Oracle® Solaris 11.1 でのネットワークのセキュリティー保護

Part No: E36646–02
2013 年 2 月

ORACLE®

130327@25097

目次

はじめに

このガイドでは、Oracle Solaris オペレーティングシステム (Oracle Solaris OS) がインストールされ、ネットワークをセキュリティー保護する準備が整っていることを前提としています。

注 – この Oracle Solaris のリリースでは、SPARC および x86 系列のプロセッサアーキテクチャーを使用するシステムをサポートしています。サポートされるシステムについては、Oracle Solaris OS: Hardware Compatibility Listsを参照してください。本書では、プラットフォームにより実装が異なる場合は、それを特記します。

対象読者

このドキュメントは、Oracle Solaris を実行するネットワーク接続されたシステムの管理者を対象としています。このドキュメントを利用するにあたっては、UNIX のシステム管理について少なくとも 2 年の経験が必要です。UNIX システム管理のトレーニングコースに参加することも役に立ちます。

Oracle サポートへのアクセス

Oracle のお客様は、My Oracle Support を通じて電子的なサポートを利用することができます。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。

表記上の規則

次の表では、このドキュメントで使用される表記上の規則について説明します。

表 P–1 表記上の規則

字体	説明	例
`AaBbCc123`	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	`.login` ファイルを編集します。 `ls -a` を使用してすべてのファイルを表示します。 `machine_name% you have mail.`
`AaBbCc123`	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	`machine_name%` **`su`** `Password:`
aabbcc123	プレースホルダ: 実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、 `rm` *filename* と入力します。
AaBbCc123	書名、新しい単語、および強調する単語を示します。	『ユーザーズガイド』の第 6 章を参照してください。 キャッシュは、ローカルに格納されるコピーです。 ファイルを保存しないでください。 注: いくつかの強調された項目は、オンラインでは太字で表示されます。

コマンド例のシェルプロンプト

次の表に、Oracle Solaris OS に含まれるシェルの UNIX システムプロンプトおよびスーパーユーザーのプロンプトを示します。コマンド例のシェルプロンプトは、そのコマンドを標準ユーザーで実行すべきか特権ユーザーで実行すべきかを示します。

表 P–2 シェルプロンプト

シェル	プロンプト
Bash シェル、Korn シェル、および Bourne シェル	`$`
Bash シェル、Korn シェル、および Bourne シェルのスーパーユーザー	`#`
C シェル	`machine_name%`
C シェルのスーパーユーザー	`machine_name#`

◆ ◆ ◆ 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。

リンク保護の概要

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 – 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 – 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情報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、16 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。

リンク保護の構成 (タスクマップ)

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。保護のタイプが他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 つの一般的な操作を実行します。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 – 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。

タスク	説明	手順
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。	13 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。	リンク保護を削除します。	14 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。	14 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。	15 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。	16 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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    DEFAULT    POSSIBLE
vnic0     protection    rw   --       --         mac-nospoof,
                                                  restricted,
                                                  ip-nospoof,
                                                  dhcp-nospoof
```

可能なタイプの説明については、11 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
```

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          DEFAULT       POSSIBLE
vnic0    protection      rw   mac-nospoof    --            mac-nospoof,
                              restricted                   restricted,
                              ip-nospoof                   ip-nospoof,
                              dhcp-nospoof                 dhcp-nospoof
```

VALUE の下のリンク保護のタイプは、保護が有効にされることを示します。

▼ 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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    DEFAULT    POSSIBLE
vnic0      protection     rw   --       --         mac-nospoof,
                                                   restricted,
                                                   ip-nospoof,
                                                   dhcp-nospoof
```

VALUE の下にリンク保護のタイプが表示されないことは、リンク保護が無効にされていることを示します。

▼ 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 13 ページの「リンク保護を有効にする方法」に示すように、ip-nospoof 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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          DEFAULT       POSSIBLE
link     protection      rw      ...
                                 ip-nospoof                   ip-nospoof
```

VALUE の下に ip-nospoof が表示されていることは、この保護のタイプが有効にされていることを示します。

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。

▼ 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 13 ページの「リンク保護を有効にする方法」に示すように、dhcp-nospoof 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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          DEFAULT       POSSIBLE
link     protection      rw      ...
                                 dhcp-nospoof                 dhcp-nospoof
```

VALUE の下の dhcp-nospoof のリストは、この保護のタイプが有効にされていることを示します。

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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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           DEFAULT    POSSIBLE
vnic0   protection          rw       mac-nospoof     --         mac-nospoof,
                                     restricted                 restricted,
                                     ip-nospoof                 ip-nospoof,
                                     dhcp-nospoof               dhcp-nospoof
vnic0   allowed-ips         rw       10.0.0.1,       --         --
                                     10.0.0.2
vnic0   allowed-dhcp-cids   rw       hello           --         --
```

注 – allowed-ips プロパティーは、VALUE 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 リンク保護の統計情報を表示します。

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
```

詳細については、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

ネットワークのチューニング (タスク)

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 (タスクマップ)

タスク	説明	手順
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。	20 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。	21 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。	21 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。	22 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。	23 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。	23 ページの「中断中の TCP 接続の最大数を設定する方法」

タスク	説明	手順
初期の TCP 接続に対して強固な乱数を生成します。	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。	24 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。	24 ページの「ICMP リダイレクトを妨げる方法」
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。	25 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 – 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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, 2011 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, 2011 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 – _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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 詳細については、『Oracle Solaris 11.1 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 詳細については、『Oracle Solaris 11.1 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、トンネルモードの IPsec で VPN 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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　詳細については、『Oracle Solaris 11.1 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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　詳細については、『Oracle Solaris 11.1 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を RFC 6528 (http://www.ietf.org/rfc/rfc6528.txt) に準拠するように設定します。

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` 役割にこの承認が含まれています。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

1. **TCP_STRONG_ISS 変数のデフォルト値を変更します。**

```
# pfedit /etc/default/inetinit
# TCP_STRONG_ISS=1
TCP_STRONG_ISS=2
```

2. **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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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、現在の値を確認します。**

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

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

詳細については、『Oracle Solaris 11.1 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 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。

▼ 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 11.1 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 – 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 11.1 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

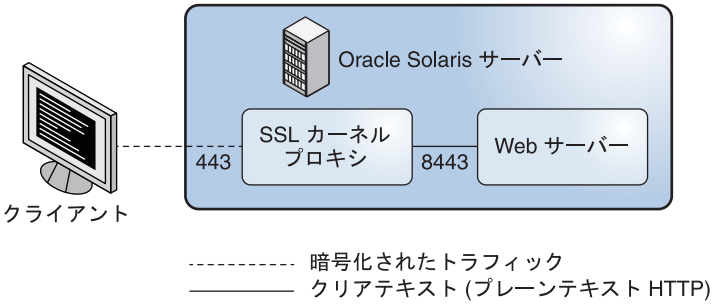
3 Web サーバーと Secure Sockets Layer プロトコル

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。

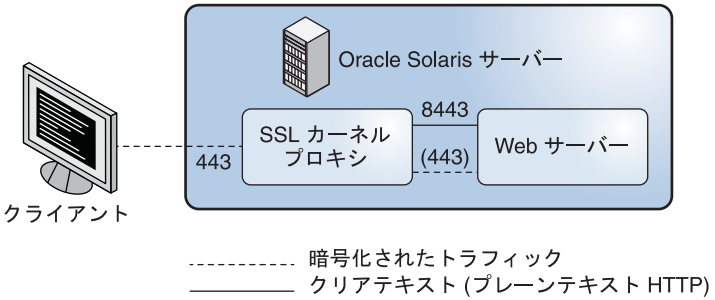
- このプロキシにより、Web サーバーのようなサーバーアプリケーションの SSL パフォーマンスが高速化するため、ユーザーレベルの SSL ライブラリに依存するアプリケーションよりも高いパフォーマンスを発揮します。アプリケーションのワークロードに応じて、パフォーマンスは 35% 以上向上する可能性があります。
- SSL カーネルプロキシ は透過的です。割り当てられた IP アドレスはありません。そのため、Webサーバーは、実際のクライアント IP アドレスと TCP ポートを参照します。
- SSL カーネルプロキシ と Web サーバーは連携するように設計されています。

  図 3–1 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 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 (タスク)

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。

- 29 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」
- 31 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」
- 33 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」
- 35 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 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 PKCS#11-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 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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、29 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」の手順 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 PKCS#11-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 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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. **Certificate Signing Request (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
      ```

      詳細については、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 – パスフレーズには空白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 **8443** ポート上でテキストを待機するように **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 kernel proxy 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 **(省略可能) 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。大域ゾーンでの `ksslcfg` コマンドによるサービスインスタンスの構成が完了すると、局所ゾーンで 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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

Oracle Solaris の IP フィルタ (概要)

この章では、Oracle Solaris の機能の 1 つである IP フィルタの概要を説明します。IP フィルタを使用したタスクについては、第 5 章「IP フィルタ (タスク)」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。

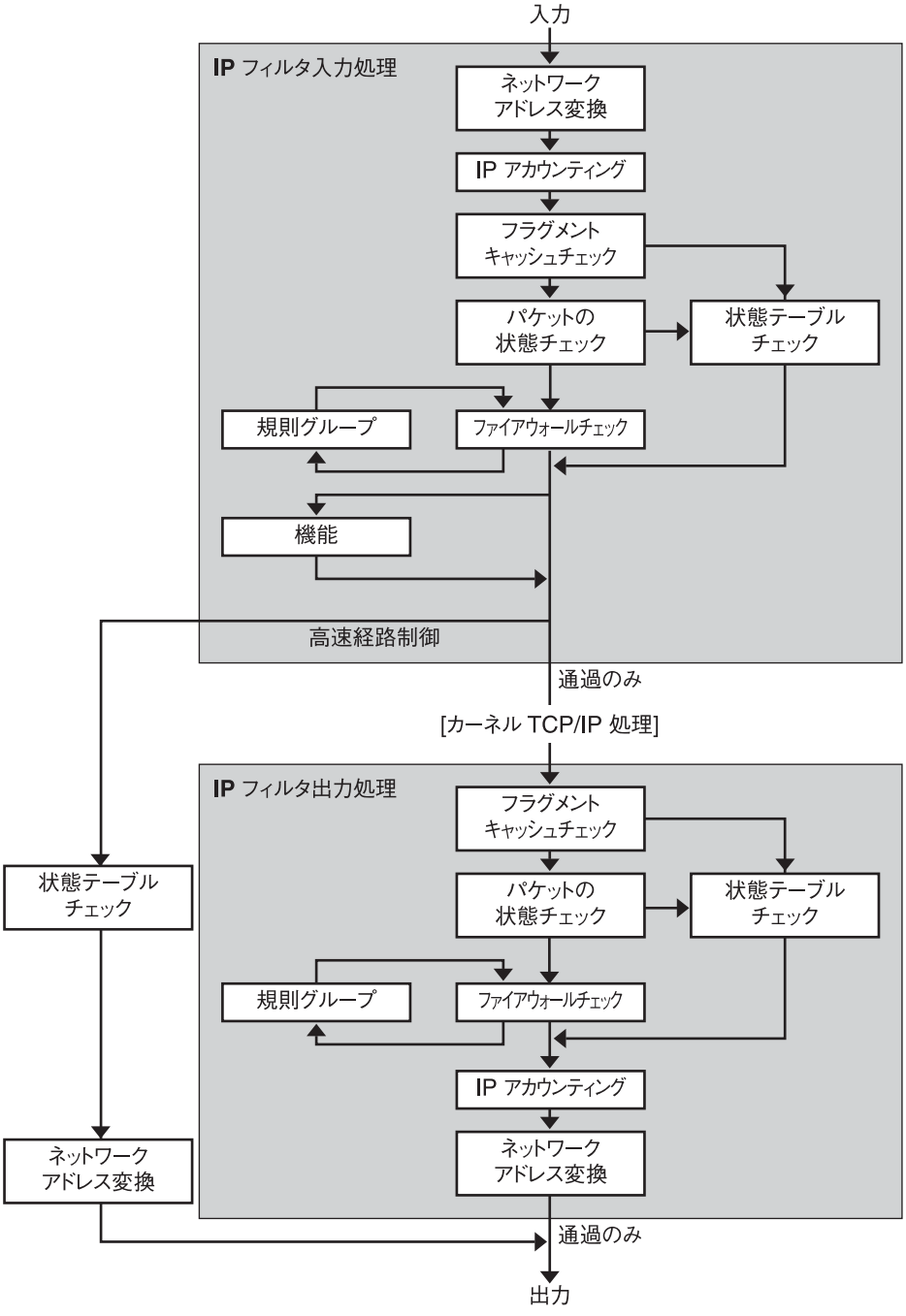
オープンソースの IP フィルタの情報ソース

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 の詳細については、『Oracle Solaris 11.1 でのサービスと障害の管理』の第 1 章「サービスの管理 (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 11.1 でのサービスと障害の管理』の第 2 章「サービスの管理 (タスク)」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)` 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、51 ページの「IP フィルタの構成」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 の権利プロファイルを含む役割になる必要があります。IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「RBAC の初期構成 (タスクマップ)」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ガイド』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、77 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

さらに、まとまった IP アドレスを参照するために、アドレスプールを作成することもできます。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、規則処理を速めるために役立ちます。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* 上の複製を任意で指定した IP アドレスに送信します。

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、以下の順序で使用してください。

注 – 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、59 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 11.1 ネットワークの構成と管理』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。詳細については、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。しかし、1 つの規則で両方のアドレスの種類を指定することはできません。アドレスの種類ごとに規則を別個に設定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まとまりに名前付けを行います。アドレスプールは、IP アドレスと規則の一致を取るため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、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/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。この時点で、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。ほかの IPv6 機能について学習するには、『System Administration Guide: IP Services』の「Major Features of IPv6」を参照してください。

注 – ジャンボグラムの詳細については、Internet Engineering Task Force (IETF) の RFC 2675、『IPv6 Jumbograms』のドキュメントを参照してください。`http://www.ietf.org/rfc/rfc2675.txt`

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を使用するために、`-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 パケットのログを記録でき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。

IPv6 の詳細は、『System Administration Guide: IP Services』の第 3 章「Introducing IPv6 (Overview)」を参照してください。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 (タスク)」を参照してください。

IP フィルタのマニュアルページ

次の表に IP フィルタに関するマニュアルページのドキュメントを示します。

マニュアルページ	説明
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

IP フィルタ (タスク)

この章では、タスクの手順をステップごとに説明します。IP フィルタの概要情報は、第 4 章「Oracle Solaris の IP フィルタ (概要)」 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 フィルタの構成

次のタスクマップに、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示します。

表 5–1 IP フィルタの構成 (タスクマップ)

タスク	手順
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。	52 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。	53 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。	55 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。	55 ページの「パケット再構築を無効にする方法」

表 5–1 IP フィルタの構成 (タスクマップ) (続き)

タスク	手順
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。	56 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 フィルタを使用して停止します。	57 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 つのファイルのプロパティーにファイルの場所が提案されています。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には、そのファイルのプロパティー値を変更します。手順については、53 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。53 ページの「IP フィルタ構成ファイルの作成方法」の手順 1 と手順 2 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確認します。

- 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、53 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 – 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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規則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、42 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、77 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 – 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 **(省略可能) 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、`/etc/ipf/ipnat.conf` など、適切な名前で NAT 規則のファイルを作成します。この名前を変更するには、次のように、`config/ipnat_config_file` サービスプロパティーの値を変更します。

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

NAT については、45 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 **(省略可能) 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、`/etc/ipf/ippool.conf` などの適切な名前でプールのファイルを作成します。この名前を変更するには、次のように、`config/ippool_config_file` サービスプロパティーの値を変更します。

```
$ 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、47 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 **(省略可能) 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。56 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 **(省略可能) 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、55 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

53 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 – `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、58 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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。

```
# 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 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 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、55 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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、58 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

次のタスクマップに、IP フィルタの規則セットに関連するタスクを示します。

表 5–2 IP フィルタ規則セットの操作 (タスクマップ)

タスク	手順
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。	59 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	59 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	60 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	61 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	61 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」 63 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	63 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	65 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	65 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。	66 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。	66 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	67 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	68 ページの「アドレスプールを削除する方法」
アドレスプールに規則を追加します。	68 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 – 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、52 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」の手順 2 で説明するように、-6 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 – 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。

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

(Edit the /etc/ipf/myorg.ipf.conf configuration file.)

