクトリ

/etc/inet/ike/publickeys ディレクトリのファイルまたは*スロット*には、公開鍵と非公開鍵のペアの公開部分とその証明書が含まれています。このディレクトリは 0755 で保護されています。ikecert certdb コマンドを使用して、そのディレクトリを読み込みます。-T オプションは、鍵を publickeys ディレクトリではなく Sun Crypto Accelerator 6000 ボード上に格納します。

スロットには、別のシステムで生成された証明書の X.509 識別名が、エンコードされた形式で含まれます。自己署名付き証明書を使用する場合、そのコマンドへの入力として、リモートシステムの管理者から受信する証明書を使用します。CA からの証明書を使用する場合、CA から受け取る 2 つの署名付き証明書をこのデータベースに格納します。CA に送信した CSR に基づいた証明書をインストールします。また、CA の証明書も格納します。

IKEv1 /etc/inet/secret/ike.privatekeys ディレクトリ

/etc/inet/secret/ike.privatekeys ディレクトリには、公開鍵/非公開鍵ペアの一部である非公開鍵ファイルが格納されています。このディレクトリは 0700 で保護されています。ikecert certlocal コマンドを実行して、ike.privatekeys ディレクトリを読み込みます。非公開鍵は、ペアとなる公開鍵、自己署名付き証明書や CA が格納されてから有効になります。ペアとなる公開鍵は、/etc/inet/ike/publickeys ディレクトリか、サポートされるハードウェアに格納されます。

IKEv1 /etc/inet/ike/crls ディレクトリ

/etc/inet/ike/crls ディレクトリには、証明書失効リスト (CRL) ファイルが含まれています。各ファイルは、/etc/inet/ike/publickeys ディレクトリにある公開証明書ファイルに対応しています。CA は証明書の CRL を提供します。ikecert certrldb コマンドを使用して、そのデータベースを読み込むことができます。

ネットワークセキュリティー用語集

3DES	Triple-DES を参照してください。
インターネットプロトコル (IP)	インターネットを介してデータをあるコンピュータから別のコンピュータに送信するための方法またはプロトコル。
解釈ドメイン (DOI)	データ形式や、ネットワークトラフィック交換タイプ、セキュリティー関連情報の命名規約を定義します。セキュリティー関連情報の例としては、セキュリティーポリシーや、暗号化アルゴリズム、暗号化モードなどがあります。
鍵管理	セキュリティーアソシエーション (SA) を管理する方法。
仮想ネットワーク	ソフトウェアおよびハードウェアのネットワークリソースとネットワーク機能を組み合わせたもの。単一のソフトウェアエンティティとしてまとめて管理されます。「*内部*」仮想ネットワークは、ネットワークリソースを単一のシステムに統合したもので、「ワンボックスネットワーク (network in a box)」と呼ばれることもあります。
仮想ネットワークインタフェース (VNIC)	物理的なネットワークインタフェースで構成されているかどうかに関係なく、仮想ネットワーク接続を提供する擬似インタフェース。排他的 IP ゾーンなどのコンテナが VNIC 上に構成されて、仮想ネットワークを形成します。
仮想プライベートネットワーク (VPN)	インターネットのような公共ネットワーク内でトンネルを利用する、単独の、安全で論理的なネットワーク。
仮想 LAN (VLAN) デバイス	IP プロトコルスタックの Ethernet (データリンク) レベルでトラフィック転送を行うネットワークインタフェース。
カプセル化	ヘッダーとペイロードを 1 番目のパケット内に配置し、そのパケットを 2 番目のパケットのペイロード内に配置すること。
カプセル化セキュリティーペイロード (ESP)	パケットに対して完全性と機密性を提供する拡張ヘッダー。ESP は、IP セキュリティーアーキテクチャー (IPsec) の 5 つのコンポーネントの 1 つです。
キーストア名	管理者がストレージ領域 (つまり、キーストア) に与える、ネットワークインタフェースカード (NIC) 上の名前。キーストア名は、「トークン」、「トークン ID」とも呼ばれます。

