

Oracle® Solaris 11.3 でのネットワークのセキュリティ保護

ORACLE®

Part No: E62586
2016年11月

Part No: E62586

Copyright © 1999, 2016, Oracle and/or its affiliates. All rights reserved.

このソフトウェアおよび関連ドキュメントの使用と開示は、ライセンス契約の制約条件に従うものとし、知的財産に関する法律により保護されています。ライセンス契約で明示的に許諾されている場合もしくは法律によって認められている場合を除き、形式、手段に関係なく、いかなる部分も使用、複写、複製、翻訳、放送、修正、ライセンス供与、送信、配布、発表、実行、公開または表示することはできません。このソフトウェアのリバース・エンジニアリング、逆アセンブル、逆コンパイルは互換性のために法律によって規定されている場合を除き、禁止されています。

ここに記載された情報は予告なしに変更される場合があります。また、誤りが無いことの保証はいたしかねます。誤りを見つけた場合は、オラクルまでご連絡ください。

このソフトウェアまたは関連ドキュメントを、米国政府機関もしくは米国政府機関に代わってこのソフトウェアまたは関連ドキュメントをライセンスされた者に提供する場合は、次の通知が適用されます。

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

このソフトウェアまたはハードウェアは様々な情報管理アプリケーションでの一般的な使用のために開発されたものです。このソフトウェアまたはハードウェアは、危険が伴うアプリケーション(人的傷害を発生させる可能性があるアプリケーションを含む)への用途を目的として開発されていません。このソフトウェアまたはハードウェアを危険が伴うアプリケーションで使用する場合、安全に使用するために、適切な安全装置、バックアップ、冗長性(redundancy)、その他の対策を講じることは使用者の責任となります。このソフトウェアまたはハードウェアを危険が伴うアプリケーションで使用したこと起因して損害が発生しても、Oracle Corporationおよびその関連会社は一切の責任を負いかねます。

OracleおよびJavaはオラクル およびその関連会社の登録商標です。その他の社名、商品名等は各社の商標または登録商標である場合があります。

Intel, Intel Xeonは、Intel Corporationの商標または登録商標です。すべてのSPARCの商標はライセンスをもとに使用し、SPARC International, Inc.の商標または登録商標です。AMD, Opteron, AMDロゴ、AMD Opteronロゴは、Advanced Micro Devices, Inc.の商標または登録商標です。UNIXは、The Open Groupの登録商標です。

このソフトウェアまたはハードウェア、そしてドキュメントは、第三者のコンテンツ、製品、サービスへのアクセス、あるいはそれらに関する情報を提供することがあります。適用されるお客様とOracle Corporationとの間の契約に別段の定めがある場合を除いて、Oracle Corporationおよびその関連会社は、第三者のコンテンツ、製品、サービスに関して一切の責任を負わず、いかなる保証もいたしません。適用されるお客様とOracle Corporationとの間の契約に定めがある場合を除いて、Oracle Corporationおよびその関連会社は、第三者のコンテンツ、製品、サービスへのアクセスまたは使用によって損失、費用、あるいは損害が発生しても一切の責任を負いかねます。

ドキュメントのアクセシビリティについて

オラクルのアクセシビリティについての詳細情報は、Oracle Accessibility ProgramのWeb サイト(<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>)を参照してください。

Oracle Supportへのアクセス

サポートをご契約のお客様には、My Oracle Supportを通して電子支援サービスを提供しています。詳細情報は(<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info>)か、聴覚に障害のあるお客様は (<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs>)を参照してください。

目次

このドキュメントの使用方法	17
1 仮想化環境でのリンク保護の使用	19
Oracle Solaris 11.3 のネットワークセキュリティーの新機能	19
リンク保護について	20
リンク保護のタイプ	20
リンク保護の構成	21
▼ リンク保護を有効にする方法	22
▼ リンク保護を無効にする方法	23
▼ 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	23
▼ 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	24
▼ リンク保護の構成と統計情報を表示する方法	25
2 ネットワークのチューニング	27
ネットワークのチューニング	27
▼ ネットワークルーティングデーモンを無効にする方法	28
▼ ブロードキャストパケット転送を無効にする方法	28
▼ エコーリクエストへの応答を無効にする方法	29
▼ 厳密なマルチホームを設定する方法	30
▼ 不完全な TCP 接続の最大数を設定する方法	30
▼ 中断中の TCP 接続の最大数を設定する方法	31
▼ 初期の TCP 接続に強固な乱数を指定する方法	31
▼ ICMP リダイレクトを妨げる方法	32
▼ 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	33
3 Web サーバーと Secure Sockets Layer プロトコル	35
SSL カーネルプロキシ は Web サーバー通信を暗号化する	35

SSL カーネルプロキシを使用して Web サーバーを保護する	37
▼ SSL カーネルプロキシ を使用するように Apache 2.2 Web サーバーを構成する方法	37
▼ SSL カーネルプロキシ を使用するように Oracle iPlanet Web Server を構成する方法	39
▼ 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	41
▼ ゾーンで SSL カーネルプロキシ を使用する方法	43
4 Oracle Solaris での OpenBSD パケットフィルタファイアウォール	45
パケットフィルタの概要	45
Oracle Solaris での PF の IP フィルタ および OpenBSD パケットフィルタとの比較	46
IP フィルタ と Oracle Solaris パケットフィルタの比較	46
Oracle Solaris パケットフィルタと OpenBSD パケットフィルタの比較	47
Oracle Solaris でパケットフィルタを使用するためのガイドライン	48
パケットフィルタファイアウォールとパケット処理	48
Oracle Solaris でのパケットフィルタファイアウォールモジュール	49
PF でのパケット処理	51
パケットフィルタの構成ファイル	54
パケットフィルタ規則の構文	54
パケットフィルタ規則のアクション	55
パケットフィルタ規則の一致パラメータ	56
パケットフィルタ規則のオプションのアクション	57
パケットフィルタのマクロとテーブル	59
IPF 規則と比較した PF 規則の例	60
パケットフィルタ規則の処理	65
パケットフィルタのリファレンス	66
5 パケットフィルタファイアウォールの構成	67
IPF 規則を PF 規則に変換するための方針	67
Oracle Solaris ファイアウォールを実行する準備	68
Oracle Solaris でのパケットフィルタサービスの実行およびモニタリング	69
▼ Oracle Solaris で PF ファイアウォールを構成する方法	70
▼ Oracle Solaris で PF ファイアウォールをモニターする方法	71
PF 構成ファイルの例	72

6 Oracle Solaris での IP フィルタファイアウォール	75
IP フィルタとは	75
IP フィルタのパケット処理	76
IP フィルタの使用ガイドライン	79
IP フィルタの構成ファイルの使用	79
IP フィルタの規則セットの使用	80
IP フィルタのパケットのフィルタリング機能の使用	80
IP フィルタの NAT 機能の使用	83
IP フィルタのアドレスプール機能の使用	85
IP フィルタ用の IPv6	86
IP フィルタのマニュアルページ	87
7 IP フィルタファイアウォールの構成	89
IP フィルタサービスの構成	89
▼ IP フィルタサービスのデフォルトを表示する方法	90
▼ IP フィルタ構成ファイルの作成方法	91
▼ IP フィルタを有効にし、リフレッシュする方法	93
▼ パケット再構築を無効にする方法	93
▼ ループバックフィルタリングを有効にする方法	94
▼ パケットフィルタリングを無効にする方法	95
IP フィルタ規則セットの操作	95
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	96
IP フィルタ用 NAT 規則の管理	103
IP フィルタのアドレスプールの管理	105
IP フィルタの統計および情報の表示	107
▼ IP フィルタの状態テーブルを参照する方法	108
▼ IP フィルタの状態統計を参照する方法	108
▼ 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	109
▼ IP フィルタの NAT 統計を参照する方法	109
▼ IP フィルタのアドレスプール統計情報を表示する方法	110
IP フィルタ用ログファイルの操作	110
▼ IP フィルタのログファイルを設定する方法	111
▼ IP フィルタのログファイルを参照する方法	112
▼ パケットログバッファをフラッシュする方法	113
▼ ロギングされたパケットをファイルに保存する方法	114
IP フィルタの構成ファイルの例	114
8 IP セキュリティーアーキテクチャーについて	119

IPsec の概要	119
IPsec パケットのフロー	121
IPsec セキュリティーアソシエーション	124
IPsec セキュリティーアソシエーションの鍵管理	124
IPsec 保護プロトコル	125
認証ヘッダー	126
カプセル化セキュリティーペイロード	126
IPsec の認証アルゴリズムと暗号化アルゴリズム	127
IPsec ポリシー	128
IPsec のトランスポートモードとトンネルモード	129
仮想プライベートネットワークと IPsec	132
IPsec と FIPS 140-2	133
IPsec と NAT 越え	133
IPsec と SCTP	134
IPsec と Oracle Solaris ゾーン	135
IPsec と仮想マシン	135
IPsec の構成コマンドとファイル	135
9 IPsec の構成	137
IPsec によるネットワークトラフィックの保護	137
▼ IPsec によって 2 つのサーバー間でネットワークトラフィックをセ キュリティー保護する方法	139
▼ IPsec を使用してほかのサーバーとの Web サーバー通信を保護する 方法	144
IPsec による VPN の保護	146
トンネルモードを使用して VPN を IPsec で保護する例	146
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	148
▼ IPsec のトンネルモードで 2 つの LAN 間の接続を保護する方法	149
IPsec のその他のタスク	154
▼ IPsec の鍵を手動で作成する方法	154
▼ ネットワークセキュリティーの役割を構成する方法	157
▼ 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	160
10 インターネット鍵交換について	163
IKE の概要	163
IKE の概念および用語	164
IKE の動作	165

IKEv2 と IKEv1 の比較	168
IKEv2 プロトコル	169
IKEv2 構成の選択	170
公開証明書の IKEv2 ポリシー	170
IKEv2 および FIPS 140-2	171
IKEv1 プロトコル	171
IKEv1 の鍵ネゴシエーション	172
IKEv1 構成の選択	173
11 IKEv2 の構成	175
IKEv2 の構成	175
事前共有鍵による IKEv2 の構成	176
▼ 事前共有鍵で IKEv2 を構成する方法	176
▼ IKEv2 で事前共有鍵を使用する場合に新しいピアを追加する方 法	179
IKEv2 の公開鍵証明書を格納するためのキーストアの初期化	181
▼ IKEv2 公開鍵証明書用キーストアを作成および使用する方法	182
公開鍵証明書による IKEv2 の構成	184
▼ 自己署名付き公開鍵証明書により IKEv2 を構成する方法	185
▼ CA の署名付き証明書で IKEv2 を構成する方法	190
▼ IKEv2 で証明書検証ポリシーを設定する方法	193
▼ IKEv2 で失効した証明書を処理する方法	194
▼ ハードウェア上で IKEv2 の公開鍵証明書を生成および格納する方 法	196
12 IKEv1 の構成	201
IKEv1 の構成	201
事前共有鍵による IKEv1 の構成	202
▼ 事前共有鍵で IKEv1 を構成する方法	202
▼ 新規ピアシステムのために IKEv1 を更新する方法	205
公開鍵証明書による IKEv1 の構成	206
▼ 自己署名付き公開鍵証明書により IKEv1 を構成する方法	207
▼ CA の署名付き証明書で IKEv1 を構成する方法	212
▼ ハードウェア上で IKEv1 の公開鍵証明書を生成および格納する方 法	217
▼ IKEv1 で失効した証明書を処理する方法	221
移動体システム用の IKEv1 の構成	223
▼ 遠隔地のシステム用に IKEv1 を構成する方法	223

接続したハードウェアを検出するように IKEv1 を構成する	229
▼ Sun Crypto Accelerator 6000 ボードを検出するように IKEv1 を構成する 方法	230
13 IPsec およびその鍵管理サービスのトラブルシューティング	233
IPsec およびその鍵管理構成のトラブルシューティング	233
▼ IPsec および IKE システムのトラブルシューティングを準備する方 法	234
▼ IPsec と IKE を実行する前にシステムをトラブルシューティングす る方法	235
▼ IPsec 実行中にシステムをトラブルシューティングする方法	236
IPsec および IKE の意味上の誤りのトラブルシューティング	240
IPsec およびその鍵サービスに関する情報の表示	241
IPsec および手動鍵サービスプロパティの表示	242
IKE 情報の表示	242
IPsec およびその鍵サービスの管理	246
IPsec およびその鍵サービスの構成および管理	246
実行中の IKE デーモンの管理	248
14 IPsec および鍵管理のリファレンス	251
IPsec リファレンス	251
IPsec のサービス、ファイル、およびコマンド	251
IPsec のセキュリティーアソシエーションデータベース	256
IPsec での鍵管理	256
IKEv2 リファレンス	257
IKEv2 のユーティリティーおよびファイル	257
IKEv2 サービス	258
IKEv2 デーモン	259
IKEv2 構成ファイル	259
IKEv2 の ikeadm コマンド	260
IKEv2 事前共有鍵ファイル	260
IKEv2 ikev2cert コマンド	260
IKEv1 リファレンス	261
IKEv1 のユーティリティーおよびファイル	261
IKEv1 サービス	262
IKEv1 デーモン	263
IKEv1 構成ファイル	264
IKEv1 ikeadm コマンド	264

IKEv1 事前共有鍵ファイル	265
IKEv1 公開鍵のデータベースおよびコマンド	265
用語集	269
索引	277

表目次

表 1	リンク保護の構成のタスクマップ	22
表 2	ネットワークのチューニングのタスクマップ	27
表 3	IP フィルタ と Oracle Solaris でのパケットフィルタの比較	46
表 4	OpenBSD PF と Oracle Solaris PF の違い	47
表 5	IP フィルタサービスの構成のタスクマップ	89
表 6	IP フィルタルールセットの操作のタスクマップ	96
表 7	IP フィルタの統計および情報の表示のタスクマップ	107
表 8	IP フィルタログファイルの操作のタスクマップ	110
表 9	IPsec で AH と ESP が提供する保護	127
表 10	IPsec の構成コマンドとファイルの例	136
表 11	IPsec によるネットワークトラフィックの保護のタスクマップ	138
表 12	IPsec のその他のタスクのタスクマップ	154
表 13	Oracle Solaris での IKEv2 および IKEv1 の実装	168
表 14	公開鍵証明書による IKEv2 の構成のタスクマップ	184
表 15	公開鍵証明書による IKEv1 の構成のタスクマップ	207
表 16	移動体システム用の IKEv1 の構成のタスクマップ	223
表 17	IKEv2 のサービス名、コマンド、構成と鍵の保管場所、およびハードウェアデバイス	257
表 18	IKEv1 のサービス名、コマンド、構成と鍵の保管場所、およびハードウェアデバイス	261
表 19	IKEv1 の <code>ikecert</code> オプションと <code>ike/config</code> エントリの対応	266

例目次

例 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	43
例 2	PF での NAT 規則	59
例 3	PF でのスパム規則	60
例 4	IP フィルタ 構成ファイルに基づいた PF 構成ファイル	72
例 5	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	98
例 6	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	98
例 7	パケットフィルタリング規則セットの削除	99
例 8	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	100
例 9	アクティブでない規則セットへの規則の追加	101
例 10	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	102
例 11	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	103
例 12	NAT 規則の削除	104
例 13	NAT 規則セットへの規則の追加	105
例 14	アドレスプールの削除	106
例 15	アドレスプールへの規則の追加	107
例 16	IP フィルタの状態テーブルの参照	108
例 17	IP フィルタの状態統計の参照	109
例 18	IP フィルタの NAT 統計の参照	110
例 19	IP フィルタのアドレスプール統計の参照	110
例 20	IP フィルタログの作成	112
例 21	IP フィルタのログファイルの参照	113
例 22	パケットログバッファのフラッシュ	113
例 23	ファイルへのロギングされたパケットの保存	114
例 24	IP フィルタのホスト構成	115
例 25	IP フィルタのサーバー構成	115
例 26	IP フィルタのルーター構成	117

例 27	ssh 接続を使用した IPsec ポリシーのリモート構成	142
例 28	FIPS 140-2 承認アルゴリズムを使用した IPsec ポリシーの構成	142
例 29	IKEv2 プロトコルのみを使用する IPsec ポリシーの構成	143
例 30	サーバーの or pass アクションを使用した、IPsec を使用するクライアントシステムへの移行	143
例 31	すべてのサブネットで使用できるトンネルの作成	147
例 32	2つのサブネットだけを接続するトンネルの作成	147
例 33	ネットワーク管理およびセキュリティーの役割の作成と割り当て	158
例 34	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	158
例 35	信頼できるユーザーに IPsec を構成および管理する権限を与える ...	158
例 36	ローカルとリモートで異なる IKEv2 共有鍵を使用する	178
例 37	ライフタイムの制限を付けた自己署名付き証明書を作成する	189
例 38	公開鍵証明書をそのフィンガープリントで検証する	190
例 39	システムが IKEv2 証明書検証を待機する時間を変更する	196
例 40	IKEv1 事前共有鍵をリフレッシュする	204
例 41	IKEv1 の構成時に rsa_encrypt を使用する	216
例 42	CRL を IKEv1 のローカルの certr1db データベースに貼り付ける	222
例 43	移動体システムから保護されたトラフィックを受信するために IKEv1 を使用する中央コンピュータを構成する	226
例 44	NAT 越しのシステムを IPsec と IKEv1 で構成する	227
例 45	移動体システムからの自己署名付き証明書の受け入れ	228
例 46	自己署名付き証明書による中央システムとの接続	228
例 47	メタスロットトークンの検索と使用	230
例 48	無効な IKEv2 構成の修正	238
例 49	一致しないルールメッセージの修正	239
例 50	実行中の IKE デーモンの新しいデバッグレベルの設定	239

このドキュメントの使用方法

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

製品ドキュメントライブラリ

この製品および関連製品のドキュメントとリソースは <http://www.oracle.com/pls/topic/lookup?ctx=E62101-01> で入手可能です。

フィードバック

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

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。

- 19 ページの「Oracle Solaris 11.3 のネットワークセキュリティの新機能」
- 20 ページの「リンク保護について」
- 21 ページの「リンク保護の構成」

Oracle Solaris 11.3 のネットワークセキュリティの新機能

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

- OpenBSD パケットフィルタは、ログに記録されたパケットを取り込み、それをシステムへの入力のために評価するネットワークファイアウォールです。PF は、排他的 IP インスタンスを含むゾーンなどの Oracle Solaris コンポーネントと連携して動作するように拡張された OpenBSD パケットフィルタ (PF), バージョン 5.5 に基づいています。ポリシーベースのルーティングオプション (PBR) については、57 ページの「パケットフィルタ規則のオプションのアクション」にある route-to の説明を参照してください。
- SSL カーネルプロキシは、TLS 1.0 および SSL 3.0 プロトコルをサポートしています。ただし、SSL 3.0 プロトコルはデフォルトで無効になっています。
- IKE および IPsec は、暗号化フレームワークの最新のアルゴリズムのメカニズムを使用できます。このリリースでは、Camellia が使用可能になっています。
- IKEv1 から IKEv2 への移行に役立てるために、IPsec 管理者は、IPsec ポリシールールが使用する必要がある IKE プロトコルのバージョンを指定できます。システムまたはシステムのネットワークに 1 つの IKE プロトコルを指定することによって、管理者は、IKEv2 をサポートできないシステムとの下位互換性を維持しながら、IKEv1 から新しい IKEv2 プロトコルにシステムを移行できます。168 ページの「IKE バージョンの指定」および例 29 「IKEv2 プロトコルのみを使用する IPsec ポリシーの構成」を参照してください。
- IPsec 管理者は、IPsec ルールに `or pass {}` 命令を使用して、暗号化されていないがそれ以外のルールは満たしているパケットを破棄せずに通過させることを指

定できます。このオプションを使用すると、IPsec クライアント、および IPsec がまだ構成されていないクライアントをサーバーがサポートできます。詳細は、例 30「サーバーの `or pass` アクションを使用した、IPsec を使用するクライアントシステムへの移行」および `ipseconf(1M)` のマニュアルページを参照してください。

注記 - Oracle Solaris の暗号化フレームワーク機能は、FIPS 140-2-2 レベル 1 で検証されています。IKE の FIPS 140-2 モードの使用については、表 13 を参照してください。ハードウェアとソフトウェアの詳細は、Oracle FIPS 140-2 ソフトウェア検証 (<http://www.oracle.com/technetwork/topics/security/fips140-software-validations-1703049.html>) を参照してください。

リンク保護について

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

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

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

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

リンク保護のタイプ

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

`mac-nospoof`

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

ip-nospoof

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

allowed-ips リンクプロパティを使用して、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せん。

allowed-dhcp-cids リンクプロパティを使用して、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防ぐよう設計されています。

注記 - リンク保護のためにドロップされたパケットは、4つの保護のタイプ (**mac_spoofed**、**dhcp_spoofed**、**ip_spoofed**、**restricted**) についてのカーネル統計によって追跡されます。これらのリンクごとの統計情報を取得するには、[25 ページの「リンク保護の構成と統計情報を表示する方法」](#)を参照してください。

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

リンク保護の構成

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

注記 - 大域ゾーンでリンク保護を構成する必要があります。

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

表 1 リンク保護の構成のタスクマップ

タスク	説明	参照先
リンク保護を有効にします。	リンクから送信されるパケットを制限し、なりすましからリンクを保護します。	22 ページの「リンク保護を有効にする方法」
リンク保護を無効にします。	リンク保護を解除します。	23 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。	23 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。	24 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。	25 ページの「リンク保護の構成と統計情報を表示する方法」

▼ リンク保護を有効にする方法

この手順では、送信パケットのタイプを制限し、リンクのなりすましを防ぎます。

始める前に Network Link Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。

1. 使用可能なリンク保護のタイプを表示します。

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

可能なタイプの説明については、20 ページの「リンク保護のタイプ」および [dladm\(1M\)](#) のマニュアルページを参照してください。

2. 1 つまたは複数の保護タイプを指定してリンク保護を有効にします。

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

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

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



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

3. リンク保護が有効にされていることを確認します。

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

▼ リンク保護を無効にする方法

この手順では、リンク保護をデフォルト値のリンク保護なしにリセットします。

始める前に Network Link Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

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

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

2. リンク保護が無効になっているかどうかを確認します。

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

▼ IP なりすましに対して保護するように IP アドレスを指定する方法

始める前に [How to Enable Link Protection](#) に示すように、[22 ページの「リンク保護を有効にする方法」](#) 保護のタイプを有効にします。

Network Link Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. IP なりすましに対して保護を有効にしていることを確認します。

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

2. **allowed-ips** リンクプロパティ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 プロパティに追加する方法を示します。

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

詳細は、[dladm\(1M\)](#) のマニュアルページを参照してください。

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

始める前に How to Enable Link Protection に示すように、[22 ページの「リンク保護を有効にする方法」](#) 保護のタイプを有効にします。

Network Link Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

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

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

2. **allowed-dhcp-cids** リンクプロパティに 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 プロパティの値として、文字列 hello を指定する方法を示します。

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

詳細は、[dladm\(1M\)](#) のマニュアルページを参照してください。

▼ リンク保護の構成と統計情報を表示する方法

始める前に Network Link Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。

1. リンク保護のプロパティ値を表示します。

```
# 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 プロパティの値を示します。

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

注記 - allowed-ips プロパティは、EFFECTIVE の下に一覧表示されているように、ip-nospoof が有効にされている場合にのみ使用されます。allowed-dhcp-cids プロパティは 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 <-----
    dhcspoofed 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 `dhcspoofed` 統計情報を検出します。

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

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

◆◆◆ 第 2 章

ネットワークのチューニング

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

ネットワークのチューニング

表 2 ネットワークのチューニングのタスマップ

タスク	説明	参照先
ネットワークルーティングデーモンを無効にします。	不審なネットワーク侵入者によるシステムへのアクセスを制限します。	28 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」
ネットワークポート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。	28 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。	29 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。	30 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。	30 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。	31 ページの「中断中の TCP 接続の最大数を設定する方法」
初期の TCP 接続に対して強固な乱数が生成されることを確認します。	RFC 6528 で規定されているシーケンス番号生成値に準拠します。	31 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。	ネットワークポートのインジケータを削除します。	32 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を強化します。	33 ページの「ネットワークパラメータをセキュアな値にリセットする方法」

▼ ネットワークルーティングデーモンを無効にする方法

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

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

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。

1. ルーティングデーモンが動作していることを確認します。

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

サービスが実行中でない場合は、ここで停止できます。

2. ルーティングデーモンを無効にします。

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

3. ルーティングデーモンが無効にされていることを確認します。

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

参照 [routeadm\(1M\)](#) のマニュアルページ

▼ ブロードキャストパケット転送を無効にする方法

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

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

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

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

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

2. 現在の値を検証します。

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

参照 [ipadm\(1M\)](#) のマニュアルページ

▼ エコーリクエストへの応答を無効にする方法

この手順を使用して、ネットワークトポロジに関する情報の流布を回避します。

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

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

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

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

参照 詳細は、『Oracle Solaris 11.3 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 プロパティは、マルチホームシステム上での IP パケットの送受信動作を制御します。

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「[割り当てられている管理権利の使用](#)」を参照してください。

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

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

2. 現在の値を確認し、可能な値に注目します。

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

参照 詳細は、『Oracle Solaris 11.3 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、[149 ページの「IPsec のトンネルモードで 2 つの LAN 間の接続を保護する方法」](#)を参照してください。

▼ 不完全な TCP 接続の最大数を設定する方法

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

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. 受信接続の最大数を設定します。

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

2. 現在の値を検証します。

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

参照 詳細は、『Oracle Solaris 11.3 カーネルのチューンアップ・リファレンスマニュアル』の「[_conn_req_max_q0](#)」および [ipadm\(1M\)](#) のマニュアルページを参照してください。

▼ 中断中の TCP 接続の最大数を設定する方法

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

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. 受信接続の最大数を設定します。

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

2. 現在の値を検証します。

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

参照 詳細は、『Oracle Solaris 11.3 カーネルのチューンアップ・リファレンスマニュアル』の「[_conn_req_max_q](#)」および [ipadm\(1M\)](#) のマニュアルページを参照してください。

▼ 初期の TCP 接続に強固な乱数を指定する方法

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

始める前に `solaris.admin.edit/etc.default/inetinit` 承認が割り当てられている管理者になる必要があります。デフォルトでは、`root` 役割がこの承認を持っています。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「[割り当てられている管理権利の使用](#)」を参照してください。

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

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

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

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

3. システムをリブートします。

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

▼ ICMP リダイレクトを妨げる方法

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

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

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

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

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

2. **ICMP リダイレクトメッセージの送信を妨げます。**

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

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

```
PROTO PROPERTY          PERM CURRENT  PERSISTENT  DEFAULT  POSSIBLE
ipv6 send_redirects     rw  off        off         on       on,off
```

詳細は、『Oracle Solaris 11.3 カーネルのチューンアップ・リファレンスマニュアル』の「」および [ipadm\(1M\)](#) のマニュアルページを参照してください。

▼ ネットワークパラメータをセキュアな値にリセットする方法

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

始める前に Network Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「割り当てられている管理権利の使用」を参照してください。

1. IP パケットに対してソースパケット転送プロパティを 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 11.3 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対策をバイパスすることを回避します。ソースルーティングされたパケットにより、パケットの送信元は、ルーターに構成されているパスと異なるパスを提案できます。

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

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

詳細は、『Oracle Solaris 11.3 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

Web サーバーと Secure Sockets Layer プロトコル

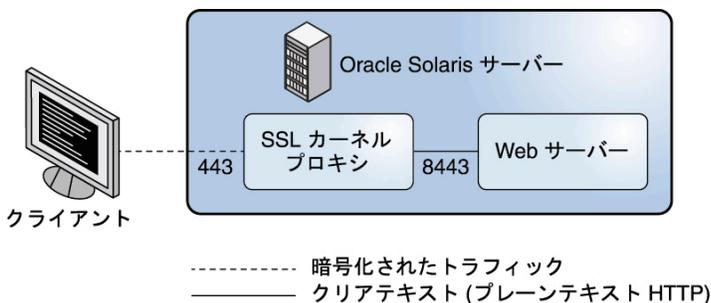
この章では、Secure Sockets Layer (SSL) プロトコルを使用して、Oracle Solaris システムの Web サーバー通信を暗号化し、高速化する方法について説明します。

- 35 ページの「SSL カーネルプロキシは Web サーバー通信を暗号化する」
- 37 ページの「SSL カーネルプロキシを使用して Web サーバーを保護する」

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 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。

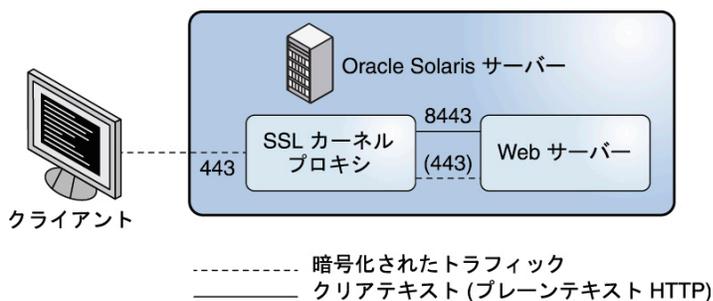
図1は、SSL カーネルプロキシ を使用している Web サーバーの基本的なシナリオを示しています。SSL カーネルプロキシ はポート 443 上に構成され、Web サーバーはポート 8443 上に構成されて、そこで暗号化されていない HTTP 通信を受信します。

- 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2は、複雑なシナリオを示しています。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 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、TLS 1.0 および SSL 3.0 プロトコルをサポートしています。ただし、SSL 3.0 プロトコルはデフォルトで無効になっています。もっとも一般的な

暗号化スイートがサポートされています。サポートされている暗号化スイートの完全なリスト、および特定の暗号化スイートまたはプロトコルを有効または無効にする手順については、[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、サポートされていないすべての暗号化スイートに対して、ユーザーレベルのサーバーにフォールバックするように構成できます。

SSL カーネルプロキシ を使用して Web サーバーを保護する

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

- [37 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」](#)
- [39 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」](#)
- [41 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」](#)
- [43 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* – 証明書と鍵の形式を定義するために、*-f* オプションとともに使用されます。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* – 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するために、*-i* オプションとともに使用されます。
- *password-file* – *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されるパスワードを取得するために、*-p* オプションとともに使用されます。*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* – PKCS #11 トークンを指定するために、*-T* オプションとともに使用されます。
- *certificate-label* – PKCS #11 トークン内の証明書オブジェクトに含まれるラベルを選択するために、*-c* オプションとともに使用されます。
- *proxy-port* – SSL プロキシポートを設定するために、*-x* オプションとともに使用されます。標準ポート 80 とは異なるポートを指定する必要があります。Web サーバーは暗号化されていないテキストトラフィックを SSL プロキシポートで待機します。一般に値は 8443 です。
- *ssl-port* – SSL カーネルプロキシの待機するポートを指定します。一般に値は 443 です。

4. SSL カーネルプロキシのサービスインスタンスを作成します。

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

- 鍵の形式として PEM または PKCS #12 を指定します。

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

- 鍵の形式として PKCS #11 を指定します。

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

5. サービスインスタンスがオンラインであることを確認します。

```
# svcs svc:/network/ssl/proxy
STATE          STIME          FMRI
```

```
online          02:22:22 svc:/network/ssl/proxy:default
```

次の出力は、サービスインスタンスが作成されなかったことを示します。

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

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

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

```
Listen proxy-port
```

7. Web サーバーの SMF 依存関係を設定します。

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

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

8. Web サーバーサービスを有効にします。

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

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

SSL カーネルプロキシは Oracle iPlanet Web Server での SSL パケットの処理速度を向上できます。この手順では、[図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 \(http://www.oracle.com/technetwork/middleware/iplanetwebserver-098726.html?ssSourceSiteId=ocomen\)](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 \(http://docs.oracle.com/cd/E18958_01/index.htm\)](http://docs.oracle.com/cd/E18958_01/index.htm) を参照してください。

Network Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。

1. Web サーバーを停止します。

管理者の Web インタフェースを使ってサーバーを停止します。手順については、[Oracle iPLANET WEB SERVER 7.0.15 \(http://docs.oracle.com/cd/E18958_01/index.htm\)](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リストについては、[37 ページの「SSL カーネルプロキシを使用するように Apache 2.2 Web サーバーを構成する方法」のステップ 3](#) を参照してください。

3. SSL カーネルプロキシのサービスインスタンスを作成します。

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

■ 鍵の形式として PEM または PKCS #12 を指定します。

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

■ 鍵の形式として PKCS #11 を指定します。

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

4. インスタンスがオンラインであることを確認します。

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

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

手順については、[Oracle iPLANET WEB SERVER 7.0.15 \(http://docs.oracle.com/cd/E18958_01/index.htm\)](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。この手順では、[図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。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

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

- a. 証明書署名要求 (CSR) を生成します。

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

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

注記 - FIPS 140-2 に準拠する場合、RSA 鍵の最小の長さは 2048 です。詳細は、『[Oracle Solaris 11.3 での FIPS 140 対応システムの使用](#)』を参照してください。

詳細は、[openssl\(5\)](#) のマニュアルページを参照してください。

- b. CSR を認証局 (CA) に送信します。
 - c. webcert.pem ファイルを CA からの署名付き証明書で置き換えます。
2. パスフレーズと公開/非公開鍵証明書で SSL カーネルプロキシ を構成します。
 - a. パスフレーズを作成、保存、保護します。

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

注記 - パスフレーズに空白文字を含めることはできません。

- b. 非公開鍵証明書と公開鍵証明書を 1つのファイルに組み合わせます。

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

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

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

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

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

```
# pfedit /etc/apache2/2.2/httpd.conf
...
## Listen 80
Listen 8443
```

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

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

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

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

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

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

- b. スクリプトを実行可能ファイルにし、ファイルを保護します。

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

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

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

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

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

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

7. Apache サービスを有効にします。

```
# svcadm enable apache22
```

8. (オプション) 2 つのポートが機能しているかどうかを確認します。

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

a. SSL カーネルプロキシ で使用可能な暗号化を使用します。

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

kstat カウンタ kssl_full_handshakes が 1 増加することにより、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が検証されます。

```
# 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 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する必要があります。大域ゾーンで ksslcfg コマンドによってサービスインスタンスを構成すると、非大域ゾーンで Web サーバーを起動できるようになります。

- インスタンスを構成する際には、`ksslcfg` コマンドを実行して特定のホスト名または IP アドレスを指定する必要があります。特に、インスタンスは IP アドレスに `INADDR_ANY` を指定できません。

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

Network Security および Zone Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. 非大域ゾーン内で Web サーバーを停止します。

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

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

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

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

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

3. 非大域ゾーンで、Web サービスインスタンスを有効にします。

たとえば、`apache-zone` で Web サービスを有効にします。

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

Oracle Solaris での OpenBSD パケットフィルタファイアウォール

この章では、Oracle Solaris のパケットフィルタ (PF) 機能の概要について説明します。PF タスクについては、第5章「パケットフィルタファイアウォールの構成」を参照してください。

この章で扱う内容は、次のとおりです。

- 45 ページの「パケットフィルタの概要」
- 46 ページの「Oracle Solaris での PF の IP フィルタ および OpenBSD パケットフィルタとの比較」
- 48 ページの「パケットフィルタファイアウォールとパケット処理」
- 54 ページの「パケットフィルタの構成ファイル」
- 54 ページの「パケットフィルタ規則の構文」
- 65 ページの「パケットフィルタ規則の処理」
- 66 ページの「パケットフィルタのリファレンス」

パケットフィルタの概要

Oracle Solaris の OpenBSD パケットフィルタ (PF) 機能は、着信パケットを取り込み、それをシステムへの入力およびシステムからの出力のために評価するネットワークファイアウォールです。PF には、ステートフルパケット検査が用意されています。IP アドレスやポート番号だけでなく、受信しているネットワークインタフェースごとにパケットを一致させることができます。

PF は、排他的 IP インスタンスを含むゾーンなどの Oracle Solaris コンポーネントと連携して動作するように拡張された OpenBSD パケットフィルタ (PF) バージョン 5.5 に基づいています。Oracle Solaris 11.3 では、PF と IP フィルタの両方をパケットのフィルタリングに使用できます。PF はより堅牢なフィルタリングモジュールであるため、ファイアウォールポリシーを IP フィルタ規則から PF に移行してください。

次の OpenBSD PF 機能は、Oracle Solaris バージョンには含まれていません。

- RFC 6146 で説明されている IPv6 と IPv4 の間のネットワークアドレス変換 (NAT-64)
- PF ファイアウォールをクラスタとして配備できる PFSYNC
- QOS (パケットのキューイング)
- Netflow 統計

Oracle Solaris での PF の IP フィルタ および OpenBSD パケットフィルタとの比較

IP フィルタ の機能とパケットフィルタ (PF) の機能は、正確には一致しません。そのため、IP フィルタ の構成をパケットフィルタの構成にマップする変換ツールはあり得ません。ネットワークポリシー (ファイアウォールポリシーを含む) をある製品から別の製品に変換する場合の最適な方針は、要件と仕様を確認してから、新しいツールを使用してポリシーを実装することです。IP フィルタ 構成ファイルのネットワークポリシーを実装する PF 構成ファイルの例については、[例4「IP フィルタ 構成ファイルに基づいた PF 構成ファイル」](#)を参照してください。