$ svcadm refresh network/ipfilter
$ ipfstat -io (Optional)
empty list for ipfilter (out)
block in log quick from 10.0.0.0/8 to any
block in quick on net11 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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

既存の規則セットに規則を追加すると、テストやデバッグ時に役に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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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、規則セットに規則を追加します。

    ```
    $ 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。
    a. 適当なファイルに規則セットを作成します。
    b. 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規則セットを作成すると、テストまたはデバッグ時に役立ちます。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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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

カーネル内で別の規則セットに切り替えると、テストまたはデバッグ時に役立ちます。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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 – 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 – 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、参照および変更します。

▼ 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 NAT 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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

既存の規則セットに規則を追加すると、テストやデバッグ時に役に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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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。
  a. 適当なファイルに追加の NAT 規則を作成します。
  b. 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、参照および変更します。

▼ 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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

既存の規則セットに規則を追加すると、テストやデバッグ時に役に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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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、規則セットに規則を追加します。

```
$ 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。
  a. 適当なファイルに追加のアドレスプールを作成します。
  b. 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。
61 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 – IP フィルタサービスをリフレッシュしたり、再起動したりしないでください。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 (タスクマップ)

タスク	手順
状態テーブルを表示します。	70 ページの「IP フィルタの状態テーブルを参照する方法」
パケットの状態に関する統計情報を表示します。	71 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。	72 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 統計情報を表示します。	72 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	72 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

- 状態テーブルを参照します。

```
$ ipfstat
```

注 – -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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

- **NAT 統計情報を表示します。**

```
$ 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 (タスクマップ)

タスク	手順
IP フィルタログファイルを別個に作成する	73 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。	74 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	76 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	76 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 フィルタのログ情報はすべて `syslogd` ファイルに記録されます。デフォルトの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個に、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オンライン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 – 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 – 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 – rsyslog サービスがオンラインである場合は、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。

/etc/syslog.conf で:

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

コマンド行で、次のコマンドを実行します。

```
# 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 73 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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** フィルタログファイルを表示できます。

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

    注 – 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 コマンドを実行すると、ipmon の別のコピーも開かれます。この場合、両方のコピーが同じログ情報を読み取るため、一方だけが特定のログメッセージを取得します。

  ログファイルの参照については、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、バッファーを消去し、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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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。ログファイルに何も保存されていない場合も、この例のようなレポートが出力されます。

```
# 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

デバッ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することができます。

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、規則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、43 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たホストマシン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
```

例 5–20 IP フィルタのホスト構成 (続き)

```
# 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
```

この規則セットは、すべてのデータが net0 インタフェースを出入りできる 2 つの制限なしの規則で開始します。2 番目の規則セットは、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。次の規則セットは、ホストマシン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マシンの構成を示しています。このマシンには、net0 ネットワークインタフェースがあります。

```
# 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 \
```

例 5–21 IP フィルタのサーバー構成 *(続き)*

```
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 と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 aren't 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
```

例 5–22 IP フィルタのルーター構成 (続き)

```
# 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

IP セキュリティーアーキテクチャー (概要)

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

この章では、次の内容について説明します。

IPsec をネットワークに実装するには、第 7 章「IPsec の構成 (タスク)」を参照してください。参考情報については、第 8 章「IP セキュリティーアーキテクチャー (リファレンス)」を参照してください。

IPsec とは

IPsec は、パケットの認証または暗号化、もしくはこの両方を実行することで、IP パケットを保護します。IPsec は IP モジュールの内側で実行されます。したがって、インターネットアプリケーションは、IPsec を使用するために自分自身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 の保護に関連する主要コンポーネントは次のとおりです。

- セキュリティープロトコル – IP データグラムの保護メカニズムです。認証ヘッダー (AH) は IP パケットのハッシュを含み、完全性を保証します。データグラムの内容は暗号化されませんが、パケットの内容が変更されていないことが受信側に保証されます。また、パケットが送信側によって送られたことも保証されます。カプセル化セキュリティーペイロード (ESP)は、IP データを暗号化し、パケット転送中に内容が分からないようにします。ESP は、認証アルゴリズムオプションによってデータの完全性も保証します。
- セキュリティーアソシエーション **(SA)** – ネットワークトラフィックの特定のフローに適用される暗号化パラメータと IP セキュリティープロトコル。各 SA は、セキュリティーパラメータインデックス (SPI) と呼ばれる一意の参照を持ちます。
- セキュリティーアソシエーションデータベース **(SADB)** – IP 宛先アドレスと索引番号にセキュリティープロトコルを関連付けるデータベースです。この索引番号は、セキュリティーパラメータインデックス (SPI)と呼ばれます。これらの 3 つの要素 (セキュリティープロトコル、宛先アドレスおよび SPI) は、正当な IPsec パケットを一意に識別します。セキュリティーアソシエーションデータベースにより、パケットの宛先に届いた保護対象のパケットは確実に受信側に認識されます。また、受信側は、このデータベースの情報を使用して、通信を復号化し、パケットが変更されていないことを確認し、パケットを再度組み立て、そのパケットを最終的な宛先に届けます。
- 鍵管理 – 暗号化アルゴリズムおよび SPI 用の鍵の生成と配布です。
- セキュリティーメカニズム – IP データグラム内のデータを保護する認証アルゴリズムと暗号化アルゴリズムです。
- セキュリティーポリシーデータベース **(SPD)** – パケットに適用される保護レベルを指定するデータベースです。SPD は、IP トラフィックをフィルタリングし、パケットの処理方法を決定します。パケットは破棄したり、問題ない場合は、通過させたりできます。また、IPsec で保護することも可能です。アウトバウンドパケットの場合、SPD と SADB が適用する保護レベルを決定します。インバウンドパケットの場合、パケットの保護レベルを許容できるかどうかの決定に SPD が役立ちます。パケットが IPsec によって保護されている場合は、パケットを復号化し、確認したあとに、SPD が参照されます。

IPsec は、セキュリティーメカニズムを IP 宛先アドレスに転送される IP データグラムに適用します。受信側ユーザーは、SADB の情報を使用して、到着パケットが正当なことを確認し、それらを復号化します。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 データグラム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 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。すべての RFC の著作権は、インターネット協会が有しています。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 (IKE)』、1998 年 11 月
- RFC 3554、『On the Use of Stream Control Transmission Protocol (SCTP) with IPsec』、2003 年 7 月

IPsec の用語

IPsec RFC は、IPsec をシステムに実装する際に分かっていると便利な用語を多数定義しています。次の例は、IPsec の用語、それらの一般的に使用されている用語を示し、各用語を定義しています。鍵のネゴシエーションで使用する用語の一覧は、表 9–1 を参照してください。

表 6–1　IPsec の用語、略語、および使用方法

IPsec の用語	略語	定義
セキュリティーアソシエーション	SA	ネットワークトラフィックの特定のフローに適用される暗号化パラメータと IP セキュリティープロトコル。SA は、セキュリティープロトコル、一意のセキュリティーパラメータインデックス (SPI)、および IP 宛先の 3 つで定義されます。
セキュリティーアソシエーションデータベース	SADB	アクティブなセキュリティーアソシエーションをすべて含むデータベース。
セキュリティーパラメータインデックス	SPI	セキュリティーアソシエーションの索引値。SPI は、同じ IP 宛先およびセキュリティープロトコルを持つ SA を区別する 32 ビットの値です。

表 6–1 IPsec の用語、略語、および使用方法 (続き)

IPsec の用語	略語	定義
セキュリティーポリシーデータベース	SPD	アウトバウンドパケットとインバウンドパケットの保護レベルが指定どおりかを判断するデータベース。
鍵交換		非対称暗号化アルゴリズムを使用して鍵を生成する処理。主な手法には、RSA と Diffie-Hellman の 2 つがあります。
Diffie-Hellman	DH	鍵生成と鍵認証を可能にする鍵交換アルゴリズム。しばしば「認証された鍵交換」と呼ばれます。
RSA	RSA	鍵生成と鍵配布を可能にする鍵交換アルゴリズム。このプロトコル名は、作成者の Rivest、Shamir、Adleman の三氏に因んでいます。
インターネットセキュリティーアソシエーションおよび鍵管理プロトコル	ISAKMP	SA 属性の形式を設定し、SA のネゴシエーション、変更、削除を行うための共通フレームワーク。ISAKMP は、IKE 交換を処理するための IETF 標準です。

IPsec パケットのフロー

図 6–1 に、アウトバウンドパケットに対して IPsec が呼び出されたときに、IP データグラム の一部として、IP アドレス設定されたパケットが送られるしくみを示します。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。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は、これ以降のセクションで説明します。

図 6–2 は、IPsec 入力プロセスを示しています。

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

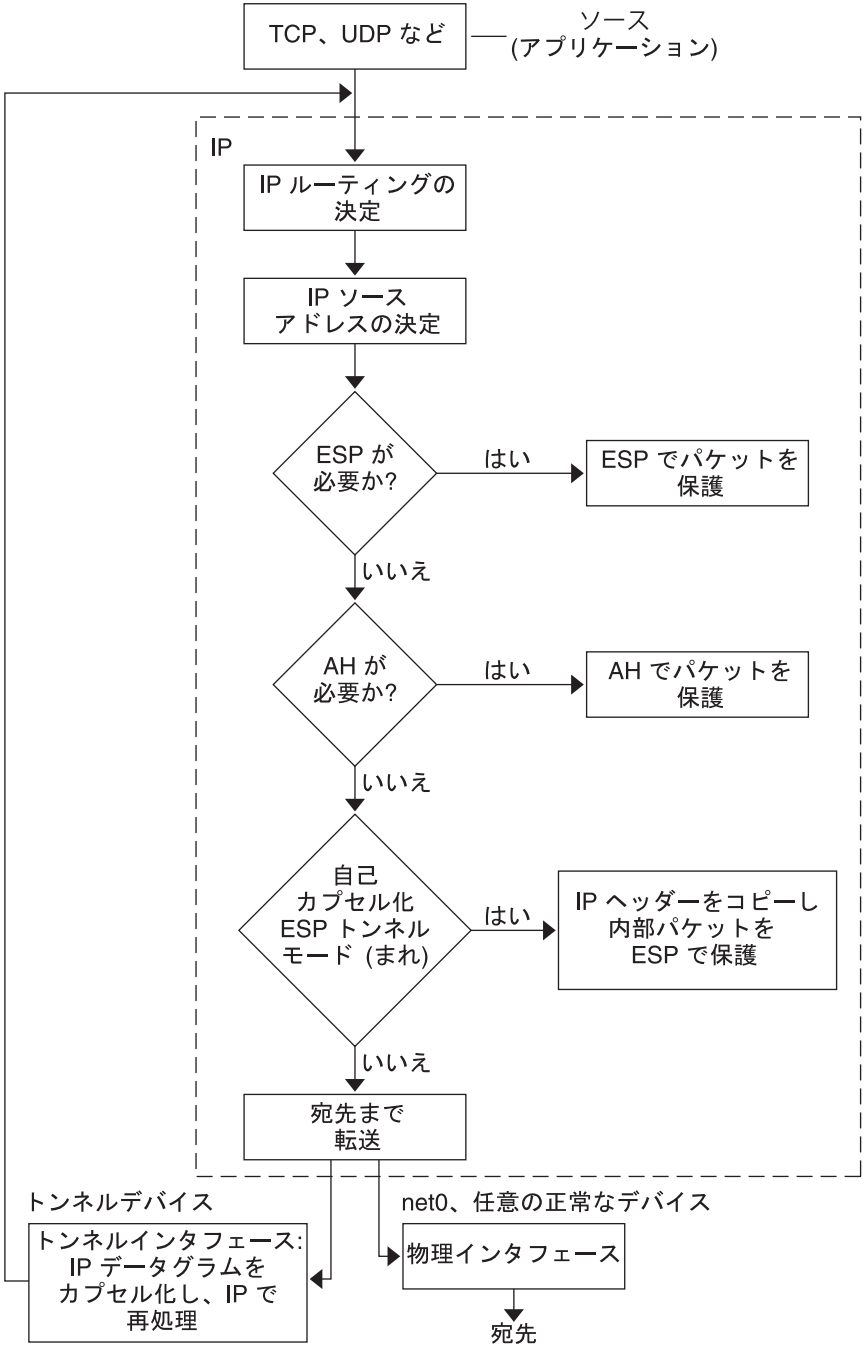
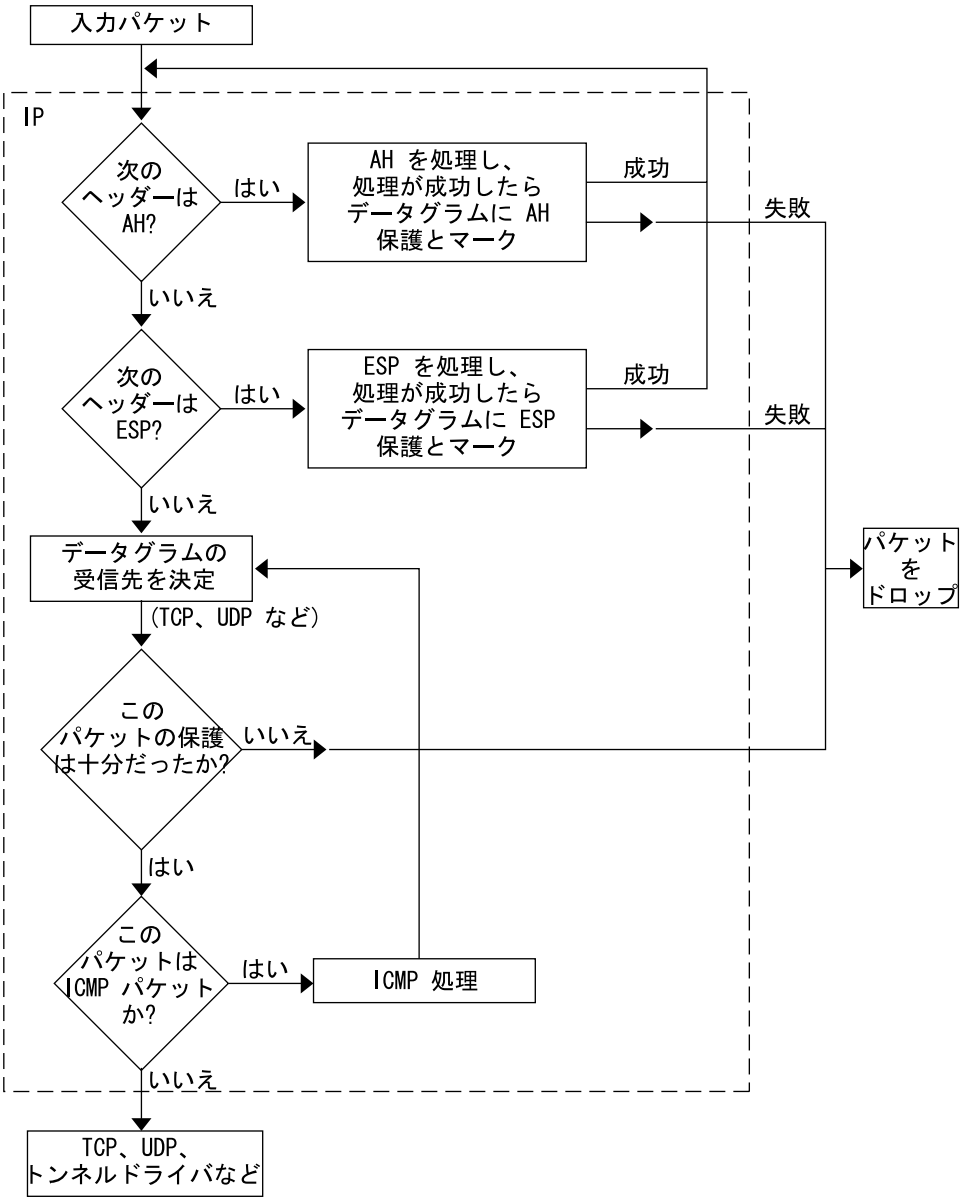