公開鍵暗号化 2 つの鍵を使用する暗号化システム。公開鍵はだれでも知ることができます。非公開鍵は、メッセージの受信者だけが知っています。IKE により、IPsec の公開鍵が提供されます。

識別名 (DN) 通常の文字列を使用して共有情報を表すための標準化された方法。識別名は、LDAP 証明書や X.509 証明書などのテクノロジーで使用されています。詳細は、A String Representation of Distinguished Names (http://www.ietf.org/rfc/rfc1779.txt)を参照してください。

証明書失効リスト (CRL) CA が無効とした公開鍵証明書のリスト。CRL は、IKE を使用して管理される CRL データベースに格納されます。

ステートフルパケットフィルタ アクティブな接続の状態をモニターし、そこから得た情報を使ってパケットフィルタを通過させるネットワークパケットを決めるファイアウォール。要求と応答を追跡、照合することによって、ステートフルパケットフィルタは、要求と一致しない応答を選別できます。

ストリーム制御転送プロトコル (SCTP) TCP と似た方法で接続指向の通信を行う転送層プロトコル。さらに、このプロトコルは、接続のエンドポイントの 1 つが複数の IP アドレスをもつことができる複数ホーム機能をサポートします。

セキュリティーアソシエーション (SA) 1 つのホストから 2 つめのホストにセキュリティー属性を指定するアソシエーション。

セキュリティーパラメータインデックス (SPI) 受信したパケットを復号化するために使用する、SADB (セキュリティーアソシエーションデータベース) 内の行を特定する整数値。

セキュリティーポリシーデータベース (SPD) パケットにどのレベルの保護を適用するかを指定するデータベース。SPD は、IP トラフィックをフィルタして、パケットを破棄すべきか、検証済みとして通過させるべきか、IPsec で保護すべきかを決めます。

双方向トンネル 双方向にパケットを送信するトンネル。

対称鍵暗号化 メッセージの送信側と受信側が 1 つの共通鍵を共有する暗号化システム。この共通鍵は、メッセージを暗号化および復号化するために使用されます。対称鍵は、IPsec での大量データ転送の暗号化に使用します。AES は対称鍵の 1 つの例です。

デジタル署名 送信側を一意に識別する、電子的に転送されたメッセージに添付されるデジタルコード。

盗聴 コンピュータネットワーク上で盗聴すること。普通のテキストによるパスワードなどの情報をネットワークから自動的に選別するプログラムの一部としてしばしば使用されます。

動的パケットフィルタ ステートフルパケットフィルタを参照してください。

トラストアンカー X.509 証明書では、認証局からのルート証明書。ルート証明書から目的の証明書までの証明書によって、トラストチェーンが確立されます。

トラストチェーン X.509 証明書では、トラストアンカーからユーザーの証明書までの証明書が、一続きの認証チェーンを提供しているという認証局からの保証。

トンネル パケットがカプセル化される間に通過するパス。カプセル化を参照してください。

IPsec では、構成されたトンネルはポイントツーポイントインタフェースです。トンネルによって、IP パケットを別の IP パケット内にカプセル化できます。

なりすまし コンピュータに不正にアクセスするために、メッセージが、信頼されるホストから来たかのように見える IP アドレスを使ってコンピュータにメッセージを送信すること。IP のなりすましを行うために、ハッカーはまず、さまざまなテクニックを使って、信頼されるホストの IP アドレスを見つけ、次にパケットヘッダーを変更します。それによって、パケットは、そのホストから来たかのように見えます。

認証局 (CA) デジタル署名および公開鍵と非公開鍵のペアの作成に使用するデジタル証明書を発行する、公証された第三者機関または企業。CA は、一意の証明書を付与された個人が当該の人物であることを保証します。

認証ヘッダー IP パケットに対して認証と完全性を提供する拡張ヘッダーですが、機密性は提供されません。

ネットワークアドレス変換 (NAT) あるネットワークで使用されている IP アドレスを、別のネットワークで認識されている異なる IP アドレスに変換すること。必要となる大域 IP アドレスの数を抑えるために使用されます。