IP フィルタ と Oracle Solaris パケットフィルタの比較

次の表は、PF の Oracle Solaris 実装と IP フィルタ を比較しています。[表4](#)では、PF の Oracle Solaris 実装と OpenBSD PF を比較します。

表 3 IP フィルタ と Oracle Solaris でのパケットフィルタの比較

ファイアウォールの機能	IP フィルタ	Oracle Solaris の PF 実装
構成ファイル	ippool.conf、ipnat.conf、ipv6.conf など複数	1つの pf.conf ファイル
規則の理解しやすさ	複雑な構文	マクロなどのショートカットやテーブルによる読みやすさ
IPv4 および IPv6 パケットのフラグメント	管理者が再構築を明示的にオンにする必要がある	IP の再構築がデフォルトでオンになっている
ループバックインタフェースの保護	set intercept_loopback true; で有効にする必要がある	ファイアウォールがループバックインタフェース上のパケットを常にインターセプトする
パッケージ名	ipfilter	firewall
OS 署名ファイル	なし	pf.os

ファイアウォールの機能	IP フィルタ	Oracle Solaris の PF 実装
pass 規則	デフォルトではステートレス	デフォルトではステートフル
権利プロファイル	IP フィルタの管理	Network Firewall Management
SMF サービス名	ipfilter	サービスを有効にする前に PF 構成を必要とする firewall と、pflog および ftp-proxy サービス

Oracle Solaris パケットフィルタと OpenBSD パケットフィルタの比較

次の表では、PF の OpenBSD 実装と Oracle Solaris バージョンの違いについて説明します。Oracle Solaris に含まれていない OpenBSD 機能については、[45 ページの「パケットフィルタの概要」](#)を参照してください。

表 4 OpenBSD PF と Oracle Solaris PF の違い

OpenBSD PF の動作	Oracle Solaris PF の動作	Oracle Solaris PF での違い
ユーザーが Web から PF をダウンロードします。	管理者が IPS パッケージとして PF をインストールします。	IPS リポジトリが、保存されたデータと移動中のデータのセキュリティーを提供します。
pf* コマンドがファイアウォールを実行します。	svc* コマンドがファイアウォールを実行します。これは SMF サービスです。	一部の PF コマンドの使用は、SMF コマンドに置き換えられています。
NAT での PF は IPv4 および IPv6 ネットワークを介して機能します。	OpenBSD が RFC 6146 で説明されている NAT-64 をサポートしているのに対して、Oracle Solaris は、 RFC 2663 で説明されている従来の NAT のみをサポートしています。	NAT での PF は、IPv4 ネットワーク上でのみ機能します。
ゾーンのプロビジョニングはありません。	PF は Oracle Solaris ゾーン内やゾーン間で機能します。	非大域ゾーンが PF を使用できます。 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ゾーンでは、ゾーン間のフィルタリングがサポートされています。
管理者は pf.conf ファイルを直接編集します。	管理者は pfconf スクリプトを使用して pf.conf ファイルを編集します。	スクリプトが構文を検証し、ファイアウォールを更新します。

詳細は、[48 ページの「Oracle Solaris でパケットフィルタを使用するためのガイドライン」](#) および [第5章「パケットフィルタファイアウォールの構成」](#)を参照してください。

Oracle Solaris でパケットフィルタを使用するためのガイドライン

PF を使用する場合は、次のガイドラインを確認してください。

- PF ファイアウォールをインストールして使用するには、70 ページの「[Oracle Solaris で PF ファイアウォールを構成する方法](#)」を参照してください。
solaris-small-server、solaris-large-server、および solaris-desktop グループパッケージは、デフォルトで IP フィルタサービスをインストールします。
- `svcadm enable firewall` などの SMF コマンドを使用して PF を管理します。 `pfctl` コマンドをいつ使用するかについては、67 ページの「[IPF 規則を PF 規則に変換するための方針](#)」を参照してください。
SMF の概要については、『[Oracle Solaris 11.3 でのシステムサービスの管理](#)』の第 1 章、「サービス管理機能の概要」を参照してください。SMF の手順については、『[Oracle Solaris 11.3 でのシステムサービスの管理](#)』の第 3 章、「サービスの管理」を参照してください。
- PF を管理するには、Network Firewall Management 権利プロファイルが割り当てられた管理者になる必要があります。root 役割にはこのプロファイルが含まれています。
ベストプラクティスは、作成したユーザーまたは役割に、Network Firewall Management 権利プロファイルを割り当てることです。役割を作成してその役割をユーザーに割り当てるには、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「役割の作成」を参照してください。
- `pf.conf` PF 構成ファイルを編集するには、Oracle Solaris から `pfconf` シェルスクリプトを使用します。このスクリプトは、`pf.conf` ファイルをエディタで開き、エディタを閉じる前に構文を検証し、エディタから `firewall` サービスをリフレッシュします。
- マクロおよびテーブルを使用して、規則の構文を簡略化し、パフォーマンスを向上させます。詳細は、59 ページの「[パケットフィルタのマクロとテーブル](#)」を参照してください。

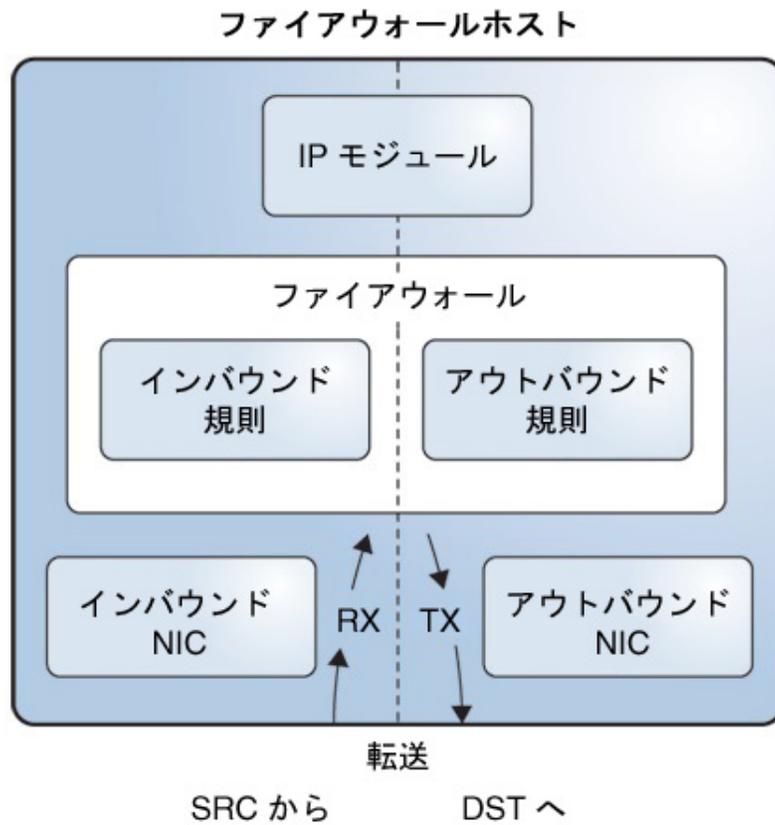
パケットフィルタファイアウォールとパケット処理

このセクションでは、パケットがファイアウォールホストに到着し、ネットワークデバイスと IP モジュールの間に位置するファイアウォールによって処理される方法について説明します。[図3](#)は、ネットワークデバイスとホストの IP スタックの間で転送されるすべてのパケットをファイアウォールモジュールがどのようにして検査できるかを示しています。[図4](#)は、ファイアウォールモジュールの内部でパケットがどのように処理されるかを示しています。

Oracle Solaris でのパケットフィルタファイアウォールモジュール

次の図は、ファイアウォールホストがパケットをリモートソースからリモート宛先にどのように転送するかを示しています。転送は、この図のいちばん下に示されています。

図 3 OpenBSD パケットファイアウォール



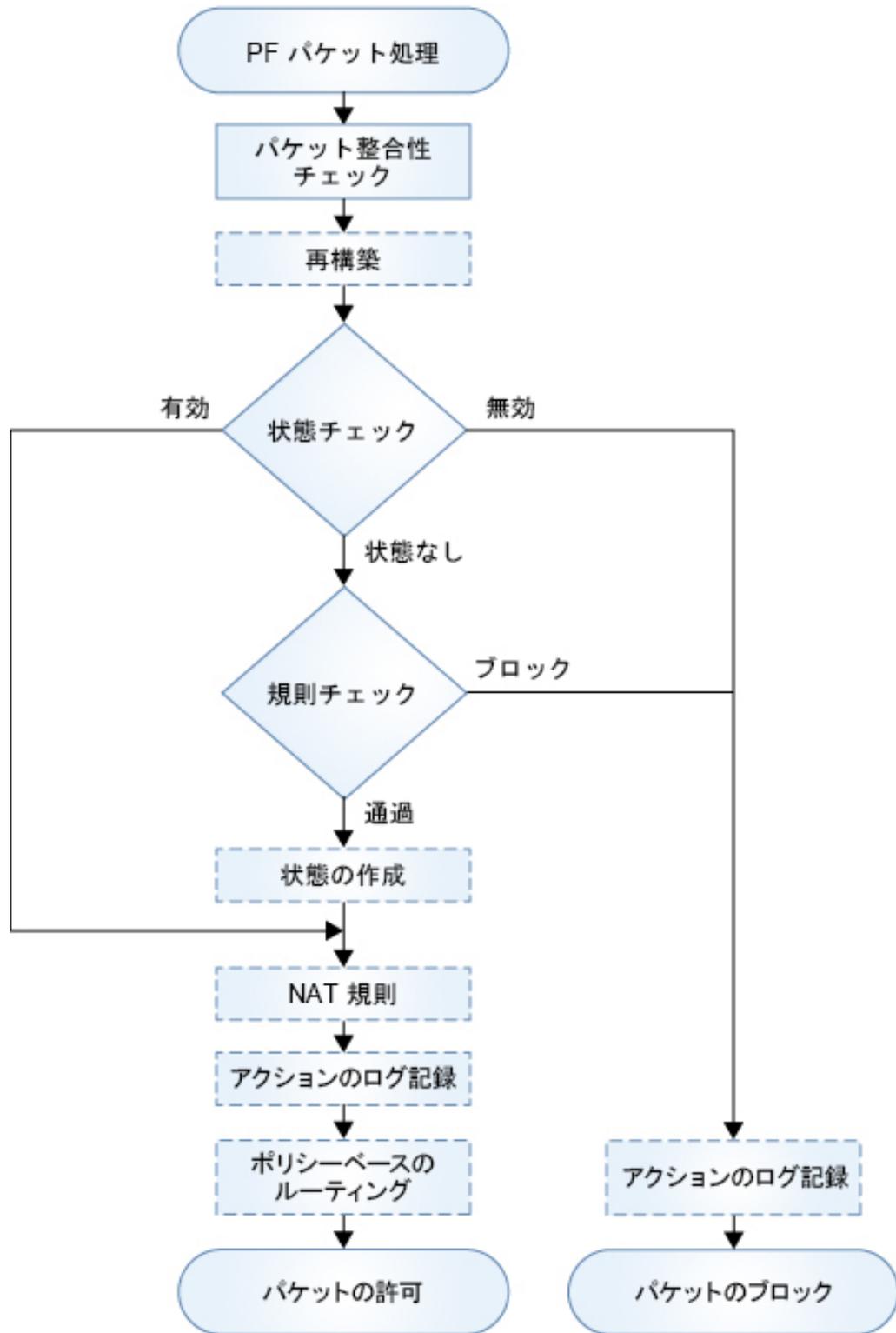
左側は受信側 (RX)、つまりインバウンドパスです。右側は送信側 (TX) です。パケットは、インバウンド NIC からホストに入力されます。インバウンドパケットが IP モジュールに向けて移動するとき、ファイアウォールはそれをインターセプトし、イ

ンバウンド規則を適用します。インバウンドポリシーがそのパケットを受け入れた場合、ファイアウォールはそのインバウンドパケットを IP モジュールに送信します。

IP モジュールは、IP ヘッダー内の宛先アドレスを使用してルートを見つけます。一致するルートが見つかった場合、IP モジュールはそのパケット (TX) を送信します。そのパケットがアウトバウンド NIC に向けて転送されるとき、ファイアウォールはそれを再度インターセプトします。このとき、ファイアウォールはそのパケットにアウトバウンド規則を適用します。アウトバウンドポリシーがそのパケットを受け入れた場合、ファイアウォールはそれをアウトバウンド NIC に送信し、それがパケットをリモート宛先に送信します。

PF でのパケット処理

図4は PF ファイアウォールモジュールによるパケット検査プロセスを示しています。破線のボックスは、管理オプションを示します。PF 規則には、IP パケットのフラグメントの再構築、NAT 規則の処理、アクションのログ記録、および状態の作成のためのオプションを含めることができます。



パケットがファイアウォールに入力されるとすぐに、ファイアウォールは、基本的なパケット整合性チェックを実行します。また、管理者が `set reassemble yes` ディレクティブを指定している場合は、フラグメントの完全なパケットへの再構築も行います。これらの整合性チェックにより、無効なフラグが設定されたパケットは破棄されます。次に、ファイアウォールは、そのパケットを既存の状態に一致させようとし

ます。

- 状態は見つかりましたが、そのパケットに対して無効です。たとえば、TCP パケットのシーケンス番号が予測された範囲内がない可能性があります。
- 状態は見つかり、有効であるため、そのパケットは受け入れられます。
- 状態が見つかりません。その場合、ファイアウォールは、一致する規則を見つけようとします (規則のチェック)。

状態が見つからない最後のケースの場合、ファイアウォールは、一致する規則を見つけるために規則のリストを検索します。一致する規則が見つからなかった場合、ファイアウォールは、それ以上のアクションなしでそのパケットを受け入れます。

注記 - 図4では、「それ以上のアクションなし」のケースを示していません。

規則が見つかった場合は、その規則に `block` または `pass` のどちらかのポリシーアクションが付属しています。デフォルトでは、PF の `pass` 規則は、一致するパケットに関する状態を作成します。状態の作成を回避するには、特定の規則で `no state` オプションを使用します。

パケットが `block` 規則に一致した場合、そのパケットは破棄されます。 `block` と `pass` のどちらのアクションも、パケットをログに記録するためのオプションの `log` アクションを受け付けます。

`pass` 規則では、そのパケットに対するオプションの NAT アクションやルーティングアクションが追加される可能性があります。

- NAT アクションは、発信元アドレス、宛先アドレス、またはその両方のアドレスを変更します。
- ルーティングアクション (ポリシーベースのルーティング (PBR) と呼ばれます) は、パケットをその宛先に転送するためのアウトバウンド NIC を選択します。PBR は、IP モジュールがパケットをルーティングするために使用する 2 つのパラメータ (送信インタフェースと次のホップアドレス) をオーバーライドできます。それ以外の場合は、IP モジュールがルーティングに関するすべての決定を行います。

PBR は、IP モジュールでのルーティングより柔軟です。IP モジュールのルーティングテーブルは、宛先アドレスのみを使用して、パケットの転送にどのネットワークインタフェースを使用するかを決定します。ファイアウォールは、パケットをどのネットワークインタフェースで転送するかを決定するときに、パケットヘッダー内のほぼすべてのフィールドを使用できます。

図3に示すように、受け入れられたパケットがファイアウォールを離れると、インバウンドパケットが IP モジュールに転送され、アウトバウンドパケットが対応する NIC (TX) に転送されます。

パケットフィルタの構成ファイル

PF は、すべてのファイアウォール構成情報のために `pf.conf` ファイルを使用します。 `svc:/network/firewall` サービスは、 `firewall/rules` サービスプロパティに指定されているファイルの場所から規則をロードします。デフォルトでは、このファイルの場所は `/etc/firewall/pf.conf` です。サービスプロパティ値を一覧表示するには、71 ページの「Oracle Solaris で PF ファイアウォールをモニターする方法」を参照してください。

このファイルを編集するには、 `pfconf` スクリプトを使用します。このファイルを保存すると、このスクリプトはファイアウォール規則の構文を検証したあと、ファイアウォールサービスをリフレッシュして新しい規則を有効にします。サンプルの規則については、60 ページの「IPF 規則と比較した PF 規則の例」、59 ページの「パケットフィルタのマクロとテーブル」、および72 ページの「PF 構成ファイルの例」を参照してください。詳細は、 `pfctl(1M)` のマニュアルページを参照してください。

パケットフィルタ規則の構文

PF 規則の構文は一見、IPF の構文に似ています。

```
action match-parameter optional-action-1 optional-action-2 ...
```

ただし、同一の構文の結果が非常に異なる場合があります。IPF 規則は、容易に PF 規則に変換されません。たとえば、次の規則セットは PF と IPF で有効です。ただし、IPF ではクライアント `192.168.1.2` が Secure Shell を使用して `172.16.1.2` に接続できるのに対して、PF ではできません。この解釈の違いは、61 ページの「状態の一致での PF と IPF の違い」で説明されています。

```
block in from 172.16.1.2 to 192.168.1.2
block in from 192.168.1.2 to 172.16.1.2
pass in proto tcp from 192.168.1.2 to 172.16.1.2 port = 22 keep state
```

注記 - デフォルトでは、構成ファイル内のど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。

PF での規則は、アクション、一致パラメータ、およびオプションのアクションを使用して、パケットを処理し、そのパケットが受け入れられるか、または破棄されるかを

判定します。パケットが規則に一致した場合、PFはそのパケットにアクションを適用します。規則は、次の要素を順番に使用して記述します。

1. その規則をアクションで開始します。リストについては、55 ページの「パケットフィルタ規則のアクション」を参照してください。
2. 必要なパラメータを一致させます。リストについては、56 ページの「パケットフィルタ規則の一致パラメータ」を参照してください。
3. 必要なオプションのアクションを含めます。リストについては、57 ページの「パケットフィルタ規則のオプションのアクション」を参照してください。

PF 規則の詳細な文法と構文については、`pf.conf(5)` のマニュアルページを参照してください。IPF 規則と PF 規則の間でポリシーまたは構文が異なる規則の例については、60 ページの「IPF 規則と比較した PF 規則の例」を参照してください。

パケットフィルタ規則のアクション

各規則がアクションを開始します。パケットが規則に一致した場合、PFはそのパケットにアクションを適用します。次のリストには、パケットに適用される一般的に使用されているアクションが含まれています。また、そのアクションが IP フィルタにもあったかどうかを示されています。

set	PF にのみあります。さまざまな PF ファイアウォールパラメータを構成します。たとえば、 <code>reassemble yes</code> 引数は、IP フラグメントの再構築を有効にします。 <code>set reassemble yes</code>
anchor	PF にのみあります。一致するパケットにさらに適用されるべき新しい規則セットを開きます。
block	パケットはフィルタを通過できません。 <code>## without an in, out, or on match parameter, ## blocks all traffic on all interfaces block all</code>
pass	パケットはフィルタを通過します。すべてのパケット (着信および送信) に適用されます。 <code>pass all</code>
match	パケットの <code>block</code> または <code>pass</code> 状態を変更することなく、きめ細かいフィルタリングを提供します。

`match` 規則は、最後に一致した規則についてだけでなく、パケットが規則に一致するたびにパラメータが設定されるという点で、`block` 規則や `pass` 規則とは異なります。これらのステッキパラメータは、明示的にオーバーライドされるまで有効です。影響を受けるパラメータは、`nat-to`、`binat-to`、`rdr-to`、および `scrub` です。

パケットフィルタ規則の一致パラメータ

このリスト内のキーワードは、パケットが規則に一致するかどうかを判定する条件を定義します。

<code>in</code>	規則がインバウンドパケットに一致した場合にのみ、その規則に適用されます。 <code>pass in from any to any port = 22</code>
<code>out</code>	規則がアウトバウンドパケットに一致した場合にのみ、その規則に適用されます。 <code>pass out log on net0</code>
<code>on interface-name</code>	指定されたインタフェースを出入りするパケットを一致させます。 <code>set skip on lo0</code>
<code>from ip-address</code>	規則が、指定された発信元 IP アドレスからのパケットに一致した場合にのみ、その規則に適用されます。次の例は、整形式の規則では <code>in</code> 、 <code>out</code> 、または <code>on</code> を使用する必要がないことを示しています。この規則は、ホストのすべてのインタフェース上のすべてのインバウンドおよびアウトバウンドパケットに一致します。 <code>pass from any to any</code> <code>any</code> 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け入れるために使用されます。IP アドレスは、ポート番号で変更できます。次の規則の例は、ポート 22 に到着するすべてのインバウンドパケットに一致します。 <code>pass in from any to any port = 22</code>
<code>flags a/b any</code>	設定されている TCP フラグに基づいて TCP パケットを一致させます。 この規則は、 <code>b</code> のセットのうち <code>a</code> のセットのフラグが設定されている TCP パケットにのみ適用されます。 <code>b</code> にないフラグは無視されます。

これらのフラグは、(F)IN、(S)YN、(R)ST、(P)USH、(A)CK、(U)RG、(E)CE、および C(W)R です。

次の例では、いくつかのサンプルのフラグ設定について説明します。

- `flags S/SA` – ステートフル接続のデフォルト設定。
- `flags any` – どのフラグもチェックされされません。
- `flags S/SS` – SYN が設定されます。その他のフラグは無視されます。

SYN と ACK をともにパケットに一致させることはできません。SYN+PSH、SYN+RST、および SYN をパケットに一致させることができますが、SYN+ACK、ACK+RST、および ACK はできません。

- `flags /SFRA-a` のセットが指定されていない場合は、デフォルトで「なし」になります。このフィルタを使用する前に、すべての `a` フラグを設定解除する必要があります。

さらに詳細な使用方法や、フラグ設定の影響については、[pf.conf\(5\)](#) のマニュアルページを参照してください。

<code>icmp-type</code>	ICMP タイプに基づいてパケットを一致させます。このキーワードは <code>proto</code> オプションが <code>icmp</code> に設定されているときに使用され、 <code>flags</code> オプションが指定されているときは使用されません。
<code>proto</code>	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 <code>tcp/udp</code> は、TCP または UDP パケットを一致させます。
<code>tos</code>	16 進数または 10 進数の整数として表されたサービスタイプの値に基づいてパケットを一致させます。
<code>ttl</code>	生存期間の値を基に、パケットの一致を取ります。パケットに格納されている生存期間の値は、そのパケットが破棄される前に移動できるホップの数を示します。
<code>group</code>	実効グループ ID で着信パケットを一致させます。
<code>user</code>	実効ユーザー ID で着信パケットを一致させます。

パケットフィルタ規則のオプションのアクション

このリスト内のキーワードは、追加のオプションのアクションを定義します。

allow-opts	<p>pass 規則に対して指定されると、パケットに IP オプションまたはルーティング拡張ヘッダーが含まれている場合でも、最後に一致した規則に基づいてそのパケットのフィルタの通過を許可します。allow-opts が指定されていない場合、PF は、IP オプションを含む IPv4 パケットおよびルーティング拡張ヘッダーを含む IPv6 パケットをブロックします。</p>
keep keep-options	<p>keep state の省略形。特定の規則に一致するパケットのために保持される情報を決定します。PF での pass 規則はデフォルトで状態を作成するため、keep オプションは、保持される状態のためのオプション (sloppy や if-bound など) をさらに指定するために役立ちます。</p>
nat-to	<p>パケットが特定のインタフェース上を移動したときに IP アドレスを変更するように指定します。この手法により、変換しているホスト上の 1 つまたは複数の IP アドレスで、内部ネットワーク上のより大きな範囲のシステムのネットワークトラフィックをサポートできます。内部ネットワークのアドレスは、RFC 1918: プライベートインターネットのためのアドレス割り当て で定義されているアドレス範囲に準拠しているべきです。</p>
rdr-to	<p>パケットを別の宛先の、異なっている可能性のあるポートにリダイレクトします。rdr-to は、1 つのポートだけでなくポート範囲を指定できます。</p>
log	<p>一致するパケットをログに記録します。log が match アクションで使用されていると、状態が保持されない場合でも、match でパケットがただちにログに記録されます。一致する match 規則が複数ある場合は、パケットが複数回ログに記録されます。</p>
quick	<p>パケットが規則に一致する場合は、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。</p>
route-to	<p>一致するパケットを指定されたインタフェース上のアウトバウンドキューに移動します。さらに、route-to は、指定されたインタフェースを経由して、各パケットを異なるインタフェースまたは異なる次のホップに送信できます。ポリシーベースのルーティング (PBR) を実装します。</p> <p>route-to は、次の引数を受け入れます。(インタフェース名 + 次のホップアドレス) のペアには 2 つの構文バリエーションがあります。</p> <ul style="list-style-type: none"> ■ <i>interface-name</i> ■ (<i>interface-name next-hop-address</i>) または <i>next-hop-address@interface-name</i>

- { (interface1-name next-hop1-address), (interface2-name next-hop2-address) [, ...] } または { next-hop1-address@interface1-name, next-hop2-address@interface2-name [, ...] }

複数のインタフェースを指定するときに使用するインタフェースを選択する方法については、<http://www.openbsd.org/faq/pf/pools.html> を参照してください。

パケットフィルタのマクロとテーブル

PF には、PF 規則の読みやすさ、拡張性、および再利用性を向上させるためのマクロとテーブルが用意されています。マクロとテーブルは、リストを受け入れます。

- マクロ – 1 つまたは複数の項目を 1 つの項目として処理されるように定義します。理解しやすい名前を指定します。たとえば、次の規則は、2 つの理解しやすいマクロを使用しています。

```
emailserver = 192.1.2.223
email = "{ smtp, pop3, imap }" ## mail services
pass in from any to $emailserver port = $email
```

- テーブル – 規則セット全体をリロードすることなく操作できる IP アドレスのリスト。高速な検索を促進します。

テーブルは、規則または規則のセットによって定義されます。あるテーブルを参照するすべての規則が削除された場合、そのテーブルは破棄されます。その規則が削除されても存続するテーブルを作成するには、`persist` キーワードを使用します。

たとえば、次の `clients` テーブルはクライアントのリストです。クライアントの範囲内の 1 つのアドレスは許可されません。

```
table <clients> persist { 192.168.2.0/24, !192.168.2.5 }
```

例 2 PF での NAT 規則

この例は、NAT 規則を構築する方法を示しています。この規則は、アウトバウンドパケット内の発信元アドレスを `net2` インタフェースにバインドされたアドレスに変換します。これにより、`192.168.1.0/24` の発信元アドレスを持つ `net2` デバイスから発信されるパケットが書き換えられ、その発信元アドレスが外部に対して `net2` として示されます。

```
## NAT rule that externally shows net2 as source address
pass out on net2 from 192.168.1.0/24 to any nat-to (net2)
```

例 3 PF でのスパム規則

この例は、PF でのテーブルの使用を示しています。管理者が、迷惑メールを送信する IP アドレスの spam テーブルを作成します。次のファイアウォール規則は、spam テーブル内のすべてのアドレスからの着信パケットをブロックします。

```
table <spam> { 192.168.1.0/24 }
block in from <spam> to any
```

新しい迷惑メール発信元をブロックするために、管理者は、このテーブルのみを更新します。

```
# pfctl -t spam -T add 172.16.0.0/16
```

pfctl コマンドは、規則セットを変更することなくファイアウォール構成を更新します。カーネルで、このテーブルは次のように読み取られます。

```
table <spam> { 192.168.1.0/24 172.16.0.0/16 }
```

PF 規則の詳細な文法と構文については、[pf.conf\(5\)](#) のマニュアルページを参照してください。ipadm コマンドの引数については、[ipadm\(1M\)](#) のマニュアルページを参照してください。

IPF 規則と比較した PF 規則の例

PF と IPF では、デフォルトのフィルタリング動作は異なりますが、同様の構文を使用します。PF 構成ファイル内の変更されていない IPF 規則によって、間違っただセキュリティポリシーが実装される可能性があります。次の例は、相違点のいくつかを示しています。IPF 規則と PF 規則の違いをテストする方法については、[67 ページの「IPF 規則を PF 規則に変換するための方針」](#)を参照してください。

PF ではループバックインタフェースのフィルタリングがデフォルトでオン

IPF とは異なり、PF は、デフォルトではループバックインタフェースにバインドされたパケットを処理します。pass all などの単純な規則であっても、PF が有効になっていると、ループバックトラフィックのパフォーマンスが大幅に低下する場合があります。次の 2 つの機能の衝突がパフォーマンス低下の原因です。

- PF での pass 規則のデフォルトであるステートフルパケット検査
- Oracle Solaris に固有の機能である [TCP Fusion](#)

TCP Fusion は、パケット処理を簡略化することによって、ループバックにバインドされた接続のスループットを向上させます。ただし、この簡略化により、状態の検証の

ために PF が使用する TCP シーケンスの番号付けが壊れます。TCP Fusion で処理されたパケットのシーケンス番号は無効であるため、PF はそのパケットを破棄します。TCP はこれらのパケットを再送信しますが、それらのパケットは引き続き状態の検証に失敗するため、スループットが低下します。

PF 規則でループバックポリシーを実装するには、次の点を明確にする必要があります。ポリシーでループバックパケットを処理しない場合は、ループバックパケットを処理するデフォルトをオーバーライドする必要があります。

```
## PF version - no filtering of packets on loopback interface
set skip on lo0
```

ポリシーでループバックパケットを処理する場合は、2つのオプションがあります。

- ループバックインタフェースのパケットを処理する pass 規則の状態を保持しない。

```
## PF version - no state on loopback interface
pass in on lo0 from any to any no state
```

- ループバックインタフェースのパケットのステートフル検査を実行する場合は、TCP シーケンス番号を無視する。

```
## PF version - no TCP sequence number inspection on loopback interface
pass in on lo0 from any to any keep state (sloppy)
```

状態の一致での PF と IPF の違い

PF と IPF は、異なる方法で特定のパケットを状態に一致させます。192.168.1.0/24 および 172.16.0.0/16 ネットワーク間でトラフィックを転送するルーター上の次の規則セットを考えてみます。

```
## Valid rule set in IPF and PF

block in from 192.168.1.0/24 to any
block in from 172.16.0.0/16 to any
pass in from 192.168.1.2 to 172.16.1.2 keep state
```

これらの規則は両方のファイアウォールに対して有効ですが、これらは異なるポリシーを実装します。

- IPF では、192.168.1.2 クライアントは 172.16.1.2 サーバーに到達でき、サーバーの応答パケットはこのクライアントに到達できます。
 1. block 規則は、192.168.1.0/24 および 172.16.0.0/16 ネットワーク間のすべてのトラフィックを阻止します。
 2. pass 規則は、192.168.1.2 クライアントが 172.16.1.2 サーバーに接続することを許可します。
 3. pass 規則内の keep state アクションは、サーバーの応答がこのクライアントに到達することを許可します。

192.168.1.2 から送信された最初の要求パケットが `pass` 規則に一致するとすぐに、状態が作成されます: 192.168.1.2 -> 172.16.1.2。この状態はまた、サーバーからクライアントに返された逆方向のパケットにも一致します: 172.16.1.2 -> 192.168.1.2。

- PF 規則の処理では、192.168.1.2 クライアントは 172.16.1.2 サーバーに接続できません。
1. PF での状態の検査はより厳格です。pass 規則の `in` 一致パラメータは、PF にインバウンドパケットのみを一致させるよう指示します。そのため、pass in 規則によって作成される状態は、次の 2 種類のパケットにのみ一致します。
 - クライアントからサーバーに送信されたインバウンド転送パケット:
192.168.1.2 -> 172.16.1.2
 - サーバーからクライアントに送信された逆方向のアウトバウンドパケット:
172.16.1.2 -> 192.168.1.2
 2. サーバーから PF ファイアウォールに応答が到着したとき、PF はそのパケットを逆方向のパケットではなく、はじめてのインバウンドパケットとして認識するため、このパケットは `pass in` 規則によって作成される状態に一致しません。規則の処理は、引き続きそのパケットに一致する規則を探して、そのパケットを転送するか、または破棄するかを判定します。このパケットは 2 番目の `block` 規則である `block in from 172.16.0.0/16 to any` に一致するため、PF はサーバーから送信された応答を破棄します。

PF で、IPF の場合と同様にルーティングされるトラフィックに対しては、次の 2 つの規則の構文が考えられます。

- 転送アウトバウンドパケットに関する状態を作成する規則を追加します。この状態により、サーバーからの逆方向のインバウンドパケットを有効にできます。

```
## PF rule set1 enforces IPF rule set policy

block in from 192.168.1.0/24 to any
block in from 172.16.0.0/16 to any
pass in from 192.168.1.2 to 172.16.1.2 keep state
pass out from 192.168.1.2 to 172.16.1.2 keep state
```

`pass` 規則は、次の 2 つの状態を作成します。

```
192.168.1.2 -> 172.16.1.2 @ in
192.168.1.2 -> 172.16.1.2 @ out
```

これらの状態は、サーバーからの逆方向のインバウンドパケットを許可します。

```
in: 172.16.1.2 -> 192.168.1.2
```

- パケットの方向を使用しない、より単純な規則セットを記述します。

```
## PF rule set2 enforces IPF rule set policy
```

```
block in from 192.168.1.0/24 to any
block in from 172.16.0.0/16 to any
pass from 192.168.1.2 to 172.16.1.2 keep state
```

in パラメータのない pass 規則は、方向には関係なく、クライアントから送信されたパケットに一致します。そのため、192.168.1.2 からのパケットをはじめのインバウンドパケットとしてインターセプトしたとき、PF は状態を作成します。

```
192.168.1.2 -> 172.16.1.2 @ in
```

そのパケットは、ファイアウォールによってルーティングされたあと、PF によって再度、今回はアウトバウンドパケットとしてインターセプトされます。そのパケットが pass 規則に一致すると、PF は 2 番目の状態を作成します。

```
192.168.1.2 -> 172.16.1.2 @ out
```

PF でのネットワークアドレス変換

PF の OpenBSD バージョンが [RFC 6146](#) で説明されている NAT-64 をサポートしているのに対して、PF の Oracle Solaris バージョンは、[RFC 2663](#) で説明されている従来の NAT のみをサポートしています。

NAT は通常、プライベートアドレスの範囲を持つネットワークとインターネットの間にマッピングを設定します。NAT を使用すると、プライベートネットワーク内のホストはインターネット上の任意のホストと通信できます。標準的な配備では、このマッピングは 1 対多です。つまり、プライベートネットワーク内の多くのホストが、インターネットサーバーに接続するための 1 つのパブリック IP アドレスを共有します。

PF での NAT 処理は、オプションの規則アクションです。PF での NAT 規則は実質的に、ファイアウォールにアクションタイプ (rdr-to または nat-to) および一致するパケットの方向 (インバウンドまたはアウトバウンド) に応じて IP アドレスを変更するよう指示する状態を作成します。

PF では、nat-to および rdr-to アクションは、パケットが nat-to または rdr-to のオプションのアクションを含む規則に一致するとすぐに実行されます。以降の規則には、すでに変換されたパケットが表示されます。また、どちらのアクションでも、一致するパケットに関する状態が作成されます。

- **nat-to** – 通常、アウトバウンドパケットに使用されます。プライベートネットワーク内のネットワーククライアントがインターネットサーバーと通信できるようにします。nat-to は、規則内の追加のパラメータに従って、一致するパケット内の発信元アドレスとポートを上書きするようファイアウォールに指示します。

たとえば、次の規則は、パケット内の 192.168.1.0/24 のアドレス範囲にあるプライベート発信元アドレスを、net0 インタフェースにバインドされたパブリック IP アドレスに変更します。

```
pass out on net0 from 192.168.1.0/24 to any nat-to (net0)
```

- **rdr-to** – 通常、インバウンドパケットに使用されます。rdr-to のオプションのアクションは、インターネット上のクライアントが、プライベート IP アドレスを持つネットワーク内の PF ファイアウォールの背後にあるサーバーと通信できるようにするという点で、nat-to の反対です。rdr-to は、一致するパケット内の宛先アドレスを変更します。

たとえば、次の規則は、インバウンド SMTP トラフィックを、ファイアウォールの背後にあるプライベートネットワーク上の SMTP サーバーの IP アドレスである 172.16.1.2:2525 にリダイレクトします。この規則は、net0 にバインドされたアドレスと同じ宛先 IP アドレスを持ち、かつ宛先ポートが 25 である、net0 インタフェースに到着するすべての TCP インバウンドパケットに一致します。rdr-to アクションは、172.16.1.2:2525 を持つパケット内の宛先アドレスとポートを上書きします。

```
pass in on net0 inet proto tcp from any to (net0) port = 25 rdr-to 172.16.1.2 port 2525
```

rdr-to または nat-to アクションからの変更は、影響を受けるパケットが状態を作成する pass または block 規則に一致していないかぎり、以降の規則にただちに表示されます。その場合、PF を離れるパケットは、rdr-to または nat-to オプションには影響されません。log アクションは、引き続きそのパケットをログに記録します。

```
## Block packets going to 192.168.2.2
match out to 192.168.2.2 nat-to 192.168.2.1
block from 192.168.2.1
```

match および pass アクションを使用した規則の同等

pass または block 規則を必要とする match 規則のすべてのパラメータは、状態を作成します。時には、pass 規則は、match と pass 規則を合わせたものと同等になる場合があります。たとえば、次の PF 規則セットは同等です。

```
webserver = "192.168.2.7"
webports = "{ http, https }"
emailserver = "192.168.2.5"
email = "{ smtp, pop3, imap }"

### Rule set 1 - match action then pass action
match in on $ext_if proto tcp to $ext_if port $webports rdr-to $webserver
match in on $ext_if proto tcp to $ext_if port $email rdr-to $emailserver

pass proto tcp from any to $webserver port $webports
pass proto tcp from any to $emailserver port $email
pass proto tcp from $emailserver to any port smtp
```

```
### Rule set 2 - no match action, only pass action
pass in on $ext_if inet proto tcp to $ext_if port $webports rdr-to $webserver
pass in on $ext_if inet proto tcp to $ext_if port $email rdr-to $mailserver
pass on $int_if inet proto tcp to $webserver port $webports
pass on $int_if inet proto tcp to $mailserver port $email
```