図 6–2 IPsec をインバウンドパケットプロセスに適用



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、通信するホストが認識するセキュリティープロパティーを示します。1 つの SA は、1 方向のデータを保護します。つまり、1 つのホストかグループ (マルチキャスト) アドレスのどちらかです。大部分の通信がピアツーピアかクライアントサーバーなので、両方向のトラフィックの安全性を確保するために 2 つの SA が必要です。

次の 3 つの要素は、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)

任意の 32 ビット値の SPI は、AH パケットまたは ESP パケットで転送されます。AH および ESP によって保護される範囲については、 `ipsecah(7P)` と `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。

SA 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管理できます。たとえば、IKE アプリケーションと `ipseckeys` 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、127 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

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。このキーイング素材の管理を鍵管理と呼びます。IKE (インターネット鍵交換) プロトコルにより、鍵管理が自動的に行われます。また、`ipseckey` コマンドを指定して、鍵管理を手動で行うこともできます。

IPv4 と IPv6 パケットの SA は、どちらの鍵管理方法も使用できます。手動で鍵管理を行う決定的な理由がないかぎり、IKE をお勧めします。

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:default` サービス – 自動鍵管理のための SMF サービスです。`ike` サービスは `in.iked` デーモンを実行して自動鍵管理を提供します。IKE については、第 9 章「インターネット鍵交換 (概要)」を参照してください。`in.iked` デーモンの詳細については、`in.iked(1M)` のマニュアルページを参照してください。`ike` サービスについては、171 ページの「IKE サービス」を参照してください。
- `svc:/network/ipsec/manual-key:default` サービス – 手動での鍵管理のための 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` コマンドについては、127 ページの「IPsec の SA を生成するためのユーティリティー」を参照してください。`ipseckey` コマンドオプションの詳細な説明については、`ipseckey(1M)` のマニュアルページを参照してください。

IPsec の保護メカニズム

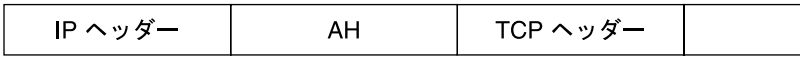
IPsec は、データを保護するために次の 2 つのセキュリティープロトコルを提供しています。

- 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、認証アルゴリズムでデータを保護します。ESP は、暗号化アルゴリズムでデータを保護します。ESP は認証メカニズムと組み合わせて使用可能であり、またそうすべきです。NAT をトラバースしない場合は、ESP と AH を組み合わせることができます。それ以外の場合、ESP で認証アルゴリズムと暗号化メカニズムを使用できます。AES-GCM のような複合モードのアルゴリズムでは、単一のアルゴリズム内で暗号化と認証が提供されます。

認証ヘッダー

認証ヘッダーは、IP データグラムに対するデータ認証、強力な完全性、再送保護を供給します。AH では大部分の IP データグラムを保護します。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)モジュールは、ESP がカプセル化した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、保護される対象は、データグラムのうち ESP がカプセル化した部分だけです。ESP は、保護されたパケットの完全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 はデータをカプセル化します。したがって、次の図に示されているように、ESP が保護するのはデータグラム内の EPS の開始点以降のデータのみです。

暗号化部分

TCP パケットでは、ESP は TCP ヘッダーとそのデータだけをカプセル化します。パケットが IP 内 IP データグラムの場合、ESP は内部 IP データグラムを保護します。ソケット別ポリシーでは、「自己カプセル化」ができるため、必要に応じて ESP では IP オプションをカプセル化できます。

自己カプセル化が設定されている場合は、IP 内 IP データグラムを構築するために 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、データグラム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、データグラム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、94 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 と ESP が提供する保護を比較しています。

表 6–2 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	IP ヘッダーからトランスポートヘッダーまでのパケットを保護	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
ESP	データグラムの ESP 開始後のパケットを保護	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 セキュリティープロトコルは、認証と暗号化という 2 種類のアルゴリズムを提供しています。AH モジュールは、認証アルゴリズムを使用します。ESP モジュールは、暗号化アルゴリズムと認証アルゴリズムを使用します。`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アルゴリズムにアクセスします。暗号化フレームワークは、その他のサービスに加えて、アルゴリズムの集中リポジトリを提供します。このフレームワークによって、IPsec は、高性能な暗号ハードウェアアクセラレータを利用できます。

詳細については、次を参照してください。

- 『Oracle Solaris 11.1 の管理: セキュリティーサービス』の第 11 章「暗号化フレームワーク (概要)」
- 『Oracle Solaris 11 セキュリティーサービス開発ガイド』の第 8 章「Oracle Solaris 暗号化フレームワークの紹介」

IPsec での認証アルゴリズム

認証アルゴリズムは、データとキーを基に整合性チェックサムの値、つまり、「ダイジェスト」を生成します。AH モジュールは、認証アルゴリズムを使用します。ESP モジュールも、認証アルゴリズムを使用します。

IPsecでの暗号化アルゴリズム

暗号化アルゴリズムは、キーでデータを暗号化します。IPsecのESPモジュールは、暗号化アルゴリズムを使用します。暗号化アルゴリズムでは、「ブロックサイズ」ごとにデータを処理します。

IPsecの保護ポリシー

IPsecの保護ポリシーは、どのセキュリティーメカニズムも使用できます。IPsec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、システム共通ポリシーをアウトバウンドデータグラムとインバウンドデータグラムに適用します。アウトバウンドデータグラムは、保護付きまたは保護なしで送信されます。保護が適用されると、特定アルゴリズムか汎用アルゴリズムのどちらかになります。システムで認識されるデータがあるため、アウトバウンドデータグラムにはその他の規則も適用できます。インバウンドデータグラムの処理は、受理されるか拒絶されるかのどちらかです。インバウンドデータグラムの受理か拒絶を決定する基準はいくつかありますが、場合によってはその基準が重複したり競合することがあります。競合の解決に当たっては、どの規則の構文解析を最初に行うかが決定されます。ポリシーのエントリによって、そのトラフィックがすべてのほかのポリシーを省略すると指示されている場合を除いて、トラフィックは自動的に受理されます。

データグラムを保護する通常のポリシーを省略することもできます。それには、システム規模ポリシーに例外を指定するか、ソケット単位ポリシーで省略を要求します。システム内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。

`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。これらのモードは、パケットのエンコードには影響を与えません。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されます。内側のパケットが IP パケットである場合に、モードによってポリシーの適用方法が次のように異なります。

- トランスポートモードでは、外側のヘッダーによって、内側の IP パケットを保護する IPsec ポリシーが決まります。
- 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。

トランスポートモードでは、外側のヘッダー、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。同様に、次のヘッダーが IP ヘッダーであれば、外側のヘッダーと内側の IP ヘッダーを使用して IPsec ポリシーを決定できます。

トンネルモードは IP 内 IP データグラムに対してのみ機能します。トンネルモードのトンネリングは、自宅のコンピュータから中央コンピュータに接続する場合に役立ちます。トンネルモードでは、IPsec ポリシーは内側の IP データグラム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、ポリシーを適用することができます。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 データグラム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、108 ページの「IPsec による VPN の保護」を参照してください。

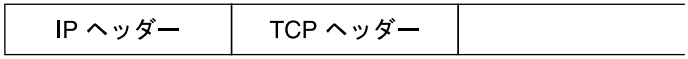
Oracle Solaris では、IP トンネルネットワークインタフェースにのみトンネルモードを適用できます。トンネルインタフェースの詳細は、『Oracle Solaris 11.1 ネットワークの構成と管理』の第 6 章「IP トンネルの構成」を参照してください。`ipsecconf` コマンドには、IP トンネルネットワークインタフェースを選択するための `tunnel`

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。

トランスポートモードでは、ESP または AH、あるいはその両方を使用してデータグラムを保護できます。

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



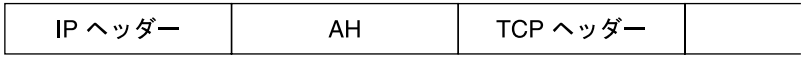
トランスポートモードで、ESP は次の図のようにデータを保護します。網かけされた領域は、パケットの暗号化された部分を示します。

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

トランスポートモードで、AH は次の図のようにデータを保護します。

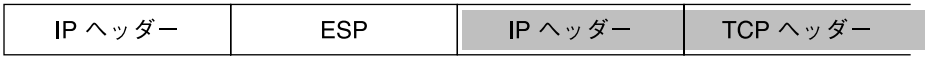
図 6–5 認証ヘッダーで保護されたパケット



AH 保護では、トランスポートモードの場合でも、IP ヘッダーの大部分が保護されます。

トンネルモードでは、データグラム全体が IPsec ヘッダーの保護下にあります。図 6–3 のデータグラムは、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 6–6 トンネルモードで保護された IPsec パケット



暗号化部分

ipsecconf コマンドには、トンネルをトンネルモードまたはトランスポートモードで設定するためのキーワードが用意されています。

- 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については、105 ページの「IPsec を使って Web 以外のトラフィックから Web サーバーを保護する方法」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。
- トンネル構成の例については、111 ページの「トンネルモードの IPsec で VPN 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

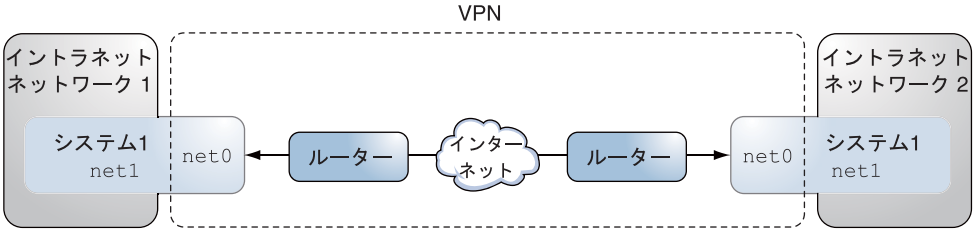
構成したトンネルは、ポイントツーポイントインタフェースです。トンネルによって、IP パケットを別の IP パケット内にカプセル化できます。トンネルの構成には、トンネルソースとトンネル宛先が必要です。詳細については、『Oracle Solaris 11.1 ネットワークの構成と管理』の「IP トンネルを作成および構成する方法」を参照してください。

トンネルは、IP への物理インタフェースのようなものを作成します。この物理的リンクの完全性は、基本になるセキュリティープロトコルによって異なります。セキュリティーアソシエーション (SA) を確実に行えば、信頼性の高いトンネルになります。トンネルのデータパケットのソースはトンネル宛先で指定したピアでなければなりません。この信頼関係があるかぎり、インタフェース別 IP 送信を利用して仮想プライベートネットワーク (VPN) を作成できます。

VPN に IPsec 保護を追加できます。IPsec が接続の安全性を確保します。たとえば、分離したネットワークを持つ複数のオフィスを VPN テクノロジを使用して接続している組織は、IPsec を追加して 2 つのオフィス間のトラフィックをセキュリティー保護できます。

次の図は、ネットワークシステムに配備した IPsec で、2 つのオフィスが VPN を形成する方法を示しています。

図 6–7 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、111 ページの「トンネルモードの IPsec で VPN を保護する方法」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、システムは、システムが NAT デバイスの背後にある場合も、リモートネットワークから安全に接続を行うことができます。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 は、Network Address Translation (ネットワークアドレス変換) の略語です。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。詳細は、45 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。

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 ヘッダーに依存しますので、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` から取得できます。

- 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

NAT を通して IPsec を使用するには、162 ページの「移動体システム用の IKE の構成 (タスクマップ)」を参照してください。

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 については、RFC と『System Administration Guide: IP Services』の「SCTP Protocol」を参照してください。

IPsec と Oracle Solaris ゾーン

共有 IP ゾーンについては、IPsec の構成は大域ゾーンから行います。IPsec ポリシー構成ファイル `ipsecinit.conf` は、大域ゾーンだけに存在します。このファイルには、大域ゾーンに適用するエントリだけでなく、非大域ゾーンに適用するエントリも含めることができます。

排他的 IP ゾーンについては、IPsec は非大域ゾーンごとに構成されます。

IPsec をゾーンで使用する方法については、101 ページの「IPsec によるトラフィックの保護」を参照してください。ゾーンについては、『Oracle Solaris 11.1 の管理: Oracle Solaris ゾーン、Oracle Solaris 10 ゾーン、およびリソース管理』の第 15 章「Oracle Solaris ゾーンの紹介」を参照してください。

IPsec と論理ドメイン

IPsec は論理ドメインで動作します。論理ドメインは、IPsec を含む Oracle Solaris バージョン (Oracle Solaris 10 リリースなど) を実行している必要があります。

論理ドメインを作成するには、Oracle VM Server for SPARC (以前の名称は Logical Domains) を使用する必要があります。論理ドメインを構成する方法については、『Oracle VM Server for SPARC 2.2 管理ガイド』を参照してください。

IPsec ユーティリティーおよび IPsec ファイル

表 6–3 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 11.1 でのサービスと障害の管理』の第 1 章「サービスの管理 (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、101 ページの「IPsec による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 8 章「IP セキュリティーアーキテクチャー (リファレンス)」を参照してください。

表 6–3 選択される IPsec ユーティリティーとファイルのリスト

IPsec ユーティリティー、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`	IKE を使用した IPsec SA の自動管理用の SMF サービス。	`smf(5)`, `in.iked(1M)`
`/etc/inet/ipsecinit.conf` ファイル	IPsec ポリシーファイル。	`ipsecconf(1M)`
	SMF `policy` サービスはシステムのブート時にこのファイルを使用して IPsec ポリシーを構成します。	
`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。	