ネットワークインタフェースカード (NIC) ネットワークへのインタフェースになる、ネットワークアダプタカード。NIC によっては、`igb` カードなど複数の物理インタフェースを装備できるものもあります。

パケット IP パケットを参照してください。

パケット 通信回線上で、1 単位として送られる情報の集合。IP ヘッダー や ペイロード を含みます。

パケットフィルタ 指定するパケットのファイアウォールの通過を許可するようにも許可しないようにも構成できるファイアウォール機能。

パケットヘッダー IP ヘッダーを参照してください。

ハッシュ値 テキストの文字列から生成される数値。ハッシュ関数は、転送されるメッセージが改ざんされないようにするために使用します。一方向のハッシュ関数の例としては、MD5 と SHA-1 があります。

非対称鍵暗号化 メッセージの送受信側で異なる鍵を使用してメッセージの暗号化および暗号解除を行う暗号化システム。非対称鍵を使用して、対称鍵暗号に対するセキュアなチャネルを作成します。Diffie-Hellman アルゴリズムは、非対称鍵プロトコルの例です。対称鍵暗号化と比較してください。

ファイアウォール 組織のプライベートネットワークやイントラネットをインターネットから切り離し、外部からの進入を防止するためのデバイスまたはソフトウェア。ファイアウォールには、フィルタリングや、プロキシサーバー、NAT (ネットワークアドレス変換) などを組み込むことができます。

物理インタフェース リンクへのシステムの接続。この接続は通常、デバイスドライバとネットワークインタフェースカード (NIC) として実装されます。NIC によっては、`igb` のように複数の接続点を持つものもあります。

ブロードキャストアドレス アドレスのホスト部分のビットがすべてゼロ (10.50.0.0) か 1 (10.50.255.255) である IPv4 ネットワークアドレス。ローカルネットワーク上のマシンからブロードキャストアドレスに送信されたパケットは、同じネットワーク上のすべてのマシンに配信されます。

プロキシサーバー Web ブラウザなどのクライアントアプリケーションと別のサーバーの間にあるサーバー。要求をフィルタするために使用されます (たとえば、特定の Web サイトへのアクセスを防ぐ)。

ペイロード パケットで伝送されるデータ。ペイロードには、パケットを宛先に送るために必要なヘッダー情報は含まれません。

マーカー 1. diffserv アーキテクチャーおよび IPQoS のモジュールの 1 つ。パケットの転送方法を指示する値を IP パケットの DS フィールドに付けます。IPQoS 実装では、このマーカーモジュールは `dscpmk` です。

2. IPQoS 実装のモジュールの 1 つで、ユーザー優先順位の値を Ethernet パケットの仮想 LAN タグに付けます。ユーザー優先順位の値は、VLAN デバイスを備えたネットワーク上でパケットが転送される方法を示します。このモジュールは `dlcosmk` と呼ばれます。

マルチキャストアドレス 特定の方法でインタフェースのグループを特定する IPv6 アドレス。マルチキャストアドレスに送信されるパケットは、グループにあるすべてのインタフェースに配信されます。IPv6 マルチキャストアドレスには、IPv4 ブロードキャストアドレスに似た機能があります。

マルチホームホスト 複数の物理インタフェースをもち、パケット転送を行わないシステム。マルチホームホストではルーティングプロトコルを実行できます。

メッセージ認証コード (MAC) データの整合性を保証し、データの出所を明らかにするコード。MAC は盗聴行為には対応できません。

ラベル 1. IKEv2 ルールのキーワードで、その値は、`auth_method` が `preshared` の場合には共有鍵ファイルの `label` キーワードの値に一致する必要があります。