パケットフィルタ規則の処理

PF は、「最後に一致した」ポリシーに従って規則を処理します。つまり、パケットに対するポリシー決定は、そのパケットに一致した最後の規則によって決定されます。このポリシーでは、規則の最適な順序として、最初を一般に適用可能な規則にし、規則セット内のあとの方をより詳細な一致パラメータにすることを推奨しています。

たとえば、次の規則セットを考えてみます。

```
block in from any to any
pass in from any to any port = 22
pass in from any to any port = 25
```

最初の規則は、すべてのインバウンドパケットをブロックします。続く規則は、宛先ポート番号を一致させます。

そのため、パケットがポート 22 に到着すると、PF はそれを次のように処理します。

1. ブロックの規則を適用します。
2. ポート 22 の規則を適用します。これが、そのパケットに一致します。PF は、その一致をメモリー内に保持します。
3. ポート 25 の規則を適用します。これは、そのパケットに一致しません。

PF は、そのパケットをすべての規則に対してチェックしたあと、ポート 22 の規則が成功したためにそのパケットを転送します。

`quick` キーワードは例外です。規則に `quick` キーワードが含まれている場合、その規則のアクションがただちに実行され、それ以上の規則はそのパケットに適用されません。

```
block in from any to any
pass in quick from any to any port = 22
pass in from any to any port = 25
```

ファイアウォールを介して同じパケットを通し、この規則セットを使用した場合、PF は 2 番目の規則が適用されるとただちにそのパケットに対して `pass` アクションを実行します。`quick` キーワードは、そのパケットのそれ以上のチェックを停止するため、3 番目の規則は適用されません。

詳細は、[pfctl\(1M\)](#) および [pf.conf\(5\)](#) のマニュアルページを参照してください。

パケットフィルタのリファレンス

次の PF のマニュアルページは、Oracle Solaris のために記述または変更されていません。

<code>pf(5)</code>	PF サービスについて説明します。
<code>pfctl(1M)</code>	PF 規則を管理したり、チューニング可能パラメータを表示したり、その他のタスクを実行したりします。
<code>pf.conf(5)</code>	パケットフィルタリング規則セットを格納します。デフォルトでは、ループバックトラフィックを許可する規則を提供します。
<code>pf.os(5)</code>	OS シグニチャー情報を格納するオペレーティングシステム (OS) フィンガープリントデータベース。

次のリファレンスは、追加の説明、例、チュートリアル、およびマニュアルページを提供します。

- Hansteen, Peter N.M. [The Book of PF, 3rd Edition](#). No Starch Press. ISBN 978-1-59327-589-1、2014 年。
- [PF: The OpenBSD Packet Filter \(http://www.openbsd.org/faq/pf/index.html\)](http://www.openbsd.org/faq/pf/index.html)
- [OpenBSD のマニュアルページ検索 \(http://man.openbsd.org/\)](http://man.openbsd.org/)
- [Oracle Solaris での PF ファイアウォール \(https://blogs.oracle.com/solarisfw/\)](https://blogs.oracle.com/solarisfw/)

パケットフィルタファイアウォールの構成

この章では、OpenBSD パケットフィルタ (PF) ソフトウェアを使用してファイアウォールを実装するための手順について説明します。概要については、[第4章「Oracle Solaris での OpenBSD パケットフィルタファイアウォール」](#)を参照してください。

この章で扱う内容は、次のとおりです。

- 67 ページの「IPF 規則を PF 規則に変換するための方針」
- 68 ページの「Oracle Solaris ファイアウォールを実行する準備」
- 69 ページの「Oracle Solaris でのパケットフィルタサービスの実行およびモニタリング」
- 72 ページの「PF 構成ファイルの例」

IPF 規則を PF 規則に変換するための方針

PF 規則に移行する IPF 規則がある場合は、どのようなポリシーを適用しようとしているかを明確にし、そのポリシーが反映された PF 規則を作成する必要があります。

いくつかの PF 機能は、規則の一部をテストしたり、規則の構文や実行の順序を表示したり、その他の役立つ移行タスクを実行したりするために役立ちます。

- PF では規則を異なるファイルに配置できますが、この配置はデフォルトではありません。これを行うには、PF 構成ファイルに `INCLUDE` 文を追加します。たとえば、`pfconf` スクリプトを実行し、`INCLUDE` 文を挿入します。

```
# pfconf
...
include "/etc/firewall/pfnat.conf"
```

- `pfctl` コマンドのオプションを使用して、ファイアウォールポリシーをテストしたり表示したりできます。
 - メインの規則セットを表示するには、`-a` および `-sr` オプションを使用します。

```
$ pfctl -a '*' -sr
```

- 規則をカーネルにロードすることなく規則ファイルの構文をチェックするには、`-n` オプションを使用します。たとえば、次のコマンドは、`/etc/firewall/test` ディレクトリにある `pf.conf` ファイル内の規則の構文をチェックします。

```
$ pfctl -n -f /etc/firewall/test/pf.conf
```

- デバッグレベルを設定するには、`-x` オプションを使用します。デフォルトのデバッグレベルは `error` です。

```
# pfctl -x debug
# dmesg
```

デバッグメッセージはコンソールにのみ出力されます。`dmseg` コマンドは、システムバッファ内の最新の診断メッセージを検索し、それを標準出力に出力します。

- デバッグレベルを設定したあと、デバッグに役立つ出力を含めるには、`-g` オプションを使用します。

```
$ pfctl -x debug -g -f testfile -n
```

- 詳細出力を表示するには、`-v` および `-vv` オプションを使用します。

```
$ pfctl -vv
```

- 状態を表示するとき、状態に対して DNS 逆引き検索を実行するには `-r` オプションを使用します。

その他のオプションについては、[pfctl\(1M\)](#) のマニュアルページを参照してください。

PF 構成ファイルに変更された IP フィルタ 構成ファイル全体の例については、[例4「IP フィルタ 構成ファイルに基づいた PF 構成ファイル」](#)を参照してください。

Oracle Solaris ファイアウォールを実行する準備

`firewall` パッケージ内の `pf.conf` ファイルは、構文的に無効なファイルです。Oracle Solaris では、このファイルを提供し、これを有効なファイルと置き換えるようファイアウォール管理者に勧めます。ただし、無効なファイルが有効な状態でファイアウォールサービスを有効にした場合、`firewall` サービスが機能低下状態になり、基本的な保護規則セットをロードします。この規則セットでは、独自のサイトのファイアウォールポリシーファイルを作成する時間があります。同様に、無効な `pf.conf` ファイルを使用してシステムをリブートした場合、Oracle Solaris は基本的な保護規則セットをロードします。

基本的な規則セットは、次のとおりです。

```
## ignore loopback traffic
```

```
set skip on lo0
set reassemble yes no-df
## block everything unless told otherwise
block
## accept incoming SSH connections
pass in proto tcp to any port 22
## allow incoming messages from DHCP
pass in inet proto udp from port 67 to port 68
pass in inet6 proto udp from port 547 to port 546
## packet too big - needed for PMTUD
pass in inet6 proto ipv6-icmp icmp6-type 2
## router advertisement
pass in inet6 proto ipv6-icmp icmp6-type 134
## neighbor solicitation
pass in inet6 proto ipv6-icmp icmp6-type 135
## neighbor advertisement
pass in inet6 proto ipv6-icmp icmp6-type 136
## allow all connections initiated from this system,
## this includes DHCP requests
pass out
```

この基本的な構成では、Secure Shell を使用してリモートでファイアウォール構成を変更できます。さらに、この構成では、一部のネットワークトラフィック許可し、ループバックトラフィックを無視します。

Oracle Solaris でのパケットフィルタサービスの実行およびモニタリング

Oracle Solaris でのパケットフィルタは、多くの Oracle Solaris 機能を利用しています。

- PF は、IPS リポジトリからインストールするパッケージです。
インストール後、`/etc/firewall` ディレクトリには2つのファイルが含まれています。ファイアウォール規則のための `pf.conf` ファイルは空です。`pf.os` ファイルは、リモートオペレーティングシステムのためのパッシブオペレーティングシステム (OS) フィンガープリンティングを提供します。これは、レイヤー3およびレイヤー4ヘッダー内のデータからフィンガープリントを決定します。
このパッケージにはまた、ファイアウォール規則を編集するための `pfconf` ヘルパースクリプトも用意されています。このスクリプトは、規則をエディタにロードします。エディタを終了すると、`pfconf` は規則の構文を検証し、`firewall` サービスを新しい規則でリフレッシュします。
- PF は、SMF `firewall` サービスとして実行されます。
ほかのすべての SMF サービスと同様に、このサービスは、`svcadm` コマンドを使用して無効および有効にします。ファイルの場所および権利は、このサービスのプロパティです。
- Network Firewall Management 権利プロファイルは、管理者をファイアウォールの構成およびモニタリングに必要な権利のみに制限します。

この権利プロファイルは、ファイアウォールの管理に責任を負う信頼できるユーザーに割り当てることができます。

- Oracle Solaris での PF では `pfctl` コマンドを使用できますが、基本的なファイアウォール管理には `svcadm` コマンドと `pfconf` スクリプトを使用してください。 `pfctl` コマンドは、ファイアウォールの統計を収集したり、状態を表示したり、アドレステーブルを管理したりするために役立ちます。

▼ Oracle Solaris で PF ファイアウォールを構成する方法

PF をファイアウォールとして実行するには、`firewall` パッケージをインストールし、ポリシーが反映されるようにファイアウォールを構成したあと、`firewall` サービスを有効にします。PF の `firewall` パッケージは、デフォルトでインストールされ、有効になっている IP フィルタの `ipfilter` パッケージと共存できます。ただし、Oracle Solaris イメージ上で同時に有効にできるファイアウォールは 1 つだけです。

始める前に `firewall` パッケージをインストールするには、ソフトウェアインストールに関連する権利プロファイルが割り当てられた管理者になる必要があります。`firewall` サービスを構成するには、Network Firewall Management 権利プロファイルが割り当てられた管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。

1. PF パッケージをインストールします。

```
$ pkg install firewall
```

2. `pfconf` スクリプトを使用して、パケットフィルタリング規則セットを作成または更新します。

```
# pfconf
```

`pfconf` スクリプトは、PF 構成ファイルの場所としてサービスプロパティ `rules` を使用します。

サンプル規則については、59 ページの「パケットフィルタのマクロとテーブル」および 60 ページの「IPF 規則と比較した PF 規則の例」を参照してください。

3. 最初に `ipfilter` サービスを無効にしてから、PF を有効にします。

```
# svcadm disable network/ipfilter  
# svcadm enable network/firewall
```

注記 - PF 構成ファイルを構成しない場合は、PF がファイルをインストールし、firewall サービスを degraded 状態にします。詳細は、68 ページの「[Oracle Solaris ファイアウォールを実行する準備](#)」を参照してください。

4. (オプション) この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/firewall
```

このコマンドはカーネルからすべての規則を削除し、このサービスを無効にします。

▼ Oracle Solaris で PF ファイアウォールをモニターする方法

モニタリングには、ファイアウォールサービスのプロパティーの表示や、実行中の規則または可能性のある規則セットとしての規則の表示が含まれます。

始める前に Network Firewall Management 権利プロファイルが割り当てられた管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. firewall サービスのステータスを検査します。

- パケットフィルタサービスが有効になっているかどうかを判定します。

```
$ svcs -x firewall:default
svc:/network/firewall:default (Network Firewall)
State: disabled since Fri Apr 10 10:10:50 2015
Reason: Disabled by an administrator.
See: http://oracle.com/msg/SMF-8000-05
See: pf.conf(5)
See: /var/svc/log/network-firewall:default.log
Impact: This service is not running.
```

デフォルトでは、ipfilter ファイアウォールサービスは実行されています。

- パケットフィルタサービスの構成ファイルの名前と場所を一覧表示します。

```
$ svccfg -s firewall:default listprop | grep firewall
firewall          application
firewall/fingerprints  astring      /etc/firewall/pf.os
firewall/rules      astring      /etc/firewall/pf.conf
firewall/value_authorization  astring      solaris.smf.value.network.firewall
restarter/logfile   astring      /var/svc/log/network-firewall:default.log
```

2. ファイアウォール規則を検査します。

- ファイアウォール構成内の現在の規則を検査します。

次の例は、カーネルにロードされた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。

```
# pfctl -s rules
empty list for firewall(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
```

- 可能性のあるファイアウォール構成が構文的に正しいことを確認します。

次のコマンドは、規則ファイルの構文をチェックします。このチェックでは、規則はカーネルにロードされません。

```
# pfctl -n -f /test/firewall/pf.conf
```

PF 構成ファイルの例

このセクションの例では、PF 規則と規則セットを示します。

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

構文情報の詳細は、54 ページの「パケットフィルタ規則の構文」および `pf.conf(5)` のマニュアルページを参照してください。

例 4 IP フィルタ 構成ファイルに基づいた PF 構成ファイル

次に、IP フィルタ 構成ファイルを示します。

```
## IP Filter configuration file
# Anything in that we initiate is okay
pass out quick all keep state keep frags
# Drop all NETBIOS traffic but don't log it.
block in quick from any to any port = 137 #netbios-ns
block in quick from any to any port = 138 #netbios-dgm
block in quick from any to any port = 139 #netbios-ssn
# Allow incoming IKE/IPsec
pass in quick proto udp from any to any port = ike
pass in quick proto udp from any to any port = 4500
pass in quick proto esp from any to any
# Allow ping and ICMP destination unreachable / fragmentation needed
pass in quick proto icmp from any to any icmp-type echo
pass in quick proto icmp from any to any icmp-type 3
```

```
# Allow routing info
# pass in quick proto udp from any to port = route
# pass in quick proto icmp from any to any icmp-type 9 # routeradvert
# pass in quick proto igmp from any to any
# Allow incoming SSH
pass in quick proto tcp from any to any port = ssh
# Allow DHCP - required
pass in quick proto udp from any to port = bootpc
# Block and log everything else that comes in
block in log all
block in from any to 255.255.255.255
block in from any to 127.0.0.1/32
```

PF 構成ファイルは次のとおりです。

```
## PF configuration file with identical policy
## to preceding IP Filter configuration file
anchor "_auto/*"
## anchor "_static/*"
set skip on lo0
## reassemble no allows an attacker to fragment
## packets in a smart way and bypass the firewall.
## yes no-df allows reassembly and clears the
## dont-fragment bit.
set reassemble yes no-df
block log
## Allow incoming SSH
pass in proto tcp to any port 22
## equals: pass in proto tcp from any to any port = 22 flags S/SA
## Allow DHCP
pass in inet proto udp from port 67 to port 68
pass in inet6 proto udp from port 547 to port 546
## Allow NAT traversal, hence _auto above
pass in inet proto udp from any to any port 500
pass in inet6 proto udp from any to any port 4500
## Allow incoming IKE/IPsec
pass in inet proto esp from any to any
pass in inet6 proto esp from any to any
## Too-big packets must be allowed by firewall.
## However, the later 'pass out' rule allows all
## inbound responses, including to 'icmp6-type 2'
## messages. So, remove IPF 'pass in' rule in PF.
## pass in inet6 proto ipv6-icmp icmp6-type 2
## Allow routing info
pass in inet6 proto ipv6-icmp icmp6-type 134
pass in inet6 proto ipv6-icmp icmp6-type 135
pass in inet6 proto ipv6-icmp icmp6-type 136
# Anything in that we initiate is okay
pass out
```


Oracle Solaris での IP フィルタファイアウォール

この章では、Oracle Solaris の IP フィルタ機能の概要を説明します。IP フィルタのタスクについては、[第7章「IP フィルタファイアウォールの構成」](#)を参照してください。

この章では、次の内容について説明します。

- [75 ページの「IP フィルタとは」](#)
- [76 ページの「IP フィルタのパケット処理」](#)
- [79 ページの「IP フィルタの使用ガイドライン」](#)
- [79 ページの「IP フィルタの構成ファイルの使用」](#)
- [80 ページの「IP フィルタの規則セットの使用」](#)
- [86 ページの「IP フィルタ用の IPv6」](#)
- [87 ページの「IP フィルタのマニュアルページ」](#)

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。

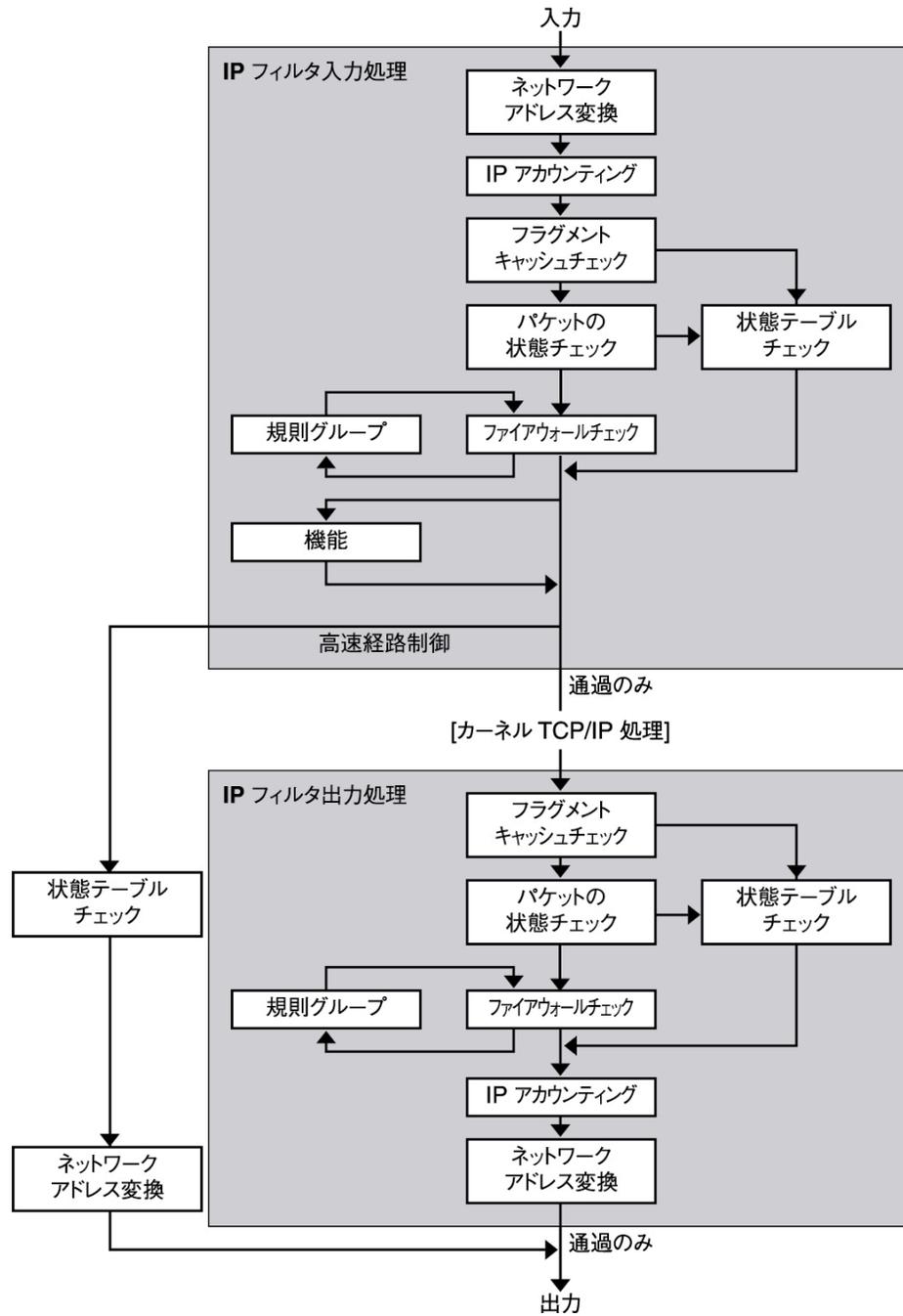
IP フィルタは、オープンソースの IP フィルタソフトウェアから派生したものです。オープンソースの IP フィルタのライセンス契約の条件、作者、および著作権宣言文を参照するためのデフォルトパスは、`/usr/lib/ipf/IPFILTER.LICENCE` です。Oracle

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。

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 5 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` のどちらかが指定されているかに応じて自動的に通されるかブロックされます。

- **ファイアウォールチェック**

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

- **グループ**

グループを使用すると、ツリー形式で規則セットを作成できます。

- **機能**

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

- **高速経路制御**

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

- **IP 認証**

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

IP フィルタの使用ガイドライン

- IP フィルタは SMF サービス `svc:/network/ipfilter` によって管理されます。SMF の完全な概要については、『[Oracle Solaris 11.3 でのシステムサービスの管理](#)』の第 1 章、「サービス管理機能の概要」を参照してください。SMF に関連するステップごとの手順については、『[Oracle Solaris 11.3 でのシステムサービスの管理](#)』の第 3 章、「サービスの管理」を参照してください。
- IP フィルタでは、構成ファイルを直接編集する必要があります。
- IP フィルタは Oracle Solaris の一部としてインストールされます。システムが自動ネットワーク接続を使用するように構成されている場合、デフォルトで、IP フィルタサービスは有効にされています。自動ネットワークプロファイルは、`nwam(5)` および `netadm(1M)` のマニュアルページに説明するように、このファイアウォールを有効にします。自動的にネットワーク接続されるシステムでのカスタム構成の場合、IP フィルタサービスは有効にされません。このサービスを有効にするための関連タスクについては、[89 ページの「IP フィルタサービスの構成」](#)を参照してください。
- IP フィルタを管理するには、`root` 役割になるか、IP Filter Management 権利プロファイルが割り当てられている必要があります。IP Filter Management 権利プロファイルは、自分が作成したユーザーまたは役割に割り当てることができます。役割を作成してその役割をユーザーに割り当てするには、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「役割の作成」を参照してください。
- Oracle Solaris クラスタソフトウェアは、スケーラブルサービス用の IP フィルタによるフィルタリングはサポートしませんが、フェイルオーバーサービス用の IP フィルタはサポートします。IP フィルタをクラスタ内で構成するときのガイドラインおよび制限については、『[Oracle Solaris Cluster ソフトウェアのインストール](#)』の「Oracle Solaris OS の機能制限」を参照してください。
- ゾーン間のフィルタリングは、IP フィルタ規則が実装されているゾーンが、システム上のほかのゾーンの仮想ルーターとして機能する場合にかぎりサポートされません。

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、[114 ページの「IP フィルタの構成ファイルの例」](#)を参照してください。詳細については、`svc.ipfd(1M)` のマニュアルページを参照してください。

IP フィルタの規則セットの使用

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

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

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

IP フィルタのパケットのフィルタリング機能の使用

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

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

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

IP フィルタは、構成された規則リストの最初から最後まで規則を処理してから、パケットの通過またはブロックを行います。IP フィルタは、パケットを通過させるかどうかを決めるフラグを維持管理しています。フラグは、規則セット全体を調べ、最後に一致した規則を基にパケットを通過させるか、ブロックするかを決定します。

このプロセスには、2つの例外があります。最初の例外は、パケットが quick キーワードを含む規則に一致した場合です。規則が quick キーワードを含む場合は、その規則に対する処理が実行され、それ以降の規則はチェックされません。2番目の例外

は、パケットが `group` キーワードを含む規則に一致した場合です。パケットがグループに一致すると、グループでタグ付けされた規則だけがチェックされます。

パケットのフィルタリング規則の構成

パケットのフィルタリング規則を作成するには、次の構文を使用します。

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

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

<code>block</code>	パケットはフィルタを通過できません。
<code>pass</code>	パケットはフィルタを通過します。
<code>log</code>	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、 <code>ipmon</code> コマンドを使用します。
<code>count</code>	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、 <code>ipfstat</code> コマンドを使用します。
<code>skip number</code>	フィルタは <code>number</code> フィルタリング規則をスキップします。
<code>auth</code>	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、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。

<code>log</code>	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、 <code>ipmon</code> コマンドを使用します。
<code>quick</code>	パケットが一致した場合、 <code>quick</code> 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。
<code>on interface-name</code>	パケットが指定したインターフェースを出入りする場合だけ、規則を適用します。

`dup-to interface-name` パケットをコピーし、`interface-name` 上の複製をオプションで指定した IP アドレスに送信します。

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

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

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

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

`tos` 16 進数または 10 進数の整数で表されたサービスタイプの値を基に、パケットをフィルタリングします。

`tTL` 生存期間の値を基に、パケットの一致を取ります。パケットに保存されている生存期間の値は、破棄される前にパケットがネットワーク上に存在できる期間を示します。

`proto` 特定のプロトコルに対して一致を取ります。/etc/protocols ファイルに指定されている任意のプロトコル名を使用したり、そのプロトコルを表す 10 進数の数を指定したりできます。キーワード `tcp/udp` は、TCP または UDP パケットとの一致を取るために使用できます。

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

<code>with</code>	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せん。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 <code>not</code> または <code>no</code> と記述します。
<code>flags</code>	設定されている TCP フラグを基にフィルタリングを行う TCP で使用します。TCP フラグの詳細については、 ipf(4) のマニュアルページを参照してください。
<code>icmp-type</code>	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 <code>proto</code> オプションが <code>icmp</code> に設定されているときに使用され、 <code>flags</code> オプションが指定されているときは使用されません。
<code>keep keep-options</code>	保存しておくパケットの情報を決定します。使用可能な <code>keep-options</code> には <code>state</code> オプションが含まれます。 <code>state</code> 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。
<code>head number</code>	番号 <code>number</code> 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。
<code>group number</code>	デフォルトグループではなく、グループ番号 <code>number</code> 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、[96 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`) については、『[Oracle Solaris 11.3 でのネットワーク配備の計画](#)』の第 1 章、「[ネットワーク配備の計画](#)」を参照してください。

IP フィルタの NAT 機能の使用

NAT は、発信元 IP アドレスと宛先 IP アドレスをほかのインターネットアドレスまたはイントラネットアドレスに変換するマッピング規則を設定します。これらの規則

は、受信 IP パケットまたは発信 IP パケットの発信元アドレスおよび宛先アドレスを変更し、パケットを送信します。また、NAT を使用して、あるポートから別のポートにトラフィックの方向を変更することもできます。NAT は、パケットに修正または方向の変更が行われても、パケットの完全性を維持します。

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

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

NAT 規則の構成

次の構文で NAT 規則を作成します。

command interface-name parameters

1. 各規則の冒頭には、次のコマンドのいずれかが記述されています。

<code>map</code>	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。
<code>rdr</code>	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。
<code>bimap</code>	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。
<code>map-block</code>	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。

<code>ipmask</code>	ネットワークマスクを指定します。
<code>dstipmask</code>	<code>ipmask</code> が変換されるアドレスを指定します。

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

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

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

次の規則は IPv6 アドレスに適用されます。

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

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

IP フィルタのアドレスプール機能の使用

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

アドレスプール構成規則は、IP フィルタサービスによって読み込まれるファイル内に置くことができます。ファイルを作成して、そのパス名をサービスの `config/ippool_config_file` プロパティ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
```

<code>table</code>	複数のアドレスへの参照を定義します。
<code>role</code>	IP フィルタでのプールの役割を指定します。参照できる役割は <code>ipf</code> だけです。
<code>type</code>	プールの保存形式を指定します。
<code>number</code>	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 192.168.1.0 のグループをプール番号 13 で参照する場合、アドレスプールの構成ファイルに次の規則を含めます。

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

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

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

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

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

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

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

IP フィルタ用の IPv6

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

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

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

IPv6 に関連する IP フィルタのタスクは、IPv4 とほとんど変わりません。もっとも大きな違いは、特定のコマンドで -6 オプションを使用することです。ipf コマンドと ipfstat コマンドには、IPv6 パケットフィルタリングで使用するための -6 オプションが用意されています。IPv6 パケットフィルタリング規則をロードおよびフラッシュするときは、ipf コマンドで -6 オプションを使用します。IPv6 統計を表示するとき

は、`ipfstat` コマンドに `-6` オプションを使用します。`ipmon` コマンドと `ippool` コマンドでも IPv6 がサポートされますが、IPv6 をサポートするためのオプションは指定しません。`ipmon` コマンドは、IPv6 パケットのロギングに対応するように拡張されています。`ippool` コマンドでは、IPv6 アドレスのプールをサポートしています。IPv4 アドレス用と IPv6 アドレス用の個別のプールを作成することも、IPv4 アドレスと IPv6 アドレスの両方を格納するプールを作成することもできます。

再利用可能な IPv6 パケットフィルタリング規則を作成するには、特定の IPv6 ファイルを作成する必要があります。次に、そのパス名を IP フィルタサービスの `config/ip6_config_file` プロパティの値として設定します。デフォルト値は `/etc/ipf/ipf6.conf` です。

IP フィルタに関連するタスクについては、[第7章「IP フィルタファイアウォールの構成」](#)を参照してください。

IP フィルタのマニュアルページ

次のマニュアルページで IP フィルタについて説明しています。

ipf(1M)	IP フィルタ規則を管理し、チューニング可能パラメータを表示し、他のタスクを実行します。
ipf(4)	IP フィルタパケットのフィルタリング規則を作成するための文法と構文を含む。
ipfilter(5)	IP フィルタソフトウェアについて説明します。
ipfs(1M)	リブート後も、NAT 情報と状態テーブル情報を保存し、復元します。
ipfstat(1M)	パケット処理の統計情報を取得して表示します。
ipmon(1M)	ログデバイスを開き、パケットフィルタリングと NAT の両方について記録されたパケットを表示します。
ipnat(1M)	NAT 規則を管理し、NAT 統計情報を表示します。
ipnat(4)	NAT 規則を作成するための文法と構文を含む
ippool(1M)	アドレスプールを作成し、管理します。
ippool(4)	IP フィルタアドレスプールを作成するための文法と構文を含む。

[svc.ipfd\(1M\)](#) IP フィルタサービスの構成に関する情報を提供します。

IP フィルタファイアウォールの構成

この章では、IP フィルタのタスクの手順をステップごとに説明します。概要情報については、[第6章「Oracle Solaris での IP フィルタファイアウォール」](#)を参照してください。

この章で扱う内容は、次のとおりです。

- [89 ページの「IP フィルタサービスの構成」](#)
- [95 ページの「IP フィルタ規則セットの操作」](#)
- [107 ページの「IP フィルタの統計および情報の表示」](#)
- [110 ページの「IP フィルタ用ログファイルの操作」](#)
- [114 ページの「IP フィルタの構成ファイルの例」](#)

IP フィルタサービスの構成

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

表 5 IP フィルタサービスの構成のタスクマップ

タスク	参照先
IP フィルタが使用するファイルおよびサービスのステータスを表示します。	90 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。	91 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。	93 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。	93 ページの「パケット再構築を無効にする方法」
システム上のゾーン間のトラフィックをフィルタリングします。	94 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」

タスク	参照先
IP フィルタの使用を停止します。	95 ページの「パケットフィルタリングを無効にする方法」

▼ IP フィルタサービスのデフォルトを表示する方法

始める前に `ipfstat` コマンドを実行するには、IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. IP フィルタサービスの構成ファイル名と場所を表示します。

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

最初の3つのファイルプロパティには、デフォルトのファイルの場所があります。これらのファイルは作成するまで存在しません。構成ファイルの場所を変更する場合は、そのファイルのプロパティ値を変更する必要があります。手順については、[91 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を変更します。[91 ページの「IP フィルタ構成ファイルの作成方法」](#)のステップ1およびステップ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判定します。

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

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

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

```
# ipfstat -io
```

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

- ポリシーを変更するには、[91 ページの「IP フィルタ構成ファイルの作成方法」](#)を参照してください。

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

▼ IP フィルタ構成ファイルの作成方法

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

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. IP フィルタサービスのポリシーファイルの場所を指定します。

このファイルには、パケットフィルタリング規則セットが含まれます。

a. まず、ポリシーファイルを `custom` に設定します。

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

b. 次に、場所を指定します。

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

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

2. パケットフィルタリング規則セットを作成します。

パケットのフィルタリングについては、[80 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、[114 ページの「IP フィルタの構成ファイルの例」](#) および `/etc/nwam/loc/NoNet/ipf.conf` ファイルを参照してください。

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

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

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

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

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

NAT については、[83 ページの「IP フィルタの NAT 機能の使用」](#)を参照してください。

4. (オプション) アドレスプール構成ファイルを作成します。

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

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

アドレスプールには、IPv4 アドレスと IPv6 アドレスの任意の組み合わせを含めることができます。アドレスプールの詳細については、[85 ページの「IP フィルタのアドレスプール機能の使用」](#)を参照してください。

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

システムに構成されているゾーン間のトラフィックのフィルタリングを行う場合は、ループバックフィルタリングを有効にする必要があります。[94 ページの「ループバックフィルタリングを有効にする方法」](#)を参照してください。ゾーンに適用する規則セットも定義する必要があります。

6. (オプション) 断片化されたパケットの再構築を無効にします。

デフォルトで、フラグメントは、IP フィルタで再構築されます。デフォルトを変更するには、[93 ページの「パケット再構築を無効にする方法」](#)を参照してください。

▼ IP フィルタを有効にし、リフレッシュする方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

91 ページの「[IP フィルタ構成ファイルの作成方法](#)」を完了しています。

1. IP フィルタを有効にします。

最初に IP フィルタを有効にするには、次のコマンドを入力します。

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

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

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

注記 - refresh コマンドは一時的にファイアウォールを無効にします。ファイアウォールを保持するには、規則を追加するか、新しい構成ファイルを追加します。例と手順については、95 ページの「[IP フィルタ規則セットの操作](#)」を参照してください。

▼ パケット再構築を無効にする方法

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

始める前に IP Filter Management 権利プロファイルと `solaris.admin.edit/path-to-IPFilter-policy-file` 承認が割り当てられている管理者になる必要があります。root 役割には、これらの権利がすべて含まれています。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. IP フィルタを無効にします。

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

2. IP フィルタポリシーファイルの先頭に次の規則を追加します。

```
set defrag off;
```

次のように、`pfedit` コマンドを使用します。

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

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

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

3. IP フィルタを有効にします。

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

4. パケットが再構築中でないことを確認します。

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

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

▼ ループバックフィルタリングを有効にする方法

始める前に IP Filter Management 権利プロファイルと `solaris.admin.edit/path-to-IPFilter-policy-file` 承認が割り当てられている管理者になる必要があります。`root` 役割には、これらの権利がすべて含まれています。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. IP フィルタが実行中の場合は、停止させます。

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

2. IP フィルタポリシーファイルの先頭に次の規則を追加します。

```
set intercept_loopback true;
```

次のように、`pfedit` コマンドを使用します。

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

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

```
...
#set defrag off;
#
# Enable loopback filtering to filter between zones
#
```

```
set intercept_loopback true;
#
# Define policy
#
block in all
block out all
other rules
```

3. IP フィルタを有効にします。

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

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

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

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

▼ パケットフィルタリングを無効にする方法

この手順は、カーネルからすべての規則を削除し、サービスを無効にします。この手順を使用する場合、適切な構成ファイルで IP フィルタを有効にし、パケットフィルタリングと NAT を再起動する必要があります。詳細については、[93 ページの「IP フィルタを有効にし、リフレッシュする方法」](#)を参照してください。

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

- サービスを無効にするには、svcadm コマンドを使用します。

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

サービスをテストまたはデバッグするために、サービスの実行中に規則セットを削除できます。詳細については、[95 ページの「IP フィルタ規則セットの操作」](#)を参照してください。

IP フィルタ規則セットの操作

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

- テスト目的

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

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

表 6 IP フィルタルールセットの操作のタスクマップ

タスク	参照先
アクティブなパケットフィルタリング規則セットを表示します。	97 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	97 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	97 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	99 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	99 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」 100 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	101 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	102 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	103 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。	103 ページの「IP フィルタで NAT 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。	104 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	105 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	105 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。	106 ページの「規則をアドレスプールに追加する方法」

IP フィルタのパケットフィルタリング規則セットの管理

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

注記 - 次の手順に、IPv4 ネットワークの例を示します。IPv6 パケットの場合は、90 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」のステップ 2 の説明に従って、-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。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「割り当てられている管理権利の使用」を参照してください。

- アクティブなパケットフィルタリング規則セットを表示します。

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

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

▼ アクティブでないパケットフィルタリング規則セットを参照する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「割り当てられている管理権利の使用」を参照してください。

- アクティブでないパケットフィルタリング規則セットを参照します。

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

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

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

次のいずれかのタスクを実行する場合には、ここで示す手順を実行します。

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

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「割り当てられている管理権利の使用」を参照してください。