表 6–3 選択される IPsec ユーティリティーとファイルのリスト (続き)

IPsec ユーティリティー、ファイル、または サービス	説明	マニュアルページ
PF_KEY ソケットインタフェース	SA 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 キー作成コマンド。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。 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。	ipsecalgs(1M)
/etc/inet/ipsecalgs ファイル	構成されている IPsecプロトコル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/config ファイル	IKE 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。/etc/inet/ike/config ファイル内の規則およびグローバルパラメータに基づいて鍵管理が行われます。135 ページの「IKE ユーティリティーおよび IKE 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 サービスは IKE デーモン in.iked を起動して自動鍵管理を提供します。	ike.config(4)

◆ ◆ ◆ 第 7 章

IPsecの構成(タスク)

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。

- 101ページの「IPsecによるトラフィックの保護」
- 108ページの「IPsecによるVPNの保護」
- 115ページの「IPsecおよびIKEの管理」

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(概要)」を参照してください。IPsecの参考情報については、第8章「IPセキュリティーアーキテクチャー(リファレンス)」を参照してください。

IPsecによるトラフィックの保護

このセクションでは、2つのシステム間のトラフィックを保護する手順と、Webサーバーを保護する手順について説明します。VPNを保護するには、108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、115ページの「IPsecおよびIKEの管理」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** – 共有IP非大域ゾーンのIPsecポリシーと鍵を管理するには、大域ゾーンでIPsecポリシーファイルを作成し、大域ゾーンでIPsec構成コマンドを実行します。構成中の非大域ゾーンに対応する発信元アドレスを使用してください。排他的IPゾーンについては、非大域ゾーンでIPsecポリシーを構成します。
- **IPsecとRBAC** – IPsecを管理する役割を使用するには、『Oracle Solaris 11.1の管理:セキュリティーサービス』の第9章「役割に基づくアクセス制御の使用(タスク)」を参照してください。例については、117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 は、Streams Control Transmission Protocol (SCTP) アソシエーションを保護するのに使用できますが、注意が必要です。詳細は、98 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。そのような構成では、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。

タスク	説明	手順
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。	103 ページの「IPsec で 2 つのシステム間のトラフィックを保護するには」
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。	105 ページの「IPsec を使って Web 以外のトラフィックから Web サーバーを保護する方法」
IPsec ポリシーを表示します。	現在適用されている IPsec ポリシーを、適用された順に表示します。	107 ページの「IPsec ポリシーを表示するには」
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。	セキュリティーアソシエーション向けの raw データを提供します。	139 ページの「IKE の構成 (タスクマップ)」
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。	108 ページの「IPsec による VPN の保護」

▼ IPsec で 2 つのシステム間のトラフィックを保護するには

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。

- 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。
- 各システムには、AES アルゴリズムを使用した ESP 暗号化 (128 ビットの鍵が必要) と、SHA-2 メッセージダイジェストを使用した ESP 認証 (512 ビットの鍵が必要) が必要です。
- 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 – 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 IPsec ポリシーは、大域ゾーン内または排他的 IP スタックゾーン内で構成できます。共有 IP スタックゾーンのポリシーは大域ゾーン内で構成する必要があります。排他的 IP ゾーンについては、非大域ゾーンで IPsec ポリシーを構成します。

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。システムファイルを編集して鍵を作成するには、root 役割になる必要があります。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

リモートからログインする場合、セキュアなリモートログイン用の ssh コマンドを使用してください。例については、例 7–1 を参照してください。

1 各システム上で、**/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、存在しないネームサービスに依存しなくても、サービス管理機能 (SMF) でシステム名が使用できるようになります。詳細は、smf(5) のマニュアルページを参照してください。

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。

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

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}
```

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}
```

IPsec ポリシーエントリの構文については、`ipsecconf(1M)`のマニュアルページを参照してください。

4 各システムで、2 つのシステム間に **IPsec SA** ペアを追加するために **IKE** を構成します。

139 ページの「IKE の構成 (タスクマップ)」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、`ike.config(4)` のマニュアルページを参照してください。

注 – 鍵を手動で生成して維持する必要がある場合は、115 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。

```
# ipsecconf -f -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 svc:/network/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 svc:/network/ipsec/policy:default
```

7 **IPsec** の鍵を有効化します。

- **ike** サービスが有効になっていない場合は有効にします。

```
# svcadm enable svc:/network/ipsec/ike:default
```

- **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 svc:/network/ipsec/ike:default
```

手順 4 で鍵を手動で構成した場合は、115 ページの「IPsec の鍵を手動で作成する方法」を実行して鍵を有効化します。

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、121 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 接続を使用している場合に IPsec ポリシーを追加する

この例では、root 役割の管理者が、2 つのシステムの IPsec ポリシーと鍵を構成します。その際、ssh コマンドを使用して 2 番目のシステムにアクセスします。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。

- まず、前の手順の手順 1 から 手順 5 までを実行して、最初のシステムを構成します。
- 次に、別の端末ウィンドウで、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:
other-system #
```

- ssh セッションの端末ウィンドウで、手順 1 から手順 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。
- ここで ssh セッションを終了します。

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

- 最後に、手順 6 と手順 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。

▼ IPsec を使って Web 以外のトラフィックから Web サーバーを保護する方法

セキュアな Web サーバーでは、Web クライアントであれば Web サービスと通信できます。セキュアな Web サーバーでは、Web トラフィック以外のトラフィックは、セキュリティー検査を通る必要があります。次の手順には、Web トラフィックの検査省略手順が含まれています。さらに、この Web サーバーでは、セキュアでない DNS

クライアント要求を出すことができます。その他のすべてのトラフィックでは、AES と SHA-2 アルゴリズムによる ESP が必要です。

始める前に　IPsec ポリシーの構成は大域ゾーンで行う必要があります。排他的 IP ゾーンについては、非大域ゾーンで IPsec ポリシーを構成します。

103 ページの「IPsec で 2 つのシステム間のトラフィックを保護するには」を完了して、次の条件が成立しています。

- 2 つのシステム間の通信は IPsec で保護されています。
- 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。

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。システムファイルを編集するには、root 役割になる必要があります。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

リモートからログインする場合、セキュアなリモートログイン用の ssh コマンドを使用してください。例については、例 7–1 を参照してください。

1　セキュリティーポリシー検査を省略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 サーバーのポリシーを IPsec ポリシーファイルに追加します。

/etc/inet/ipsecinit.conf ファイルに次の行を追加します。

```
# 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、手順 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 -f -c /etc/inet/ipsecinit.conf
```

4　IPsec ポリシーをリフレッシュします。

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

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 svc:/network/ipsec/ike
```

鍵を手動で構成した場合は、115 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。必要に応じて、手順 6 を実行します。

6 **(省略可能) 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
#
{laddr 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有効にします。

```
remote-system # ipsecconf -f -c /etc/inet/ipsecinit.conf
remote-system # svcadm refresh svc:/network/ipsec/policy:default
```

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。

▼ IPsec ポリシーを表示するには

引数を指定しないで ipsecconf コマンドを実行すると、システムに構成されているポリシーを確認できます。

始める前に ipsecconf コマンドは大域ゾーンで実行する必要があります。排他的 IP ゾーンについては、非大域ゾーンで ipsecconf コマンドを実行します。

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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

- **IPsec ポリシーを表示します。**
  - 追加された順序でグローバルな **IPsec** ポリシーエントリを表示します。

    ```
    $ ipsecconf
    ```

    各エントリが、「インデックス」とそのあとに番号が付いて表示されます。
  - 一致した順序で **IPsec** ポリシーエントリを表示します。

    ```
    $ ipsecconf -l -n
    ```

- 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
```

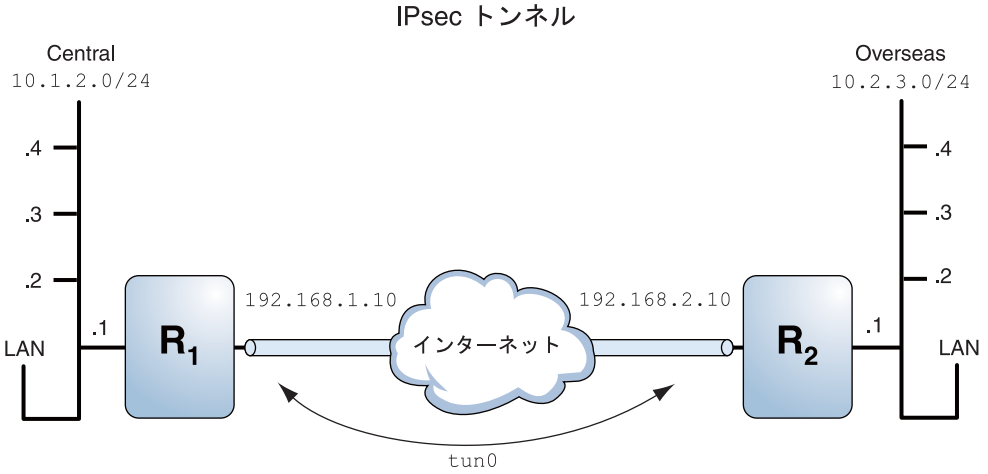
IPsec による VPN の保護

Oracle Solaris では、IPsec で保護された VPN を構成できます。トンネルは、トンネルモードまたはトランスポートモードで作成できます。詳しくは、94 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、101 ページの「IPsec によるトラフィックの保護」を参照してください。

トンネルモードのトンネル用の IPsec ポリシーの例については、108 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

図 7–1　IPsec で保護されたトンネル



次の例では、LAN のすべてのサブネットに対してトンネルを構成することを前提にしています。

```
## Tunnel configuration ##
# 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
```

例 7–2　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 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 7–3　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}
```

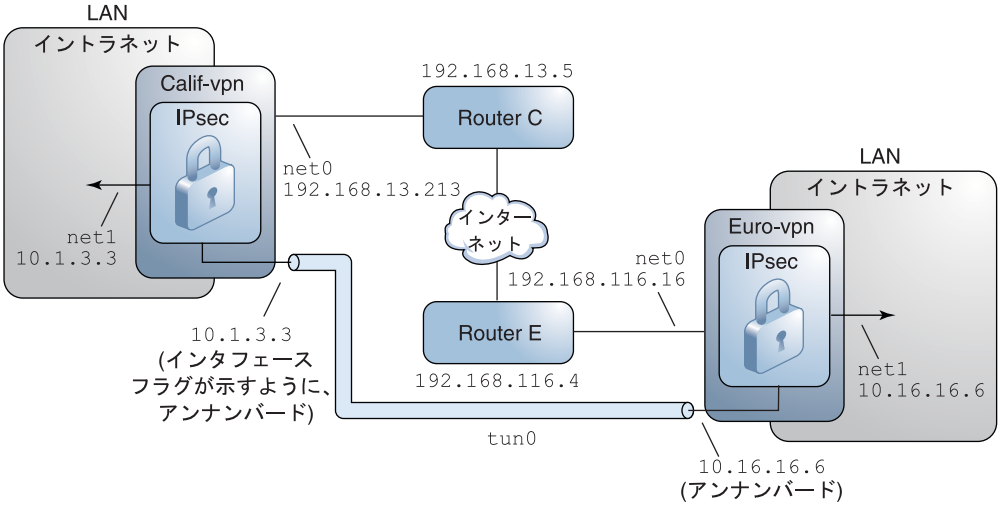
IPsec で VPN を保護するタスクのためのネットワークトポロジの説明

このセクション以降に説明する手順では、次の設定がすでになされているものとします。図 7–2 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。
- 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、SHA-2 アルゴリズムを使用した ESP 認証が必要です。この例の SHA-2 アルゴリズムでは、512 ビットの鍵が必要です。
- 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。

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



前の図に示すように、この手順では次の構成パラメータを使用します。

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。*Step 6* の 手順 6 アドレスでもある	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
システムイントラネットアドレス。*Step 6* の 手順 6 アドレスでもある	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

パラメータ	ヨーロッパ	カリフォルニア
トンネル名アドレスオブジェクト	`tun0/v4tunaddr`	`tun0/v4tunaddr`

トンネル名の詳細は、『Oracle Solaris 11.1 ネットワークの構成と管理』の「dladm コマンドによるトンネルの構成と管理」を参照してください。アドレスオブジェクトについては、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP 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。

▼ トンネルモードの IPsec で VPN 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、103 ページの「IPsec で 2 つのシステム間のトラフィックを保護するには」の手順の応用です。設定については、109 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より完全な説明については、103 ページの「IPsec で 2 つのシステム間のトラフィックを保護するには」の対応する手順を参照してください。

注 – 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 – 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 システムまたは共有 IP ゾーンの IPsec ポリシーの構成は、大域ゾーンで行う必要があります。排他的 IP ゾーンについては、非大域ゾーンで IPsec ポリシーを構成します。

構成コマンドを実行するには、Network Management および 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。システムファイルを編集するには、`root` 役割になる必要があります。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

リモートからログインする場合、セキュアなリモートログイン用の ssh コマンドを使用してください。例については、例 7–1 を参照してください。

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 厳密マルチホームをオンにします。**

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

IP 厳密宛先マルチホームをオンに設定するには、システムの着信先アドレスのうちの 1 つに宛てたパケットが正しい着信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。

ループバックマウントと ssh サービスが稼働していることを確認します。

```
# svcs | grep network
online         Aug_02   svc:/network/loopback:default
...
online         Aug_09   svc:/network/ssh:default
```

2 **IPsec ポリシーを追加します。**

/etc/inet/ipsecinit.conf ファイルを編集して、VPN 用の IPsec ポリシーを追加します。その他の例については、108 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入力します。**

```
# 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入力します。

```
# 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。

139 ページの「IKE の構成 (タスクマップ)」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、`ike.config(4)` のマニュアルページを参照してください。

注 – 鍵を手動で生成して維持する必要がある場合は、115 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検証します。

```
# ipsecconf -f -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 svc:/network/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 svc:/network/ipsec/policy:default
```

6 トンネル *tunnel-name* を作成して構成します。

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。

インタフェース `net1` がまだ存在しない場合、最初のコマンドによって作成されます。

```
# ipadm create-addr -T static -a local=10.1.3.3 net1/inside
# dladm create-iptun -T ipv4 -a local=10.1.3.3,remote=10.16.16.6 tun0
# ipadm create-addr -T static \
-a local=192.168.13.213,remote=192.168.116.16 tun0/v4tunaddr
```

b. **euro-vpn** システムで、トンネルを作成して構成します。