2. IKEv2 証明書の作成時に使用されるキーワード。この値は、キーストア内の証明書のあらゆる部分 (非公開鍵、公開鍵、および公開鍵証明書) を特定する際に便利です。

3. オブジェクトまたはプロセスの機密性レベルの強制アクセス制御 (MAC) の表示。Confidential および Top Secret はラベルの例です。ラベル付きネットワーク送信には MAC ラベルが含まれます。

4. IKEv1 ルールのキーワードで、その値はルールの取得に使用されます。

リプレー攻撃 IPsec では、パケットが侵入者によって捕捉されるような攻撃のこと。格納されたパケットは、あとで元のパケットを置き換えるか繰り返します。そのような攻撃を防止するために、パケットを保

護している秘密鍵が存在している間、値が増加を続けるフィールドをパケットに含めることができます。

リンク層 IPv4/IPv6 のすぐ下の層。

リンクローカルアドレス IPv6 では、自動アドレス構成などのために、単一リンク上でアドレスを指定するために使用することを表します。デフォルトでは、リンク - ローカル・アドレスはシステムの MAC アドレスから作成されます。

ルーター 複数のインタフェースを通常もち、ルーティングプロトコルを実行し、パケットを転送するシステム。システムが PPP リンクのエンドポイントである場合は、ルーターとしてのインタフェースを 1 つだけもつようなシステムを構成できます。

ルーター広告 ルーターが、各種のリンクパラメータおよびインターネットパラメータと共に、その存在を定期的にあるいはルーター要請メッセージに応じて通知すること。

ルーター発見 ホストが、接続されているリンク上にあるルーターを特定すること。

ルーター要請 ホストがルーターに対し、次に予定されている時間ではなく、ただちにルーター広告メッセージを送信するように要求すること。

AES Advanced Encryption Standard。対称ブロックのデータ暗号技術。2000 年の 10 月、米国政府は暗号化標準としてこのアルゴリズムの Rijndael 方式を採用しました。AES は DES に代わる米国政府の標準として採用されています。

Blowfish 32 ビットから 448 ビットまでの可変長鍵の対称ブロックの暗号化アルゴリズム。その作成者である Bruce Schneier 氏は、鍵を頻繁に変更しないアプリケーションに効果的であると述べています。

DES データ暗号化規格。1975 年に開発され、1981 年に ANSI X.3.92 として ANSI で標準化された対称鍵の暗号化方式。DES では 56 ビットの鍵を使用します。

Diffie-Hellman アルゴリズム 「公開鍵」暗号化としても知られています。1976 年に Diffie 氏と Hellman 氏が開発した非対称暗号鍵協定プロトコルです。このプロトコルを使用すると、セキュアでない伝達手段で、事前の秘密情報がなくても 2 人のユーザーが秘密鍵を交換できます。Diffie-Hellman は、IKE プロトコルで使用されます。

DSA デジタル署名アルゴリズム。512 ビットから 4096 ビットまでの可変長鍵の公開鍵アルゴリズム。米国政府標準である DSS は最大 1024 ビットです。この場合、DSA では入力に SHA-1 を使用します。

ECDSA Elliptic Curve Digital Signature Algorithm。楕円曲線数学に基づく公開鍵アルゴリズム。ECDSA 鍵サイズは、同じ長さの署名の生成に必要な DSA 公開鍵のサイズより大幅に小さくなります。

HMAC メッセージ認証を行うための鍵付きハッシュ方法。HMAC は秘密鍵認証アルゴリズムの 1 つです。HMAC は秘密共有鍵と併用して、MD5 、SHA-1 などの繰り返し暗号化のハッシュ関数で使用します。HMAC の暗号の強さは、基になるハッシュ関数のプロパティーによって異なります。

ICMP エコー要求パケット 応答を促すためにインターネット上のマシンに送信されるメッセージ。そのようなパケットは一般に“ping”パケットといわれています。

IKE インターネット鍵交換。IPsec セキュリティーアソシエーション (SA) 用の認証された鍵情報の供給を自動化します。

IP インターネットプロトコル (IP)、IPv4、IPv6 を参照してください。

IP スタック TCP/IP はしばしば「スタック」と呼ばれます。データ交換のクライアントエンドとサーバーエンドですべてのデータが通過する層 (TCP、IP、場合によってはそのほかを含む) のことを意味します。