1. 次の手順から 1 つを選択します。

- まったく異なる規則セットをアクティブにする場合は、別個のファイルに新規規則セットを作成します。
- 構成ファイル内の現在の規則セットを更新します。

2. 現在の規則セットを削除し、新しい規則セットをロードします。

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

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

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

例 5 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 6 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。

必要に応じて、アクティブな規則セットを一覧表示します。

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

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

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

▼ パケットフィルタリング規則セットを削除する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「割り当てられている管理権利の使用」を参照してください。

- 規則セットを削除します。

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

-a 全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 7 パケットフィルタリング規則セットの削除

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

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

▼ アクティブなパケットフィルタリング規則セットに規則を追加する方法

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

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「割り当てられている管理権利の使用」を参照してください。

- 次のいずれかの方法で規則をアクティブな規則セットに追加します。

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

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

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

- 次のコマンドを実行します。
 1. 適当なファイルに規則セットを作成します。
 2. 作成しておいた規則をアクティブな規則セットに追加します。

```
# ipf -f filename
```

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

filename が、IP フィルタ構成ファイルのいずれかのプロパティの値である場合、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 8 アクティブなパケットフィルタリング規則セットへの規則の追加

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

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

▼ アクティブでないパケットフィルタリング規則セットに規則を追加する方法

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

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

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

```
# ipf -I -f filename
```

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

例 9 アクティブでない規則セットへの規則の追加

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

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

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

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

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

- アクティブな規則セットとアクティブでない規則セットを切り替えます。

```
# ipf -s
```

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

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

例 10 アクティブなパケットフィルタリング規則セットとアクティブでないパケットフィルタリング規則セットの切り替え

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

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

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

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

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

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

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

- 全削除コマンドで、アクティブでない規則セットを指定します。

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

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

例 11 カーネルからのアクティブでないパケットフィルタリング規則セットの削除

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

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

IP フィルタ用 NAT 規則の管理

次の手順で IP フィルタの NAT 規則を管理、表示、および変更します。

▼ IP フィルタでアクティブな NAT規則を表示する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

- **アクティブな NAT 規則を参照します。**

次の例は、アクティブな NAT 規則セットからの出力を示しています。

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

List of active sessions:
```

▼ IP フィルタで 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。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

- **カーネルから NAT 規則を削除します。**

```
# ipnat -FC
```

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

例 12 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:
# ipnat -C
1 entries flushed from NAT list
# ipnat -l
List of active MAP/Redirect filters:

List of active sessions:
```

▼ NAT パケットフィルタリング規則に規則を追加する方法

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

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

● 次のいずれかの方法で規則をアクティブな規則セットに追加します。

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

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

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

- 次のコマンドを実行します。
 1. 適当なファイルに追加の NAT 規則を作成します。
 2. 作成しておいた規則をアクティブな NAT 規則に追加します。

```
# ipnat -f filename
```

filename の規則がアクティブな NAT 規則の最後に追加されます。

filename が、IP フィルタ構成ファイルのいずれかのプロパティの値である場合、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 13 NAT 規則セットへの規則の追加

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

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

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

List of active sessions:
```

IP フィルタのアドレスプールの管理

次の手順でアドレスプールを管理、表示、および変更します。

▼ アクティブなアドレスプールを参照する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

- **アクティブなアドレスプールを参照します。**

次の例は、アクティブなアドレスプールの内容を参照する方法を示しています。

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

▼ アドレスプールを削除する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

- **現在のアドレスプールのエントリを削除します。**

```
# ippool -F
```

例 14 アドレスプールの削除

次の例は、アドレスプールを削除する方法を示しています。

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

▼ 規則をアドレスプールに追加する方法

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

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. 次のいずれかの方法で規則をアクティブな規則セットに追加します。

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

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

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

- 次のコマンドを実行します。

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

```
# ippool -f filename
```

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

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

- a. 新しいパケットフィルタリング規則にプールを追加します。

- b. 現在の規則セットに新しいパケットフィルタリング規則を追加します。
[99 ページの「アクティブなパケットフィルタリング規則セットに規則を追加する方法」](#)の指示に従います。

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

例 15 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 -1
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 -1
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 7 IP フィルタの統計および情報の表示のタスクマップ

タスク	参照先
状態テーブルを表示します。	108 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。	108 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。	109 ページの「IP フィルタのチューニング可能パラメータを表示する方法」
NAT 統計を表示します。	109 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	110 ページの「IP フィルタのアドレスプール統計情報を表示する方法」

▼ IP フィルタの状態テーブルを参照する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「割り当てられている管理権利の使用」を参照してください。

- 状態テーブルを参照します。

```
# ipfstat
```

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

例 16 IP フィルタの状態テーブルの参照

次の例に、状態テーブルの出力を示します。

```
# ipfstat
bad packets:          in 0    out 0
  IPv6 packets:      in 56286 out 63298
  input packets:     blocked 160 passed 11 nomatch 1 counted 0 short 0
output packets:      blocked 0 passed 13681 nomatch 6844 counted 0 short 0
  input packets logged: blocked 0 passed 0
  output packets logged: blocked 0 passed 0
  packets logged:    input 0 output 0
  log failures:      input 0 output 0
fragment state(in):  kept 0 lost 0 not fragmented 0
fragment reassembly(in): bad v6 hdr 0 bad v6 ehdr 0 failed reassembly 0
fragment state(out): kept 0 lost 0 not fragmented 0
packet state(in):    kept 0 lost 0
packet state(out):   kept 0 lost 0
ICMP replies:        0      TCP RSTs sent: 0
Invalid source(in):  0
Result cache hits(in): 152      (out): 6837
IN Pullups succeeded: 0      failed: 0
OUT Pullups succeeded: 0      failed: 0
Fastroute successes: 0      failures: 0
TCP cksum fails(in): 0      (out): 0
IPF Ticks:           14341469
Packet log flags set: (0)
                      none
```

▼ IP フィルタの状態統計を参照する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護』の「割り当てられている管理権利の使用」を参照してください。

- 状態統計を参照します。

```
# ipfstat -s
```

例 17 IP フィルタの状態統計の参照

次の例に、状態統計情報の出力を示します。

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

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

▼ IP フィルタのチューニング可能パラメータを表示する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。

- IP フィルタのカーネルチューニング可能パラメータを表示します。

次の出力は、切り詰められています。

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

▼ IP フィルタの NAT 統計を参照する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。

- NAT 統計を表示します。

```
# ipnat -s
```

例 18 IP フィルタの NAT 統計の参照

次の例に、NAT 統計情報の例を示します。

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

▼ IP フィルタのアドレスプール統計情報を表示する方法

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

- アドレスプール統計の参照

```
# ippool -s
```

例 19 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 8 IP フィルタログファイルの操作のタスクマップ

タスク	参照先
IP フィルタログファイルを別個に作成する	111 ページの「IP フィルタのログファイルを設定する方法」

タスク	参照先
状態、NAT、および通常のログファイルを表示します。	112 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の消去	113 ページの「パケットログバッファ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	114 ページの「ロギングされたパケットをファイルに保存する方法」

▼ IP フィルタのログファイルを設定する方法

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

始める前に `root` 役割になる必要があります。

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

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

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

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

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

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

3. 新規ログファイルを作成します。

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

4. `system-log` サービスの構成情報をリフレッシュします。

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

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

例 20 IP フィルタログの作成

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

syslog.conf を編集します。

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

次に、コマンド行でファイルを作成してサービスを再起動します。

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

▼ IP フィルタのログファイルを参照する方法

始める前に [111 ページの「IP フィルタのログファイルを設定する方法」](#) を完了しています。

IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

- **状態、NAT、または通常のログファイルを参照します。**

ログファイルを参照するには、適切なオプションと共に次のコマンドを入力してください。

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

S 状態ログファイルを表示します。

N NAT ログファイルを表示します。

I 通常の IP ログファイルを表示します。

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

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

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

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

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

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

例 21 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をフラッシュする方法

この手順では、バッファをクリアし、画面に出力を表示します。

始める前に IP Filter Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

● パケットログバッファの消去

```
# ipmon -F
```

例 22 パケットログバッファのフラッシュ

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

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

▼ ロギングされたパケットをファイルに保存する方法

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

始める前に root 役割になる必要があります。

- **ロギングされたパケットをファイルへ保存します。**

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

Control-C を入力して、コマンド行のプロンプトに戻って、このプロシーチ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 23 ファイルへのロギングされたパケットの保存

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

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

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

IP フィルタの構成ファイルの例

次の例に、単一のホスト、サーバー、ルーターに適用するパケットフィルタリング規則を示します。

構成ファイルは、次のような標準的な UNIX 構文規則に従っています。

- シャープ記号 (#) は、コメントを含む行を示します。
- 規則とコメントは、同一の行に共存できます。

- 規則を読みやすくするために、不要な空白を使用できます。
- 複数行に渡って規則を記述できます。行の最後のバックスラッシュ (\) は、ルールが次の行に続いていることを示します。

構文についての詳細については、[81 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 24 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システムの構成を示しています。

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

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

# block and log untrusted internal IPs. 0/32 is notation that replaces
# address of the system 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 番目のルールセットは、プライベートアドレス空間 10.0.0.0 および 172.16.0.0 からの受信パケットがファイアウォールの中に入るのをブロックします。次のルールセットは、ホストシステムからの特定の内部アドレスをブロックします。そして、最後の規則セットは、ポート 6000 およびポート 111 から受信されるパケットをブロックします。

例 25 IP フィルタのサーバー構成

この例は、Web サーバーとして機能するホストシステムの構成を示しています。このシステムには、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 \
to net0/32 port = 113 flags S keep state group 100
pass in 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 26 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 are not on the
# loopback interface should not exist.
block in log quick on net0 from 127.0.0.0/8 to any
block in log quick on net0 from any to 127.0.0.0/8
block in log quick on net1 from 127.0.0.0/8 to any
block in log quick on net1 from any to 127.0.0.0/8

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

IP セキュリティーアーキテクチャーについて

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

この章で扱う内容は、次のとおりです。

- 119 ページの「IPsec の概要」
- 121 ページの「IPsec パケットのフロー」
- 124 ページの「IPsec セキュリティーアソシエーション」
- 125 ページの「IPsec 保護プロトコル」
- 128 ページの「IPsec ポリシー」
- 129 ページの「IPsec のトランスポートモードとトンネルモード」
- 132 ページの「仮想プライベートネットワークと IPsec」
- 133 ページの「IPsec と FIPS 140-2」
- 133 ページの「IPsec と NAT 越え」
- 134 ページの「IPsec と SCTP」
- 135 ページの「IPsec と Oracle Solaris ゾーン」
- 135 ページの「IPsec と仮想マシン」
- 135 ページの「IPsec の構成コマンドとファイル」

ネットワークに IPsec を実装するには、[第9章「IPsec の構成」](#)を参照してください。参照情報については、[第14章「IPsec および鍵管理のリファレンス」](#)を参照してください。

IPsec の概要

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

IPsec では次の用語を使用します。

- **セキュリティプロトコル** – IP パケットに適用される保護。authentication header (AH) は、IP ヘッダーを含む完全なパケットのハッシュである整合性チェックベクトル (ICV) を追加することで IP パケットを保護します。受信者はパケットが変更されていないことを保証されます。暗号化による機密性の確保は行われません。
encapsulating security payload (ESP) は IP パケットのペイロードを保護します。パケットのペイロードは機密性を確保するために暗号化が可能で、ICV を使用してデータの整合性を保証できます。
- **セキュリティアソシエーション (SA)** – 特定の SA と特定のトラフィックフローを一致させるのに使用される、暗号化パラメータ、鍵、IP セキュリティプロトコル、IP アドレス、IP プロトコル、ポート番号などのパラメータ。
- **セキュリティアソシエーションデータベース (SADB)** – セキュリティアソシエーションを格納するデータベース。SA は、セキュリティパラメータインデックス (SPI)、セキュリティプロトコル、および宛先 IP アドレスから参照されます。これらの 3 つの要素は IPsec SA を一意に識別します。IPsec ヘッダー (ESP または AH) を持つ IP パケットをシステムが受け取ると、そのシステムは SADB で一致する SA を検索します。一致する SA が見つかったら、IPsec はその SA を使用してパケットを復号化および検証できます。検証に失敗した場合または一致する SA が見つからない場合は、パケットが破棄されます。
- **鍵管理** – 暗号化アルゴリズムによって使用される鍵のセキュアな生成および配布と、それらの格納に使用される SA の生成。
- **セキュリティポリシーデータベース (SPD)** – IP トラフィックに適用するセキュリティポリシーを指定するデータベース。SPD は、トラフィックをフィルタリングしてパケットの処理方法を決定します。パケットは、破棄するか、検証済みとして通過させることができます。または、パケットを IPsec で保護する、つまりセキュリティポリシーを適用することができます。
アウトバウンドパケットの場合、IPsec ポリシーは IP パケットに IPsec を適用するべきかどうかを決定します。IPsec を適用すると、IP モジュールが SADB で一致する SA を検索し、この SA を使用してポリシーを適用します。
インバウンドパケットの場合、IPsec ポリシーによって、受け取ったパケットの保護レベルが適切であることが確認されます。ポリシーが特定の IP アドレスからのパケットを IPsec で保護することを要求する場合、システムは保護されていないパケットをすべて破棄します。インバウンドパケットが IPsec によって保護されている場合、IP モジュールが SADB で一致する SA を検索し、その SA をパケットに適用します。

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

IPsec パケットのフロー

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

図7は、IPsec インバウンドプロセスを示しています。

図 6 アウトバウンドパケットプロセスに適用された IPsec

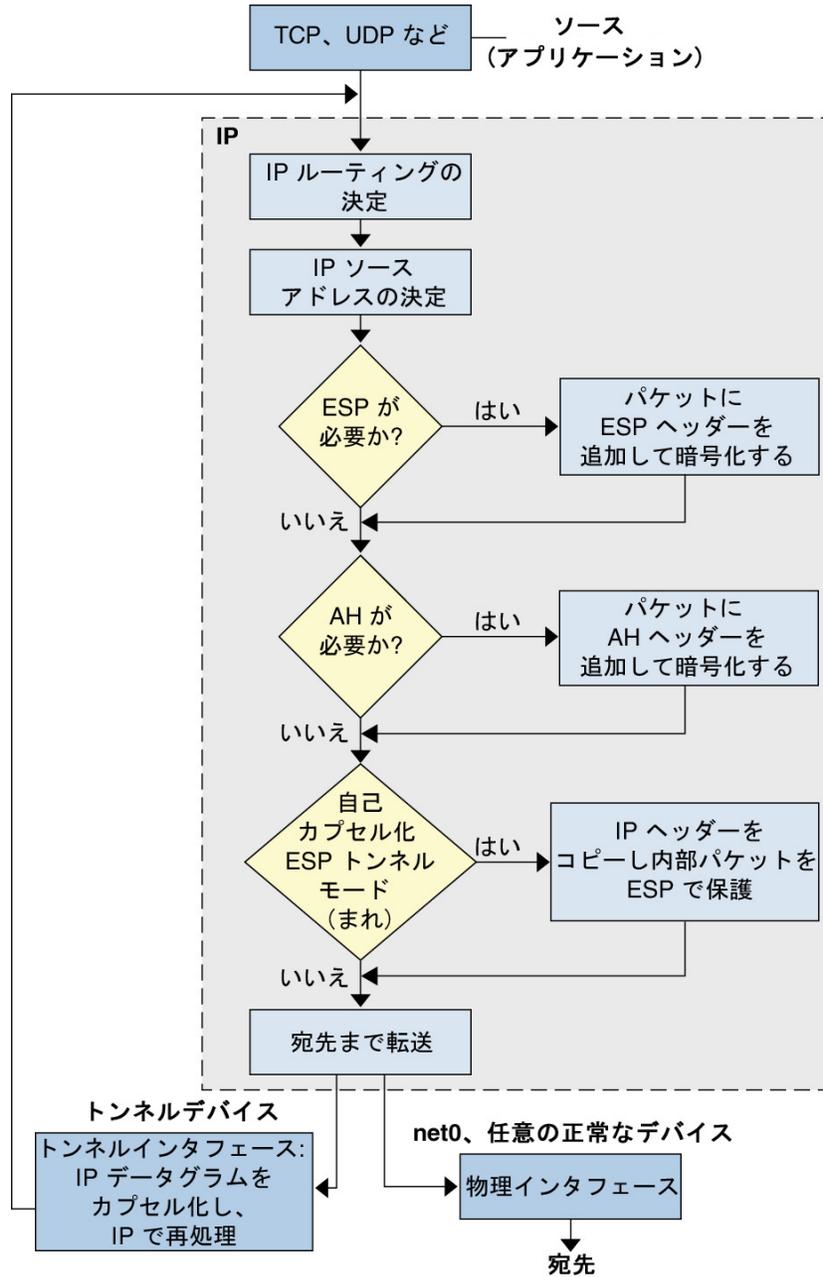
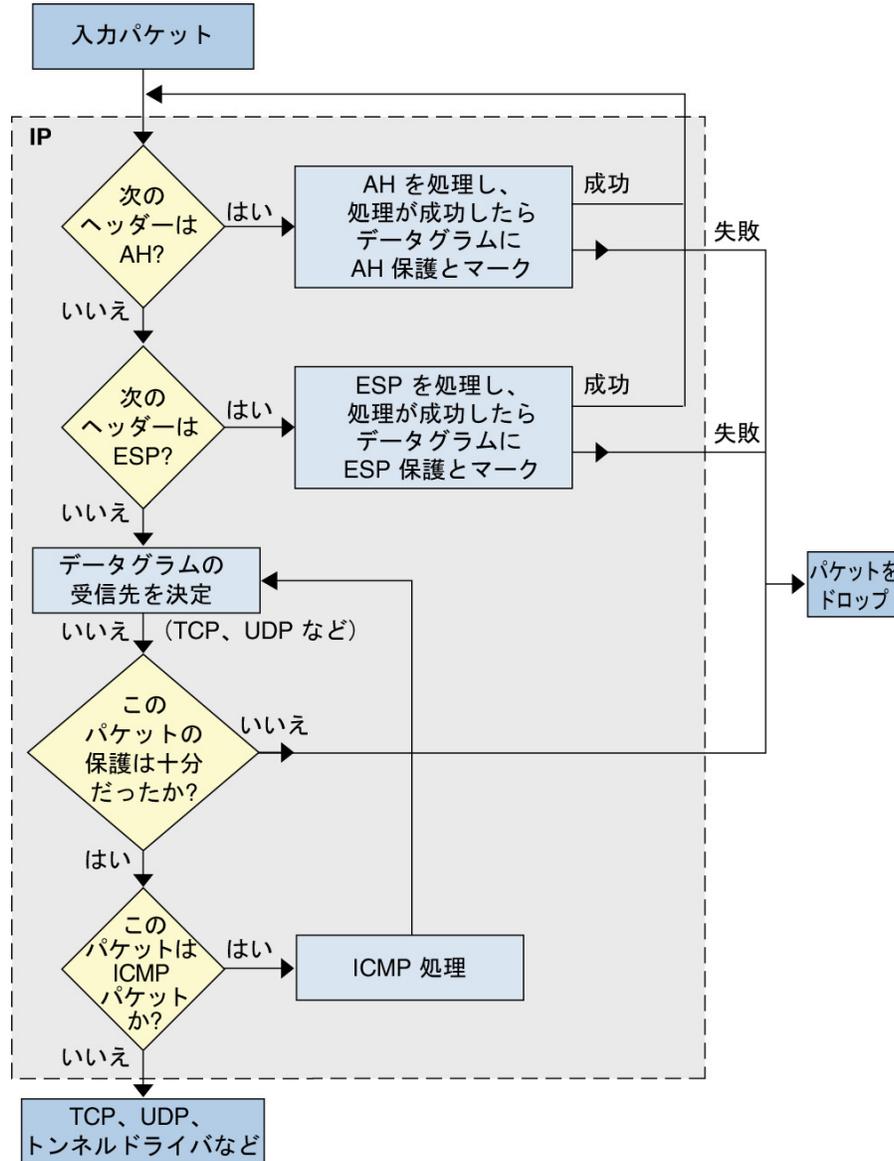


図 7 IPsec をインバウンドパケットプロセスに適用



IPsec セキュリティアソシエーション

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

あわせて、次の3つの要素が IPsec SA を一意に識別します。

- セキュリティープロトコル (AH または ESP)
- 宛先 IP アドレス
- security parameter index (SPI)

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

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

IPsec SADB のより完全な説明については、[256 ページの「IPsec のセキュリティアソシエーションデータベース」](#)を参照してください。

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

IPsec セキュリティアソシエーションの鍵管理

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

IPsec SA を生成するための IKE

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。Oracle Solaris のこのリリースは、IKE プロトコルの IKE バージョン 2 (IKEv2) および IKE バージョン 1 (IKEv1) をサポートしています。

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

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

詳細は、[165 ページの「IKE の動作」](#)を参照してください。

IKE を構成するには、[第11章「IKEv2 の構成」](#)を参照してください。IKEv2 プロトコルをサポートしていないシステムと通信する場合は、[第12章「IKEv1 の構成」](#)の手順に従います。

IPsec SA を生成するための手動鍵

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

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

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

IPsec 保護プロトコル

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

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

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

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

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

認証ヘッダー

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

IP ヘッダー	AH	TCP ヘッダー	
---------	----	----------	--

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。tunnel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

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

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

IP ヘッダー	ESP	TCP ヘッダー	
---------	-----	----------	--

■ 暗号化部分

TCP パケットでは、ESP ヘッダーが認証され、そのヘッダーが TCP ヘッダーおよびそのデータをカプセル化します。パケットが IP 内 IP パケットの場合、ESP は内部 IP パ

ケットを保護します。ソケット別ポリシーでは、「自己カプセル化」ができるため、ESP は必要に応じて IP オプションをカプセル化できます。

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

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

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

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

さらに詳しくは、[129 ページの「IPsec のトランスポートモードとトンネルモード」](#)を参照してください。

AH と ESP を使用する場合のセキュリティー上の考慮事項

次の表では、AH が提供する保護と ESP が提供する保護を比較します。

表 9 IPsec で AH と ESP が提供する保護

プロトコル	パケットの範囲	保護	対象となる攻撃
AH	パケットの IP ヘッダーからトランスポートデータの最後までを保護	強力な完全性およびデータ認証を提供します。 <ul style="list-style-type: none"> ■ 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	パケットの ESP ヘッダーからトランスポートデータの最後までを保護	暗号化オプションで、IP ペイロードを暗号化します。機密性を確保します 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 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	盗聴 リプレー、カットアンドペースト リプレー、カットアンドペースト、盗聴

IPsec の認証アルゴリズムと暗号化アルゴリズム

IPsec セキュリティーでは、認証と暗号化という 2 種類のアロリズムを使用します。AH プロトコルは認証アルゴリズムを使用します。ESP プロトコルは、暗号化アルゴリズムと認証アルゴリズムを使用できます。ipsecalg コマンド

ドを使用すると、システムのアルゴリズムとそれらのプロパティ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を検索することもできます。

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

詳細については、次を参照してください。

- 『Oracle Solaris 11.3 での暗号化と証明書の管理』 の第 1 章、「暗号化フレームワーク」
- 『Oracle Solaris 11 セキュリティ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 ポリシー

IPsec ポリシーはソケット単位レベルおよびシステム全体レベルで適用できます。

IPsec は、IPsec のポリシールールに一致するアウトバウンドパケットとインバウンドパケットにポリシーを適用します。各ポリシールールは 1 つ以上のアクションを持つことができます。アクションには、特定のアルゴリズムを使用してパケットを暗号化したり、暗号化せずにパケットを受け渡したりするものがあります。許容される複数のアルゴリズムを指定するために、ポリシールールで複数のアクションを使用することもあります。

IPsec ポリシールールには 3 つの部分があります。

- セレクタ - ルールがネットワークパケットに一致するかどうかを判別します。トラフィックセレクタとも呼ばれます。空白 ({}) セレクタはすべてのトラフィックに一致します。セレクタは、アクションとパラメータのペアを複数持つことができます。
- アクション - トラフィックがセレクタに一致したときに適用されます。アクションの例には `ipsec` および `pass` があります。
- アクションパラメータ - アクションの追加の指定。 `pass` や `drop` のような単純なアクションにパラメータはありません。 `ipsec` などのアクションは、暗号化パラメータを指定できます。

IPsec ポリシーはインバウンドとアウトバウンドの両方のパケットに適用されます。すべてのルールに一致しないパケットは受け渡されます。パケットが複数のルールに一致する場合は、最初に一致したものが使用されます。

ルールは下記の順序で処理されます。

1. ソケット単位のルール

2. システム全体の pass、bypass、および drop ルール
3. ESP を使用するシステム全体の ipsec ルール
4. AH を使用するシステム全体の ipsec ルール

bypass および or pass オプションによって、パケットに適用される IPsec ポリシーに対して、例外を指定できます。

- IPsec ルールのすべてまたは一部を bypass できます。バイパスルールに一致するパケットは IPsec 保護が適用されずに通過することが許可され、パケットに一致するほかの IPsec ポリシールールは適用されません。たとえば、Web クライアントからのパケットを暗号化する必要がなくなることがあります。144 ページの「IPsec を使用してほかのサーバーとの Web サーバー通信を保護する方法」を参照してください。
- IPsec ポリシールールの or pass アクションを使用すると、ルールの以前のアクションと一致する IPsec 以外のパケットをシステムに渡すことができます。encrypt アクションと or pass アクションを持つ IPsec ポリシールールは、暗号化されたパケット、およびクライアントシステムからの暗号化されていないパケットを受け入れます。

or pass アクションを使用すると、IPsec で構成されているクライアントに加えて、IPsec で構成されていないクライアントにサーバーが対応できます。1つの使用例としては、すべてのシステムに IPsec を構成するためにネットワークを移行している場合があります。このオプションは、すべてのトラフィックを暗号化する必要がある環境には適していません。例については、例30「サーバーの or pass アクションを使用した、IPsec を使用するクライアントシステムへの移行」を参照してください。

ipsecinit.conf ファイルと ipsecconf コマンドを使用して、IPsec ポリシーを構成します。詳細と例については、[ipsecconf\(1M\)](#) のマニュアルページと第9章「IPsec の構成」を参照してください。

IPsec のトランスポートモードとトンネルモード

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

128 ページの「IPsec ポリシー」で紹介されたトラフィックセクタは、パケットがポリシールールに一致するかどうかを判別します。セクタには次が含まれます。

- 発信元 IP アドレス

- 着信先 IP アドレス
- プロトコル番号 (該当する場合)
- ポート番号 (該当する場合)

IPsec ポリシールールを照合するパターンは、これらのセレクタのサブセットで構成されます。

トランスポートモードでは、トラフィックセレクタは外側の IP ヘッダーに対して照合されます。トンネルモードでは、内側の IP ヘッダーに対して照合されます。トンネルモードは、エンドシステムとセキュリティーゲートウェイのような中間システムとのあらゆる組み合わせに適用できます。

トランスポートモードでは、IP ヘッダー、次のヘッダー、および次のヘッダーでサポートされるすべてのポートを使用して、IPsec ポリシーを適用できるかどうかを決定できます。実際、IPsec は 2 つの IP アドレスの間で異なるトランスポートモードポリシーを適用でき、ポート単位まで細かく設定できます。たとえば、次のヘッダーが TCP であれば、ポートをサポートするので、外側の IP アドレスの TCP ポートに対して IPsec ポリシーを設定できます。

トンネルモードは、IP 内 IP パケットに対してのみ機能します。トンネルモードでは、IPsec ポリシーは内側の IP パケットの内容に適用されます。内側の IP アドレスごとに異なるポリシーを適用できます。つまり、内側の IP ヘッダー、その次のヘッダー、および次のヘッダーでサポートされるポートを使用して、ポリシーを適用できます。

トンネルモードでは、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。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、[146 ページの「IPsec による VPN の保護」](#)を参照してください。

IPsec ポリシーが IP トンネルのトラフィックに適用されるとき、そのトンネルのトラフィックを IPsec ポリシールールにリンクするために IP トンネルインタフェースの名前が使用されます。IPsec ポリシーには、IP トンネルネットワークインタフェースを選択するための `tunnel` キーワードが用意されています。規則内に `tunnel` キーワードが含まれている場合は、その規則に指定されているすべてのセレクタが内側のパケットに適用されます。

トンネルインタフェースについては、『[Oracle Solaris 11.3 での TCP/IP ネットワーク、IPMP、および IP トンネルの管理](#)』の第 4 章、「[IP トンネルの管理について](#)」を参照してください。

次の図は、保護されているパケットおよび保護されていないパケットを示しています。

図8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 8 TCP 情報を伝送する保護されていない IP パケット



図9は、トランスポートモードでデータを保護する ESP を示しています。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 9 TCP 情報を伝送する保護された IP パケット



■ 暗号化部分

図10は、トンネルモードではパケット全体が ESP ヘッダーの内側にあることを示しています。図8のパケットは、外側の IPsec ヘッダーと、この場合は ESP によって、トンネルモードで保護されます。

図 10 トンネルモードで保護された IPsec パケット



■ 暗号化部分

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

- ソケットごとのポリシーの詳細については、[ipsec\(7P\)](#)のマニュアルページを参照してください。
- ソケットごとのポリシーの例は、[144 ページの「IPsec を使用してほかのサーバーとの Web サーバー通信を保護する方法」](#)を参照してください。
- トンネルの詳細については、[ipsecconf\(1M\)](#)のマニュアルページを参照してください。
- トンネル構成の例は、[149 ページの「IPsec のトンネルモードで 2 つの LAN 間の接続を保護する方法」](#)を参照してください。

仮想プライベートネットワークと IPsec

仮想プライベートネットワーク (VPN) という用語は、多くの場合、インターネットなどのよりパブリックなネットワーク上に構築される、プライベートでセキュアなポイントツーポイントネットワークを説明するのに使用されます。このポイントツーポイントネットワーク、つまり VPN を使用して、プライベートネットワーク上のシステム同士、またはプライベートネットワーク上のシステムネットワーク同士を接続できます。

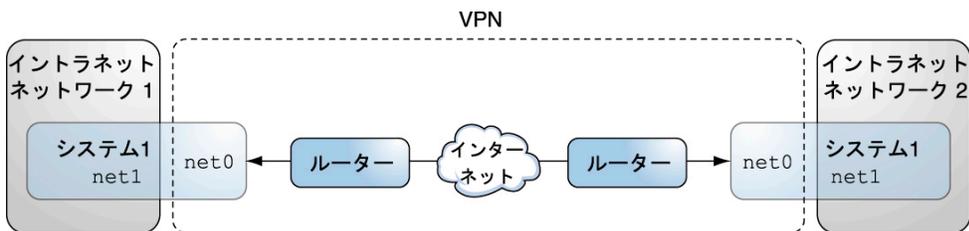
構成された tunnel は、ポイントツーポイントインタフェースです。トンネルによって、IP パケットを別の IP パケット内にカプセル化できます。トンネルの構成には、トンネルソースとトンネル宛先が必要です。詳細は、『[Oracle Solaris 11.3 での TCP/IP ネットワーク、IPMP、および IP トンネルの管理](#)』の「[IP トンネルを作成および構成する方法](#)」を参照してください。

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

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

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

図 11 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、[149 ページの「IPsec のトンネルモードで 2 つの LAN 間の接続を保護する方法」](#)を参照してください。

IPsec と FIPS 140-2

FIPS 140-2 が有効化されたシステムでは、証明書を作成して IPsec を構成するとき、ユーザーが FIPS 140-2 承認アルゴリズムのみを使用するように選択する責任があります。このガイドの手順と例では、アルゴリズム any が指定されている場合を除き、FIPS 140-2 承認アルゴリズムを使用します。

注記 - FIPS 140-2 で検証された暗号化のみを使用するという厳格な要件がある場合は、Oracle Solaris 11.3 SRU 5.6 リリースを実行している必要があります。 Oracle は、この特定のリリースでの暗号化フレームワークに対する FIPS 140-2 の検証を完了しました。後のリリースは、この検証された基盤の上に構築されており、パフォーマンス、機能、および信頼性に対応するソフトウェアの機能強化を含んでいます。これらの機能強化を利用するために、可能な場合は常に Oracle Solaris を FIPS 140-2 モードで構成するようにしてください。

次のメカニズムは、IPsec で利用可能であり、FIPS 140-2 モードの Oracle Solaris で使用することが承認されています。

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

Oracle Solaris についての FIPS 140-2 承認アルゴリズムの最終的なリストについては、『[Oracle Solaris 11.3 での FIPS 140 対応システムの使用](#)』の「[Oracle Solaris システムでの FIPS 140-2 レベル 1 証明書のリファレンス](#)」を参照してください。

IPsec と NAT 越え

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

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

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

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

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

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

IPsec と SCTP

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

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

IPsec と Oracle Solaris ゾーン

IPsec は、排他的 IP ゾーンと呼ばれる Oracle Solaris ゾーンでサポートされています。各ゾーンは独自の IPsec ポリシーおよび IKE 構成を持ち、別個のホストのように扱われます。

Trusted Extensions の共有 IP ゾーンはそれら独自の IP スタックを持たないため、ゾーンごとの IPsec をサポートしません。共有 IP ゾーンの場合、IPsec ポリシーと IKE 構成は大域ゾーンで実行されます。共有 IP ゾーンの IPsec ポリシールールは、ゾーンの共有 IP アドレスのルールです。

詳細は、『[Oracle Solaris ゾーン](#)の紹介』の第 1 章、「[Oracle Solaris ゾーン](#)の概要」および『[Trusted Extensions 構成と管理](#)』の「[ラベル付きゾーンへのアクセス](#)」を参照してください。

IPsec と仮想マシン

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

IPsec の構成コマンドとファイル

[表10](#)では、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 11.3](#)でのシステムサービスの管理』の第 1 章、「[サービス管理機能の概要](#)」を参照してください。

IPsec をネットワークに実装する手順については、[137 ページ](#)の「[IPsec によるネットワークトラフィックの保護](#)」を参照してください。

IPsec ユーティリティとファイルの詳細については、[第14章](#)「[IPsec および鍵管理のリファレンス](#)」を参照してください。

表 10 IPsec の構成コマンドとファイルの例

IPsec のコマンド、ファイル、またはサービス	説明	マニュアルページ
svc:/network/ipsec/ipsecalgs	IPsec アルゴリズムを管理する SMF サービス。	ipsecalgs(1M)
svc:/network/ipsec/manual-key	手動で鍵が設定された IPsec SA を管理する SMF サービス。	ipseckey(1M)
svc:/network/ipsec/policy	IPsec ポリシーを管理する SMF サービス。	smf(5) 、 ipseconf(1M)
svc:/network/ipsec/ike: ikev2、svc:/network/ipsec/ike: default	IKE を使用した IPsec SA 自動管理用の SMF サービスインスタンス。	smf(5) 、 in.ikev2d(1M) 、 in.iked(1M)
/etc/inet/ipsecinit.conf ファイル	IPsec ポリシーファイル。	ipseconf(1M)
ipseconf コマンド	SMF <code>policy</code> 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。 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。	ipseconf(1M)
PF_KEY ソケットインタフェース	SMF <code>policy</code> サービスによって、システムブート時に IPsec ポリシーを構成するために使用されます。	pf_key(7P)
ipseckey コマンド	セキュリティアソシエーションデータベース (SADB) のインタフェース。手動と自動の鍵管理を処理します。	ipseckey(1M)
/etc/inet/secret/ipseckey ファイル	IPsec SA 鍵作成コマンド。ipseckey は、PF_KEY インタフェースに対するコマンド行フロントエンドです。ipseckey は、SA を作成、破棄、または修正できます。 手動で鍵が設定された SA を含みます。	
ipsecalgs コマンド	SMF <code>manual-key</code> 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。 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の一覧を参照および変更するときに役立ちます。	ipsecalgs(1M)
/etc/inet/ipsecalgs ファイル	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。 構成されている IPsec メカニズムとアルゴリズム定義を含みます。このファイルは、ipsecalgs コマンドによって管理されず。手動では絶対に編集しないでください。	
/etc/inet/ike/ikev2.config ファイル	IKEv2 の構成とポリシーファイル。鍵管理はこのファイルのルールとグローバルパラメータに基づきます。257 ページの「IKEv2 のユーティリティおよびファイル」を参照してください。	ikev2.config(4)
/etc/inet/ike/config ファイル	IKEv1 の構成とポリシーファイル。デフォルトでは、このファイルはありません。鍵管理はこのファイルのルールとグローバルパラメータに基づきます。261 ページの「IKEv1 のユーティリティおよびファイル」を参照してください。 このファイルが存在する場合、svc:/network/ipsec/ike:default サービスは IKEv1 デーモン in.iked を起動します。	ike.config(4)

IPsec の構成

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

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

IPsec の概要については、第8章「IP セキュリティーアーキテクチャーについて」を参照してください。IPsec の参照情報については、第14章「IPsec および鍵管理のリファレンス」を参照してください。

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

IPsec によるネットワークトラフィックの保護

このセクションの手順では、2つのシステム間のトラフィックおよび Web サーバーをセキュリティー保護できます。VPN を保護するには、146 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、154 ページの「IPsec のその他のタスク」を参照してください。

次の情報は、すべての IPsec 構成タスクで使用されます。

- **IPsec とゾーン** – 各システムは大域ゾーンまたは排他的 IP ゾーンです。詳細は、135 ページの「IPsec と Oracle Solaris ゾーン」を参照してください。
- **IPsec および FIPS 140-2 モード** – IPsec 管理者は、Oracle Solaris 用の FIPS 140-2 承認アルゴリズムを選択する責任があります。この章の手順と例では、アルゴリズム「any」が指定されている場合を除き、FIPS 140-2 承認アルゴリズムを使用しません。

- **IPsec と RBAC** – IPsec を管理する役割を使用するには、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の第3章、「Oracle Solaris での権利の割り当て」を参照してください。例については、157 ページの「ネットワークセキュリティーの役割を構成する方法」を参照してください。
- **IPsec と SCTP** – IPsec を使用して Streams Control Transmission Protocol (SCTP) アソシエーションを保護できますが、注意が必要です。詳細は、134 ページの「IPsec と SCTP」を参照してください。
- **IPsec と Trusted Extensions のラベル** – Oracle Solaris の Trusted Extensions 機能が構成されたシステムでは、IPsec パケットにラベルを追加できます。詳細については、『Trusted Extensions 構成と管理』の「ラベル付き IPsec の管理」を参照してください。
- **IPv4 および IPv6 アドレス** – このガイドの IPsec の例では、IPv4 アドレスを使用しています。Oracle Solaris は IPv6 アドレスもサポートします。IPv6 ネットワークの IPsec を構成するには、例で IPv6 アドレスに読み替えてください。トンネルを IPsec で保護するときに、内部アドレスや外部アドレスで IPv4 アドレスと IPv6 アドレスを混在させることができます。このタイプの構成では、たとえば IPv4 ネットワーク上で IPv6 トンネリングを行うことができます。

次のタスクマップに、1 台以上のシステム間で IPsec を設定する手順を示します。また、[ipseconf\(1M\)](#)、[ipseckey\(1M\)](#)、[ipadm\(1M\)](#) の各マニュアルページでも、それぞれの「例」セクションで役立つ手順について説明しています。

表 11 IPsec によるネットワークトラフィックの保護のタスクマップ

タスク	説明	参照先
システム間のトラフィックを保護します。	あるシステムから別のシステムへのパケットを保護します。	139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」
リモートから IPsec を構成します。	リモートシステムに接続してそれらに IPsec を構成するには、ssh コマンドを使用します。	例27 「ssh 接続を使用した IPsec ポリシーのリモート構成」
FIPS 140-2 モードで動作するシステムに IPsec を構成します。	IPsec に FIPS 140-2 アルゴリズムのみを選択します。	例28 「FIPS 140-2 承認アルゴリズムを使用した IPsec ポリシーの構成」
IPsec ルールに使用する IKE プロトコルのバージョンを指定します。	すべてが IKEv2 であるネットワークへの移行に役立ちます。	例29 「IKEv2 プロトコルのみを使用する IPsec ポリシーの構成」
IPsec ルールの or pass アクションを使用します。	すべてのシステムが IPsec によって保護されるネットワークに移行する場合に役立ちます。	例30 「サーバーの or pass アクションを使用した、IPsec を使用するクライアントシステムへの移行」
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。	144 ページの「IPsec を使用してほかのサーバーとの Web サーバー通信を保護する方法」
IKE を使用して IPsec SA 用のキーイング素材を自動作成します。	IPsec SA の作成に推奨される方法。	175 ページの「IKEv2 の構成」 および 201 ページの「IKEv1 の構成」

タスク	説明	参照先
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。	146 ページの「IPsec による VPN の保護」
手動鍵管理を設定します。	IKE を使用せずに IPsec SA の生データを提供します。	154 ページの「IPsec の鍵を手動で作成する方法」

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

この手順では、次の設定がすでになされているものとしてします。

- システムに静的 IP アドレスがアサイン済みで、ネットワーク構成プロファイル DefaultFixed を実行している。netadm list コマンドが Automatic を返す場合は、[netcfg\(1M\)](#) のマニュアルページで詳細を確認してください。
- 2 つのシステムが host1 および host2 と名付けられている。
- 各システムが IP アドレスを持っています。これは、IPv4 アドレス、IPv6 アドレスのどちらでもかまいませんし、その両方でもかまいません。この手順では、IPv4 アドレスを使用します。
- 各システムは大域ゾーンまたは排他的 IP ゾーンである。詳細は、[135 ページの「IPsec と Oracle Solaris ゾーン」](#) を参照してください。
- 各システムはトラフィックを AES アルゴリズムで暗号化し、SHA-2 で認証する。

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

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

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

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

- 構成コマンドを実行するには、Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。
- この管理役割では、IPsec 関連のシステムファイルを編集し、pfedit コマンドを使用して鍵を作成できます。

- `hosts` ファイルを編集するには、`root` 役割になるか、そのファイルを編集するための明示的なアクセス権を持っている必要があります。例35「信頼できるユーザーに IPsec を構成および管理する権限を与える」を参照してください。

詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、セキュアなりモートログイン手順について、例27「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.3 での Secure Shell アクセスの管理』の「Secure Shell を使用して ZFS をリモートで管理する方法」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、ローカルネームサービスがネットワークに接続されたネームサービスに依存することなくシステム名を IP アドレスに解決できるようになります。

a. `host2` 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 host1
192.168.116.16 host1
```

b. `host1` 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 sytem2
192.168.13.213 host2
```

2. 各システムで、IPsec ポリシーファイルを作成します。

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

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

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

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

a. `host1` システムで、次のポリシーを追加します。