```
# ipadm create-addr -T static -a local=10.16.16.6 net1/inside
# dladm create-iptun -T ipv4 -a local=10.16.16.6,remote=10.1.3.3 tun0
# ipadm create-addr -T static \
-a local=192.168.116.16,remote=192.168.13.213 tun0/v4tunaddr
```

注 – 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。

これらのコマンドについては、dladm(1M) および ipadm(1M) のマニュアルページ、および『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP 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 管理: ネットワークインタフェースとネットワーク仮想化』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=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 転送をオンに設定しておきます。

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 転送はオンに設定しておきます。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 転送はオンのままである必要があります。net0 インタフェースの IP 転送はオフです。そのため、「外部」から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。「外部」とはインターネットを意味します。

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 転送がオフになっていても、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、インターネットルーティング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。詳細については、route(1M) と in.routed(1M) のマニュアルページを参照してください。

IPsec および IKE の管理

次のタスクマップでは、ユーザーが 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ます。

タスク	説明	手順
手動によるセキュリティーアソシエーションの作成または置き換えを行います。	次のようなセキュリティーアソシエーション向けのローデータを提供します。 ■ IPsec アルゴリズム名とキー作成素材 ■ セキュリティーパラメータインデックス (SPI) ■ 発信元と着信先の IP アドレス、およびその他のパラメータ	115 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。	117 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」
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、および再起動を行うコマンドを、いつどのように使用するかについて説明します。サービスのプロパティー値を変更するコマンドについても説明します。	119 ページの「IPsec および IKE サービスを管理する方法」
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 データグラム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。	121 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 の鍵を手動で作成する方法

次の手順では、103 ページの「IPsec で 2 つのシステム間のトラフィックを保護するには」の手順 4 で使用するキーイング素材を提供します。partym と enigma という 2 つ

のシステムの鍵を生成しようとしています。一方のシステムで鍵を生成してから、このシステムの鍵を両方のシステムで使用します。

始める前に　非大域ゾーンのキーイング素材を手動で管理するには、大域ゾーン内にいる必要があります。

root 役割になる必要があります。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

1　SA の鍵情報を生成します。

a. 必要な鍵を決定します。

16 進のアウトバウンドトラフィックと、同じく 16 進のインバウンドトラフィックには、それぞれ 3 種類の乱数が必要です。つまり、1 台のシステムで次の数値を生成する必要があります。

- spi キーワードの値として、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。
- AH の SHA-2 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。
- ESP の 3DES アルゴリズム用の 2 つの 16 進数の乱数。各数字の長さは 168 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。

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 11.1 の管理: セキュリティーサービス』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　各システム上で root 役割になり、IPsec の手動鍵ファイルに鍵を追加します。

a. **enigma** システム上で、**/etc/inet/secret/ipseckeys** ファイルの内容が次のようになるように編集します。

```
# 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 3des \
   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 3des \
   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
```

c. ファイルの構文を検証します。

```
# ipseckey -c -f /etc/inet/secret/ipseckeys
```

注 – 両システムの鍵情報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。

  ```
  # svcadm enable svc:/network/ipsec/manual-key:default
  ```

- **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。

▼ 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 の役割によるアクセス制御 (RBAC) 機能で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
Console User:RO::Manage System as the Console User...
Network Management:RO::Manage the host and network configuration...
Network Autoconf Admin:RO::Manage Network Auto-Magic configuration via nwamd...
Network Autoconf User:RO::Network Auto-Magic User...
Network ILB:RO::Manage ILB configuration via ilbadm...
Network LLDP:RO::Manage LLDP agents via lldpadm...
Network VRRP:RO::Manage VRRP instances...
Network Observability:RO::Allow access to observability devices...
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。

```
% getent exec_attr | grep "Network Management"
...
Network Management:solaris:cmd:::/sbin/dlstat:euid=dladm;egid=sys
...
Network Management:solaris:cmd:::/usr/sbin/snoop:privs=net_observability
Network Management:solaris:cmd:::/usr/sbin/spray:euid=0 ...
```

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、手順 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。

4 **Network Management** 権利プロファイルを含むネットワークセキュリティーの役割を作成します。

Network Management プロファイルに加え、Network Security または Network IPsec Management 権利プロファイルを持つ役割は、`ipadm`、`ipseckey`、`snoop` コマンドなどを適切な特権で実行できます。

役割の作成、役割のユーザーへの割り当て、ネームサービスでの変更の登録については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「RBAC の初期構成 (タスクマップ)」を参照してください。

例 7–4 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。

- 最初の役割には LinkWifi という名前を付けます。
  - 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。
  - その後、該当するユーザーにこの LinkWifi 役割を割り当てます。
- 2 番目の役割には IPsec Administrator という名前を付けます。
  - 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。
  - その後、該当するユーザーにこの IPsec Administrator 役割を割り当てます。

▼ IPsec および IKE サービスを管理する方法

次の手順では、IPsec の管理、IKE の管理、および手動での鍵管理に SMF サービスを使用する代表的な方法について説明します。デフォルトでは、`policy` サービスと `ipsecalgs` サービスは有効になっています。また、デフォルトでは、`ike` サービスと `manual-key` サービスは無効になっています。

始める前に `root` 役割になる必要があります。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

1 **IPsec** ポリシーを管理するには、次のいずれかを実行します。

- **`ipsecinit.conf`** ファイルに新しいポリシーを追加したあと、**`policy`** サービスをリフレッシュします。

```
# svcadm refresh svc:/network/ipsec/policy
```

- サービスのプロパティーの値を変更したあと、プロパティーの値を表示し、**`policy`** サービスをリフレッシュしてから再起動します。

```
# svccfg -s policy setprop config/config_file=/etc/inet/MyIpsecinit.conf
# svccfg -s policy listprop config/config_file
config/config_file  astring  /etc/inet/MyIpsecinit.conf
# svcadm refresh svc:/network/ipsec/policy
# svcadm restart svc:/network/ipsec/policy
```

2 鍵を自動的に管理するには、次のいずれかを実行します。

- **/etc/inet/ike/config** ファイルにエントリを追加したあと、**ike** サービスを有効にします。

```
# svcadm enable svc:/network/ipsec/ike
```

- **/etc/inet/ike/config** ファイルのエントリを変更したあと、**ike** サービスを再起動します。

```
# svcadm restart svc:/network/ipsec/ike:default
```

- サービスのプロパティーの値を変更したあと、プロパティーの値を表示し、サービスをリフレッシュしてから再起動します。

```
# svccfg -s ike setprop config/admin_privilege = astring: "modkeys"
# svccfg -s ike listprop config/admin_privilege
config/admin_privilege  astring  modkeys
# svcadm refresh svc:/network/ipsec/ike
# svcadm restart svc:/network/ipsec/ike
```

- **ike** サービスを停止するには、無効にします。

```
# svcadm disable svc:/network/ipsec/ike
```

3 鍵を手動で管理するには、次のいずれかを実行します。

- **/etc/inet/secret/ipseckeys** ファイルにエントリを追加したあと、**manual-key** サービスを有効にします。

```
# svcadm enable svc:/network/ipsec/manual-key:default
```

- **ipseckeys** ファイルを変更したあと、サービスをリフレッシュします。

```
# svcadm refresh manual-key
```

- サービスのプロパティーの値を変更したあと、プロパティーの値を表示し、サービスをリフレッシュしてから再起動します。

```
# svccfg -s manual-key setprop config/config_file=/etc/inet/secret/MyIpseckeyfile
# svccfg -s manual-key listprop config/config_file
config/config_file  astring  /etc/inet/secret/MyIpseckeyfile
# svcadm refresh svc:/network/ipsec/manual-key
# svcadm restart svc:/network/ipsec/manual-key
```

- 鍵を手動で管理できないようにするには、**manual-key** サービスを無効にします。

```
# svcadm disable svc:/network/ipsec/manual-key
```

4 **IPsec** のプロトコルとアルゴリズムのテーブルを変更した場合は、**ipsecalgs** サービスをリフレッシュします。

```
# svcadm refresh svc:/network/ipsec/ipsecalgs
```

注意事項 サービスのステータスを調べるには、`svcs` *service* コマンドを使用します。サービスが `maintenance` (保守) モードになっている場合は、`svcs -x` *service* コマンドの出力に表示されるデバッグのヒントに従ってください。

▼ 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。`AH:` 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。`ESP:` 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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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:       Type root password
#
```

2. **`partym` システムから、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 -v enigma
Using device /dev/bge (promiscuous mode)
```

3. **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。

```
% ssh enigma
Password:       Type your password
% su -
Password:       Type root password
# ping partym
```

4 **snoop** の出力を調べます。

partym システムで、冒頭の IP ヘッダー情報のあとに AH と ESP 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

IP セキュリティーアーキテクチャー (リファレンス)

この章では、次の内容について説明します。

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 (タスク)」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 (概要)」を参照してください。

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。初期値は /etc/inet/ipsecinit.conf 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 – 手動での鍵管理を有効にします。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、171 ページの「IKE サービス」を参照してください。

SMF の詳細については、『Oracle Solaris 11.1 でのサービスと障害の管理』の第 1 章「サービスの管理 (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 データグラムすべてがポリシーに沿っているかどうかを検査します。転送されたデータグラムは、このコマンドで追加されたポリシー検査の対象外になります。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) を構成します。 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受け入れます。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受け入れます。

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、101 ページの「IPsec によるトラフィックの保護」を参照してください。

ipsecinit.conf 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 コマンドを使用して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 does 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。詳細については、connect(3SOCKET) と 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。

ネーミングシステムを保護してください。次の 2 つの条件に該当する場合、そのホスト名は信頼できません。

- ソースアドレスが、ネットワークを介して参照できるホストである。
- ネーミングシステムの信頼性に問題がある。

セキュリティーの弱点の多くは、実際のツールではなく、ツールの使用方法にあります。ipsecconf コマンドを使用するときは注意が必要です。もっとも安全な操作モードのために、ssh を使用するか、コンソールなど物理的に接続された TTY を使用してください。

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ありません。ただし、この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。決して直接には編集しないでください。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。一般的な意味では、DOI は、データ形式、ネットワークトラフィック交換タイプ、および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。

具体的には、ISAKMP DOI は、有効な IPsec アルゴリズムとそのプロトコル (PROTO_IPSEC_AH と PROTO_IPSEC_ESP) の命名規則と番号付け規則を定義します。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は、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の第 11 章「暗号化フレームワーク (概要)」を参照してください。

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。SADB の保守は、1 つまたは複数の (そしておそらくは協力する) ユーザープロセスがメッセージを特殊なソケット経由で送信することによって行われます。SADB を保守するこの方法は、route(7P) のマニュアルページで説明している方法に類似しています。root 役割だけがデータベースにアクセスできます。

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 の SA を生成するためのユーティリティー

IKE プロトコルは、IPv4 アドレスおよび IPv6 アドレスの鍵を自動的に管理します。IKEを設定する方法については、第 10 章「IKE の構成 (タスク)」を参照してください。手動キーイングユーティリティーは ipseckey コマンドです (ipseckey(1M) のマニュアルページを参照)。

セキュリティーアソシエーションデータベース (SADB) を手動で生成するには、ipseckey コマンドを使用します。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。

ipseckey コマンドを使用すると、鍵が手動で追加された場合でも、IKE によって追加された場合でも、システムで認識されているすべての SA を表示できます。-c オプションを指定して ipseckey コマンドを実行すると、引数として指定した鍵ファイルの構文がチェックされます。

ipseckey コマンドで追加された IPsec SA には持続性がなく、システムのリブート時に失われます。手動で追加した SA をシステムのブート時に有効にするには、/etc/inet/secret/ipseckeys ファイルにエントリを追加してから、svc:/network/ipsec/manual-key:default サービスを有効にします。手順については、115 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。

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。

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 – 可能であれば、ipseckey による手動のキーイングではなく、IKE を使用してください。

鍵情報を扱う場合および ipseckey コマンドを使用する場合には、次のことに注意してください。

- 鍵情報をリフレッシュしているかどうか。定期的に鍵をリフレッシュすることが、セキュリティーの基本作業となります。鍵をリフレッシュすることで、アルゴリズムと鍵の脆弱性が暴かれないように保護し、公開された鍵の侵害を制限します。
- TTY がネットワークに接続されているか。ipseckey コマンドは対話モードで実行されているか。
  - 対話モードの場合、鍵情報のセキュリティーは、TTY のトラフィックに対応するネットワークパスのセキュリティーになります。平文の telnet や rlogin セッションでは、ipseckey コマンドを使用しないでください。
  - ローカルウィンドウでも、ウィンドウを読み取ることのできる隠密プログラムからの攻撃には無防備です。
- -f オプションを使用しているか。ファイルはネットワークを介してアクセスされているか。ファイルは外部から読み取り可能か。
  - ネットワークマウントファイルが読み取られている場合、不正に読み取られる可能性があります。外部から読み取れるファイルに鍵情報を保存して使用しないでください。
  - ネーミングシステムを保護してください。次の 2 つの条件に該当する場合、そのホスト名は信頼できません。
    - ソースアドレスが、ネットワークを介して参照できるホストである。
    - ネーミングシステムの信頼性に問題がある。

セキュリティーの弱点の多くは、実際のツールではなく、ツールの使用方法にあります。ipseckey コマンドを使用するときには注意が必要です。もっとも安全な操作モードのために、ssh を使用するか、コンソールなど物理的に接続された TTY を使用してください。

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。しかし、AH はデータを暗号化しません。したがって、AH で保護されたトラフィックは snoop コマンドで読み取ることができます。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、121 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使用可能です。

◆ ◆ ◆ 第 9 章

9 インターネット鍵交換 (概要)

インターネット鍵交換 (IKE) は、IPsec の鍵管理を自動化します。Oracle Solaris は IKEv1 を実装します。IKE について説明するこの章の内容は次のとおりです。

IKE を実装する手順については、第 10 章「IKE の構成 (タスク)」を参照してください。参照情報については、第 11 章「インターネット鍵交換 (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 (概要)」を参照してください。

IKE による鍵管理

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。自動鍵管理では、鍵の作成、認証、および交換にセキュアな通信チャネルを要求します。Oracle Solaris ではインターネット鍵交換バージョン 1 (IKE) を使用して鍵管理を自動化します。IKE を使用すれば、セキュアなチャネルを大量のトラフィックに割り当てるために容易にスケーリングできます。IPv4 および IPv6 パケットの IPsec SA では、IKE の利点を生かすことができます。

IKE では、ハードウェアアクセラレーションとハードウェアストレージを利用できます。ハードウェアアクセラレータによって、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。

IKE の鍵ネゴシエーション

IKE デーモン in.iked は、安全な方法で IPsec SA のキーイング素材をネゴシエートし、認証します。デーモンは OS によって提供される内部機能から鍵用のランダムシードを使用します。IKE は、PFS (Perfect Forward Secrecy) をサポートしています。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。

IKE の鍵用語について

次の表は、鍵ネゴシエーションで使用される用語と、一般的に使われるその略語、各用語の定義と使用についてまとめたものです。

表 9–1 鍵ネゴシエーションの用語、略語、使用

鍵ネゴシエーションの用語	略語	定義と使用
鍵交換		非対称暗号化アルゴリズムの鍵を生成する処理。主な 2 つの方法は、RSA と Diffie-Hellman プロトコルです。
Diffie-Hellman アルゴリズム	DH	鍵生成と鍵認証を提供する鍵交換アルゴリズム。しばしば「認証された鍵交換」と呼ばれます。
RSA アルゴリズム	RSA	鍵生成と鍵転送を提供する鍵交換アルゴリズム。このプロトコル名は、作成者の Rivest、Shamir、Adleman の三氏に因んでいます。
Perfect Forward Secrecy	PFS	認証された鍵交換だけに適用されます。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。
Oakley グループ		フェーズ 2 の鍵を安全な方法で確立する 1 つの手法。Oakley グループは PFS のネゴシエーションに使用されます。The Internet Key Exchange (IKE) (http://www.faqs.org/rfcs/rfc2409.html) のセクション 6 を参照してください。

IKE フェーズ 1 交換

フェーズ 1 交換は、「メインモード」といわれているものです。フェーズ 1 交換では、IKE は公開鍵暗号方式を使用して、ピア IKE エンティティーと IKE 自体を認証します。その結果がインターネットセキュリティーアソシエーションと鍵管理プロトコル (ISAKMP) セキュリティーアソシエーション (SA) で、IKE で IP データグラムの

鍵情報のネゴシエーションを行うためのセキュアなチャネルとなります。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。

IKE がフェーズ 1 交換で鍵情報をネゴシエートする方法は構成可能です。IKE では、`/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)
- PFS (Perfect Forward Secrecy) を使用する場合
- 影響を受けるインタフェース
- セキュリティープロトコルとそのアルゴリズム
- 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付きであっても、公開鍵インフラ (認証局 (CA)) 組織の PKI が発行したものであってもかまいません。

IKE フェーズ 2 交換

フェーズ 2 交換は「クイックモード」といいます。フェーズ 2 交換では、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。また、フェーズ 1 交換で作成したセキュアなチャネルを使用して、鍵情報の伝送を保護します。IKE 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。また、IKE デーモンは、鍵を一定の割合 (構成可能) でリフレッシュします。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。

IKE 構成の選択

`/etc/inet/ike/config` 構成ファイルには、IKE ポリシーのエントリが含まれています。2 つの IKE デーモンを相互に認証するためには、これらのエントリが有効でなければなりません。さらに、鍵情報も必要です。構成ファイルのエントリは、フェーズ 1 交換を認証するための鍵情報の使用方法を決定します。選択肢は、事前共有鍵か公開鍵証明書のどちらかです。

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。公開鍵証明書は自己署名付きにするか、PKI 組織から発行できます。詳細は、`ike.config(4)` のマニュアルページを参照してください。

IKE と事前共有鍵認証

事前共有鍵は、複数のピアシステムを認証するために使用されます。事前共有鍵は、1 つのシステム上の管理者によって作成される 16 進数または ASCII 文字列です。この鍵はその後、ピアシステムの管理者によってセキュアな方法で対域外で共有されます。事前共有鍵が傍受者によって傍受されると、その傍受者はピアシステムの 1 つを偽装できる可能性があります。

この認証方法を使用するピア上の事前共有鍵は、同一である必要があります。これらの鍵は、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、`ike.preshared(4)` のマニュアルページを参照してください。

IKE と公開鍵証明書

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。公開鍵では、鍵の認証とネゴシエーションに Diffie-Hellman アルゴリズム (DH) を使用します。公開鍵証明書には、2 つの方法があります。公開鍵証明書は、自己署名付きにすることも、認証局 (CA) が認証することもできます。

自己署名付き公開鍵証明書は、自ら (管理者) が作成します。`ikecert certlocal -ks` コマンドを実行して、システムの公開鍵と非公開鍵のペアの非公開部分を作成します。そのあと、管理者は、リモートシステムから X.509 形式で自己署名付き証明書の出力を取得します。リモートシステムの証明書は、鍵のペアの公開部分の `ikecert certdb` コマンドに入力されます。自己署名付き証明書は、通信するシステムの `/etc/inet/ike/publickeys` ディレクトリに保存されます。証明書をシステムに接続されているハードウェアに保存したい場合は、`-T` オプションを指定します。

自己署名付き証明書は、事前共有鍵と CA 間の中間ポイントになります。事前共有鍵とは異なり、自己署名付き証明書は移動体システムまたは再番号付け可能なシステムで使用できます。固定番号を使用しないで、システムの証明書に自己署名するには、`DNS` (`www.example.org`) または `email` (`root@domain.org`) の代替名を使用します。

公開鍵は、PKI または CA 組織で配信できます。公開鍵とそれに関連する CA は、`/etc/inet/ike/publickeys` ディレクトリに格納されます。証明書をシステムに接続されているハードウェアに保存したい場合は、`-T` オプションを指定します。また、ベンダーは証明書失効リスト (CRL) も発行します。管理者は鍵と CA を格納するだけでなく、CRL を `/etc/inet/ike/crls` ディレクトリに格納する責任があります。

CA には、サイトの管理者ではなく、外部の機関によって認証されるといった特長があります。その点では、CA は公証された証明書となります。自己署名付き証明書と同様に、CA は移動体システムまたは再番号付け可能なシステムで使用できます。その一方、自己署名付き証明書とは異なり、CA は通信する多くのシステムを保護するために容易にスケーリングできます。

IKE ユーティリティーおよび IKE ファイル

次の表は、IKE ポリシーの構成ファイルや、IKE キーの格納場所、IKE を実装する各種の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 11.1 でのサービスと障害の管理』の第 1 章「サービスの管理 (概要)」を参照してください。

表 9–2 IKE 構成ファイル、鍵の格納場所、コマンド、サービス

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	IKE 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 (IKE) 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	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 IKE ポリシー構成ファイル。インバウンド IKE 要求のマッチングとアウトバウンド IKE 要求の準備に関するサイトの規則が含まれています。 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 コマンドで変更することができます。	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情報が含まれます。事前共有鍵を使って IKE 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)
publickeys ディレクトリ	/etc/inet/ike ディレクトリ内のディレクトリ。公開鍵と証明書ファイルが格納されています。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	/etc/inet/ike ディレクトリ内のディレクトリ。公開鍵や証明書ファイルの失効リストが格納されています。	ikecert(1M)

表 9–2 IKE 構成ファイル、鍵の格納場所、コマンド、サービス (続き)

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
Sun Crypto Accelerator 6000 ボード	オペレーティングシステムの処理を少なくすることで公開鍵操作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)

◆ ◆ ◆ 第 10 章

IKE の構成 (タスク)

この章では、使用するシステムにあわせて Internet Key Exchange (IKE) を構成する方法について説明します。IKE の構成が完了すると、そのネットワークにおける 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。

IKE の概要については、第 9 章「インターネット鍵交換 (概要)」を参照してください。IKE の参照情報については、第 11 章「インターネット鍵交換 (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。

IKE 情報の表示

フェーズ 1 IKE ネゴシエーションで使用できるアルゴリズムおよびグループを表示できます。

▼ フェーズ 1 IKE 交換に使用できるグループおよびアルゴリズムの表示方法

この手順では、フェーズ 1 IKE 交換で使用できる Diffie-Hellman グループを判別します。また、IKE フェーズ 1 交換で使用可能な暗号化および認証アルゴリズムを表示します。数値は、IANA (Internet Assigned Numbers Authority) によってこれらのアルゴリズムに指定された値に一致します。

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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

1 **IKE がフェーズ 1 で使用できる Diffie-Hellman グループの一覧を表示します。**

Diffie-Hellman グループは IKE SA を設定します。