IP 内 IP カプセル化 IP パケット内で IP パケットをトンネリングするためのメカニズム。

IP パケット IP 経由で転送される情報パケット。IP パケットはヘッダーとデータを含みます。ヘッダーにはパケットのソースと宛先のアドレスが含まれます。ヘッダーのその他のフィールドは、データと付随するパケットを宛先で識別し、再結合する際に役立ちます。

IP ヘッダー インターネットパケットを固有に識別する 20 バイトのデータ。ヘッダーには、パケットの送信元と送信先のアドレスが含まれています。さらに、ヘッダー内のオプションによって、新しいバイトを追加できます。

IP リンク リンク層でノード間の通信に使用される通信設備や通信メディア。リンク層とは IPv4 および IPv6 のすぐ下の層です。例としては、Ethernet (ブリッジされたものも含む) や ATM ネットワークなどがあります。1 つまたは複数の IPv4 サブネット番号またはネットワーク接頭辞が IP リンクに割り当てられます。同じサブネット番号またはネットワーク接頭辞を複数の IP リンクに割り当てることはできません。ATM LANE では、IP リンクは 1 つのエミュレートされた LAN です。ARP を使用する場合、ARP プロトコルの有効範囲は単一の IP リンクです。

IPsec IP セキュリティー。IP パケットを保護するためのセキュリティーアーキテクチャー。

IPv4 インターネットプロトコルのバージョン 4。単に IP と呼ばれることもあります。このバージョンは 32 ビットのアドレス空間をサポートしています。

IPv6 インターネットプロトコルのバージョン 6。128 ビットのアドレス空間をサポートしています。

MD5 デジタル署名などのメッセージ認証に使用する繰り返し暗号化のハッシュ関数。1991 年に Rivest 氏によって開発されました。

NAT ネットワークアドレス変換 (NAT) を参照してください。

Perfect Forward Secrecy (PFS) PFS では、データ伝送を保護するために使用される鍵が、追加の鍵を導き出すために使用されることはありません。さらに、データ伝送を保護するために使用される鍵のソースが、追加の鍵を導き出すために使用されることはありません。したがって、PFS は以前に記録されたトラフィックの復号化を回避できます。

PFS は認証された鍵交換だけに適用されます。Diffie-Hellman アルゴリズムも参照してください。

PKI 公開鍵インフラストラクチャー。インターネットトランザクションに関係する各関係者の有効性を確認および承認する、デジタル署名、認証局、ほかの登録機関のシステム。

RSA デジタル署名と公開鍵暗号化システムを取得するための方法。その開発者である Rivest 氏、Shamir 氏、Adleman 氏によって 1978 年に最初に公開されました。

SADB セキュリティーアソシエーションデータベース。暗号化鍵と暗号化アルゴリズムを指定するテーブル。鍵とアルゴリズムは、安全なデータ転送で使用されます。

SHA-1 セキュアなハッシュアルゴリズム。メッセージ要約を作成するために 2^{64} 文字以下の長さを入力するときに操作します。SHA-1 アルゴリズムは DSA に入力されます。

smurf 攻撃 リモートロケーションから IP ブロードキャストアドレス または複数のブロードキャストアドレスに向けられた ICMP echo request パケットを使用して、深刻なネットワークの輻輳や中断を引き起こすこと。

TCP/IP TCP/IP (伝送制御プロトコル/インターネットプロトコル) は、インターネットの基本的な通信言語またはプロトコルです。プリベートネットワーク (イントラネットやエクストラネット) の通信プロトコルとしても使用されます。

Triple-DES Triple-Data Encryption Standard。対称鍵暗号化システムの 1 つ。Triple-DES では鍵の長さとして 168 ビットが必要です。Triple-DES を「3DES」と表記することもあります。

索引

あ

か

さ

た

な

は

ま

や

ら

K

U

V

W

X