```
{laddr host1 raddr host2} ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
```

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

b. `host2` システムで、同じポリシーを追加します。

```
{laddr host2 raddr host1} ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
```

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

175 ページの「IKEv2 の構成」の構成手順のいずれかに従います。IKE 構成ファイルの構文については、[ikev2.config\(4\)](#) のマニュアルページを参照してください。IKEv1 プロトコルのみをサポートしているシステムと通信している場合は、201 ページの「IKEv1 の構成」および [ike.config\(4\)](#) のマニュアルページを参照してください。

注記 - 鍵を手動で生成して維持する必要がある場合は、154 ページの「IPsec の鍵を手動で作成する方法」を参照してください。

5. IPsec ポリシーファイルの構文を確認します。

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

エラーがあれば修正し、ファイルの構文を確認してから続行します。

6. IPsec ポリシーをリフレッシュします。

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

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

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

7. IPsec の鍵をアクティブ化します。

- `ike` サービスが有効になっていない場合は有効にします。

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

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

- `ike` サービスが有効になっている場合は再起動します。

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

ステップ 4 で鍵を手動で構成した場合は、154 ページの「IPsec の鍵を手動で作成する方法」の手順を実行して鍵をアクティブ化します。

8. パケットが保護されていることを確認します。

手順については、160 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 27 ssh 接続を使用した IPsec ポリシーのリモート構成

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

1. 管理者は[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」のステップ 1 からステップ 5](#)を実行することによって最初のシステムを構成します。
2. 別の端末ウィンドウで、管理者は同じように定義されたユーザー名と ID を使用して、ssh コマンドによってリモートでログインします。

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

3. ssh セッションの端末ウィンドウで、管理者は[ステップ 1 からステップ 7](#)までを実行することによって、2 番目のシステムの IPsec ポリシーと鍵を構成します。
4. 管理者は ssh セッションを終了します。

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

5. 管理者は、[ステップ 6](#)および[ステップ 7](#)を実行することによって、最初のシステムの IPsec ポリシーを有効にします。

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

例 28 FIPS 140-2 承認アルゴリズムを使用した IPsec ポリシーの構成

この例では、管理者が、鍵の長さが 256 ビットの対称アルゴリズムを必要とするサイトのセキュリティーポリシーに従うように、FIPS 140-2 対応システムの IPsec ポリシーを構成します。

管理者は 2 つの使用可能な IPsec ポリシーを指定します。最初のポリシーは、暗号用に 256 ビットの鍵の長さで CCM モードで AES を指定します。2 番目のポリシーは、暗号用に 256 ビットの鍵の長さで AES を指定し、認証用に SHA384 を指定します。

```
{laddr host1 raddr host2} ipsec {encr_algs aes-ccm(256) sa shared} or ipsec
{laddr host1 raddr host2} ipsec {encr_algs aes(256) encr_auth_algs sha384 sa shared}
```

例 29 IKEv2 プロトコルのみを使用する IPsec ポリシーの構成

この例では、管理者は、IKEv1 プロトコルを無視するように、サーバー上の IPsec ポリシーを構成します。すべての SA は IKEv2 で作成されます。IKEv1 とのネゴシエーションの試みは失敗します。管理者は対応する IKEv2 構成ファイルを作成します。

```
## ipsecinit.conf
{raddr 10.1.2.0/24 dir both } ipsec {encr_algs aes-ccm sa shared ike_version 2}
```

このルールは、10.1.2.0/24 サブネット上の任意のホストが、複合モードの aes-ccm アルゴリズムを使用してサーバーと通信できることを示しています。

対応する IKEv2 構成ファイルでは、server1.example.com サーバーに接続するために、証明書がサーバーと同じトラストチェーンで署名されているアドレス範囲を有効にします。

```
## ikev2.config
    ikesa_xform { dh_group 21 auth_alg sha256 encr_alg aes }

{
    label "IKEv2-only certificate"

    request_http_certs yes
    auth_method cert
    local_id DNS = "server1.example.com"
    remote_id ANY
    local_addr 0.0.0.0/0
    remote_addr 0.0.0.0/0
}
```

server1 サーバーの構成が完了します。証明書がインストールされ、IPsec と IKEv2 が構成されているクライアントシステムは server1 と通信できます。

例 30 サーバーの or pass アクションを使用した、IPsec を使用するクライアントシステムへの移行

この例では、管理者はサブネット上のすべてのクライアントが IPsec を使用するよう徐々に構成します。or pass {} アクションを使用すると、サブネット上のすべてのクライアント (IPsec で構成されていないクライアントを含む) からサーバーがパケットを受信できます。

```
## ipsecinit.conf
{raddr 10.1.2.0/24 dir both } ipsec {encr_algs aes-ccm sa shared} or pass {}
```

or pass {} アクションは、指定された暗号化メカニズムによって暗号化されていないがそれ以外のルールを満たしているすべてのトラフィック、およびルールに一致する IPsec トラフィックを通過させます。or pass ルールを持つサーバーへの接続は、クライアントから開始される必要があります。アクションは接続が最初に確立されたときにキャッシュされます。

10.1.2.0 サブネット上のすべてのクライアントは、サーバーへの接続を確立できます。

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

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

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

始める前に この手順では、2 つのサーバー上で IPsec を構成する [139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#) の手順が実行済みで、次の条件が満たされていると想定します。

- 各システムは、固定アドレスを持つ大域ゾーンまたは排他的 IP ゾーンである。詳細は、[135 ページの「IPsec と Oracle Solaris ゾーン」](#) を参照してください。
- Web サーバーとの通信は IPsec によってすでに保護されている。
- IKE によってキーイング素材が生成されています。
- パケットが保護されていることを確認してあります。

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

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

詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなりモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#) および『[Oracle Solaris 11.3 での Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」を参照してください。

1. IPsec ポリシー検査をバイパスするサービスを指定します。

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

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

ipsecinit.conf ファイルに次の行を追加します。

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

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

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

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

3. IPsec ポリシーファイルの構文を確認します。

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

4. IPsec ポリシーをリフレッシュします。

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

5. IPsec 用の鍵をリフレッシュします。

ike サービスを再起動します。

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

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

鍵を手動で構成した場合は、[154 ページ](#)の「[IPsec の鍵を手動で作成する方法](#)」の手順に従います。

これで設定が完了しました。

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

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

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

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

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

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

7. (オプション) IPsec ポリシーエントリ (トンネル単位のエントリを含む) を一致が発生した順序で表示します。

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

IPsec による VPN の保護

IPsec を使用して VPN を保護できます。背景については、[129 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、[137 ページの「IPsec によるネットワークトラフィックの保護」](#)を参照してください。

トンネルモード用の IPsec ポリシーの例については、[146 ページの「トンネルモードを使用して VPN を IPsec で保護する例」](#)を参照してください。

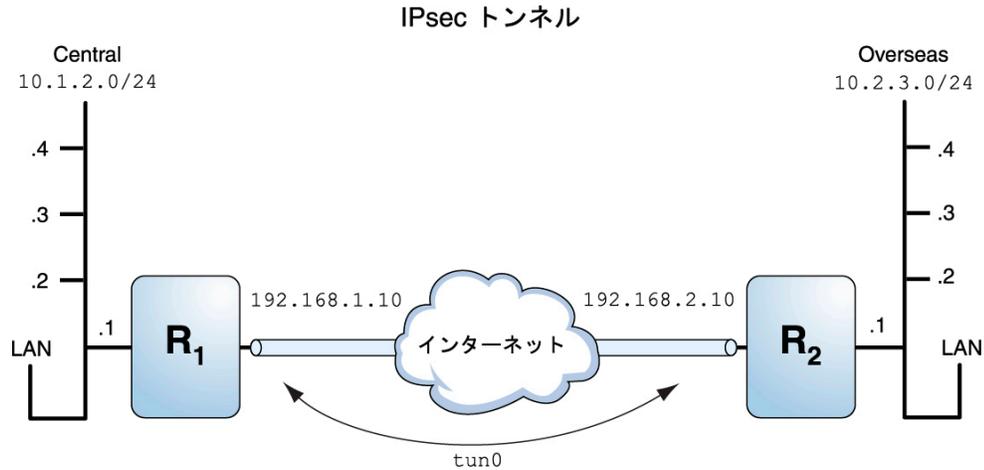
トンネルモードを使用して VPN を IPsec で保護する例

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

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

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

図 12 IPsec で保護されたトンネル



次の例はこの図に基づいています。

例 31 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12にある Central LAN のローカル LAN からのすべてのトラフィックをルーター 1 からルーター 2 にトンネリングしたあと、Overseas LAN のすべてのローカル LAN に配信できます。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 32 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、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}
```

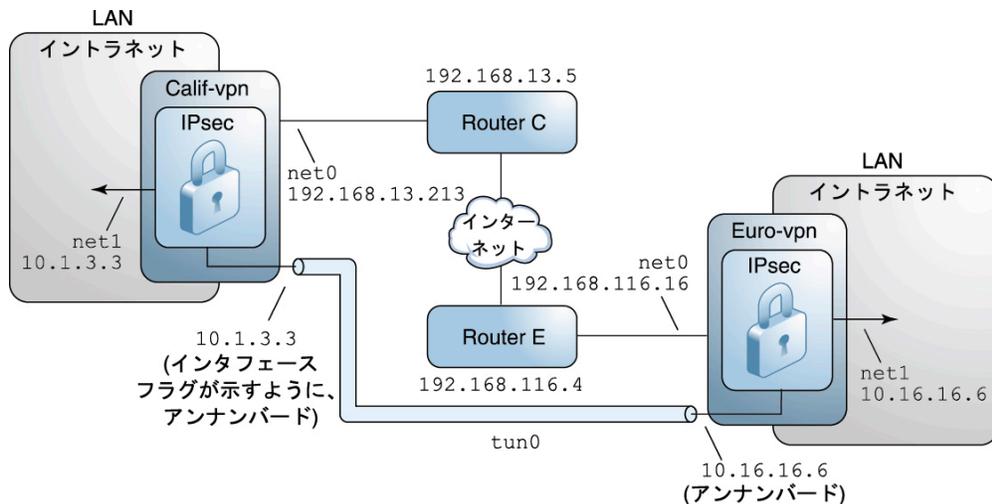
IPsec で VPN を保護するタスクのためのネットワークポロジの説明

このセクションの手順では、次の設定を前提としています。ネットワークの図については、[図13](#)を参照してください。

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

次の図は、手順内で使用される構成パラメータを示しています。

図 13 インターネット経由で接続されたオフィス間の VPN の例



構成パラメータを次の表に示します。

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

トンネル名については、『Oracle Solaris 11.3 での TCP/IP ネットワーク、IPMP、および IP トンネルの管理』の「IP トンネルの管理」を参照してください。アドレスオブジェクトについては、『Oracle Solaris 11.3 でのネットワークコンポーネントの構成と管理』の「IPv4 インタフェースを構成する方法」および `ipadm(1M)` のマニュアルページを参照してください。

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

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

この手順は、139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」の手順の応用です。設定については、148 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ポロジの説明」を参照してください。

特定のコマンドを実行する理由の詳細については、139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」の対応する手順を参照してください。

注記 - 両方のシステムでこの手順を実行してください。

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

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

始める前に 各システムは大域ゾーンまたは排他的 IP ゾーンである。詳細は、[135 ページの「IPsec と Oracle Solaris ゾーン」](#)を参照してください。

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

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

詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、セキュアなりモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#)および『Oracle Solaris 11.3 での Secure Shell アクセスの管理』の「Secure Shell を使用して ZFS をリモートで管理する方法」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 厳密宛先マルチホームを有効にします。

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

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

hostmode1 パラメータを strong に設定したときは、ある特定のインタフェースに到着するパケットには、そのインタフェースのローカル IP アドレスの 1 つが指定されている必要があります。その他のパケットは、システムのほかのローカルアドレスが指定されているものも含めてすべて捨てられます。

c. ほとんどのネットワークサービスが無効になっていることを確認します。

ssh サービスが実行中であることを確認します。

```
% svcs | grep network
...
online          Aug_09   svc:/network/ssh:default
```

2. /etc/inet/ipsecinit.conf ファイルに VPN の IPsec ポリシーを追加します。

その他の例については、[146 ページの「トンネルモードを使用して VPN を IPsec で保護する例」](#)を参照してください。

このポリシーでは、ローカル LAN 上のシステムとゲートウェイの内部 IP アドレスの間に IPsec 保護は必要でないため、bypass 文を追加します。

a. euro-vpn システムで、ipsecinit.conf ファイルに次のエントリを追加します。

```
# LAN traffic to and from this host can bypass IPsec.
{laddr 10.16.16.6 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-2.
{tunnel tun0 negotiate tunnel}
ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
```

b. calif-vpn システムで、ipsecinit.conf ファイルに次のエントリを追加します。

```
# LAN traffic to and from this host can bypass IPsec.
{laddr 10.1.3.3 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-2.
{tunnel tun0 negotiate tunnel}
ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
```

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

[175 ページの「IKEv2 の構成」](#)の構成手順のいずれかに従って、IKE を構成します。IKE 構成ファイルの構文については、[ikev2.config\(4\)](#) のマニュアルページを参照してください。IKEv1 プロトコルのみをサポートしているシステムと通信している場合は、[201 ページの「IKEv1 の構成」](#) および [ike.config\(4\)](#) のマニュアルページを参照してください。

注記 - 鍵を手動で生成して維持する必要がある場合は、[154 ページの「IPsec の鍵を手動で作成する方法」](#)を参照してください。

4. IPsec ポリシーファイルの構文を確認します。

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

エラーがあれば修正し、ファイルの構文を確認してから続行します。

5. IPsec ポリシーをリフレッシュします。

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

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

```
# svcadm enable ipsec/policy
```

6. トンネル tun0 を作成および構成します。

次のコマンドは、内部および外部インタフェースを構成し、tun0 トンネルを作成し、そのトンネルに IP アドレスを割り当てます。

a. calif-vpn システムで、トンネルを作成して構成します。

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

最初のコマンドは、IP インタフェース net1 を作成します。2 番目のコマンドは、net1 にアドレスを追加します。3 番目のコマンドは、IP インタフェース tun0 を作成します。4 番目のコマンドは、トンネルリンク内でカプセル化されている IP アドレスを追加します。詳細は、[dladm\(1M\)](#) および [ipadm\(1M\)](#) のマニュアルページを参照してください。

b. euro-vpn システムで、トンネルを作成して構成します。

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

注記 - ipadm コマンドの -T オプションは、作成するアドレスのタイプを指定します。dladm コマンドの -T オプションは、トンネルを指定します。

これらのコマンドについては、[dladm\(1M\)](#) と [ipadm\(1M\)](#) のマニュアルページおよび『Oracle Solaris 11.3 でのネットワークコンポーネントの構成と管理』の「IPv4 インタフェースを構成する方法」を参照してください。カスタマイズされた名前については、『Oracle Solaris 11.3 でのネットワークコンポーネントの構成と管理』の「Oracle Solaris のネットワークデバイスとデータリンク名」を参照してください。

7. 各システムで転送を構成します。

```
# ipadm set-ifprop -m ipv4 -p forwarding=on net1
# ipadm set-ifprop -m ipv4 -p forwarding=on tun0
# ipadm set-ifprop -m ipv4 -p forwarding=off net0
```

IP 転送とは、別のインタフェースから到着したパケットを転送できることを意味します。IP 転送はまた、送信するパケットがもともとは別のインタフェースから発信されたパケットである可能性も意味します。パケットを正しく転送するには、受信インタフェースと送信インタフェースの IP 転送を有効にしておきます。

net1 インタフェースはイントラネットの内部にあるため、net1 の IP 転送は有効にしておきます。tun0 はインターネットを通してこれら 2 つのシステムを接続するため、tun0 の IP 転送は有効にしておく必要があります。net0 インタフェースの IP 転送は無効になっているため、インターネット上の外部の脅威によってパケットが保護イントラネットに侵入するのを防ぐことができます。

8. 各システムで、プライベートインタフェースの広告を禁止します。

```
# ipadm set-addrprop -p private=on net0
```

net0 の IP 転送が無効になっていても、ルーティングプロトコルの実装によっては、このインタフェースを通知することがあります。たとえば、in.routed プロトコルは、イントラネット内のピアにパケットが転送される際に net0 を有効なインタフェースとして通知する場合があります。インタフェースの「private」フラグを設定して、このような通知が行われないようにします。

9. ネットワークサービスを再起動します。

```
# svcadm restart svc:/network/initial:default
```

10. net0 インタフェース経由のデフォルトルートを手動で追加します。

デフォルトルートは、インターネットに直接アクセスできるルーターでなければなりません。

a. calif-vpn システムで次のルートを追加します。

```
# route -p add net default 192.168.13.5
```

b. euro-vpn システムで、次のルートを追加します。

```
# route -p add net default 192.168.116.4
```

net0 インタフェースはイントラネットの一部ではありませんが、net0 は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。詳細は、[route\(1M\)](#) および [in.routed\(1M\)](#) のマニュアルページを参照してください。

IPsec のその他のタスク

次のタスクマップは、IPsec を管理するときに使用する可能性のあるタスクを示したものです。

表 12 IPsec のその他のタスクのタスクマップ

タスク	説明	参照先
手動による IPsec SA の作成または置き換えを行います。	IPsec SA の生データを提供します。	154 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。	157 ページの「ネットワークセキュリティーの役割を構成する方法」
すべてのネットワーク管理タスクを処理できる権利プロファイルを作成します。	ネットワーク管理を実行できるが、root 役割より権限が少ない役割を作成します。	例35「信頼できるユーザーに IPsec を構成および管理する権限を与える」
IPsec がパケットを保護しているかどうかを検査します。	snoop の出力を調べ、IP パケットがどのように保護されているかを示すヘッダーをチェックします。	160 ページの「IPsec によってパケットが保護されていることを確認する方法」
IPsec と鍵情報を一連の SMF サービスとして管理します。	サービスの有効化、無効化、リフレッシュ、および再起動を行います。サービスのプロパティー値も変更します。	242 ページの「IPsec および手動鍵 サービスプロパティーの表示」

▼ IPsec の鍵を手動で作成する方法

次の手順は、鍵の管理に使用しているのが IKE だけではない場合に IPsec 鍵を提供します。

ipseckey コマンドを使用して追加された IPsec SA には永続性がなく、システムのリブート時に失われます。IPsec SA に永続性を持たせるために、`/etc/inet/secret/ipseckey` ファイルにエントリを追加します。



注意 - 手動で鍵処理をする必要がある場合は、生成する鍵が確実にセキュアであるように細心の注意を払う必要があります。これらの鍵は、データのセキュリティー保護に実際に使用されます。

始める前に 共有 IP ゾーンの鍵情報の手動管理は、大域ゾーンで行う必要があります。排他的 IP ゾーンの場合は、鍵情報をその排他的 IP ゾーンで構成します。

root 役割になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。

1. IPsec SA の鍵を生成します。

鍵は、`ipsecinit.conf` ファイル内の特定のポリシーをサポートしている必要があります。たとえば、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#)のポリシーを使用することもできます。

```
{laddr host1 raddr host2} ipsec {encr_algs aes encr_auth_algs sha512 sa shared}
```

このポリシーは AES および SHA-2 アルゴリズムを使用します。

a. 必要な鍵を決定します。

SA 用に、`aes`、`sha512`、および `security parameter index (SPI)` の鍵を生成する必要があります。

- SPI の値として、2 つの 16 進数の乱数。1 つはアウトバウンドトラフィック用です。もう 1 つはインバウンドトラフィック用です。それぞれの乱数の最大桁数は 8 桁です。
- SHA-2 認証アルゴリズム用の 2 つの 16 進数の乱数。各数字の長さは 512 文字でなければいけません。もう 1 つは `dst host1` 用です。もう 1 つは `dst host2` 用です。
- AES 暗号化アルゴリズム用の 2 つの 16 進数の乱数。各数字の長さは 128 文字でなければいけません。もう 1 つは `dst host1` 用です。もう 1 つは `dst host2` 用です。

注記 - `ipsecalgs -l` コマンドは、アルゴリズムの鍵のサイズを表示します。手動鍵を使用する場合はこの手順に従い、SHA512 および AES アルゴリズムを使用してください。手動鍵には、弱いアルゴリズム、複合モードのアルゴリズム、または GMAC アルゴリズムは使用しないでください。

b. 必要な鍵を生成します。

- 乱数発生関数がすでにある場合は、それを使用してください。
- 『[Oracle Solaris 11.3 での暗号化と証明書の管理](#)』の「[pktool コマンドを使用して対称鍵を生成する方法](#)」とそのセクションの IPsec の例に示すように、`pktool` コマンドを使用します。

2. IPsec の手動鍵ファイルに鍵を追加します。

a. `host1` システムの `/etc/inet/secret/ipseckeys` ファイルを次のように編集します。

```
## ipseckeys - This file takes the file format documented in
## ipseckey(8).
# Note that naming services might not be available when this file
# loads, just like ipsecinit.conf.
#
# Backslashes indicate command continuation.
#
```

```

# for outbound packets on host1
add esp spi 0x8bcd1407 \
  src 192.168.116.16 dst 192.168.13.213 \
  encr_alg aes \
  auth_alg sha512 \
  encrkey abcdefabcdefabcdefabcdefabcdefab... \
  authkey 12345678128456789123456789123456...
#
# for inbound packets
add esp spi 0xnntnnnnn \
  src 192.168.13.213 dst 192.168.116.16 \
  encr_alg aes \
  auth_alg sha512 \
  encrkey fedcbafedcbafedcbafedcbafedcbafe... \
  authkey 98765432123456789876543212345678...

```

b. 読み取り専用ファイルを保護します。

```
# chmod 400 /etc/inet/secret/ipseckeys
```

pfedit -s コマンドを使用して ipseckeys ファイルを作成した場合は、権限が正しく設定されています。詳細は、[pfedit\(1M\)](#) のマニュアルページを参照してください。

c. ファイルの構文を確認します。

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

注記 - 2つのシステムの鍵は同じでなければなりません。

3. IPsec の鍵をアクティブにします。

■ manual-key サービスが有効になっていない場合は有効にします。

```

% svcs manual-key
STATE          STIME      FMRI
disabled      Apr_10    svc:/network/ipsec/manual-key:default
# svcadm enable ipsec/manual-key

```

■ manual-key サービスが有効になっている場合はリフレッシュします。

```
# svcadm refresh ipsec/manual-key
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#)を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#)を参照してください。

▼ ネットワークセキュリティの役割を構成する方法

Oracle Solaris の権利機能を使用してシステムを管理している場合は、ネットワーク管理の役割またはネットワークセキュリティの役割を提供するためにこの手順を使用します。

始める前に 役割を作成して割り当てるには、root 役割になる必要があります。標準ユーザーは、使用可能な権利プロファイルの内容を一覧表示できます。

1. 使用可能なネットワーク関連の権利プロファイルを一覧表示します。

```
% getent prof_attr | grep Network | more
...
Network Management:RO::Manage the host and network configuration...
Network Security:RO::Manage network and host security...:profiles=Network Wifi
Security,Network Link Security,Network IPsec Management...
Network Wifi Management:RO::Manage wifi network configuration...
Network Wifi Security:RO::Manage wifi network security...
Network Link Security:RO::Manage network link security...
Network IPsec Management:RO::Manage IPsec and IKE...
System Administrator:RO::Can perform most non-security administrative tasks:
profiles=...Network Management...
Information Security:RO::Maintains MAC and DAC security policies:
profiles=...Network Security...
```

Network Management プロファイルは、System Administrator プロファイルを補完するプロファイルです。System Administrator 権利プロファイルを役割に含めると、その役割は Network Management プロファイルでコマンドを実行できます。

2. Network Management 権利プロファイル内のコマンドを一覧表示します。

```
% profiles -p "Network Management" info
...
cmd=/usr/sbin/dladm
cmd=/usr/sbin/dlstat
...
cmd=/usr/sbin/svcadm
cmd=/usr/sbin/svccfg
cmd=/usr/sbin/dumppcap
```

3. サイトでのネットワークセキュリティの役割の範囲を決定します。

決定には、[ステップ 1](#) の権利プロファイルの定義を参考にしてください。

- すべてのネットワークセキュリティを扱う役割を作成する場合は、Network Security 権利プロファイルを使用します。
- IPsec と IKE だけを扱う役割を作成するには、Network IPsec Management 権利プロファイルを使用します。
- ネットワーク管理とセキュリティを処理する役割を作成するには、Network Management プロファイルに加えて、Network Security または Network IPsec Management 権利プロファイルを使用します。

4. 役割を作成して、その役割を 1 人または複数のユーザーに割り当てます。

手順については、『Oracle Solaris 11.3でのユーザーとプロセスのセキュリティ保護』の「役割の作成」および例35「信頼できるユーザーにIPsecを構成および管理する権限を与える」を参照してください。

例 33 ネットワーク管理およびセキュリティの役割の作成と割り当て

この例では、管理者が Network Management と Network Security という2つの権利プロファイルを1つの役割に割り当てます。次に、管理者は信頼できるユーザーにこの役割を割り当てます。

```
# roleadd -c "Network Mgt and Security" \  
-S ldap -K profiles="Network Management Plus" netmgtsec  
# passwd netmgtsec  
New Password: xxxxxxxx  
Confirm password: xxxxxxxx  
# usermod -R netmgtsec jdoe
```

ユーザー jdoe が netmgtsec 役割になると、jdoe はこれらのプロファイルの権利を利用できるようになります。

```
% su - netmgtsec  
Password: xxxxxxxx  
#
```

例 34 ネットワークセキュリティ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を管理し、もう一方の役割は IPsec と IKE を管理します。各役割には、シフトごとに1人、合計3人を割り当てます。管理者によって次のように役割が作成されます。

1. 最初の役割には LinkWifi という名前を付けます。
2. この役割には Network Wifi、Network Link Security、および Network Management 権利プロファイルを割り当てます。
3. 管理者は該当するユーザーにこの LinkWifi 役割を割り当てます。
4. 2番目の役割には IPsec Administrator という名前を付けます。
5. この役割には Network IPsec Management および Network Management 権利プロファイルを割り当てます。
6. 管理者は該当するユーザーにこの IPsec Administrator 役割を割り当てます。

例 35 信頼できるユーザーに IPsec を構成および管理する権限を与える

この例では、管理者が1人のユーザーに IPsec を構成および管理する役割を与えます。

Network Management と IPsec Network Management の権利プロファイルに加えて、管理者はユーザーに `hosts` ファイルを編集する権限とそのログを読み取る権限を与えます。

1. 管理者は、ファイル編集用とログ読み取り用の2つの権利プロファイルを作成します。

```
# profiles -p -S LDAP "Hosts Configuration"
profiles:Network Configuration> set desc="Edits root-owned network files"
...Configuration> add auth=solaris.admin.edit/etc/hosts
...Configuration> commit
...Configuration> end
...Configuration> exit

# profiles -p -S LDAP "Read Network Logs"
profiles:Read Network Logs> set desc="Reads root-owned network log files"
...Logs> add cmd=/usr/bin/more
...Logs:more>set privs={file_dac_read}:/var/user/ikeuser/*
...Logs:more>set privs={file_dac_read}:/var/log/ikev2/*
...Logs:more> set privs={file_dac_read}:/etc/inet/ike/*
...Logs:more> set privs={file_dac_read}:/etc/inet/secret/*
...Logs:more>end
...Logs> add cmd=/usr/bin/tail
...Logs:tail>set privs={file_dac_read}:/var/user/ikeuser/*
...Logs:tail>set privs={file_dac_read}:/var/log/ikev2/*
...Logs:tail>set privs={file_dac_read}:/etc/inet/ike/*
...Logs:tail> set privs={file_dac_read}:/etc/inet/secret/*
...Logs:tail>end
...Logs> add cmd=/usr/bin/page
...Logs:page>set privs={file_dac_read}:/var/user/ikeuser/*
...Logs:page>set privs={file_dac_read}:/var/log/ikev2/*
...Logs:page>set privs={file_dac_read}:/etc/inet/ike/*
...Logs:page> set privs={file_dac_read}:/etc/inet/secret/*
...Logs:page>end
...Logs> exit
```

この権利プロファイルによって、ユーザーは `more`、`tail`、および `page` コマンドを使用してログを読み取れるようになります。 `cat` コマンドと `head` コマンドは使用できません。

2. 管理者は、ユーザーが IPsec とその鍵サービスのすべての構成および管理タスクを実行できる権利プロファイルを作成します。

```
# profiles -p "Site Network Management"
profiles:Site Network Management> set desc="Handles all network files and logs"
...Management> add profiles="Network Management"
...Management> add profiles="Network IPsec Management"
```

```
...Management> add profiles="Hosts Configuration"
...Management> add profiles="Read Network Logs"
...Management> commit; end; exit
```

3. 管理者はプロファイルの役割を作成し、それにパスワードを割り当ててから、ネットワーキングとセキュリティーを理解している信頼できるユーザーにその役割を割り当てます。

```
# roleadd -S LDAP -c "Network Management Guru" \
-m -K profiles="Site Network Management" netadm
# passwd netadm
Password: xxxxxxxx
Confirm password: xxxxxxxx
# usermod -S LDAP -R +netadm jdoe
```

4. 帯域外で、管理者は jdoe に役割のパスワードを渡します。

▼ IPsec によってパケットが保護されていることを確認する方法

パケットが保護されていることを確認するには、snoop コマンドで接続をテストします。snoop 出力に表示される接頭辞は、次のとおりです。

- AH: 接頭辞は、AH がヘッダーを保護していることを示します。この接頭辞が表示されるのは、auth_alg を使ってトラフィックを保護している場合です。
- ESP: 接頭辞は、暗号化されたデータが送信されていることを示します。この接頭辞が表示されるのは、encr_auth_alg が encr_alg を使ってトラフィックを保護している場合です。

始める前に さらに、接続をテストするためには、両方のシステムにアクセスできなければなりません。

snoop の出力を作成するには、root 役割になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. host2 など、1 つのシステム上で root 役割になります。

```
% su -
Password: xxxxxxxx
#
```

2. (オプション) SA の詳細を表示します。

```
# ipseckey dump
```

この出力は、使用されている SA に一致する SPI 値、使用されていたアルゴリズム、鍵などを示します。

3. このシステムで、リモートシステムからパケットをスヌープする準備をします。

host2 の端末ウィンドウで、host1 システムからパケットをスヌープします。

```
# snoop -d net0 -o /tmp/snoop_capture host1
Using device /dev/xxx (promiscuous mode)
```

4. リモートシステムからパケットを送信します。

別の端末ウィンドウで、host1 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 host1 システムから host2 システムに送信します。パケットは、snoop -v host1 コマンドで取り込む必要があります。

```
host2% ssh host1
Password: xxxxxxxx
host1% su -
Password: xxxxxxxx
host1# ping host2
```

5. snoop の出力を調べます。

```
host2# snoop -i /tmp.snoop_capture -v
```

snoop の出力を Wireshark アプリケーションにロードすることもできます。詳細は、[234 ページの「IPsec および IKE システムのトラブルシューティングを準備する方法」](#) および [255 ページの「snoop コマンドと IPsec」](#) を参照してください。

このファイルで、冒頭の IP ヘッダー情報のあとに AH と ESP の情報が含まれている出力を確認します。次のような AH と ESP の情報は、パケットが保護されていることを示します。