```
# ikeadm dump groups
Value Strength Description
1     66       ietf-ike-grp-modp-768
2     77       ietf-ike-grp-modp-1024
5     91       ietf-ike-grp-modp-1536
14    110      ietf-ike-grp-modp-2048
15    130      ietf-ike-grp-modp-3072
16    150      ietf-ike-grp-modp-4096
17    170      ietf-ike-grp-modp-6144
18    190      ietf-ike-grp-modp-8192

Completed dump of groups
```

これらの値の 1 つを、次に示すように IKE フェーズ 1 変換の `oakley_group` パラメータの引数として使用します。

```
p1_xform
  { auth_method preshared oakley_group 15 auth_alg sha encr_alg aes }
```

2 **IKE がフェーズ 1 で使用できる認証アルゴリズムの一覧を表示します。**

```
# ikeadm dump authalgs
Value Name
1     md5
2     sha1
4     sha256
5     sha384
6     sha512

Completed dump of authalgs
```

これらの名前の 1 つを、次に示すように IKE フェーズ 1 変換の `auth_alg` パラメータの引数として使用します。

```
p1_xform
  { auth_method preshared oakley_group 15 auth_alg sha256 encr_alg 3des }
```

3 **IKE がフェーズ 1 で使用できる暗号アルゴリズムの一覧を表示します。**

```
# ikeadm dump encralgs
Value Name
3     blowfish-cbc
5     3des-cbc
1     des-cbc
7     aes-cbc

Completed dump of encralgs
```

これらの名前の 1 つを、次に示すように IKE フェーズ 1 変換の encr_alg パラメータの引数として使用します。

```
p1_xform
  { auth_method preshared oakley_group 15 auth_alg sha256 encr_alg aes }
```

参照　これらの値を必要とする IKE 規則の構成のタスクについては、139 ページの「IKE の構成 (タスクマップ)」を参照してください。

IKE の構成 (タスクマップ)

IKE を認証するには、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。規則として、保護しようとしているエンドポイントには、特定の IKE 認証方法を関連付けます。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、IKE 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。

IKE を構成したあと、IKE 構成を使用する IPsec タスクを実行します。次の表に、特定の IKE 構成に注目したタスクマップを示します。

タスク	説明	手順
事前共有鍵で IKE を構成します。	2 つのシステムに秘密鍵を共有させることにより、その通信を保護します。	139 ページの「事前共有鍵による IKE の構成 (タスクマップ)」
公開鍵証明書で IKE を構成します。	公開鍵証明書を使って通信を保護します。証明書は、自己署名付き、または PKI 機関の保証付きです。	144 ページの「公開鍵証明書による IKE の構成 (タスクマップ)」
NAT 境界を越えます。	IPsec と IKE を構成して、移動体システムと通信します	162 ページの「移動体システム用の IKE の構成 (タスクマップ)」
ハードウェアキーストアを使用して証明書ペアを生成するよう IKE を構成します。	Sun Crypto Accelerator 6000 ボードを使用可能にして、IKE 操作を高速化し、公開鍵証明書を格納します。	169 ページの「接続したハードウェアを検出するように IKE を構成する」

事前共有鍵による IKE の構成 (タスクマップ)

次の表に、事前共有鍵で IKE を構成および保守する手順を示します。

タスク	説明	手順
事前共有鍵で IKE を構成します。	IKE 構成ファイルと共有する 1 つの鍵を作成します。	140 ページの「事前共有鍵により IKE を構成する方法」

タスク	説明	手順
実行中の IKE システムへ事前共有鍵を追加します。	現在 IKE ポリシーを実施しているシステムに、新しい IKE ポリシーエントリと新しい鍵情報を追加します。	143 ページの「新規ピアシステムのために IKE を更新する方法」

事前共有鍵による IKE の構成

事前共有鍵は、IKE 用のもっとも簡単な認証方法です。IKE を使用するようにピアシステムを構成し、さらに、ユーザーが両方のシステムの管理者である場合、事前共有鍵を使用することをお勧めします。ただし、公開鍵認証とは異なり、事前共有鍵は IP アドレスに関連付けられます。事前共有鍵を特定の IP アドレスまたは IP アドレス範囲に関連付けることができます。事前共有鍵は、移動体システムや、番号が変更されることがあるシステムでは使用できませんが、番号の変更が、指定された IP アドレスの範囲内である場合を除きます。

▼ 事前共有鍵により IKE 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 – 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 `solaris.admin.edit/etc/inet/ike/config` 承認に加えて、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。`root` 役割には、これらの権利がすべて含まれています。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

リモートからログインする場合、セキュアなリモートログイン用の `ssh` コマンドを使用してください。例については、例 7–1 を参照してください。

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 IKE 構成例は、How to Secure Traffic Between Two Systems With IPsecの 103 ページの「IPsec で 2 つのシステム間のトラフィックを保護するには」 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 システムごとに **/etc/inet/secret/ike.preshared** ファイルを作成します。
各ファイルに事前共有鍵を書き込みます。

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 **IKE** サービスを有効にします。

```
# svcadm enable ipsec/ike
```

例 10–1 IKE 事前共有鍵をリフレッシュする

IKE 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、管理者は 192.168.13.0/24 サブネット上のすべてのホストについて有効な、事前共有鍵エントリを追加します。

```
#...
{ localidtype IP
    localid 192.168.116.0/24
    remoteidtype IP
    remoteid 192.168.13.0/24
    # enigma and partym'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 IKE サービスを再起動します。

```
# 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。

▼ 新規ピアシステムのために IKE 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。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 IKE 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。

`solaris.admin.edit/etc/inet/ike/config` 承認に加えて、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。`root` 役割には、これらの権利がすべて含まれています。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

リモートからログインする場合、セキュアなリモートログイン用の `ssh` コマンドを使用してください。例については、例 7–1 を参照してください。

1. **IPsec を使用する新規システム用の鍵を管理するための IKE の規則を作成します。**

   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 **IKE** 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/secret/ike.preshared** ファイルに追加します。

```
# 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
```

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。

公開鍵証明書による IKE の構成 (タスクマップ)

次の表に、IKE の公開鍵証明書を作成する手順を示します。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。

公開証明書は一意である必要があるため、公開鍵証明書の作成者は証明書について一意となる任意の名前を生成します。通常は、X.509 識別名が使用されます。識別用の代替名も使用できます。これらの名前の形式は、*tag=value* です。値は任意ですが、値の形式は、そのタグの種類に対応する必要があります。たとえば、`email` タグの形式は *name@ domain.suffix* です。

タスク	説明	手順
自己署名付き公開鍵証明書で IKE を構成します。	システムごとに 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 ■ ピアシステムからの公開鍵証明書	146 ページの「自己署名付き公開鍵証明書により IKE を構成する方法」
PKI 認証局で IKE を構成します。	1 つの証明書要求を作成して、そのあと、システムごとに次の 3 つの証明書を格納します。 ■ 証明書要求に応じて認証局 (CA) が作成した証明書 ■ CA からの公開鍵証明書 ■ CA からの CRL	151 ページの「CA からの署名付き証明書により IKE を構成する方法」
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する。	156 ページの「ハードウェアで公開鍵証明書を生成および格納する方法」
PKI 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。	159 ページの「証明書失効リストを処理する方法」

注 – Trusted Extensions 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 (タスクマップ)」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。

公開鍵証明書による IKE の構成

公開鍵証明書を使用すると、通信するシステムは秘密鍵情報を帯域外で共有する必要がなくなります。事前共有鍵とは異なり、公開鍵証明書は、移動体システムなど、番号が変更される可能性があるシステムでも使用できます。

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、169 ページの「接続したハードウェアを検出するように IKE を構成する」を参照してください。

▼ 自己署名付き公開鍵証明書により IKE を構成する方法

この手順では、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、`certlocal` サブコマンドを使用して参照できます。証明書ペアの公開部分は、公開証明書データベースに格納されます。これは `certdb` サブコマンドを使用して参照できます。公開部分をピアシステムと交換します。2 つの証明書を組み合わせたものが、IKE 転送を認証するために使用されます。

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 によって発行される証明書とは異なり、自己署名付き証明書は帯域外で検証する必要があります。

始める前に `solaris.admin.edit/etc/inet/ike/config` 承認に加えて、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。`root` 役割には、これらの権利がすべて含まれています。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

リモートからログインする場合、セキュアなリモートログイン用の `ssh` コマンドを使用してください。例については、例 7–1 を参照してください。

1 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 -ks -m keysize -t keytype \
-D dname -A altname \
[-S validity-start-time] [-F validity-end-time] [-T token-ID]
```