```
IP:   Time to live = 64 seconds/hops
IP:   Protocol = 51 (AH)
IP:   Header checksum = 4e0e
IP:   Source address = 192.168.116.16, host1
IP:   Destination address = 192.168.13.213 host2
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 -----
```

...

インターネット鍵交換について

インターネットキー交換 (Internet Key Exchange、IKE) は、Ipssec の鍵管理を自動化します。IKE について説明するこの章の内容は次のとおりです。

- [163 ページの「IKE の概要」](#)
- [169 ページの「IKEv2 プロトコル」](#)
- [171 ページの「IKEv1 プロトコル」](#)

IKE プロトコルの最新バージョンを実装する手順については、[第11章「IKEv2 の構成」](#)を参照してください。IKEv1 の使用を続行するには、[第12章「IKEv1 の構成」](#)を参照してください。参照情報については、[第14章「IPsec および鍵管理のリファレンス」](#)を参照してください。IPsec については、[第8章「IP セキュリティーアーキテクチャーについて」](#)を参照してください。

IKE の概要

IPsec セキュリティーアソシエーション (SA) の鍵情報を管理することを「鍵管理」といいます。自動鍵管理には、鍵の作成、認証、および交換にセキュアな通信チャンネルが必要です。Oracle Solaris では、インターネット鍵交換 (IKE) を使用して鍵管理を自動化します。IKE は、秘密鍵の手動配布に伴う管理オーバーヘッドとセキュリティーのリスクを排除します。

IKE では、ハードウェア暗号化アクセラレーションと鍵ストレージを利用できます。ハードウェア暗号化アクセラレータによって、CPU に負荷のかかる鍵操作をシステム外で処理できます。ハードウェア上での鍵の格納によって、保護機能が強化されます。

Oracle Solaris は、IKE プロトコルの 2 つのバージョンをサポートしています。

- [インターネット鍵交換プロトコルバージョン 2 \(IKEv2\)、RFC 5996 に基づく IKE バージョン 2 \(IKEv2\)](#)
- [インターネット鍵交換 \(IKE\)、RFC 2409 に基づく IKE バージョン 1 \(IKEv1\)](#)

FIPS 140-2 が有効化されたシステムでは、IKEv2 を FIPS 140-2 承認アルゴリズムのみで構成するようにしてください。詳細は、[171 ページの「IKEv2 および FIPS 140-2」](#) を参照してください。

注記 - FIPS 140-2 で検証された暗号化のみを使用するという厳格な要件がある場合は、Oracle Solaris 11.3 SRU 5.6 リリースを実行している必要があります。Oracle は、この特定のリリースでの暗号化フレームワークに対する FIPS 140-2 の検証を完了しました。後のリリースは、この検証された基盤の上に構築されており、パフォーマンス、機能、および信頼性に対応するソフトウェアの機能強化を含んでいます。これらの機能強化を利用するために、可能な場合は常に Oracle Solaris を FIPS 140-2 モードで構成するようにしてください。

IKE の概念および用語

次の概念と用語は IKE の両方のバージョンに共通です。2 つのバージョンでは、異なる形で実装されている可能性があります。

- **鍵のネゴシエーションおよび交換** – セキュアな方法で行われる、鍵情報の交換およびピアの識別情報の認証。このプロセスでは非対称暗号化アルゴリズムを使用します。主な 2 つの方法は、RSA と Diffie-Hellman プロトコルです。

IKE は IKE デーモンを実行するシステム間の IPsec SA を作成および管理します。IKE は鍵情報の伝送を保護するセキュアなチャンネルをネゴシエートします。デーモンは、`/dev/random` デバイスを使用して乱数発生関数から鍵を作成します。デーモンは、鍵を一定の割合 (構成可能) で変更します。この鍵情報は、IPsec ポリシーの構成ファイル `ipsecinit.conf` に指定されているアルゴリズムによって使用されます。

- **Diffie-Hellman (DH) アルゴリズム** – セキュアでないチャンネルで 2 つのシステムが共有シークレットをセキュアに生成できる鍵交換アルゴリズム。
- **RSA アルゴリズム** – ピアシステムの識別情報の認証 (通常は X.509 証明書の所有権の提供による) に使用される非対称鍵アルゴリズム。アルゴリズム名は、作成者の Rivest, Shamir, Adleman の三氏に因んでいます。

または、この目的に [DSA](#) または [ECDSA](#) アルゴリズムが使用されることもあります。

- **Perfect forward secrecy (PFS)** – PFS では、データ伝送を保護するために使用される鍵が、追加の鍵を導き出すために使用されることはありません。さらに、データ伝送を保護するために使用される鍵のソースが、追加の鍵を導き出すために使用されることはありません。したがって、PFS は以前に記録されたトラフィックの復号化を回避できます。
- **Oakley グループ** – PFS のネゴシエーションに使用されます。 [インターネット鍵交換 \(IKE\) RFC](#) のセクション 6 を参照してください。

- **IKE ポリシー** – IKE デーモンがピアシステムとのセキュアな鍵交換チャネルの設定を試行する際に使用可能なパラメータを定義する IKE ルールのセット。これは、IKEv2 では IKE SA、IKEv1 ではフェーズ 1 と呼ばれます。
パラメータには、アルゴリズム、鍵のサイズ、Oakley グループ、および認証方式が含まれます。Oracle Solaris IKE デーモンは認証方式として事前共有鍵および証明書をサポートしています。

IKE の動作

IKE デーモンを実行しているシステムは、このシステムと IKE デーモンを実行している別のシステムの間にセキュリティーアソシエーション (SA) を作成するのに必要なパラメータをネゴシエートできます。この SA および後続の IPsec SA のネゴシエーションに使用されるプロトコルは IKE と呼ばれます。Oracle Solaris のこのバージョンは、IKE プロトコルのバージョン 1 (IKEv1) およびバージョン 2 (IKEv2) をサポートしています。

IKE セキュリティーアソシエーション (IKEv1 では ISAKMP またはフェーズ 1 SA と呼ばれる) は、これら 2 つの IKE システム間のさらなるプロトコル交換をセキュリティー保護します。これらの交換では、暗号化アルゴリズム、IPsec ポリシー、および IPsec SA の作成に必要なその他のパラメータをネゴシエートします。

IKE デーモンを実行しているシステムは、その他のシステムの代理として IPsec SA をネゴシエートするように構成することもできます。この方法で構成されたシステムは、セキュリティーゲートウェイと呼ばれます。IKE ネゴシエーションが成功すると、IPsec SA を使用してネットワークパケットを保護できます。

IKE SA を作成するためにネゴシエートされるパラメータには、IKE 交換および一部の認証情報を保護する暗号化アルゴリズムが含まれます。認証情報は、IKE プロトコル交換を含むパケットが信頼できるかどうかを判断するために使用されます。信頼とは、そのパケットが信頼できるシステムから来たもので、そのシステムになりすまししているシステムから来たものではないことを意味します。

Oracle Solaris は、事前共有鍵と公開鍵証明書という 2 種類の IKE 認証情報をサポートしています。

IKE と事前共有鍵認証

事前共有鍵は、2 つの IKE システムのみが認識する 16 進数または ASCII 文字の文字列です。この鍵は、IKE 交換の前に両方のエンドポイントが鍵の値を認識している必要があることから、事前共有鍵と呼ばれます。この鍵は両方のシステムで IKE 構成に含まれている必要があります。事前共有鍵は IKE ペイロードの生成に使用され、IKE

プロトコルを実装するパケットを作成します。これらの IKE ペイロードを処理するシステムは、受け取るペイロードの認証に同じ鍵を使用します。

事前共有鍵は、IKE エンドポイント間で IKE プロトコルを使用して交換されることはありません。通常、鍵は電話などの異なる媒体でピアシステムと共有されます。

この認証方法を使用するピアの事前共有鍵は同一である必要があります。鍵は各システム上のファイルに格納されます。

IKE と公開鍵証明書

公開鍵証明書およびそれらのトラストチェーンが、秘密の情報を手動で交換することなくシステムをデジタルに識別するメカニズムを提供します。そのため、公開鍵証明書は事前共有鍵よりもセキュアです。

公開鍵証明書は、公開鍵の値、名前や署名者など証明書の生成に関する情報、証明書のハッシュまたはチェックサム、およびハッシュのデジタル署名をエンコードするプロブデータです。これらの値が一体となって証明書を形成します。デジタル署名は証明書が変更されていないことを保証します。

公開鍵は、非公開鍵と呼ばれる別の値から数学的に生成される値です。非公開鍵から公開鍵を生成する数学的アルゴリズムでは、公開鍵から非公開鍵を取得することはできません。そのため、公開鍵証明書は自由に共有できます。公開鍵の生成に使用されるアルゴリズムの例として、[RSA](#) および楕円曲線があります。

デジタル署名とは、[RSA](#)、[DSA](#)、[ECDSA](#) などのデジタル署名アルゴリズムによって証明書の内容を渡した結果です。これらのアルゴリズムは、証明書に含まれない非公開署名鍵を使用してデジタル署名を作成します。署名は証明書に追加されます。この場合も、証明書の内容および署名から署名鍵を計算することはできません。簡単に言うと、署名鍵から生成された公開値を使用して、証明書の署名、および証明書の内容を簡単に検証できます。

証明書は、自己署名することも(この場合、証明書の署名はその証明書の公開鍵で検証できる)、第三者が署名することもできます。第三者が証明書に署名する場合、証明書の検証に使用される公開鍵の値も公開鍵証明書として配布されます。この2つ目の証明書は、信頼される[認証局 \(CA\)](#)、または仲介者によって署名されます。この仲介者は最終的に署名機関、つまりルート CA または[トラストアンカー](#)に信頼されます。

これら公開鍵証明書のコンポーネントとそれらを実装する手順および構造を合わせて、多くの場合、公開鍵インフラストラクチャー ([PKI](#)) と呼びます。組織の PKI の範囲は異なる場合があります。単純な PKI は、ローカル使用の少数の証明書に署名する CA で構成されていることがあります。より大きな PKI になると、グローバルに認知されているトラストアンカーを正式な CA として使用します。

IKE での公開鍵証明書の使用

このセクションでは、IKE での公開鍵証明書の作成および使用の全体的な手順を説明します。特定の手順については、[176 ページの「事前共有鍵による IKEv2 の構成」](#) および [202 ページの「事前共有鍵による IKEv1 の構成」](#) を参照してください。

1. 自己署名付き証明書または認証局 (CA) による証明書を使用するには、まず公開鍵/非公開鍵ペアを生成します。
 - 自己署名付き証明書の場合は、次に IKE ピアがこれらの証明書を交換し、証明書が本物であることを帯域外で検証したあと、ピアの証明書をローカルのキーストアにインポートします。これにより、キーストアにオリジナルの自己署名付き証明書とインポートされた証明書が格納されます。
 - CA からの証明書の場合、さらにいくつかの手順を実行します。公開鍵/非公開鍵ペアを生成する際に、証明書署名要求 (CSR) も生成します。CSR には公開鍵および識別子が含まれています。一般的な識別子は、**識別名 (DN)** です。例:

```
DN=O=Example\, Inc, OU=qa, L=Silicon Valley, ST=CA, CN=enigma
```

ヒント - 別の証明書の識別子と一致する可能性を低くするために、できるだけ固有の DN またはその他の識別子を作成します。

2. 署名のために CA に CSR を送信します。

一般的なプロセスでは、Web フォームに CSR を貼り付けて、そのフォームを CA に送信します。CA から複数の署名付き証明書が送信される場合があります。
3. CA から署名付き証明書を取得したあと、それらを IKEv2 キーストアまたは IKEv1 データベースにインポートします。

CA から送信されるすべての証明書をインポートする必要があります。これらの証明書は、トラストアンカー (ルート CA) から個々に識別される署名付き証明書への「トラストチェーン」で構成されます。
4. IKE ピアでこの手順を繰り返します。
5. IKE ルール内の証明書を使用します。

DN などの識別子で証明書を指定します。CA が署名する証明書の場合、特定の CA によって署名されている証明書を受け入れるように IKE を構成できます。

失効した証明書の処理

署名付き証明書は、署名機関がその有効性を保証するため、有効として信頼されません。証明書が損なわれた場合や無効と判断された場合は、CA によって取り消されます。

CA は、多くの場合 [証明書失効リスト \(CRL\)](#) と呼ばれる失効した証明書のリストを保持しています。オンライン証明書ステータスプロトコル (OCSP) を使用して、証明書のステータスを動的に確認できます。一部の公開鍵証明書には URI が埋め込まれています。それらは CRL を確認できる Web の場所、または OCSP サーバーの Web の場所を識別します。

詳細は、[RFC 2459: 証明書と CRL プロファイル](#) および [RFC 2560: オンライン証明書ステータスプロトコル - OCSP](#) を参照してください。

公開証明書を使用するシステムでの時間の調整

公開鍵証明書には、発行日時および有効な残り時間が含まれています。そのため、証明書を生成および使用するシステム上のクロックが正確である必要があります。Network Time Protocol (NTP) ソフトウェアを使用して、システムのクロックを同期できます。Oracle Solaris のリリースにはデラウェア大学の NTP パブリックドメインソフトウェアが含まれています。ドキュメントは [NTP ドキュメント](#) の Web サイトで入手できます。service/network/ntp パッケージをインストールして高精度時間プロトコル (PTP) サービスを構成することもできます。[ネットワーク接続された測定および制御システムの高精度時刻同期プロトコルのための IEEE 1588 規格](#) を参照してください。

IKE バージョンの指定

ルールごとに特定の IKE プロトコルバージョンの使用を要求することにより、対応する IKE 構成ルールを変更せずに IPsec ルールのみを変更できます。IPsec ルールに IKEv2 プロトコルを指定することは、旧バージョンの IKEv1 プロトコルから整然とシステムを移行する際に役立つ手順です。例については、[例29 「IKEv2 プロトコルのみを使用する IPsec ポリシーの構成」](#) を参照してください。

IKEv2 と IKEv1 の比較

次の表は、Oracle Solaris システムでの IKEv2 と IKEv1 のバージョンの実装を比較しています。

表 13 Oracle Solaris での IKEv2 および IKEv1 の実装

構成コンポーネント	IKEv2	IKEv1
証明書のchain of trust	キーストア内のオブジェクトに基づいて暗黙的	ike/config ファイルの cert_trust パラメータ

構成コンポーネント	IKEv2	IKEv1
証明書の作成	ikev2cert コマンド	ikecert certlocal コマンド
証明書のインポート	ikev2cert import コマンドは PKCS #11 キーストアに証明書および鍵をインポートできます	ikecert certdb コマンドは IKE キーストアにスタンドアロンの証明書をインポートできます
証明書の所有者	ikeuser	root
証明書のポリシーファイル	kmf-policy.xml	ike/config ファイル内の一部のポリシー
証明書ストレージ	PKCS #11 ソフトトークンライブラリ	ローカルの IKEv1 データベース
構成ファイルのディレクトリ	/etc/inet/ike/	/etc/inet/ike/ および /etc/inet/secret/
構成の所有者	ikeuser アカウント	root アカウント
デーモン	in.ikev2d	in.iked
IKE メッセージ長	サイズはパス MTU に調整可能	サイズは調整不可
IKE ポリシーファイル	ike/ikev2.config	ike/config
IKE 事前共有鍵	ike/ikev2.preshared	secret/ike.preshared
IKE SA およびプロトコル交換 [†]	FIPS 140-2 承認アルゴリズムで構成可能	FIPS 140-2 承認アルゴリズムで構成不可
証明書操作 [†]	FIPS 140-2 承認アルゴリズムで構成可能	FIPS 140-2 承認アルゴリズムで構成不可
NAT ポート	UDP ポート 4500	UDP ポート 4500
ポート	UDP ポート 500	UDP ポート 500
権利プロファイル	Network IPsec Management	Network IPsec Management
サービス名 (FMRI)	svc:/ipsec/ike:ikev2	svc:/ipsec/ike:default

[†]Oracle Solaris 11.3 SRU 5.6 の暗号化フレームワーク機能は、FIPS 140-2 レベル 1 で検証されています。FIPS 140-2 モードが有効であり、暗号化フレームワークが使用されている場合は、FIPS 140-2 承認アルゴリズムが使用できます。デフォルトでは、FIPS 140-2 モードは有効になっていません。

IKEv2 プロトコル

このセクションでは、IKEv2 の実装について説明します。IKEv1 については、[171 ページの「IKEv1 プロトコル」](#)を参照してください。比較については、[168 ページの「IKEv2 と IKEv1 の比較」](#)を参照してください。両方のプロトコルに適用される情報については、[163 ページの「IKE の概要」](#)を参照してください。Oracle Solaris は、IKE プロトコルの両方のバージョンを同時にサポートします。

IKEv2 デーモン in.ikev2d は、IPsec SA の鍵情報をネゴシエートして認証します。[in.ikev2d\(1M\)](#) のマニュアルページを参照してください。

IKEv2 構成の選択

/etc/inet/ike/ikev2.config 構成ファイルには、in.ikev2d デーモンの構成が含まれています。構成はいくつかのルールで構成されています。各エントリには、このシステムが同様に構成された IKEv2 ピアに対して使用できるアルゴリズムや認証データなどのパラメータが含まれます。

in.ikev2d デーモンは、事前共有鍵 (PSK) および識別情報の公開鍵証明書をサポートしています。

[ikev2.config\(4\)](#) のマニュアルページにサンプルルールが提供されています。各ルールには一意のラベルが必要です。次は、マニュアルページのサンプルルールの説明ラベルのリストです。

- IP 識別情報および PSK 認証
- IP アドレス接頭辞および PSK 認証
- IPv6 アドレス接頭辞および PSK 認証
- DN 識別情報による証明書認証
- 多くのピア ID タイプによる証明書認証
- ワイルドカードピア ID による証明書認証
- オーバーライド変換
- 混合した認証タイプ
- ワイルドカードと要求される署名者

注記 - 事前共有鍵は、IP アドレス、DN、FQDN、E メールアドレスなど多くのピア ID タイプのいずれでも使用できます。

公開証明書の IKEv2 ポリシー

kmf-policy.xml ファイルには IKEv2 の証明書検証ポリシーが含まれています。kmfcfg dbfile=/etc/inet/ike/kmf-policy.xml policy=default コマンドは、証明書検証ポリシーを変更するときに使用します。一般的な変更には、OCSP および CRL の使用、証明書検証中のネットワークタイムアウトの時間などがあります。また、管理者はポリシーによって有効日付の適用や鍵の使用要件など、証明書検証のさまざまな要素を変更できます。証明書検証のデフォルト要件をゆるめることはお勧めしません。

IKEv2 および FIPS 140-2

FIPS 140-2 が有効になったシステムでは、証明書を作成して IKEv2 を構成するとき、ユーザーが FIPS 140-2 承認アルゴリズムのみを使用するように選択する責任があります。このガイドの手順と例では、アルゴリズム any が指定されている場合を除き、FIPS 140-2 承認アルゴリズムを使用します。

次の暗号化アルゴリズムメカニズムは IKEv2 構成および事前共有鍵ファイルで利用可能であり、FIPS 140-2 モードの Oracle Solaris で使用することが承認されています。

- 鍵の長さが 128-256 ビットの CBC モードの AES
- 3DES

次の認証アルゴリズムメカニズムは IKEv2 構成および事前共有鍵ファイルで利用可能であり、FIPS 140-2 モードの Oracle Solaris で使用することが承認されています。

- SHA1
- SHA256
- SHA384
- SHA512

次のメカニズムは IKEv2 証明書で利用可能であり、FIPS 140-2 モードの Oracle Solaris で使用することが承認されています。

- 鍵の長さが 2048-3072 ビットの RSA
- 3 つの可能な曲線とそれに関連付けられたハッシュによる ECC を使用する ECDSA

ikev2cert gencert および ikev2cert gencsr コマンドの引数は次のようになります。

- keytype=ec curve=secp256r1 hash=sha256
- keytype=ec curve= secp384r1 hash=sha384
- keytype=ec curve=secp521r1 hash=sha512

詳細は、[ikev2cert\(1M\)](#) のマニュアルページを参照してください。

Oracle Solaris についての FIPS 140-2 承認アルゴリズムの最終的なリストについては、『[Oracle Solaris 11.3 での FIPS 140 対応システムの使用](#)』の「[Oracle Solaris システムでの FIPS 140-2 レベル 1 証明書のリファレンス](#)」を参照してください。

IKEv1 プロトコル

次のセクションでは、IKEv1 の概要について説明します。IKEv1 は、より高速でセキュリティ保護された鍵管理を提供する IKEv2 に置き換わっています。IKEv2 に

については、[169 ページの「IKEv2 プロトコル」](#)を参照してください。比較については、[168 ページの「IKEv2 と IKEv1 の比較」](#)を参照してください。両方のプロトコルに共通する情報については、[163 ページの「IKE の概要」](#)を参照してください。IKEv1 と IKEv2 は同時に実行可能で、ほかのシステム上のピアプロトコルとネゴシエートできます。

IKEv1 の鍵ネゴシエーション

IKEv1 デーモン `in.iked` は、セキュアな方法で鍵をネゴシエートして IPsec SA を認証します。IKEv1 は Perfect Forward Secrecy (PFS) を提供します。PFS では、データ伝送を保護する鍵を使用しないで追加鍵を取得します。また、データ伝送の鍵の作成に使用するシードを再利用しません。[in.iked\(1M\)](#) のマニュアルページを参照してください。

IKEv1 フェーズ 1 交換

IKEv1 プロトコルには 2 つのフェーズがあります。Oracle Solaris は、メインモードフェーズ 1 交換をサポートしています。メインモード交換は、2 つのピア間で ISAKMP セキュリティーアソシエーション (SA) を作成するのに使用できるパラメータをネゴシエートします。この ISAKMP SA は非対称の暗号化を使用して鍵情報を交換し、事前共有鍵または公開鍵証明書を使用してピアを認証します。IPsec SA とは異なり、ISAKMP SA は双方向であるため、1 つの SA だけです。

フェーズ 1 交換での IKEv1 による ISAKMP SA のネゴシエート方法は構成可能です。IKEv1 は、`/etc/inet/ike/config` ファイルから構成情報を読み取ります。次の構成情報があります。

- グローバルパラメータ (公開鍵証明書の名前など)
 - Perfect Forward Secrecy (PFS) が必要かどうか
 - このシステムの IKE ピア
 - フェーズ 1 交換を保護するアルゴリズム
 - 認証方式
- 認証方式には、事前共有鍵と公開鍵証明書の 2 つがあります。公開鍵証明書は、自己署名することも [認証局 \(CA\)](#) に発行してもらうこともできます。

詳細は、[ike.config\(4\)](#) のマニュアルページを参照してください。

IKEv1 フェーズ 2 交換

フェーズ 2 交換はクイックモードと呼ばれます。クイックモード交換は、IPsec SA を作成するのに必要な IPsec アルゴリズムと鍵情報をネゴシエートします。この交換はフェーズ 1 でネゴシエートされる ISAKMP SA によって保護 (暗号化) されています。

クイックモード交換のアルゴリズムおよびセキュリティープロトコルは、IPsec ポリシーファイル `/etc/inet/ipsecinit.conf` から取得されます。

IPsec SA は期限が切れると鍵が再生成されます。SA のライフタイムは、IPsec SA の作成時に `in.iked` デーモンによって設定されます。この値は構成可能です。

詳細は、[ipsecconf\(1M\)](#) および [in.iked\(1M\)](#) のマニュアルページを参照してください。

IKEv1 構成の選択

`/etc/inet/ike/config` 構成ファイルには、`in.iked` デーモンの構成が含まれています。構成はいくつかのルールで構成されています。各エントリには、このシステムが同様に構成された IKEv1 ピアに対して使用できるアルゴリズムや認証データなどのパラメータが含まれます。`in.iked` デーモンは、事前共有鍵および識別情報の公開鍵証明書をサポートしています。

エントリ `auth_method preshared` は、事前共有鍵が使用されることを示します。`auth_method` の値が `preshared` 以外の場合には、公開鍵証明書が使用されることを示します。

IKEv1 では、事前共有鍵は特定の IP アドレスまたはアドレス範囲に関連付けられています。鍵は、各システムの `/etc/inet/secret/ike.preshared` ファイルに保存されます。

詳細は、[165 ページの「IKE の動作」](#) および [ike.config\(4\)](#) と [ike.preshared\(4\)](#) のマニュアルページを参照してください。

IKEv2 の構成

この章では、使用するシステムに合わせてインターネット鍵交換バージョン 2 (IKEv2) を構成する方法について説明します。IKEv2 を構成して有効にすると、指定された IPsec エンドポイントの鍵情報が自動的に生成されます。この章では、次の内容について説明します。

- 175 ページの「IKEv2 の構成」
- 176 ページの「事前共有鍵による IKEv2 の構成」
- 181 ページの「IKEv2 の公開鍵証明書を格納するためのキーストアの初期化」
- 184 ページの「公開鍵証明書による IKEv2 の構成」

IKE の概要については、[第10章「インターネット鍵交換について」](#)を参照してください。IKE の参照情報については、[第14章「IPsec および鍵管理のリファレンス」](#)を参照してください。詳細な手順については、[ikeadm\(1M\)](#)、[pktool\(1\)](#)、[ikev2cert\(1M\)](#)、[ikev2.config\(4\)](#)、[in.ikev2d\(1M\)](#)、および [kmfcfg\(1\)](#) のマニュアルページの例を参照してください。

IKEv2 の構成

IKE の認証には、事前共有鍵、自己署名付き証明書、および認証局 (CA) の証明書を使用できます。ルールによって、特定の認証方法と保護されているエンドポイントが関連付けられます。したがって、1つのシステムに1つまたはすべての認証方法を使用できます。IKEv2 システムで IKEv1 を実行することもできます。通常は、IKEv2 をサポートしていないシステムで通信を保護するために IKEv1 を実行します。IKEv2 は、鍵および証明書のストレージに PKCS #11 ハードウェアトークンを使用することもできます。

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

IKEv2 を構成したら、[第9章「IPsec の構成」](#)のこれらの IKEv2 ルールを使用して鍵を管理する IPsec 手順を実行します。次のセクションでは、具体的な IKEv2 構成に注目します。

事前共有鍵による IKEv2 の構成

IKEv2 を使用するようにピアシステムまたはサブネットを構成し、さらに、あなたがこれらのサブネットの管理者である場合、事前共有鍵の使用が適しています。事前共有鍵は、テストの際にも使用される場合があります。詳細は、[165 ページの「IKE と事前共有鍵認証」](#)を参照してください。

▼ 事前共有鍵で IKEv2 を構成する方法

この手順では、名前 host1 と host2 を使用しているシステムの名前に置き換えてください。両方の IKE エンドポイントを構成します。

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなりモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#)および『[Oracle Solaris 11.3 での Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」を参照してください。

1. 各システムで `/etc/inet/ike/ikev2.config` ファイルを編集します。

```
# pfedit /etc/inet/ike/ikev2.config
```

2. ファイル内に、事前共有鍵を使用するルールを作成します。

注記 - 鍵は[ステップ 4](#)で作成します。

このファイルのルールおよびグローバルパラメータは、システムの `ipseccinit.conf` ファイル内の IPsec ポリシーの鍵を管理する必要があります。次の IKEv2 構成の例では、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティ保護する方法」](#)の `ipseccinit.conf` の例の鍵を管理します。

- a. たとえば、host1 システムの `ikev2.config` ファイルを変更します。

注記 - この例では、グローバルパラメータセクションの2つの変換を示します。ピアはこれらの変換のいずれかで構成できます。特定の変換を要求するには、その変換をルールに含めます。

```
### ikev2.config file on host1, 192.168.116.16

## Global parameters
# This default value will apply to all transforms that follow
#
ikesa_lifetime_secs 3600
#
## Global transform definitions. The algorithm choices are
## based on RFC 4921.
#
## Two transforms are acceptable to this system, Group 20 and Group 19.
## A peer can be configured with 19 or 20.
## To ensure that a particular peer uses a specific transform,
## include the transform in the rule.
##
# Group 20 is 384-bit ECP - Elliptic Curve over Prime
ikesa_xform { encr_alg aes(256..256) auth_alg sha384 dh_group 20 }
# Group 19 is 256-bit ECP
ikesa_xform { encr_alg aes(128..128) auth_alg sha256 dh_group 19 }
#
## The rule to communicate with host2
## Label must be unique
{ label "host1-host2"
  auth_method preshared
  local_addr 192.168.116.16
  remote_addr 192.168.13.213
}
```

b. host2 システムの ikev2.config ファイルを変更します。

```
## ikev2.config file on host2, 192.168.13.213
## Global Parameters
#
...
ikesa_xform { encr_alg aes(256..256) auth_alg sha384 dh_group 20 }
ikesa_xform { encr_alg aes(128..128) auth_alg sha256 dh_group 19 }
...
## The rule to communicate with host1
## Label must be unique
{ label "host2-host1"
  auth_method preshared
  local_addr 192.168.13.213
  remote_addr 192.168.116.16
}
```

3. 各システムで、ファイルの構文を確認します。

```
# /usr/lib/inet/in.ikev2d -c
```

4. 各システムの /etc/inet/ike/ikev2.preshared ファイルに事前共有鍵を追加します。



注意 - このファイルは特殊な権限を持ち、ikeuser によって所有されています。このファイルは決して削除したり置き換えたりしないでください。代わりに、ファイルが元のプロパティを保持できるように pfedit コマンドを使用して内容を編集してください。

- a. たとえば、host1 システムの ikev2.preshared ファイルは次のようになります。

```
# pfedit -s /etc/inet/ike/ikev2.preshared
## ikev2.preshared on host1, 192.168.116.16
# ...
## label must match the rule that uses this key
{ label "host1-host2"
## The preshared key can also be represented in hex
## as in 0x123...
  key "Str0ng p@wd tekniq ... "
}
```

pfedit コマンドのオプションについては、[pfedit\(1M\)](#) のマニュアルページを参照してください。

- b. host2 システムでは、ikev2.preshared ファイルは固有のラベル以外はほぼ同じです。

```
## ikev2.preshared on host2, 192.168.13.213
# ...
## label must match the label of the rule that uses this key
{ label "host2-host1"
## The preshared key can also be represented in hex
## as in 0x123...
  key "Str0ng p@wd tekniq ... "
}
```

5. IKEv2 サービスインスタンスを有効にします。

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

事前共有鍵を置き換えるときは、ピアシステムで事前共有鍵ファイルを編集して ikev2 サービスを再起動します。

```
# svcadm restart ikev2
```

例 36 ローカルとリモートで異なる IKEv2 共有鍵を使用する

この例では、IKEv2 の管理者がシステムごとに事前共有鍵を作成してそれらを交換し、各鍵を事前共有鍵ファイルに追加します。事前共有鍵エントリのラベルは ikev2.config ファイル内のルールとのラベルと一致します。その後、彼らは in.ikev2d デモンを再起動します。

相手システムの手前共有鍵を受け取ったあと、管理者は ikev2.preshared ファイルを編集します。host2 のファイルは次のとおりです。

```
# pfedit -s /etc/inet/ike/ikev2.preshared
#...
{ label "host2-host1"
  ## local and remote preshared keys
  local_key "P-LongISH to ch*angE ...)"
  remote_key "E-CHaNgE lE ...)"
}
```

したがって、host1 の ikev2.preshared 鍵ファイルは次のようになります。

```
#...
{ label "host1-host2"
  ## local and remote preshared keys
  local_key "E-CHaNgE lE ...)"
  remote_key "P-LongISH to ch*angE ...)"
}
```

管理者は各システムで IKEv2 サービスインスタンスを再起動します。

```
# svcadm restart ikev2
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#)を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティ保護する方法」](#)を参照してください。

その他の例については、[ikev2.config\(4\)](#) および [ikev2.preshared\(4\)](#) のマニュアルページを参照してください。

▼ IKEv2 で事前共有鍵を使用する場合に新しいピアを追加する方法

同じピア間で動作中の構成に対して IPsec ポリシーエントリを追加した場合、IPsec ポリシーサービスをリフレッシュする必要があります。IKE の再構成または再起動は不要です。

IPsec ポリシーに新しいピアを追加した場合、IPsec の変更に加えて IKEv2 構成を変更する必要があります。

始める前に ipsecinit.conf ファイルをリフレッシュし、ピアシステムの IPsec ポリシーをリフレッシュしました。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなりモートログイン手順について、例27「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.3 での Secure Shell アクセスの管理』の「Secure Shell を使用して ZFS をリモートで管理する方法」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 IKEv2 のルールを作成します。

- a. たとえば、host1 システムで、次の規則を /etc/inet/ike/ikev2.config ファイルに追加します。

```
# pfedit ikev2.config
## ikev2.config file on host1, 192.168.116.16
...
## The rule to communicate with host3
## Label must be unique
{label "host1-host3"
  auth_method preshared
  local_addr 192.168.116.16
  remote_addr 192.168.15.7
}
```

pfedit コマンドのオプションについては、[pfedit\(1M\)](#) のマニュアルページを参照してください。

- b. host3 システムで、次の規則を追加します。

```
## ikev2.config file on host3, 192.168.15.7
...
## The rule to communicate with host1
{label "host3-host1"
  auth_method preshared
  local_addr 192.168.15.7
  remote_addr 192.168.116.16
}
```

2. (オプション) 各システムで、ファイルの構文を検証します。

```
# /usr/lib/inet/in.ikev2d -c -f /etc/inet/ike/ikev2.config
```

3. ピアシステム用の IKEv2 事前共有鍵を作成します。

- a. host1 システムで、次の情報を /etc/inet/ike/ikev2.preshared ファイルに追加します。

```
# pfedit -s /etc/inet/ike/ikev2.preshared
## ikev2.preshared on host1 for the host3 interface
...
## The rule to communicate with host3
## Label must match the label of the rule
{ label "host1-host3"
  # host1 and host3's shared key
  key "Twas brill sE5 ..."
}
```

pfedit コマンドのオプションについては、[pfedit\(1M\)](#) のマニュアルページを参照してください。

- b. host3 システムで、次の情報を `ikev2.preshared` ファイルに追加します。

```
# ikev2.preshared on host3 for the host1 interface
# ...
{ label "host3-host1"
  # host3 and host1's shared key
  key "Twas brill sE5 ..."
}
```

4. 各システムで、変更をカーネルに読み込みます。

- サービスが有効になっている場合はリフレッシュします。

```
# svcadm refresh ikev2
```

- サービスが有効になっていない場合は有効にします。

```
# svcadm enable ikev2
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#)を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#)を参照してください。

IKEv2 の公開鍵証明書を格納するためのキーストアの初期化

IKEv2 で公開証明書を使用するには、PKCS #11 キーストアを作成する必要があります。もっとも一般的に使用されるキーストアは、Oracle Solaris の暗号化フレームワーク機能が提供する `pkcs11_softtoken` を使用します。

IKEv2 の `pkcs11_softtoken` キーストアは、特殊なユーザー `ikeuser` が所有するディレクトリにあります。デフォルトのディレクトリは `/var/user/ikeuser` です。ユーザー ID `ikeuser` はシステムに用意されていますが、キーストアは自分で作成する必要があります。キーストアを作成する場合は、キーストアの PIN を作成します。IKEv2 サービスがキーストアにログインするにはこの PIN が必要です。

`pkcs11_softtoken` キーストアは、IKEv2 で使用される非公開鍵、公開鍵、および公開証明書を保持しています。これらの鍵および証明書は、`pktool` コマンドのラッパーである `ikev2cert` コマンドで管理されます。このラッパーは、`ikeuser` が所有する `pkcs11_softtoken` キーストアにすべての鍵および証明書の操作が適用されるようにします。

PIN を ikev2 サービスのプロパティー値として追加していない場合は、/var/log/ikev2/in.ikev2d.log ファイルに次のメッセージが表示されます。

```
date: (n) No PKCS#11 token "pin" property defined
for the smf(5) service: ike:ikev2
```

公開鍵証明書を使用していない場合は、このメッセージを無視してかまいません。

▼ IKEv2 公開鍵証明書用キーストアを作成および使用する方

IKEv2 で公開証明書を使用する予定がある場合は、キーストアを作成する必要があります。キーストアを使用するには、ログインする必要があります。in.ikev2d デーモンが起動したら、ユーザーまたは自動プロセスがデーモンに PIN を提供します。サイトのセキュリティーで自動ログインを許可している場合は、それを構成する必要があります。デフォルトはキーストアを使用するための対話型ログインです。

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. IKEv2 キーストアに PIN を設定します。

ikev2cert setpin コマンドを使用して IKEv2 キーストアを作成します。このコマンドは、PKCS #11 キーストアの所有者を ikeuser に設定します。

PIN に空白を使用しないでください。たとえば、WhatShouldIWrite は有効ですが、「What Should」は有効ではありません。

```
% pfbash
# /usr/sbin/ikev2cert setpin
Enter token passphrase: changeme
Create new passphrase:          Type strong passphrase
Re-enter new passphrase: xxxxxxxx
Passphrase changed.
```



注意 - このパスフレーズを安全な場所に保管してください。キーストアを使用するにはそれが必要です。

2. 自動的に、または対話形式でキーストアにログインします。

自動ログインが推奨されます。サイトのセキュリティーポリシーが自動ログインを許可していない場合は、in.ikev2d デーモンを再起動する際に対話形式でキーストアにログインする必要があります。

- 自動ログインを有効にするようにキーストアを構成します。

- a. PIN を `pkcs11_softtoken/pin` サービスプロパティの値として追加します。

```
# svccfg -s ike:ikev2 editprop
```

一時編集ウィンドウが開きます。

- b. `setprop pkcs11_token/pin =` 行のコメントを解除します。

```
# setprop pkcs11_token/pin = astring: ()      Original entry
setprop pkcs11_token/pin = astring: ()      Uncommented entry
```

- c. 括弧を **ステップ 1** の PIN に置き換えます。

```
setprop pkcs11_token/pin = astring: PIN-from-Step-1
```

コロンと PIN の間にスペースを残します。

- d. ファイル最下部の `refresh` 行のコメントを解除して、変更を保存します。

```
# refresh
refresh
```

- e. (オプション) `pkcs11_token/pin` プロパティの値を検証します。

`pkcs11_token/pin` プロパティは、`ikeuser` 所有のキーストアにアクセスするときに確認される値を保持します。

```
# svccfg -s ike:ikev2 listprop pkcs11_token/pin
pkcs11_token/pin    astring    PIN
```

- 自動キーストアログインが構成されていない場合は、キーストアに手動でログインします。

`in.ikev2d` デーモンが起動するたびにこのコマンドを実行します。

```
# pfbash
# ikeadm -v2 token login "Sun Metaslot"
Enter PIN for PKCS#11 token 'Sun Metaslot':    Type the PIN from Step 1
ikeadm: PKCS#11 operation successful
```

3. (オプション) キーストアに PIN が設定されていることを確認します。

```
# ikev2cert tokens
Flags: L=Login required I=Initialized X=User PIN expired S=S0 PIN expired
Slot ID   Slot Name           Token Name           Flags
-----
1         Sun Crypto Softtoken Sun Software PKCS#11 softtoken LI
```

Flags 列の LI は、PIN が設定されていることを示します。

4. `pkcs11_softtoken` を手動でログアウトするには、`ikeadm` コマンドを使用します。

```
# ikeadm -v2 token logout "Sun Metaslot"
ikeadm: PKCS#11 operation successful
```

2つのサイト間の通信を一定の時間に制限するためにログアウトする場合があります。ログアウトによって秘密鍵が使用できなくなるため、新しい IKEv2 セッションを開始できません。既存の IKEv2 セッションは、`ikeadm delete ikesa` コマンドでセッション鍵を削除しないかぎり続きます。事前共有鍵ルールは引き続き機能します。[ikeadm\(1M\)](#) のマニュアルページを参照してください。

公開鍵証明書による IKEv2 の構成

公開証明書は大規模な配備に適しています。詳細については、[166 ページの「IKE と公開鍵証明書」](#)を参照してください。

公開鍵証明書は、暗号化フレームワークによってソフトトークンキーストアに格納されます。ハードウェアが接続されているシステムでは、証明書をハードウェア上で生成および格納することもできます。手順については、[196 ページの「ハードウェア上で IKEv2 の公開鍵証明書を生成および格納する方法」](#)を参照してください。

背景情報については、[165 ページの「IKE の動作」](#)を参照してください。

次のタスクマップは、IKEv2 の公開鍵証明書の作成手順を一覧表示したものです。この手順には、システムに Sun Crypto Accelerator 6000 ボードが接続されている場合にハードウェアキーストアに証明書を格納する方法が含まれています。

表 14 公開鍵証明書による IKEv2 の構成のタスクマップ

タスク	説明	参照先
証明書のキーストアを作成します。	IKEv2 の証明書を格納する PKCS #11 キーストアを初期化します。	181 ページの「IKEv2 の公開鍵証明書を格納するためのキーストアの初期化」
自己署名付き公開鍵証明書で IKEv2 を構成します。	自分で署名した公開鍵証明書を作成します。証明書をピアにエクスポートし、ピアの証明書をインポートします。	185 ページの「自己署名付き公開鍵証明書により IKEv2 を構成する方法」
CA の証明書で IKEv2 を構成します。	CSR を作成して、返されたすべての証明書をキーストアにインポートする必要があります。その後、IKE ピアの証明書を検証してインポートします。	190 ページの「CA の署名付き証明書で IKEv2 を構成する方法」
失効した証明書の処理方法を構成します。	CRL が使用されているか、および OCSP サーバーがポーリングされているか(ネットワーク遅延の処理方法を含む)を確認します。	193 ページの「IKEv2 で証明書検証ポリシーを設定する方法」
接続されたハードウェアのキーストアに証明書のストレージを構成します。	Sun Crypto Accelerator 6000 ボードを取り付けて、それを使用するように IKEv2 を構成します。	196 ページの「ハードウェア上で IKEv2 の公開鍵証明書を生成および格納する方法」

▼ 自己署名付き公開鍵証明書により IKEv2 を構成する方法

この手順では、公開鍵証明書を作成して署名します。非公開鍵および証明書は、IKEv2 の PKCS #11 ソフトトークンキーストアに格納されます。公開鍵証明書を IKE ピアに送信し、次に相手の公開証明書を送信してもらいます。

自己署名付き証明書を使用するすべての IKE システムでこの手順を実行します。

始める前に 証明書を使用するには、[182 ページの「IKEv2 公開鍵証明書用キーストアを作成および使用する方法」](#)を完了している必要があります。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルを使用している必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなりモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#)および『[Oracle Solaris 11.3 での Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」を参照してください。

1. キーストアに自己署名付き証明書を作成します。

ikev2cert gencert コマンドの引数の説明については、[pktool\(1\)](#) のマニュアルページで pktool gencert keystore=pkcs11 サブコマンドを確認してください。

subject 引数の形式については、[167 ページの「IKE での公開鍵証明書の使用」](#)を参照してください。

注記 - 証明書にラベルを付けます。ラベルはローカルキーストア内で証明書およびそれに対応する鍵を識別します。

a. たとえば、host2 システムでは、コマンドは次のようになります。

```
# pfbash
# ikev2cert gencert \
  label="IThost2" \
  subject="O=exampleco, OU=IT, C=US, CN=host2" \
  serial=0x87654321
  keytype=rsa
  keylen=2048
Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
```

次のエラーメッセージは、PIN の入力ミスまたはキーストアが初期化されていないことを示しています。

```
Error creating certificate and keypair:
keystore error: CKR_PIN_INCORRECT
```

```
libkmf error: KMF_ERR_AUTH_FAILED

Error creating certificate and keypair:
keystore error: CKR_PIN_EXPIRED: PIN expired and must be changed
libkmf error: KMF_ERR_BAD_PARAMETER: invalid parameter
```

ヒント - `pktool` コマンド構文の表示は、証明書エントリの一部が誤って入力されていることを示します。許可されていないアルゴリズムの使用、二重引用符と等号の抜け、およびその他の入力ミスがないかコマンドを確認してください。無効な引数を見つける 1 つの方法として、コマンドを取得したあと、引数を 1 回につき 1 つずつ削除します。

b. host1 システムでは、コマンドは次のようになります。

```
# ikev2cert gencert \
label=IHost1 \
subject="O=exampleco, OU=IT, C=US, CN=host1" \
serial=0x86428642
keytype=rsa
keylen=2048
Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
```

2. (オプション) 鍵および証明書を一覧表示します。

```
host1 # /usr/sbin/ikev2cert list objtype=both
Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
No.      Key Type      Key Len.      Key Label
-----
Asymmetric private keys:
1)      RSA                IHost1
Asymmetric public keys:
1)      RSA                IHost1
Certificates:
1) X.509 certificate
Label: IHost1
Subject: C=US, O=exampleco, OU=IT, CN=host1
Issuer: C=US, O=exampleco, OU=IT, CN=host1
Not Before: April 10 21:49:00 2014 GMT
Not After: April 10 21:49:00 2015 GMT
Serial: 0x86426420
Signature Algorithm: sha1WithRSAEncryption
X509v3 Subject Key Identifier:
 34:7a:3b:36:c7:7d:4f:60:ed:ec:4a:96:33:67:f2:ac:87:ce:35:cc
SHA1 Certificate Fingerprint:
 68:07:48:65:a2:4a:bf:18:f5:5b:95:a5:01:42:c0:26:e3:3b:a5:30
```

ヒント - デフォルトのハッシュアルゴリズムは SHA1 です。強力な署名アルゴリズムで証明書を作成するには、`keytype` オプションと、`keytype=rsa hash=sha384` などの別のハッシュアルゴリズムを使用します。オプションについては、[pktool\(1\)](#)のマニュアルページを参照してください。

3. 相手システムに証明書を送信します。

a. 各システムで、証明書だけをファイルにエクスポートします。

`outformat=pem` オプションを使用すると、公開証明書が直接インポートに適した形式のファイルに書き出されます。ラベルはキーストア内で証明書を識別します。

```
# cd /tmp
# ikev2cert export objtype=cert outformat=pem outfile=filename label=label
Enter PIN for Sun Software PKCS#11 softtoken:xxxxxxx
```

b. 証明書を電子メール、sftp、または ssh でほかのシステムに送信します。

たとえば、両方のシステムを管理している場合は `sftp` コマンドを使用して相手システムの証明書を取得します。

```
host1 # sftp jdoe@host2:/tmp/IHost2.pem /tmp/IHost2.pem.cert
host2 # sftp jdoe@host1:/tmp/IHost1.pem /tmp/IHost1.pem.cert
```

パスワードを入力するよう求められます。この例では、`jdoe` にパスワードを指定してください。

4. 証明書が同一であることを確認します。

キーストアに証明書をロードする前に、正しい証明書を受け取っていることを確認する必要があります。

a. 各システムにエクスポートされたファイルのダイジェストを作成します。

たとえば、`host2` の管理者は、`host2` の証明書を含むファイルのダイジェストをほかの管理者に電子メールで送信します。`host1` の管理者は、`host1` の証明書ファイルのダイジェストを電子メールで送信します。

```
host2 # digest -a sha1 /tmp/IHost2.pem
c6dbef4136c0141ae62110246f288e5546a59d86
```

```
host1 # digest -a sha1 IHost1.pem
6b288a6a6129d53a45057065bd02b35d7d299b3a
```

b. 相手のシステムで、最初のシステムの証明書を含むファイルに `digest` コマンドを実行します。

```
host1 # digest -a sha1 /tmp/IHost2.pem.cert
c6dbef4136c0141ae62110246f288e5546a59d86
```

```
host2 # digest -a sha1 /tmp/IHost1.pem.cert
6b288a6a6129d53a45057065bd02b35d7d299b3a
```

ダイジェストは一致する必要があります。一致しない場合は、ファイルをキーストアにインポートしないでください。証明書の妥当性を検証する別の方法については、[例38「公開鍵証明書とそのフィンガープリントで検証する」](#)を参照してください。

5. 検証のあと、相手システムの証明書を自身のキーストアにインポートします。

証明書をキーストアにインポートする場合、システム上で証明書を一意に識別するラベルを証明書に割り当てる必要があります。ラベルは、公開鍵とその公開鍵証明書を関連付けます。

```
host1 # ikev2cert import label=IHost2 infile=/tmp/IHost2.pem.cert
host2 # ikev2cert import label=IHost1 infile=/tmp/IHost1.pem.cert
```

6. (オプション) キーストア内のオブジェクトを一覧表示します。

このリストと [ステップ 2](#) のリストを比較します。たとえば、host1 キーストアに host2 の証明書が追加されています。

```
host1 # /usr/sbin/ikev2cert list objtype=both
Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
No.      Key Type      Key Len.      Key Label
-----
Asymmetric private keys:
1)      RSA              IHost1
Asymmetric public keys:
1)      RSA              IHost1
Certificates:
1) X.509 certificate
Label: IHost1
Subject: C=US, O=exampleco, OU=IT, CN=host1
Issuer: C=US, O=exampleco, OU=IT, CN=host1
Not Before: April 10 21:49:00 2014 GMT
Not After: April 10 21:49:00 2015 GMT
Serial: 0x86426420
Signature Algorithm: sha1WithRSAEncryption
X509v3 Subject Key Identifier:
34:7a:3b:36:c7:7d:4f:60:ed:ec:4a:96:33:67:f2:ac:87:ce:35:cc
SHA1 Certificate Fingerprint:
68:07:48:65:a2:4a:bf:18:f5:5b:95:a5:01:42:c0:26:e3:3b:a5:30

2) X.509 certificate
Label: IHost2
Subject: C=US, O=exampleco, OU=IT, CN=host2
Issuer: C=US, O=exampleco, OU=IT, CN=host2
Not Before: April 10 21:40:00 2014 GMT
Not After: April 10 21:40:00 2015 GMT
Serial: 0x87654321
Signature Algorithm: sha1WithRSAEncryption
X509v3 Subject Key Identifier:
ae:d9:c8:a4:19:68:fe:2d:6c:c2:9a:b6:06:55:b5:b5:d9:d9:45:c6
SHA1 Certificate Fingerprint:
83:26:44:29:b4:1f:af:4a:69:0d:87:c2:dc:f4:a5:1b:4f:0d:36:3b
```

7. 各システムで、証明書を IKEv2 ルールで使用します。

pfedit コマンドを使用して /etc/inet/ike/ikev2.config ファイルを編集します。

- a. たとえば、host2 システムでは、ikev2.config ファイルのルールは次のように表示されます。

```
## ... Global transform that applies to any rule without a declared transform
ikesa_xform { dh_group 21 auth_alg sha512 encr_alg aes }
## ... Any self-signed
## end-entity certificates must be present in the keystore or
## they will not be trusted.
```

```

{
  label "host2-host1"
  auth_method cert
  local_id DN = "O=exampleco, OU=IT, C=US, CN=host2"
  remote_id DN = "O=exampleco, OU=IT, C=US, CN=host1"
}
...

```

- b. host1 システムでは、ikev2.config ファイルの local_id の値に host1 の証明書の DN を使用します。

リモートパラメータには、host2 の証明書の DN をその値として使用します。label キーワードの値がローカルシステム上で一意であることを確認します。

```

...
ikesa_xform { dh_group 21 auth_alg sha512 encr_alg aes }
...
{
  label "host1-host2"
  auth_method cert
  local_id DN = "O=exampleco, OU=IT, C=US, CN=host1"
  remote_id DN = "O=exampleco, OU=IT, C=US, CN=host2"
}
...

```

8. (オプション) 各システムで、ikev2.config ファイルの妥当性を確認します。

```
# /usr/lib/inet/inikev2.d -c
```

続行する前に、入力ミスや不正確な記述を修正してください。

9. 各システムで、IKEv2 サービスインスタンスの状態を確認します。

```
# svcs ikev2
STATE          STIME          FMRI
disabled       Sep_07         svc:/network/ipsec/ike:ikev2
```

10. 各システムで、IKEv2 サービスインスタンスを有効にします。

```
host2 # svcadm enable ipsec/ike:ikev2

host1 # svcadm enable ipsec/ike:ikev2
```

例 37 ライフタイムの制限を付けた自己署名付き証明書を作成する

この例では、証明書が 2 年間有効であることを管理者が指定します。

```
# ikev2cert gencert \
> label=DBAuditV \
> serial=0x12893467235412 \
> subject="O=exampleco, OU=DB, C=US, CN=AuditVault" \
> altname=EMAIL=auditV@example.com \
> keytype=rsa hash=sha512 keylen=3072 \
> lifetime=2-year
```

例 38 公開鍵証明書をそのフィンガープリントで検証する

この例では、管理者が証明書のフィンガープリントを使用して証明書を検証します。この方法のデメリットは、管理者がフィンガープリントを表示する前に、ピアの証明書をキーストアにインポートする必要があることです。

管理者は証明書をインポートして、`ikev2cert list objtype=cert` コマンドでそれを一覧表示したあと、その出力から証明書のフィンガープリントをコピーして相手のシステム管理者に送信します。

```
SHA1 Certificate Fingerprint:
      83:26:44:29:b4:1f:af:4a:69:0d:87:c2:dc:f4:a5:1b:4f:0d:36:3b
```

検証が失敗した場合は、証明書をインポートした管理者が証明書とその公開鍵をキーストアから削除する必要があります。

```
# ikev2cert delete label=label-name
Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
1 public key(s) found, do you want to delete them (y/N) ? y
1 certificate(s) found, do you want to delete them (y/N) ? y
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#)を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#)を参照してください。

▼ CA の署名付き証明書で IKEv2 を構成する方法

多くの通信システムを保護する組織は、通常、認証局 (CA) による公開証明書を使用します。背景情報については、[166 ページの「IKE と公開鍵証明書」](#)を参照してください。

CA の証明書を使用するすべての IKE システムでこの手順を実行します。

始める前に 証明書を使用するには、[182 ページの「IKEv2 公開鍵証明書用キーストアを作成および使用する方法」](#)を完了している必要があります。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなリモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#) および『[Oracle Solaris 11.3 での](#)

[Secure Shell アクセスの管理](#) の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」を参照してください。

1. 書き込み可能なディレクトリに変更します。

次のエラーメッセージは、CSR ファイルをディスクに書き込めないことを示している場合があります。

```
Warning: error accessing "CSR-file"
```

たとえば、/tmp ディレクトリを使用します。

```
# cd /tmp
```

2. 証明書署名要求を作成します。

ikev2cert gencsr コマンドを使用して、証明書署名要求 (CSR) を作成します。このコマンドの引数については、[pktool\(1\)](#) のマニュアルページで `pktool gencsr keystore=pkcs11` サブコマンドを確認してください。

たとえば、次のコマンドは host2 システム上に CSR を含むファイルを作成します。

```
# pfbash
# /usr/sbin/ikev2cert gencsr \
keytype=rsa
keylen=2048
label=Example2m \
outcsr=/tmp/Example2mcsr1 \
subject="C=US, O=Example2Co\, Inc., OU=US-Example1m, CN=Example1m"
Enter PIN for Sun Software PKCS#11 softtoken: xxxxxxxx
```

3. (オプション) CA の Web フォームに貼り付けるために、CSR の内容をコピーします。

```
# cat /tmp/Example2mcsr1
-----BEGIN CERTIFICATE REQUEST-----
MIICKDCCAXoCAQAwTzELMAkGA1UEBhMCVVMxGzAZBgNVBAoTElBhcnR5Q292tcGFu
eSwgSw5jLjESMBAGA1UECzMJVVMtUGFydH1tMQ8wDQYDQDEwZQYXJ0eW0wgGEi
MA0GCQSqGSIB3DQEBAQUAA4IBDwAwggEKAoIBAQCMBINmgZ4XWUv2q1fshZUN/SLb
WNLXZxdKwt5e71o0w0jyby69eL7HE0QBUij73nTkXE3n4gx0jBZE+hvJ6G0CbREA
jgSqUp2US7Bn9XEcXRrs0c7MCFPrsA+hVIcNHpKNseU0U/rg+wzoo5hA1ixtWuXH
bYDeEWQI5t1ZgDZoCwGrdHEjwVYHfvz+a0WBjyZBY0ueBhXaa68QqS0SnRVDX56Q
3p4H/AR4h0dcSja72XmMKPU5p3RVb8n/hrfKj1DjiGYXD4D+WZxQ65xxCcnALvVH
nZHU1AtP7QHx4RX1QVNNwEsY6C95RX9297rNwLsYvp/86xWrqkT1NqVAeUKhAgMB
AAEwCwYJKoZiHvcNAQEF44IBAQB3R6rmZdqcgN8Tomyj2CFTdyAWixkIATXpLM1
GL5ghrnDvadD61M+vS1yhF1IcSNM8fLRrCHIktAmb8ITnggJ//rzbHq3jd1a/iQt
kgGoTXfz8j6B57Ud61+MBLiBSBy0QK4GIg80j1b9Kk5HsZ48mIoI/Qb7FFW4p9dB
JEU0eYhkaGtWJ21YNNvKg0e0cnSZy+xP9wa9WpfdS04TicLDw0Yq7koNnfL0IB
Fj2bt/wI7iZ1DcpwphsiwnW9K9YynAJZzHd1ULVpn5Kd7vSRz9youLLzSb+9i1g0
E43Dw0hRk6P/Uq0N4e1Zca4otezNxyEq1PZI7pJ5u000sb1w
-----END CERTIFICATE REQUEST-----
```

4. CSR を certificate authority (CA) に送信します

CA から CSR の送信方法を指示されることがあります。ほとんどの機関は、Web サイトに送信フォームを掲載しています。フォームの記入に当たっては、その送信が正当なものであることを証明する必要があります。通常は、CSR をフォームに貼り付けま

ヒント - 証明書を貼り付けるための拡張ボタンがある Web フォームもあります。CSR は PKCS#10 形式で生成されます。したがって、Web フォームの PKCS#10 と記載された部分を探します。

5. CA から受け取る各証明書をキーストアにインポートします。

`ikev2cert import` は証明書をキーストアにインポートします。

a. 公開鍵と CA から受信した証明書をインポートします。

```
# ikev2cert import objtype=cert label=Example1m1 infile=/tmp/Example1m1Cert
```

ヒント - 管理を容易にするために、インポートした証明書には元の CSR と同じラベルを割り当てます。

b. CA からのルート証明書をインポートします。

```
# ikev2cert import objtype=cert infile=/tmp/Example1m1CAcert
```

c. 中間 CA 証明書をキーストアにインポートします。

ヒント - 管理を容易にするために、インポートした中間証明書には元の CSR と同じラベルを割り当てます。

CA が各中間証明書を個別のファイルで送信してきた場合は、前述の証明書をインポートしたようにそれらをインポートします。ただし、CA が証明書チェーンを PKCS#7 ファイルとして送信している場合は、個々の証明書はそのファイルから抽出したあと、前述の証明書をインポートしたように各証明書をインポートする必要があります。

注記 - `openssl` コマンドを実行するには、`root` 役割になる必要があります。[openssl\(5\)](#) のマニュアルページを参照してください。

```
# openssl pkcs7 -in pkcs7-file -print_certs
# ikev2cert import objtype=cert label=Example1m1 infile=individual-cert
```

6. 証明書検証ポリシーを設定します。

証明書に CRL または OCSP のセクションが含まれている場合、サイト要件に従って証明書検証ポリシーを構成する必要があります。手順については、[193 ページの「IKEv2 で証明書検証ポリシーを設定する方法」](#)を参照してください。

7. 証明書を使用するすべての IKE システムで手順を完了したら、すべてのシステムで `ikev2` サービスを有効にします。

ピアシステムには、ルート証明書および構成済みの `ikev2.config` ファイルが必要です。

- 次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#)を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#)を参照してください。

▼ IKEv2 で証明書検証ポリシーを設定する方法

IKEv2 システムでの証明書の処理方法に関するいくつかの要素を構成できます。

- 始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなりモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#)および『[Oracle Solaris 11.3 での Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」を参照してください。

1. デフォルトの証明書検証ポリシーを表示します。

証明書ポリシーはインストール時に `/etc/inet/ike/kmf-policy.xml` ファイルに設定されます。このファイルは `ikeuser` によって所有され、`kmfcfg` コマンドを使用して変更します。デフォルトの証明書検証ポリシーは、CRL を `/var/user/ikeuser/crls` ディレクトリにダウンロードします。OCSP の使用もデフォルトで有効になっています。サイトがインターネットへの接続にプロキシを要求する場合は、プロキシを構成する必要があります。[194 ページの「IKEv2 で失効した証明書を処理する方法」](#)を参照してください。

```
# pfbash
# kmfcfg list dbfile=/etc/inet/ike/kmf-policy.xml policy=default
Policy Name: default
Ignore Certificate Validity Dates: false      Unknown purposes or applications for the certificate
Ignore Unknown EKUs: false
Ignore Trust Anchor in Certificate Validation: false
Trust Intermediate CAS as trust anchors: false
Maximum Certificate Path Length: 32
Certificate Validity Period Adjusted Time leeway: [not set]
Trust Anchor Certificate: Search by Issuer
Key Usage Bits: 0      Identifies critical parts of certificate
Extended Key Usage Values: [not set]        Purposes or applications for the certificate
HTTP Proxy (Global Scope): [not set]
Validation Policy Information:
```

```

Maximum Certificate Revocation Responder Timeout: 10
Ignore Certificate Revocation Responder Timeout: true
OCSP:
  Responder URI: [not set]
  OCSP specific proxy override: [not set]
  Use ResponderURI from Certificate: true
  Response lifetime: [not set]
  Ignore Response signature: false
  Responder Certificate: [not set]
CRL:
  Base filename: [not set]
  Directory: /var/user/ikeuser/crls
  Download and cache CRL: true
  CRL specific proxy override: [not set]
  Ignore CRL signature: false
  Ignore CRL validity date: false
IPsec policy bypass on outgoing connections: true
Certificate to name mapper name: [not set]
Certificate to name mapper pathname: [not set]
Certificate to name mapper directory: [not set]
Certificate to name mapper options: [not set]

```

2. 証明書で、変更する検証オプションを示す機能を確認します。

たとえば、CRL または OCSP URI を含む証明書は、使用する URI を指定して証明書の失効ステータスを確認する検証ポリシーを使用できます。タイムアウトを構成することもできます。

3. 構成可能なオプションについては、[kmfcfg\(1\)](#) のマニュアルページを確認してください。

4. 証明書検証ポリシーを構成します。

サンプルポリシーについては、[194 ページ](#)の「IKEv2 で失効した証明書を処理する方法」を参照してください。

▼ IKEv2 で失効した証明書を処理する方法

失効した証明書とは、なんらかの理由で効力を失った証明書です。失効した証明書を使用していると、セキュリティーのリスクとなります。証明書の失効を確認する際のオプションがあります。静的なリストを使用するか、HTTP プロトコルを介して失効を動的に確認できます。

始める前に CA から証明書を受け取ってインストールしていること。

失効した証明書を確認する CRL および OCSP メソッドに精通していること。詳細およびポインタについては、[166 ページ](#)の「IKE と公開鍵証明書」を参照してください。

Network IPsec Management 権利プロファイルが割り当てられている管理者になり、プロファイルシェルを使用する必要があります。詳細は、『[Oracle Solaris 11.3 でのユー](#)

『[ユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. **CA から受け取った証明書内の CRL および OCSP セクションを探します。**
CSR のラベルから証明書を識別できます。

```
# pfbash
# ikev2cert list objtype=cert | grep Label:
Enter PIN for Sun Software PKCS#11 softtoken:
Label: Example1m1
```

たとえば、次の出力の抜粋は、証明書内の CRL および OCSP URI の部分を表示しています。

```
# ikev2cert list objtype=cert label=Example1m1
X509v3 extensions:
...
X509v3 CRL Distribution Points:
Full Name:
URI:http://onsitecrl.PKI.example.com/OCIPsec/LatestCRL.cr1
X509v3 Authority Key Identifier:
...
Authority Information Access:
OCSP - URI:http://ocsp.PKI.example.com/revokes/
X509v3 Certificate Policies:
Policy: 2.16.840.1.113733.1.7.23.2
```

CRL Distribution Points エントリで、URI 値はこの組織の CRL が Web 上のファイルで利用可能であることを示しています。OCSP エントリは、個々の証明書のステータスを OCSP サーバーから動的に判断できることを示しています。

2. **プロキシを指定して CRL または OCSP サーバーの使用を有効にします。**

```
# kmfcfg modify \
dbfile=/etc/inet/ike/kmf-policy.xml \
policy=default \
http-proxy=www-proxy.ja.example.com:80
```

プロキシがオプションのサイトでは、プロキシを指定する必要はありません。

3. **証明書検証ポリシーが更新されていることを確認します。**
たとえば、OCSP が更新されたことを確認します。

```
# kmfcfg list \
dbfile=/etc/inet/ike/kmf-policy.xml \
policy=default
...
OCSP:
Responder URI: [not set]
Proxy: www-proxy.ja.example.com:80
Use ResponderURI from Certificate: true
Response lifetime: [not set]
Ignore Response signature: false
Responder Certificate: [not set]
```

4. **IKEv2 サービスを再起動します。**

```
# svcadm restart ikev2
```

5. (オプション) CRL または OCSP の使用を停止します。

- CRL の使用を停止するには、次を入力します。

```
# pfexec kmfcfg modify \  
dbfile=/etc/inet/ike/kmf-policy.xml policy=default \  
crl-none=true
```

crl-none=true 引数は、ローカルキャッシュからダウンロードした CRL を使用するようにシステムに強制します。

- OCSP の使用を停止するには、次を入力します。

```
# pfexec kmfcfg modify \  
dbfile=/etc/inet/ike/kmf-policy.xml policy=default \  
ocsp-none=true
```

例 39 システムが IKEv2 証明書検証を待機する時間を変更する

この例では、管理者が証明書の検証の待機を 20 秒に制限します。

```
# kmfcfg modify dbfile=/etc/inet/ike/kmf-policy.xml policy=default \  
cert-revoke-responder-timeout=20
```

デフォルトでは、応答がタイムアウトすると、ピアの認証が成功します。ここでは、認証に失敗すると接続が拒否されるポリシーを管理者が構成します。この構成では、OCSP または CRL サーバーが応答しなくなると証明書検証が失敗します。

```
# kmfcfg modify dbfile=/etc/inet/ike/kmf-policy.xml policy=default \  
ignore-cert-revoke-responder-timeout=false
```

ポリシーをアクティブ化するには、管理者が IKEv2 サービスを再起動します。

```
# svcadm restart ikev2
```

▼ ハードウェア上で IKEv2 の公開鍵証明書を作成および格納する方法

公開鍵証明書はまた、接続されたハードウェアに格納できます。Sun Crypto Accelerator 6000 ボードによってストレージが提供され、公開鍵の操作をシステムからこのボードにオフロードできます。

ハードウェア上で公開鍵証明書を作成および格納することは、システム上で公開鍵証明書を生成および格納することと似ています。ハードウェアでは、ハードウェアキーストアの識別に `ikev2cert gencert token=hw-keystore` コマンドが使用されます。

始める前に この手順では、Sun Crypto Accelerator 6000 ボードがシステムに接続されていると仮定します。また、この手順ではボード用のソフトウェアがインストールされ、ハード

ウェアキーストアが構成されているものと仮定します。手順については、[Sun Crypto Accelerator 6000 ボード製品のライブラリドキュメント \(http://docs.oracle.com/cd/E19321-01/index.html\)](http://docs.oracle.com/cd/E19321-01/index.html)を参照してください。これらの手順にはキーストアの設定が含まれています。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなりモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#)および『[Oracle Solaris 11.3 での Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」を参照してください。

1. 接続されている Sun Crypto Accelerator 6000 ボードのトークン ID を持っていることを確認します。

```
# pfbash
# ikev2cert tokens

Flags: L=Login required I=Initialized X=User PIN expired S=S0 PIN expired
Slot ID  Slot Name                               Token Name                               Flags
-----  -
1         sca6000                                       sca6000                                  LI
2         n2cp/0 Crypto Accel Bulk 1.0               n2cp/0 Crypto Accel Bulk 1.0
3         ncp/0 Crypto Accel Asym 1.0                ncp/0 Crypto Accel Asym 1.0
4         n2rng/0 SUNW_N2_Random_Number_Ge          n2rng/0 SUNW_N2_RNG
5         Sun Crypto Softtoken                       Sun Software PKCS#11 softtoken         LI
```

2. 自己署名付き証明書または CSR を生成して、トークン ID を指定します。

注記 - Sun Crypto Accelerator 6000 ボードは、RSA で最大 2048 ビットの鍵をサポートします。DSA の場合、このボードは最大 1024 ビットの鍵をサポートします。

次のオプションのいずれかを選択します。

- 自己署名付き証明書の場合は、この構文を使用します。

```
# ikev2cert gencert token=sca6000 keytype=rsa \
hash=sha256 keylen=2048 \
subject="CN=FortKnox, C=US" serial=0x6278281232 label=goldrepo
Enter PIN for sca6000: See Step 3
```

- 証明書署名要求の場合は、この構文を使用します。

```
# ikev2cert gencsr token=sca6000 -i
> keytype=
> hash=
> keylen=
> subject=
> serial=
> label=
> outcsr=
```

Enter PIN for sca6000 token: See Step 3

ikev2cert コマンドの引数については、[pktool\(1\)](#) のマニュアルページを参照してください。

3. PIN のプロンプトに、Sun Crypto Accelerator 6000 ユーザー名、コロン、およびユーザーのパスワードを入力します。

注記 - キーストアのユーザー名とパスワードを知っている必要があります。

Sun Crypto Accelerator 6000 ボードがユーザー admin、パスワード inThe%4ov で構成されている場合は、次を入力します。

```
Enter PIN for sca6000 token: admin:inThe%4ov
-----BEGIN X509 CERTIFICATE-----
MIIBuDCCASECAQAwSTELMAkGA1UEBhMCMVVMxFTATBgNVBAoTDFBhcnR5Q29tcGFu
...
oKUDBbZ90/pLWYGr
-----END X509 CERTIFICATE-----
```

4. 通信先に証明書を送信します。

次のオプションのいずれかを選択します。

- リモートシステムに自己署名付き証明書を送信します。
証明書は、電子メールのメッセージに貼り付けることもできます。
- CA に証明書署名要求を送信します。
CA の指示に従って CSR を送信します。詳細な説明は、[167 ページの「IKE での公開鍵証明書の使用」](#)を参照してください。

5. 証明書をハードウェアキーストアにインポートします。

CA から受け取った証明書をインポートして、[ステップ 3](#) のユーザーおよび PIN を指定します。

```
# ikev2cert import token=sca6000 infile=/tmp/DCA.ACCEL.CERT1
Enter PIN for sca6000 token: Type user:password
# ikev2cert import token=sca6000 infile=/tmp/DCA.ACCEL.CA.CERT
Enter PIN for sca6000 token: Type user:password
```

6. 自動的に、または対話形式で使用するハードウェアキーストアを有効にします。

自動ログインが推奨されます。サイトのセキュリティポリシーが自動ログインを許可していない場合は、in.ikev2d デーモンを再起動する際に対話形式でキーストアにログインする必要があります。

- キーストアへの自動ログインを構成します。
 - a. PIN を pkcs11_token/uri サービスプロパティの値として追加します。

このプロパティの説明については、[258 ページの「IKEv2 サービス」](#)を参照してください。

```
# svccfg -s ike:ikev2 editprop
```

一時編集ウィンドウが開きます。

- b. `setprop pkcs11_token/uri` = 行のコメントを解除し、括弧を次の形式のトークンの名前に置き換えます。

```
# setprop pkcs11_token/uri = ()      Original entry
setprop pkcs11_token/uri = pkcs11:token=sca6000
```

- c. `setprop pkcs11_token/pin` = 行のコメントを解除し、括弧を[ステップ 3](#)の `username:PIN` に置き換えます。

```
# setprop pkcs11_token/pin = ()      Original entry
setprop pkcs11_token/pin = admin:PIN-from-Step-3
```

- d. ファイル最下部の `refresh` 行のコメントを解除して、変更を保存します。

```
# refresh
refresh
```

- e. (オプション) `pkcs11_token` プロパティの値を確認します。

```
# svccfg -s ikev2 listprop pkcs11_token
pkcs11_token/pin      astring      username:PIN
pkcs11_token/uri     astring      pkcs11:token=sca6000
```

- 自動ログインが構成されていない場合は、ハードウェアキーストアに手でログインします。

`in.ikev2d` デーモンが起動するたびにこのコマンドを実行します。

```
# pfexec ikeadm -v2 token login sca6000
Enter PIN for sca6000 token: admin:PIN-from-Step-3
ikeadm: sca6000 operation successful
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#)を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#)を参照してください。

IKEv1 の構成

この章では、使用するシステムに合わせてインターネット鍵交換バージョン 1 (IKEv1) を構成する方法について説明します。IKEv1 を構成すると、ネットワークでの IPsec の鍵情報が自動的に生成されます。この章では、次の内容について説明します。

- 202 ページの「事前共有鍵による IKEv1 の構成」
- 206 ページの「公開鍵証明書による IKEv1 の構成」
- 223 ページの「移動体システム用の IKEv1 の構成」
- 229 ページの「接続したハードウェアを検出するように IKEv1 を構成する」

注記 - IKEv2 のみを実装する予定である場合は、[第11章「IKEv2 の構成」](#)に進みます。

IKE の概要については、[第10章「インターネット鍵交換について」](#)を参照してください。IKE の参照情報については、[第14章「IPsec および鍵管理のリファレンス」](#)を参照してください。詳細な手順については、[ikeadm\(1M\)](#)、[ikecert\(1M\)](#)、および [ike.config\(4\)](#) のマニュアルページで用例のセクションを参照してください。

注記 - これらのタスクは、システムが静的 IP アドレスを割り当てられ、ネットワーク構成プロファイル DefaultFixed を実行していることを前提としています。netadm list コマンドが Automatic を返す場合は、[netcfg\(1M\)](#) のマニュアルページで詳細を確認してください。

IKEv1 の構成

IKE の認証には、事前共有鍵、自己署名付き証明書、および認証局 (CA) の証明書を使用できます。ike/config ファイル内のルールが、特定の IKEv1 認証方法と IKEv1 ピアを関連付けます。したがって、1つのシステムに1つまたはすべての IKE 認証方法を使用できます。PKCS #11 ライブラリへのポインタによって、IKEv1 は、接続されたハードウェアアクセラレータを使用できます。

IKEv1 を構成したら、[第9章「IPsec の構成」](#)の IKEv1 構成を使用する IPsec タスクを実行します。

事前共有鍵による IKEv1 の構成

IKEv1 を使用するようにピアシステムまたはサブネットを構成し、さらに、あなたがこれらのサブネットの管理者である場合、事前共有鍵の使用が適しています。事前共有鍵は、テストの際にも使用される場合があります。詳細は、[165 ページの「IKE と事前共有鍵認証」](#)を参照してください。

▼ 事前共有鍵で IKEv1 を構成する方法

IKE 実装では、鍵の長さが異なるさまざまなアルゴリズムが提供されます。鍵の長さは、サイトのセキュリティーに応じて選択します。一般的に、鍵の長さが長いほど、セキュリティーが高くなります。

この手順では、ASCII 形式の鍵を生成します。

これらの手順には、システム名 `host1` および `host2` を使用します。名前 `host1` と `host2` を使用しているシステムの名前に置き換えてください。

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

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなりモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#)および『[Oracle Solaris 11.3 での Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」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 IKEv1 構成の例は、[139 ページの「IPsec によって 2 つのサーバー間でネットワークラフィックをセキュリティー保護する方法」](#)の `ipsecinit.conf` の例で機能します。

- a. たとえば、host1 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 host1, 192.168.116.16

## Global parameters
#
## Defaults that individual rules can override.
p1_xform
{ auth_method preshared oakley_group 14 auth_alg sha encr_alg 3des }
p2_pfs 14
#
## The rule to communicate with host2
# Label must be unique
{ label "host1-host2"
  local_addr 192.168.116.16
  remote_addr 192.168.13.213
  p1_xform
  { auth_method preshared oakley_group 14 auth_alg sha256 encr_alg aes }
  p2_pfs 14
}
```