`-ks`	自己署名付き証明書を作成します。
`-m` *keysize*	鍵のサイズです。*keysize* は、512、1024、2048、3072、4096 のいずれかです。
`-t` *keytype*	使用するアルゴリズムのタイプを指定します。*keytype* は `rsa-sha1`、`rsa-md5`、`dsa-sha1` のいずれかです。
`-D` *dname*	証明書主体の X.509 識別名です。*dname* の一般的な形式は次のとおりです: C=*country* (国)、O=*organization* (組織)、OU=*organizational unit* (組織単位)、CN=*common name* (共通名)。有効なタグは、C、O、OU、CN です。
`-A` *altname*	証明書の代替名です。*altname* の形式は `tag=value` です。有効なタグは `IP`、`DNS`、`email`、および `DN` です。
`-S` *validity-start-time*	証明書の有効期間の開始時間を絶対値または相対値で指定します。
`-F` *validity-end-time*	証明書の有効期間の終了時間を絶対値または相対値で指定します。

-T *token-ID* PKCS #11 ハードウェアトークンで鍵を生成できるようにします。その後、証明書はハードウェアに格納されます。

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。

```
# ikecert certlocal -ks -m 2048 -t rsa-sha1 \
-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-----
```

注 – -D および -A オプションの値は任意です。値は証明書を識別するためだけに使用されます。これらは 192.168.13.213 などのシステムを識別するためには使用されません。実際、これらは固有の値であるため、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-sha1 \
-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。この証明書を電子メールにペーストしても安全です。手順 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@ja.enigmaexample.com
From: admin@us.partyexample.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@us.partyexample.com
From: admin@ja.enigmaexample.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** で読み取り可能な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
```

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 通信するシステムの管理者と一緒に、証明書がその管理者のものであることを確認します。

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-sha1
        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-enigmax"
 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-enigmax 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
...
```

6 ピアシステムで、**IKE** を有効にします。

```
partym # svcadm enable ipsec/ike

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。

▼ CA からの署名付き証明書により IKE を構成する方法

認証局 (CA) からの公開鍵証明書では、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。

始める前に `solaris.admin.edit/etc/inet/ike/config` 承認に加えて、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。`root` 役割には、これらの権利がすべて含まれています。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

リモートからログインする場合、セキュアなリモートログイン用の `ssh` コマンドを使用してください。例については、例 7–1 を参照してください。

1 **`ikecert certlocal -kc`** コマンドを使用して、証明書要求を作成します。

コマンドの引数については、手順 1 in 146 ページの「自己署名付き公開鍵証明書により IKE を構成する方法」 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、次のコマンドでは、**`partym`** システム上に証明書要求が作成されます。

```
# ikecert certlocal -kc -m 2048 -t rsa-sha1 \
> -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. 次のコマンドでは、**`enigma`** システム上に証明書要求が作成されます。

```
# ikecert certlocal -kc -m 2048 -t rsa-sha1 \
> -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 この証明書要求を **PKI** 機関に送信します。

証明書要求の送信方法については PKI に問い合わせてください。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、証明書要求をフォームに貼り付けます。要求を受け取った機関は、それをチェックしてから、次の 2 つの証明書オブジェクトと、証明書失効リストを発行します。

- 公開鍵証明書 – この証明書は機関に送信した要求に基づいて作成されます。送信した証明書要求も、公開鍵証明書の一部として含まれます。この証明書によって一意に識別されます。
- 認証局 – 機関の署名。CA によって公開鍵証明書が正規のものであることが確認されます。
- 証明書失効リスト (CRL) – 機関が無効にした証明書の最新リストです。CRL へのアクセスが公開鍵証明書に組み込まれている場合には、CRL が別個の証明書オブジェクトとして送信されることはありません。

  CRL の URI が公開鍵証明書に組み込まれている場合には、IKE は CRL を自動的に取り出すことができます。同様に、DN (LDAP サーバー上のディレクトリ名) エントリが公開鍵証明書に組み込まれている場合には、IKE は、指定された LDAP サーバーから CRL を取得し、キャッシュできます。

  公開鍵証明書に組み込まれている URI と DN エントリの例については、159 ページの「証明書失効リストを処理する方法」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、134 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。

詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。リモートからログインする場合、セキュアなリモートログイン用の `ssh` コマンドを使用してください。例については、例 7–1 を参照してください。

b. **PKI** 機関から受け取った公開鍵証明書を追加します。

```
# ikecert certdb -a < /tmp/PKIcert.eml
```

c. **PKI** 機関の **CA** を追加します。

```
# ikecert certdb -a < /tmp/PKIca.eml
```

d. **PKI 機関が証明書失効リスト (CRL) を送信してきている場合は、これを `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
<Control>-D
```

4 **`cert_root` キーワードを使用して、`/etc/inet/ike/config` ファイルの PKI 機関を識別します。**

PKI 機関が提供する名前を使用します。

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 PKI
# This is the X.509 distinguished name for the CA that it issues.

cert_root "C=US, O=ExamplePKI\, Inc., OU=PKI-Example, CN=Example PKI"

## 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-enigmax - Example PKI"
 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 – `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、次の条件を満たしている必要があります。

- `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=ExamplePKI\, Inc., OU=PKI-Example, CN=Example PKI"
...
{
 label "JA-enigmax to US-partym - Example PKI"
 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 CRL を処理する方法を IKE に伝えます。

適切なオプションを選択します。

- **No CRL available**

  PKI 機関が CRL を提供しない場合、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=ExamplePKI\, Inc., OU=PKI-Example,...
  ignore_crls
  ...
  ```

  ignore_crls キーワードにより、IKE は CRL を検索しなくなります。

- **CRL available**

  PKI 機関から CRL の一元的な配布ポイントを知らされている場合は、ike/config ファイルを変更してこの場所を指定できます。

  例については、159 ページの「証明書失効リストを処理する方法」を参照してください。

例 10–2 IKE の構成時における rsa_encrypt の使用

ike/config ファイルで auth_method rsa_encrypt を使用する場合には、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。

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、電子メールに貼り付けることもできます。

   たとえば、partym の管理者は次のような電子メールを送信します。

```
To: admin@ja.enigmaexample.com
From: admin@us.partyexample.com
Message: -----BEGIN X509 CERTIFICATE-----
MII...
----END X509 CERTIFICATE-----
```

enigma の管理者は次のような電子メールを送信します。

```
To: admin@us.partyexample.com
From: admin@ja.enigmaexample.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、IKE 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、IKE ピアが互いの公開鍵を知っておく必要があります。

よって、/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 証明書
- ピアの公開鍵証明書

トラブルシューティング – IKE ペイロードは 3 つの証明書を持っており、大きくなりすぎて、rsa_encrypt が暗号化できないことがあります。「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。

▼ ハードウェアで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) に準拠して実装されているライブラリ、すなわち PKCS #11 ライブラリであること。

  設定の手順については、169 ページの「Sun Crypto Accelerator 6000 ボードを検出するように IKE を構成する方法」を参照してください。

`solaris.admin.edit/etc/inet/ike/config` 承認に加えて、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。`root` 役割には、これらの権利がすべて含まれています。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

リモートからログインする場合、セキュアなリモートログイン用の `ssh` コマンドを使用してください。例については、例 7–1 を参照してください。

1 自己署名付き証明書または証明書要求を作成して、トークン **ID** を指定します。

次のオプションのいずれかを選択します。

注 – 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。

- 自己署名付き証明書の場合、次の構文を使用する

```
# ikecert certlocal -ks -m 2048 -t rsa-sha1 \
> -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。

- 証明書要求の場合、次の構文を使用します。

```
# ikecert certlocal -kc -m 2048 -t rsa-sha1 \
> -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、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 – PIN の応答は、ディスク上に「クリアテキストとして」格納されます。

パスワードの入力後、証明書が印刷されます。

```
Enter PIN for PKCS#11 token: ikemgr:rgm4tigt
-----BEGIN X509 CERTIFICATE-----
MIIBuDCCASECAQAwSTELMAkGA1UEBhMCVVMxFTATBgNVBAoTDFBhcnR5Q29tcGFu
...
oKUDBbZ9O/pLWYGr
-----END X509 CERTIFICATE-----
```

3 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、電子メールに貼り付けることもできます。

- **PKI** を処理する機関に証明書要求を送信します。

  証明書要求は、PKI 機関の指示に従って送信します。詳細については、手順 2 of 151 ページの「CA からの署名付き証明書により IKE を構成する方法」を参照してください。

4 システム上で、**/etc/inet/ike/config** ファイルを編集して、証明書が認識されるようにします。

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

  リモートシステムの管理者がパラメータ cert_trust、remote_id、および remote_addr 用に提供する値を使用します。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-enigmax 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}
}
```

- 証明書リクエスト

  PKI 機関が 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 PKI
# This is the X.509 distinguished name for the CA that it issues.

cert_root "C=US, O=ExamplePKI\, Inc., OU=PKI-Example, CN=Example PKI"

...
{
 label "JA-enigmax to US-partym - Example PKI"
 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。

手順 2 で応答したように、PIN 要求に応答します。

注 – 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。

- **PKI 機関からの証明書**

  機関が証明書要求から生成した証明書を追加して、認証局 (CA) を追加します。

  ```
  # 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
  ```

  PKI 機関からの証明書失効リスト (CRL) を追加する方法については、159 ページの「証明書失効リストを処理する方法」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。

▼ 証明書失効リストを処理する方法

証明書失効リスト (CRL) には、認証局が発行した証明書のうち、期限切れになったりセキュリティーが低下したりした証明書が含まれます。CRL を処理する方法には、次の 4 つがあります。

- CA 機関が CRL を発行しない場合、CRL を無視するように IKE に指示する必要があります。このオプションについては、手順 5 in 151 ページの「CA からの署名付き証明書により IKE を構成する方法」を参照してください。
- CA から受け取った公開鍵証明書に URI (Uniform Resource Indicator) のアドレスが組み込まれている場合は、URI から CRL にアクセスするように IKE に指示することができます。

- CA から受け取った公開鍵証明書に 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は、LDAP サーバーから CRL にアクセスするように IKE に指示することができます。
- CRL は `ikecert certrldb` コマンドへの引数として指定できます。例については、例 10–3 を参照してください。

次の手順に、中央の配布ポイントから CRL を使用するように IKE に指示する手順を示します。

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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

1　CA から受信した証明書を表示する

```
# ikecert certdb -lv certspec
```

`-l`　IKE 証明書データベースにある証明書を一覧表示します。

`-v`　証明書を冗長モードで一覧表示します。このオプションは慎重に使用してください。

certspec　IKE 証明書データベース内の証明書と一致するパターンです。

たとえば、次の証明書は Oracle が発行しました。詳細は変更されています。

```
# ikecert certdb -lv example-protect.oracle.com
Certificate Slot Name: 0   Type: dsa-sha1
   (Private key in certlocal slot 0)
 Subject Name: <O=Oracle, CN=example-protect.oracle.com>
 Issuer Name: <CN=Oracle CA (Cl B), O=Oracle>
 SerialNumber: 14000D93
   Validity:
      Not Valid Before: 2011 Sep 19th, 21:11:11 GMT
      Not Valid After:  2015 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 = example-protect.oracle.com
      Key Usage: DigitalSignature KeyEncipherment
      [CRITICAL]
   CRL Distribution Points:
      Full Name:
         URI = #Ihttp://www.oracle.com/pki/pkismica.crl#i
         DN = <CN=Oracle CA (Cl B), O=Oracle>
      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 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 own web proxy
  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 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.oracle.com:389,ldap2.oracle.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 をローカルの certrldb データベースに貼り付ける

使用する機関の証明書に一元的な配布ポイントが含まれていない場合は、機関の CRL を手動でローカルの certrldb データベースに追加できます。機関の説明に従って CRL をファイルに抽出し、それを ikecert certrldb -a コマンドでデータベースに追加します。

```
# ikecert certrldb -a < Oracle.Cert.CRL
```

移動体システム用の IKE の構成 (タスクマップ)

次の表に、中央サイトにリモートからログインするシステムを処理するように、IKE を構成する手順を示します。

タスク	説明	手順
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。	162 ページの「遠隔地のシステム用に IKE 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 IKE 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
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 IKE 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
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 IKE 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
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 IKE 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

移動体システム用の IKE の構成

適切に構成することで、ホームオフィスやノートブックから IPsec と IKE を使用して、会社の中央コンピュータと通信できます。公開鍵認証方法と結びついたブランケット IPsec ポリシーを使用すると、遠隔地のシステムは中央システムとのトラフィックを保護できます。

▼ 遠隔地のシステム用に IKE を構成する方法

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 PKI 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。

始める前に　root 役割になる必要があります。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。リモートからログインする場合、セキュアなリモートログイン用の ssh コマンドを使用してください。例については、例 7–1 を参照してください。

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. IKE 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、提供することは可能です。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. IKE 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*

証明書を使用して、移動体システムを認証します。

```
## /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** サービスを使用可能にします。

```
# svcadm enable svc:/network/ipsec/ike
```

例 10–4 移動体システムからの IPsec トラフィックを受信するための中央コンピュータの構成

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 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=ExamplePKI Inc, OU=PKI-Example, CN=Example PKI"
#
# 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 越しのシステムの IPsec による構成

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=ExamplePKI Inc, OU=PKI-Example, CN=Example PKI"
#
# 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。オフサイトシステムを設定する方法については、例 10–7 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 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。

接続したハードウェアを検出するように IKE を構成する

公開鍵証明書は接続されたハードウェア上にも格納できます。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 IKE 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 Board Version 1.1 User's Guide (http://download.oracle.com/docs/cd/E19321-01/820-4144-12/820-4144-12.pdf)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。詳細については、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

リモートからログインする場合、セキュアなリモートログイン用の ssh コマンドを使用してください。例については、例 7–1 を参照してください。

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。**

   IKE 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。PKCS #11 ライブラリがリンクされていることを確認するには、次のコマンドを実行します。

   ```
   $ 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、IKE 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、156 ページの「ハードウェアで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。

◆ ◆ ◆ 第 11 章

11 インターネット鍵交換 (リファレンス)

この章では、IKE に関する次のリファレンス情報について説明します。

IKE の実装方法については、第 10 章「IKE の構成 (タスク)」を参照してください。概要については、第 9 章「インターネット鍵交換 (概要)」を参照してください。

IKE サービス

svc:/network/ipsec/ike:default サービス –サービス管理機能 (SMF) では、IKE を管理するための ike サービスが提供されています。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、IKE 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 – IKE 構成ファイルの場所です。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です。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の「オブジェクトタイプ」に記載されてい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です。初期値は base です。ほかの値は modkeys と keymat です。詳細は、173 ページの「ikeadm コマンド」を参照してください。

SMF の詳細については、『Oracle Solaris 11.1 でのサービスと障害の管理』の第 1 章「サービスの管理 (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。

IKE デーモン

in.iked 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。そのデーモンは、セキュリティー保護された通信を行うすべてのシステムで実行する必要があります。

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、ike サービスを有効にしたら、システムブート時に in.iked デーモンが実行されます。

IKE デーモンを実行すると、システムは、フェーズ 1 交換でそのピア IKE エンティティーに対してそのシステム自体を認証します。そのピアは、認証方式として IKE ポリシーファイルに定義されています。そのあと、デーモンはフェーズ 2 のキーが設定します。ポリシーファイルで指定した時間間隔で、IKE キーが自動的にリフレッシュされます。in.iked デーモンを実行すると、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 IKE デーモンをサポートします。ikeadm コマンドを使用すると、IKE ポリシーの表示および一時的な変更を行うことができます。IKE ポリシーを永続的に変更するには、ike サービスのプロパティーを変更する必要があります。IKE サービスのプロパティーを変更するには、119 ページの「IPsec および IKE サービスを管理する方法」を参照してください。ikeadm コマンドは、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、および証明書キャッシュを表示するためにも使用できます。

ikecert コマンドを実行すると、公開鍵データベースを表示および変更できます。このコマンドでは、ローカルデータベース ike.privatekeys と publickeys を管理します。公開鍵の操作とハードウェア上の公開鍵のストレージも管理します。

IKE 構成ファイル

IKE 構成ファイル /etc/inet/ike/config は、IPsec ポリシーファイル /etc/inet/ipsecinit.conf の中で保護されているインタフェースの鍵を管理します。

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、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。IKE ポリシーファイルの例については、139 ページの「事前共有鍵による IKE の構成 (タスクマップ)」を参照してください。IKE 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。

IKE 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 データグラムを保護します。IPsec SA の作成時に perfect forward security (PFS) を使用するかどうかは、IKE ポリシーファイルで決まります。

/etc/inet/ike/config ファイルには、RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki) に準拠して実装されているライブラリへのパスを含めることができます。IKE は、この PKCS #11 ライブラリを使って、アクセラレータおよび鍵ストレージ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 ファイルのセキュリティーと同様です。詳細は、125 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。

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 作成に関する統計情報の表示
- IKE プロトコル交換のデバッグ
- IKE デーモンオブジェクトの表示。たとえば、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、および証明書キャッシュがあります。

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	鍵情報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。
modkeys レベル	事前共有鍵の削除、変更、追加ができます。
keymat レベル	ikeadm コマンドで実際の鍵情報を表示できます。

特権を一時的に変更する場合は、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。手順については、119 ページの「IPsec および IKE サービスを管理する方法」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、128 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。

IKE 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.preshared ファイルに Internet Security Association and Key Management Protocol (ISAKMP) SA の事前共有鍵が含まれ、ipseckeys ファイルに IPsec SA の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。

- ike.preshared ファイルは、事前共有鍵を必要とする ike/config ファイルの構成時に作成します。IKE 認証用として、ike.preshared ファイルに ISAKMP SA の鍵情報を入力します。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。
- ipseckeys ファイルには、IPsec SA の鍵情報が含まれています。手動によるファイルの管理の例については、115 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。IKE デーモンでは、このファイルを使用しません。IKE によって IPsec SA に対して生成される鍵情報は、カーネルに保存されます。

IKE 公開鍵のデータベースおよびコマンド

ikecert コマンドを実行すると、ローカルシステムの公開鍵データベースを操作できます。このコマンドは、ike/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。IKE ではそ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、certrldb をそれぞれ実行して、3 つのデータベースを処理します。

ikecert コマンドは鍵の格納処理も行います。鍵は、ディスク、接続された Sun Crypto Accelerator 6000 ボード、またはソフトトークンキーストアに格納できます。ソフトトークンキーストアは、ハードウェアデバイスと通信するために、暗号化フレームワークのメタスロットを使用しているときに使用できます。ikecert コマンドは、PKCS #11 ライブラリを使用して鍵の格納場所を見つけます。

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。

ikecert tokens コマンド

tokens 引数を使用すると、使用可能なトークン ID がリストされます。トークン ID により、ikecert certlocal コマンドと ikecert certdb コマンドは、公開鍵証明書と証明書要求を生成します。証明書と証明書要求も、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するか、接続された Sun Crypto Accelerator 6000 ボードに格納することができます。ikecert コマンドは、PKCS #11 ライブラリを使用して証明書の格納場所を見つけます。

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。また、自己署名付き証明書または証明書要求のいずれかを作成できます。-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 オプションを選択すると、証明書要求が作成されます。鍵はシステムの /etc/inet/secret/ike.privatekeys ディレクトリに格納されます。-T オプションを指定した場合は、システムに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 コマンドへのサブコマンド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 11–1　ikecert オプションと ike/config エントリの対応表

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。

表 11–1 ikecert オプションと ike/config エントリの対応表 (続き)

ikecert オプション	ike/config エントリ	説明
-t dsa-sha1	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	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。
-t rsa-md5 および -t rsa-sha1	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 コマンドを指定して証明書要求を実行する場合、そのコマンド出力を PKI 機関または認証局 (CA) に送信します。会社が独自の PKI を運営している場合は、出力を PKI 管理者に送信します。PKI 機関、CA、または PKI の管理者はこれに基づいて証明書を作成します。PKI または CA から返された証明書は、certdb サブコマンドに渡されます。PKI 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。

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。また、リモートシステムで ikecert certlocal -ks コマンドを実行して作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、146 ページの「自己署名付き公開鍵証明書により IKE を構成する方法」を参照してください。さらに、PKI または 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、151 ページの「CA からの署名付き証明書により IKE 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。

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。PKI によって CRL が提供されるときに、ikecert certrldb コマンドを指定して CRL データベースにその CRL を格納します。手順については、159 ページの「証明書失効リストを処理する方法」を参照してください。

/etc/inet/ike/publickeys ディレクトリ

/etc/inet/ike/publickeys ディレクトリの複数のファイルまたは「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 に送信した証明書署名要求に基づいた証明書を格納します。また、CA の証明書も格納します。

/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。

/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鍵証明書ファイルに対応しています。PKI 機関により、それらの証明書の CRL が提供されます。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。

用語集

3DES Triple-DES を参照してください。

AES Advanced Encryption Standard の略。対称 128 ビット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。

CA 認証局 (CA) を参照してください。

DES Data Encryption Standard。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。

diffserv モデル IP ネットワークで差別化サービスを実装するための IETF (Internet Engineering Task Force) のアーキテクチャー標準。主なモジュールとして、クラシファイア、メーター、マーカー、スケジューラ、およびドロッパがあります。IPQoS では、クラシファイア、メーター、およびマーカーの各モジュールを実装します。diffserv モデルについては、RFC 2475 (*An Architecture for Differentiated Services*) に解説されています。

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。

DS コードポイント **(DSCP)** IP ヘッダーの DS フィールドに含まれていて、パケットの転送方法を指示する 6 ビットの値。

header IP ヘッダーを参照してください。

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 インターネット制御メッセージプロトコル (Internet Control Message Protocol)。エラーの処理や制御メッセージの交換に使用されます。

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。

IPQoS diffserv モデル 標準に加えて、仮想 LAN に対するフローアカウンティングや 802.1 D マーカーの実装を行うソフトウェア機能。IPQoS を使用すると、IPQoS 構成ファイル内に定義したとおりに、さまざまなレベルのネットワークサービスを顧客やアプリケーションに提供できます。

IPsec IP セキュリティー。IP データグラム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。

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 データグラム IP 経由で転送される情報パケット。IP データグラムはヘッダーとデータを含みます。ヘッダーにはデータグラム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には、複数のデータグラムを宛先で識別し、再結合するための情報が含まれます。

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 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。

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。

MTU	最大転送単位。リンクに転送できるサイズ (オクテット単位)。たとえば、Ethernet の MTU は 1500 オクテットです。
NAT	ネットワークアドレス変換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。 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	Public Key Infrastructure。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。
SA	セキュリティーアソシエーション (SA)を参照してください。
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。
SCTP	「ストリーム制御転送プロトコル」を参照してください。
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。
SPD	セキュリティーポリシーデータベース (SPD) を参照してください。
SPI	セキュリティーパラメータインデックス (SPI) を参照してください。
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。
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。
エニーキャストアドレス	(一般的に別のノードに属する) インタフェースグループに割り当てられる IPv6 アドレス。エニーキャストアドレスに送られたパケットは、そのアドレスを持つ、プロトコルに基づき「もっとも近い」インタフェースに配送されます。パケットのルートは、ルーティングプロトコルの距離測定に応じて決定されます。

エニーキャストグループ	同じエニーキャスト IPv6 アドレスからなるインタフェースグループ。IPv6 の Oracle Solaris 実装は、エニーキャストアドレスやグループの作成をサポートしていません。ただし、Oracle Solaris IPv6 ノードはトラフィックをエニーキャストグループに送信できます。
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。
回復検出	障害の発生後、NIC や NIC から第 3 層の装置への経路が、正しく動作し始めたことを検出する処理。
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 **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。
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。
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)**	データグラムに対して認証と完全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。
逆方向トンネル	モバイルノードの気付アドレスで始まり、ホームエージェントで終わるトンネル。
近傍検索	接続されているリンク上にあるほかのホストをホストが特定できるようにするための IP メカニズム。
近傍通知	近傍要請メッセージに対する応答、またはデータリンク層アドレスの変更を通知するために、ノードが自発的に近傍通知メッセージを送ること。
近傍要請	近傍のリンク層アドレスを決定するために、ノードによって送信される要請。また、キャッシュされたリンク層アドレスによって近傍が到達可能であるかを確認します。

クラス	IPQoS では、似たような特性を共有するネットワークフローのグループ。クラスは、IPQoS 構成ファイル内に定義します。
クラスレスドメイン間ルーティング **(CIDR)** アドレス	ネットワーククラス (クラス A、B、C) に基づかない IPv4 アドレス形式。CIDR アドレスの長さは 32 ビットです。標準的な IPv4 10 進ドット表記形式にネットワーク接頭辞を付加したものを使用します。この接頭辞はネットワーク番号とネットワークマスクを定義します。
結果 **(outcome)**	トラフィックの計測結果に基づいて実行されるアクション。IPQoS メーターには、赤、黄、および緑の 3 種類の結果 (outcome) があり、IPQoS 構成ファイル内に定義されます。
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。
再実行攻撃	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。
最小カプセル化	ホームエージェント、外来エージェント、およびモバイルノードによってサポートされる任意の形態の IPv4 内 IPv4 トンネリング。最小カプセル化は、IP 内 IP カプセル化よりも 8 ないし 12 バイト少ないオーバヘッドしか持ちません。
サイトローカルアドレス	単一サイト上でアドレスを指定するために使用します。
自動構成	ホストが、サイト接頭辞とローカル MAC アドレスからその IPv6 アドレスを自動的に構成する処理。
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。
スタック	IP スタックを参照してください。
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監視し、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。
ステートレス自動構成	ホストがそれ自身の IPv6 アドレスを生成する処理。その生成は、ホスト自身の MAC アドレスと、ローカル IPv6 ルーターによって表明される IPv6 接頭辞を結合することによって行われます。
ストリーム制御転送プロトコル	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。

スプーフィング	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。
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。
セレクタ	ネットワークストリームからトラフィックを選択するために、特定クラスのパケットに適用される条件を具体的に定義する要素。セレクタは、IPQoS 構成ファイル内のフィルタ句に定義します。
専用アドレス	インターネット経由で経路制御ができない IP アドレス。プライベートアドレスは、インターネット接続を必要としない社内ネットワークのホストで使用できます。このようなアドレスは Address Allocation for Private Internets (`http://www.ietf.org/rfc/rfc1918.txt?number=1918`) で定義され、しばしば "1918" アドレスと呼ばれます。
双方向トンネル	双方向にデータグラムを送信するトンネル。
待機	グループ内のほかの物理インタフェースに障害が発生するまでデータの伝送には使用されない物理インタフェース。
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。対称鍵システムの一例として DES があります。
データグラム	IP データグラムを参照してください。
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。
デュアルスタック	IPv4 と IPv6 に関するネットワーク層の TCP/IP プロトコルスタック。このスタック以外は同一です。Oracle Solaris のインストール時に IPv6 を使用可能にすると、ホストはデュアルスタックバージョンの TCP/IP を受け取ります。
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。

動的再構成 **(DR)**	進行中の操作にほとんど、またはまったく影響を与えることなく、システムを実行しながらシステムを再構成できるようにする機能。Oracle からの Sun プラットフォームのすべてが、DR をサポートしているわけではありません。Oracle からの Sun プラットフォームの一部は、NIC など特定のタイプのハードウェアの DR のみをサポートする場合があります。
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。
トンネル	カプセル化される間データグラムが通過するパス。カプセル化を参照してください。
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 データグラムに対し認証と完全性を提供する拡張ヘッダー。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。
ノード	IPv6 では、IPv6 が有効なシステムのこと。ホストかルーターかは問いません。
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。
フィルタ	クラスの特性を IPQoS 構成ファイル内に定義するための規則セット。IPQoS システムでは、IPQoS 構成ファイル内に定義されたフィルタに適合するトラフィックフローを選択して処理します。パケットフィルタを参照してください。

負荷分散	インバウンドまたはアウトバウンドのトラフィックを一連のインタフェースに分散する処理。負荷分散を使用すると、より高いスループットを達成できます。ただし、負荷分散が行われるのは、データが複数の接続を経由して複数の標識に送信される場合だけです。負荷分散には、インバウンドトラフィック用のインバウンド負荷分散とアウトバウンドトラフィック用のアウトバウンド負荷分散の 2 種類があります。
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。
フローアカウンティング	IPQoS では、トラフィックフローに関する情報を蓄積、記録する処理のこと。フローアカウンティングを確立するには、`flowacct` モジュールのパラメータを IPQoS 構成ファイル内に定義します。
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)。
プロトコルスタック	IP スタックを参照してください。
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。
ホスト	パケット転送を行わないシステム。Oracle Solaris がインストールされると、システムはデフォルトでホストになります。つまり、このシステムはパケットを転送できません。通常、ホストは 1 つの物理インタフェースをもちます。ただし、複数のインタフェースをもつこともできます。
ホップ	2 つのホストを分離するルーターの数を判別するための手段。たとえば、始点ホストと終点ホストが 3 つのルーターで分離されている場合、ホストは互いに 4 ホップ離れています。
ホップ単位動作 **(Per-Hop Behavior、PHB)**	トラフィッククラスに割り当てられる優先順位。PHB は、そのクラスのフローに割り当てられる、ほかのトラフィッククラスに対する相対的な優先度を示します。
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。ユーザー優先順位の値を Ethernet データグラムの仮想 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データグラム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。
メーター	特定クラスのトラフィックフローの速度を測定する diffserv アーキテクチャーのモジュール。IPQoS 実装には、`tokenmt` および `tswtclmt` という 2 つのメーターがあります。
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。
ユーザー優先順位	サービスクラスのマークを実装する 3 ビットの値。VLAN デバイスのネットワーク上で Ethernet データグラムが転送される方法を定義します。
ユニキャストアドレス	IPv6 が有効なノードの単一インタフェースを識別する IPv6 アドレス。ユニキャストアドレスは、サイト接頭辞や、サブネット ID、インタフェース ID などからなります。
リダイレクト	特定の終点に到達するために、ホストに対して最適な最初のホップノードを、ルーターが通知すること。
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。
リンク層	IPv4/IPv6 のすぐ下の層。
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。
ローカル使用アドレス	ローカルの経路制御可能な範囲だけを対象とするユニキャストアドレス (サブネット内またはネットワーク内)。また、ローカルまたはグローバルな一意の範囲を対象とすることもできます。

索引

K

L

M

T

U

V

W

X

あ