- b. host2 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 host2, 192.168.13.213
## Global Parameters
#
p1_xform
{ auth_method preshared oakley_group 14 auth_alg sha encr_alg 3des }
p2_pfs 14

## The rule to communicate with host1
# Label must be unique
{ label "host2-host1"
  local_addr 192.168.13.213
  remote_addr 192.168.116.16
  p1_xform
  { auth_method preshared oakley_group 14 auth_alg sha256 encr_alg aes }
  p2_pfs 14
}
```

3. 各システムで、ファイルの構文を確認します。

```
# /usr/lib/inet/in.iked -c -f /etc/inet/ike/config
```

4. 事前共有鍵を各システムの `/etc/inet/secret/ike.preshared` ファイルに追加します。

- a. たとえば、host1 システムの `ike.preshared` ファイルは次のようになります。

```
## ike.preshared on host1, 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 0x123...
  key "Str0ng p@wd tekniq ... "
```

```
}

```

b. host2 システムの `ike.preshared` ファイルは次のようになります。

```
## ike.preshared on host2, 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 0x123...
  key "Str0ng p@wd tekniq ... "
}
```

5. IKEv1 サービスを有効にします。

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

例 40 IKEv1 事前共有鍵をリフレッシュする

IKEv1 管理者が事前共有鍵をリフレッシュするときは、ピアシステム上のファイルを編集し、`in.iked` デーモンを再起動します。

最初に、事前共有鍵を使用する 2 つのサブネット内のすべてのシステムで、管理者が事前共有鍵エントリを変更します。

```
# pfedit -s /etc/inet/secret/ike.preshared
...
{ localidtype IP
  localid 192.168.116.0/24
  remoteidtype IP
  remoteid 192.168.13.0/24
  # The two subnet's shared passphrase for keying material
  key "P-LongISH to ch*angE ...)"
}
```

次に、管理者は各システムの IKEv1 サービスを再起動します。

`pfedit` コマンドのオプションについては、[pfedit\(1M\)](#) のマニュアルページを参照してください。

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

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#)を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティ保護する方法」](#)を参照してください。

▼ 新規ピアシステムのために IKEv1 を更新する方法

同じピア間で動作中の構成に対して IPsec ポリシーエントリを追加した場合、IPsec ポリシーサービスをリフレッシュする必要があります。IKEv1 の再構成または再起動は不要です。

IPsec ポリシーに新しいピアを追加した場合、IPsec の変更に加えて IKEv1 構成を変更する必要があります。

始める前に ipsecinit.conf ファイルをリフレッシュし、ピアシステムの IPsec ポリシーをリフレッシュしました。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなりモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#)および『[Oracle Solaris 11.3 での Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」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 IKEv1 のルールを作成します。**
 - a. **たとえば、host1 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 host1, 192.168.116.16
...
## The rule to communicate with host3

{label "host1-to-host3"
 local_addr 192.168.116.16
 remote_addr 192.168.15.7
 p1_xform
 {auth_method preshared oakley_group 14 auth_alg sha256 encr_alg aes}
 p2_pfs 14
 }
```

- b. **host3 システムで、次の規則を追加します。**

```
### ike/config file on host3, 192.168.15.7
...
## The rule to communicate with host1

{label "host3-to-host1"
 local_addr 192.168.15.7
 remote_addr 192.168.116.16
 p1_xform
 {auth_method preshared oakley_group 14 auth_alg sha256 encr_alg aes}
 p2_pfs 14
 }
```

2. ピアシステム用の IKEv1 事前共有鍵を作成します。

- a. **host1** システムで、`/etc/inet/secret/ike.preshared` ファイルに次の情報を追加します。

```
## ike.preshared on host1 for the host3 interface
##
{ localidtype IP
  localid 192.168.116.16
  remoteidtype IP
  remoteid 192.168.15.7
  # host1 and host3's shared key
  key "Twas brill sE5 ..."
}
```

- b. **host3** 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 host3 for the host1 interface
##
{ localidtype IP
  localid 192.168.15.7
  remoteidtype IP
  remoteid 192.168.116.16
  # host3 and host1's shared key
  key "Twas brill sE5 ..."
}
```

3. 各システムで、ike サービスをリフレッシュします。

```
# svcadm refresh ike:default
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#)を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#)を参照してください。

公開鍵証明書による IKEv1 の構成

公開鍵証明書を使用すると、通信するシステムが秘密鍵情報を帯域外で共有する必要がなくなります。認証局 (CA) からの公開鍵証明書では、通常、外部機関とのネゴシエーションが必要となります。この証明書は非常に簡単に拡大できるため、通信するシステムを数多く保護できます。

公開鍵証明書はまた、接続されたハードウェア内で生成して格納できます。手順については、[229 ページの「接続したハードウェアを検出するように IKEv1 を構成する」](#)を参照してください。

すべての証明書には、X.509 distinguished name (DN)、形式の一意の名前がありますまた、証明書には、電子メールアドレス、DNS 名、IP アドレスなどのサブジェクトの

別名が1つまたは複数ある場合があります。IKEv1 構成内の証明書は、完全な DN またはサブジェクトの別名のいずれかによって識別できます。これらの別名の形式は `tag=value` で、値の形式はそのタグのタイプに対応します。たとえば、`email` タグの形式は `name@domain.suffix` です。

次のタスクマップは、IKEv1 の公開鍵証明書の作成手順を一覧表示したものです。これらの手順には、接続されたハードウェア上で証明書を高速化および格納する方法が含まれます。

表 15 公開鍵証明書による IKEv1 の構成のタスクマップ

タスク	説明	参照先
自己署名付き公開鍵証明書で IKEv1 を構成します。	システムごとに鍵および2つの証明書を作成および格納します。 <ul style="list-style-type: none"> ■ 自己署名付き証明書およびその鍵 ■ ピアシステムからの公開鍵証明書 	207 ページの「自己署名付き公開鍵証明書により IKEv1 を構成する方法」
認証局で IKEv1 を構成します。	証明書署名要求を作成して、CA からの証明書を各システムに配置します。167 ページの「IKE での公開鍵証明書の使用」を参照してください。	212 ページの「CA の署名付き証明書で IKEv1 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。 <ul style="list-style-type: none"> ■ ローカルハードウェアで自己署名付き証明書を生成してから、リモートシステムからの公開鍵をハードウェアに追加する。 ■ ローカルハードウェアで証明書署名要求を生成してから、CA からの公開鍵証明書をハードウェアに追加します。 	217 ページの「ハードウェア上で IKEv1 の公開鍵証明書を生成および格納する方法」
CA からの証明書失効リスト (CRL) を更新します。	中央の配布ポイントから CRL にアクセスします。	221 ページの「IKEv1 で失効した証明書を処理する方法」

注記 - 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」の手順に従います。

公開鍵証明書は Trusted Extensions システムの大域ゾーン内で管理されます。Trusted Extensions は証明書を管理および格納する方法を変更しません。

▼ 自己署名付き公開鍵証明書により IKEv1 を構成する方法

この手順では、公開鍵/非公開鍵および証明書、すなわち証明書ペアを作成します。非公開鍵はディスク上のローカル証明書データベースに格納され、`ikecert certlocal`

コマンドを使用して参照できます。公開鍵および証明書は、公開証明書データベースに格納されます。これは `ikecert certdb` コマンドを使用して参照できます。公開証明書をピアシステムと交換します。この2つの証明書が IKEv1 転送の認証に使用されます。

自己署名付き証明書は、CA からの公開鍵証明書よりもオーバーヘッドが少ないのですが、あまり簡単には拡大できません。CA から発行される証明書とは異なり、自己署名付き証明書は証明書を交換した2人の管理者によって確認される必要があります。

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。

リモートで管理する場合は、セキュアなりモートログイン手順について、例27「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.3 での Secure Shell アクセスの管理』の「Secure Shell を使用して ZFS をリモートで管理する方法」を参照してください。

1. 各 IKEv1 システムで、自己署名付き証明書を `ike.privatekeys` データベース内に作成します。

`ikecert certlocal` コマンドの引数については、`ikecert(1M)` のマニュアルページを参照してください。

a. たとえば、`host2` システムでは、コマンドは次のようになります。

```
# ikcert certlocal -ks -m 2048 -t rsa-sha512 \
-D "O=exampleco, OU=IT, C=US, CN=host2" \
-A IP=192.168.13.213
Creating private key.
Certificate added to database.
-----BEGIN X509 CERTIFICATE-----
MIIC1TCCAb2gAwIBAgIEfdZgKjANBgkqhkiG9w0BAQUFADAaMRgwFgYDVQQDEw9T
a...+
ZBGi4QkNdI3f
-----END X509 CERTIFICATE-----
```

ここでは

- ks 自己署名付き証明書を作成します。
- m *keysize* 鍵のサイズを指定します。
- t *keytype* 使用するアルゴリズムのタイプを指定します。
- D *dname* 証明書サブジェクトの X.509 識別名 (DN) を指定します。例は、167 ページの「IKE での公開鍵証明書の使用」を参照してください。

`-A altname` 証明書の別名またはニックネームを指定します。 `altname` の形式は `tag=value` です。有効なタグは IP、DNS、email、および DN です。

注記 -D および -A オプションの値は、192.168.13.213 のように、システムではなく、証明書だけを識別する名前です。実際、これらは証明書ニックネームのため、正しい証明書がピアシステムにインストールされていることを帯域外で検証する必要があります。

b. host1 システムでは、コマンドは次のようになります。

```
# ikcert certlocal -ks -m 2048 -t rsa-sha512 \
-D "O=exampleco, OU=IT, C=US, CN=host1" \
-A IP=192.168.116.16
Creating private key.
Certificate added to database.
-----BEGIN X509 CERTIFICATE-----
MIIC1TCCAb2gAwIBAgIEB15JnjANBgkqhkiG9w0BAQUFADAaMRgwFgYDVQQDEw9T
...
y85m6LHJYtC6
-----END X509 CERTIFICATE-----
```

2. 証明書を保存し、リモートシステムに送信します。

出力は、証明書の公開部分のエンコード済みバージョンです。この証明書は、電子メールのメッセージに貼り付けても安全です。[ステップ 4](#) で示すように、受け取り側は正しい証明書をインストールしていることを帯域外で検証する必要があります。

a. たとえば、host2 証明書の公開部分を host1 管理者に送信します。

```
To: admin@host1.ja.example.com
From: admin@host2.us.example.com
Message: -----BEGIN X509 CERTIFICATE-----
MIIC1TCCAb2gAwIBAgIEfdZgKjANBgkqhkiG9w0BAQUFADAaMRgwFgYDVQQDEw9T
a...+
ZBGi4QkNdI3f
-----END X509 CERTIFICATE-----
```

b. host1 管理者から、host1 証明書の公開部分が送信されます。

```
To: admin@host2.us.example.com
From: admin@example.ja.example.com
Message: -----BEGIN X509 CERTIFICATE-----
MIIC1TCCAb2gAwIBAgIEB15JnjANBgkqhkiG9w0BAQUFADAaMRgwFgYDVQQDEw9T
...
y85m6LHJYtC6
-----END X509 CERTIFICATE-----
```

3. 各システムで、受け取った証明書を公開鍵データベースに追加します。

a. root が読み取るファイルに管理者の電子メールを保存します。

b. ファイルを `ikecert` コマンドにリダイレクトします。

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

このコマンドは、BEGIN タグと END タグの間にあるテキストをインポートします。

4. その証明書がその管理者からのものであることを相手の管理者に確認します。

たとえば、ほかの管理者に電話して、自分が持つ公開証明書のハッシュが、ほかの管理者のみが持つ非公開証明書のハッシュに一致することを検証できます。

a. `host2` 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 distinguished name (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。

```
host2 # 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=host2>   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
host2 # ikecert certlocal -l
Local ID Slot Name: 0   Key Type: rsa
  Key Size: 2048
  Public key hash: 80829EC52FC5BA910F4764076C20FDCF   Note 3

Local ID Slot Name: 1   Key Type: rsa-sha512
  Key Size: 2048
  Public key hash: FEA65C5387BBF3B2C8F16C019FEBC388

Local ID Slot Name: 2   Key Type: rsa
  Key Size: 2048
  Public key hash: 2239A6A127F88EE0CB40F7C24A65B818
```

このチェックでは、`host2` 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. `host1` システムが `host2` 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。

前のステップの `host2` における Note 3 のハッシュと、`host1` における Note 4 を比較します。

```

host1 # 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=host1>
  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=host2>
  Key Size: 2048
  Public key hash: 80829EC52FC5BA910F4764076C20FDCF      Note 4

```

host1 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、前のステップの host2 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、host2 システムの 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=host1"
# 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 14 auth_alg sha256 encr_alg 3des }
p2_pfs 14

{
  label "US-host2 to JA-host1;"
  local_id_type dn
  local_id "O=exampleco, OU=IT, C=US, CN=host2"
  remote_id "O=exampleco, OU=IT, C=US, CN=host1"
  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. **host1** システムで、**ike/config** ファイルにローカルパラメータの **host1** 値を追加します。

リモートパラメータには、**host2** 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-host1 to US-host2"
  local_id_type dn
  local_id "O=exampleco, OU=IT, C=US, CN=host1"
  remote_id "O=exampleco, OU=IT, C=US, CN=host2"

  local_addr 192.168.116.16
  remote_addr 0.0.0.0/0

  p1_xform
  {auth_method rsa_sig oakley_group 2 auth_alg sha256 encr_alg aes}
}
```

6. **ピアシステムで、IKEv1 を有効にします。**

```
host2 # svcadm enable ipsec/ike:default
host1 # svcadm enable ipsec/ike
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#)を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#)を参照してください。

▼ CA の署名付き証明書で IKEv1 を構成する方法

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなリモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#)および『[Oracle Solaris 11.3 での Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」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署名要求 (CSR) を作成します。コマンドの引数については、[207 ページの「自己署名付き公開鍵証明書により IKEv1 を構成する方法」](#)の **ステップ 1** を参照してください。

```
# ikcert certlocal -kc -m keysize -t keytype \
```

```
-D dname -A altname
```

a. たとえば、次のコマンドは `host2` システムに CSR を作成します。

```
# ikecert certlocal -kc -m 2048 -t rsa-sha384 \
> -D "C=US, O=Example2Co\, Inc., OU=US-Example2m, CN=Example2m" \
> -A "DN=C=US, O=Example2Co\, Inc., OU=US-Example2m"
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-----
MIIBYjCCATMCAQAuZELMAkGA1UEBhMCVVMxHTAbBgNVBAoTFEV4YVw1wbGVDb21w
...
IcM+tw0ThRrfuJX9t/Qa1R/KxR1MA3zck080m09X
-----END CERTIFICATE REQUEST-----
```

b. 次のコマンドは `host1` システムに CSR を作成します。

```
# ikecert certlocal -kc -m 2048 -t rsa-sha384 \
> -D "C=JA, O=Example1Co\, Inc., OU=JA-Example1x, CN=Example1x" \
> -A "DN=C=JA, O=Example1Co\, Inc., OU=JA-Example1x"
Creating software private keys.
...
Finished successfully.
-----BEGIN CERTIFICATE REQUEST-----
MIIBuDCASECAQAwSTELMAkGA1UEBhMCVVMxFTATBgNVBAoTDFBhcnR5Q29tcGFu
...
8q1qdjaStLGfhD00
-----END CERTIFICATE REQUEST-----
```

2. CA に CSR を送信します。

CA から CSR の送信方法を指示されることがあります。ほとんどの機関は、Web サイトに送信フォームを掲載しています。フォームの記入に当たっては、その送信が正当なものであることを証明する必要があります。通常は、CSR をフォームに貼り付けます。要求が組織によって確認されている場合は、組織から署名付き証明書が発行されます。詳細は、[167 ページの「IKE での公開鍵証明書の使用」](#)を参照してください。

3. 各証明書をシステムに追加します。

`ikecert certdb -a` コマンドの `-a` オプションは、張り付けられたオブジェクトをシステムの適切な証明書データベースに追加します。詳細については、[166 ページの「IKE と公開鍵証明書」](#)を参照してください。

a. 管理者になります。

詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。リモートで管理する場合は、セキュアなリモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#)および『[Oracle Solaris 11.3 での Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」を参照してください。

- b. 公開鍵と CA から受け取った証明書を追加します。

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

- c. CA の公開証明書を追加します。

中間証明書の追加が必要な場合もあります。

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

- d. CA から証明書失効リストが送られている場合は、その CRL を `certrldb` データベースに追加します。

```
# ikecert certrldb -a
Press the Return key
Paste the CRL
-----BEGIN CRL-----
...
-----END CRL----
Press the Return key
Press Control-D
```

4. `/etc/inet/ike/config` ファイル内の `cert_root` キーワードを使用して、証明書を発行した CA を識別します。

CA の証明書の識別名 (DN) を使用します。

- a. たとえば、`host2` システムの `ike/config` ファイルは次のようになります。

```
# Trusted root cert
# This certificate is from Example CA
# This is the X.509 distinguished name for the CA's cert

cert_root "C=US, O=ExampleCA\, Inc., OU=CA-Example, CN=Example CA"

## Parameters that may also show up in rules.

p1_xform
{ auth_method rsa_sig oakley_group 14 auth_alg sha384 encr_alg aes}
p2_pfs 14

{
  label "US-host2 to JA-host1 - Example CA"
  local_id_type dn
  local_id "C=US, O=Example2Co, OU=US-Example2m, CN=Example2m"
  remote_id "C=JA, O=Example1Co, OU=JA-Example1x, CN=Example1x"

  local_addr 192.168.13.213
  remote_addr 192.168.116.16

  p1_xform
  {auth_method rsa_sig oakley_group 2 auth_alg sha256 encr_alg aes}
}
```

注記 - `auth_method` パラメータのすべての引数は同じ行になければなりません。

b. host1 システム上で、同様なファイルを作成します。

特に、host1 ike/config ファイルは次を行う必要があります。

- cert_root には同じ値を使用する。
- ローカルパラメータには host1 値を使用する。
- リモートパラメータには host2 値を使用する。
- label キーワードには一意の値を作成する。この値は、リモートシステムの label 値とは異なる値でなくてはなりません。

```
...
cert_root "C=US, O=ExampleCA\, Inc., OU=CA-Example, CN=Example CA"
...
{
label "JA-host1 to US-host2 - Example CA"
local_id_type dn
local_id "C=JA, O=Example1Co, OU=JA-Example1x, CN=Example1x"
remote_id "C=US, O=Example2Co, OU=US-Example2m, CN=Example2m"

local_addr 192.168.116.16
remote_addr 192.168.13.213
...

```

5. 失効した証明書の処理について、IKEv1 ポリシーを設定します。

適切なオプションを選択します。

■ OCSP 利用不可

公開鍵証明書には OCSP サーバーに到達するための URI が記述されているものの、使用しているシステムがインターネットに接続できない場合は、キーワード ignore_ocsp を ike/config ファイルに追加します。

```
# Trusted root cert
...
cert_root "C=US, O=ExampleCA\, Inc., OU=CA-Example,..."
ignore_ocsp
...
```

ignore_ocsp キーワードは、証明書が有効であると仮定するように IKEv1 に伝えます。

■ CRL 利用不可

CA が CRL の信頼できるソースを提供しない場合、または使用しているシステムがインターネットに接続して CRL を取得できない場合は、キーワード ignore_cr1s を ike/config ファイルに追加します。

```
# Trusted root cert
...
cert_root "C=US, O=ExampleCA\, Inc., OU=CA-Example,..."
ignore_cr1s
...
```

- **CRL または OCSP の URI を利用可能**

CA が失効した証明書の中央配布ポイントを提供している場合は、`ike/config` ファイルを変更して URI を使用できます。

例は、[221 ページの「IKEv1 で失効した証明書を処理する方法」](#)を参照してください。

例 41 IKEv1 の構成時に `rsa_encrypt` を使用する

`ike/config` ファイルで `auth_method rsa_encrypt` を使用する場合は、ピアの証明書を `publickeys` データベースに追加する必要があります。

1. その証明書をリモートシステムの管理者に送信します。

証明書は、電子メールのメッセージに貼り付けることもできます。

たとえば、`host2` の管理者が次のメッセージを送信します。

```
To: admin@host1.ja.example.com
From: admin@host2.us.example.com
Message: -----BEGIN X509 CERTIFICATE-----
MII...
```

`host1` の管理者が次のメッセージを送信します。

```
To: admin@host2.us.example.com
From: admin@host1.ja.example.com
Message: -----BEGIN X509 CERTIFICATE-----
MII
...
```

2. システムごとに、電子メールで送信された証明書をローカルの `publickeys` データベースに追加します。

```
# ikecert certdb -a < /tmp/saved.cert.eml
```

RSA 暗号化の認証方法は、IKE 内の識別子を盗聴者から隠します。`rsa_encrypt` メソッドはピアの識別子を隠すため、IKEv1 はピアの証明書を取得できません。結果として、`rsa_encrypt` メソッドでは、IKEv1 ピアが互いの公開鍵を知っておく必要があります。

そのため、`/etc/inet/ike/config` ファイルの `auth_method` に `rsa_encrypt` を指定する場合には、ピアの証明書を `publickeys` データベースに追加する必要があります。この結果、`publickeys` データベースには、通信するシステムペアごとに3つ以上の証明書が存在することになります。

- ユーザーの公開鍵証明書
- CA の証明書チェーン

- ピアの公開鍵証明書

トラブルシューティング – IKEv1 ペイロードには 3 つ以上の証明書が含まれており、`rsa_encrypt` で暗号化するには大きくなりすぎることがあります。「`authorization failed` (承認に失敗しました)」や「`malformed payload` (ペイロードが不正です)」などのエラーは、`rsa_encrypt` メソッドがペイロード全体を暗号化できないことを示します。証明書を 2 つしか必要としない `rsa_sig` などのメソッドを使用して、ペイロードのサイズを減らします。

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#)を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#)を参照してください。

▼ ハードウェア上で IKEv1 の公開鍵証明書を作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作成および格納することは、システム上で公開鍵証明書を作成および格納することと似ています。ハードウェア上では、`ikecert certlocal` および `ikecert certdb` コマンドがハードウェアを識別しなければなりません。トークン ID に `-T` オプションを指定すると、コマンドがハードウェアを識別するようになります。

始める前に

- ハードウェアの構成が完了していること。
- `/etc/inet/ike/config` ファイルの `pkcs11_path` キーワードが別のライブラリを指している場合を除き、ハードウェアは `/usr/lib/libpkcs11.so` ライブラリを使用します。ライブラリが、RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki) 規格、つまり PKCS #11 ライブラリに準拠して実装されている必要があります。

設定の手順については、[230 ページの「Sun Crypto Accelerator 6000 ボードを検出するように IKEv1 を構成する方法」](#)を参照してください。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなリモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#) および『[Oracle Solaris 11.3 での Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」を参照してください。

1. 自己署名付き証明書または CSR を生成し、トークン ID を指定します。

注記 - Sun Crypto Accelerator 6000 ボードは、RSA で最大 2048 ビットの鍵をサポートします。DSA の場合、このボードは最大 1024 ビットの鍵をサポートします。

次のオプションのいずれかを選択します。

■ **自己署名付き証明書の場合は、この構文を使用します。**

```
# ikecert certlocal -ks -m 2048 -t rsa-sha512 \  
> -D "C=US, O=PartyCompany, OU=US-Partym, CN=Partym" \  
> -a -T dca0-accel-stor IP=192.168.116.16  
Creating hardware private keys.  
Enter PIN for PKCS#11 token:      Type user:password
```

-T オプションの引数は、接続された Sun Crypto Accelerator 6000 ボードのトークン ID です。

■ **CSR の場合は、この構文を使用します。**

```
# ikecert certlocal -kc -m 2048 -t rsa-sha512 \  
> -D "C=US, O=PartyCompany, OU=US-Partym, CN=Partym" \  
> -a -T dca0-accel-stor IP=192.168.116.16  
Creating hardware private keys.  
Enter PIN for PKCS#11 token:      Type user:password
```

ikecert コマンドの引数については、[ikecert\(1M\)](#) のマニュアルページを参照してください。

2. **PIN のプロンプトに、Sun Crypto Accelerator 6000 ユーザー名、コロン、およびユーザーのパスワードを入力します。**

Sun Crypto Accelerator 6000 ボードのユーザー ikemgr のパスワードが rgm4tigt の場合、次のように入力します。

```
Enter PIN for PKCS#11 token: ikemgr:rgm4tigt
```

注記 - ikecert コマンドに -p オプションを付けて入力すると、PKCS #11 トークンがディスクにクリアテキストとして格納され、root 権限によって保護されます。PIN をディスクに保存しない場合は、in.iked コマンドの実行後に ikeadm コマンドを使用してトークンをロック解除する必要があります。

パスワードの入力後、証明書が次を出力します。

```
Enter PIN for PKCS#11 token: ikemgr:rgm4tigt  
-----BEGIN X509 CERTIFICATE-----  
MIIBuDCCASECAQAwSTELMAkGA1UEBhMCVVmXFTATBgNVBAoTDFBhcnR5Q29tcGFu  
....  
oKUDBbZ90/pLWYGr  
-----END X509 CERTIFICATE-----
```

3. **相手に証明書を送信します。**

次のオプションのいずれかを選択します。

- リモートシステムに自己署名付き証明書を送信します。
証明書は、電子メールのメッセージに貼り付けることもできます。
- **CSR を認証局 (CA) に送信します。**
CA の指示に従って証明書要求を送信します。詳細な説明は、[212 ページの「CA の署名付き証明書で IKEv1 を構成する方法」](#)のステップ 2 を参照してください。

4. システム上で、証明書を認識するように `/etc/inet/ike/config` ファイルを編集します。

次のオプションのどちらか 1 つを選択します。

■ 自己署名付き証明書

`cert_trust`、`remote_id`、および `remote_addr` パラメータには、リモートシステムの管理者から提供される値を使用します。たとえば、`host1` 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-host1 to US-host2"
  local_id_type dn
  local_id "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax"
  remote_id "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"

  local_addr 192.168.116.16;
  remote_addr 192.168.13.213

  p1_xform
  {auth_method rsa_sig oakley_group 2 auth_alg sha256 encr_alg aes}
}
```

■ 証明書要求

CA が `cert_root` キーワードの値として提供する名前を入力します。たとえば、`host1` システムの `ike/config` ファイルは次のようになります。

```
# Trusted root cert
# This certificate is from Example CA
# This is the X.509 distinguished name for the CA that it issues.

cert_root "C=US, O=ExampleCA\, Inc., OU=CA-Example, CN=Example CA"

...
{
  label "JA-host1 to US-host2 - Example CA"
  local_id_type dn
```

```
local_id "C=JA, O=EnigmaCo, OU=JA-Enigma, CN=Enigma"
remote_id "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"

local_addr 192.168.116.16
remote_addr 192.168.13.213

pi_xform
{auth_method rsa_sig oakley_group 2 auth_alg sha256 encr_alg aes}
}
```

5. 通信先から受け取った証明書をハードウェアに格納します。

ステップ 2 で応答したように、PIN 要求に応答します。

注記 - 公開鍵証明書は、公開鍵を生成したハードウェアに追加する必要があります。

■ 自己署名付き証明書。

リモートシステムの自己署名付き証明書を追加します。この例では、証明書は DCA.ACCEL.STOR.CERT ファイルに格納されています。

```
# ikecert certdb -a -T dca0-accel-stor < DCA.ACCEL.STOR.CERT
Enter PIN for PKCS#11 token:      Type user:password
```

自己署名付き証明書が `rsa_encrypt` を `auth_method` パラメータの値として使用していた場合、ピアの証明書をハードウェア格納場所に追加します。

■ CA からの証明書。

CA があなたの証明書要求と組織の証明書から生成した証明書を追加します。

中間証明書の追加が必要な場合もあります。

```
# ikecert certdb -a -T dca0-accel-stor < DCA.ACCEL.STOR.CERT
Enter PIN for PKCS#11 token:      Type user:password
```

```
# ikecert certdb -a -T dca0-accel-stor < DCA.ACCEL.STOR.CA.CERT
Enter PIN for PKCS#11 token:      Type user:password
```

CA からの証明書失効リスト (CRL) を追加するには、221 ページの「IKEv1 で失効した証明書を処理する方法」を参照してください。

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、146 ページの「IPsec による VPN の保護」を参照してください。IPsec ポリシーのその他の例については、139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」を参照してください。

▼ IKEv1 で失効した証明書を処理する方法

失効した証明書とは、なんらかの理由で効力を失った証明書です。失効した証明書を使用していると、セキュリティのリスクとなります。証明書の失効を確認する際のオプションがあります。静的なリストを使用するか、HTTP プロトコルを介して失効を動的に確認できます。失効した証明書を処理する 4 つの方法があります。

- 証明書に CRL または OCSP の Uniform Resource Indicator (URI) が組み込まれている場合に、CRL または OCSP を無視するように IKEv1 に指示できます。このオプションは [ステップ 5](#) に示されています。
- CA からの公開鍵証明書に URI のアドレスが組み込まれている場合に、その URI から CRL または OCSP にアクセスするように IKEv1 に指示できます。
- CA からの公開鍵証明書に LDAP サーバーの DN (ディレクトリ名) エントリが組み込まれている場合に、その LDAP サーバーから CRL にアクセスするように IKEv1 に指示できます。
- `ikecert certtrldb` コマンドの引数として CRL を指定できます。例については、[例42「CRL を IKEv1 のローカルの certtrldb データベースに貼り付ける」](#)を参照してください。

始める前に Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

1. CA から受け取った証明書を表示します。

`ikecert certdb` コマンドの引数については、[ikecert\(1M\)](#) のマニュアルページを参照してください。

たとえば、次の証明書は企業の PKI から発行されました。詳細は変更されています。

```
# ikecert certdb -lv cert-protect.example.com
Certificate Slot Name: 0   Type: dsa-sha256
  (Private key in certlocal slot )
Subject Name: <O=Example, CN=cert-protect.example.com>
Issuer Name: <CN=ExampleCo CO (C1 B), O=Example>
SerialNumber: 14000D93
Validity:
  Not Valid Before: 2013 Sep 19th, 21:11:11 GMT
  Not Valid After:  2017 Sep 18th, 21:11:11 GMT
Public Key Info:
  Public Modulus (n) (2048 bits): C575A...A5
  Public Exponent (e) ( 24 bits): 010001
Extensions:
  Subject Alternative Names:
    DNS = cert-protect.example.com
  Key Usage: DigitalSignature KeyEncipherment
  [CRITICAL]
CRL Distribution Points:
  Full Name:
    URI = #Ihttp://www.example.com/pki/pkismica.crl#i
    DN = <CN=ExampleCo CO (C1 B), O=Example>
  CRL Issuer:
```

```

Authority Key ID:
Key ID:          4F ... 6B
SubjectKeyID:   A5 ... FD
Certificate Policies
Authority Information Access
    
```

CRL Distribution Points エントリに注目してください。

- URI エントリは、この機関の CRL が Web 上にあることを示しています。
- DN エントリは、CRL が LDAP サーバー上にあることを示しています。一度、IKE がアクセスすると、CRL は将来に備えてキャッシュに格納されます。

CRL にアクセスするには、配布ポイントまで到達する必要があります。

2. 中央の配布ポイントから CRL にアクセスするには、次のメソッドのうちの 1 つを選択します。

■ URI を使用します。

キーワード `use_http` をホストの `/etc/inet/ike/config` ファイルに追加します。たとえば、`ike/config` ファイルは次のようになります。

```

# Use CRL or OCSP from organization's URI
use_http
...
    
```

■ Web プロキシを使用します。

キーワード `proxy` を `ike/config` ファイルに追加します。キーワード `proxy` は、次のように引数として URL を取ります。

```

# Use web proxy to reach CRLs or OCSP
proxy "http://proxy1:8080"
    
```

■ LDAP サーバーを使用します。

ホストの `/etc/inet/ike/config` ファイルの `ldap-list` キーワードに引数として LDAP サーバーの名前を指定します。LDAP サーバーの名前は、使用する機関にたずねてください。`ike/config` ファイルのエントリは次のようになります。

```

# Use CRL from organization's LDAP
ldap-list "ldap1.example.com:389,ldap2.example.com"
...
    
```

IKE は CRL を取り出し、証明書の期限が切れるまで CRL を保持します。

例 42 CRL を IKEv1 のローカルの `certrldb` データベースに貼り付ける

CA の CRL を中央配布ポイントから使用できない場合は、ローカルの `certrldb` データベースに CRL を手動で追加できます。CA の指示に従って CRL をファイルに抽出し、その CRL を `ikecert certrldb -a` コマンドでデータベースに追加します。

```

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

移動体システム用の IKEv1 の構成

ソースと宛先を識別するために、IPsec と IKE は一意の ID を必要とします。一意の IP アドレスを持たない遠隔地のシステムまたは移動体システムの場合、別の種類の ID を使用する必要があります。システムを一意に識別するために、DNS、DN、または email などの ID の種類を使用できます。

一意の IP アドレスを持つ遠隔地のシステムまたは移動体システムで、別の種類の ID で構成するようにします。たとえば、システムが NAT 越しに中央システムに接続しようとした場合、そのシステムの一意なアドレスは使用されません。NAT ボックスが任意の IP アドレスを割り当てるため、中央システムは認識できません。

事前共有鍵は固定 IP アドレスを必要とするため、事前共有鍵も移動体システム用の認証メカニズムとしては機能しません。自己署名付き証明書、または CA からの証明書を使用すると、移動体システムは中央サイトと通信できます。

次のタスクマップは、リモートから中央サイトにログインするシステムを処理するために IKEv1 を構成する手順を一覧表示したものです。

表 16 移動体システム用の IKEv1 の構成のタスクマップ

タスク	説明	参照先
オフサイトから中央サイトへ通信します。	遠隔地のシステムが中央サイトと通信できるようにします。遠隔地のシステムは移動体システムの可能性もあります。	223 ページの「遠隔地のシステム用に IKEv1 を構成する方法」
移動体システムからのトラフィックを受信する中央システムで CA の公開証明書と IKEv1 を使用します。	固定 IP アドレスを持たないシステムからの IPsec トラフィックを受信するゲートウェイシステムを構成します。	例43「移動体システムから保護されたトラフィックを受信するために IKEv1 を使用する中央コンピュータを構成する」
固定 IP アドレスを持たないシステムで CA の公開証明書と IKEv1 を使用します。	中央サイト(会社の本社など)とのトラフィックを保護するように、移動体システムを構成します。	例44「NAT 越しのシステムを IPsec と IKEv1 で構成する」
移動体システムからのトラフィックを受信する中央システムで自己署名付き証明書と IKEv1 を使用します。	移動体システムから IPsec トラフィックを受信するように、ゲートウェイシステムを自己署名付き証明書で構成します。	例45「移動体システムからの自己署名付き証明書の受け入れ」
固定 IP アドレスを持たないシステムで自己署名付き証明書と IKEv1 を使用します。	中央サイトとのトラフィックを保護するように、移動体システムを自己署名付き証明書で構成します。	例46「自己署名付き証明書による中央システムとの接続」

▼ 遠隔地のシステム用に IKEv1 を構成する方法

始める前に root 役割になる必要があります。詳細は、『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』の「割り当てられている管理権利の使用」を参照してください。リモートで管理する場合は、セキュアリモートログイン手順について、例 27「ssh 接続を使用した IPsec ポリシーのリモート構成」および『Oracle Solaris 11.3 で

の [Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」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. IKEv1 構成ファイルを構成します。

DNS は中央システムを識別します。証明書を使用して、システムを認証します。

```
## /etc/inet/ike/ike.config on central
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
#
# Use web proxy
proxy "http://somecache.domain:port/"
#
# Use LDAP server
ldap_server "ldap-server1.domain.org, ldap2.domain.org:port"
#
# List CA-signed certificates
cert_root "C=US, O=Domain Org, CN=Domain STATE"
#
# List self-signed certificates - trust server and enumerated others
#cert_trust "DNS=central.domain.org"
#cert_trust "DNS=mobile.domain.org"
#cert_trust "DN=CN=Domain Org STATE (CLASS), O=Domain Org"
#cert_trust "email=root@central.domain.org"
#cert_trust "email=user1@mobile.domain.org"
#

# Rule for mobile systems with certificate
{
  label "Mobile systems with certificate"
  local_id_type DNS
  # CA's public certificate ensures trust,
  # so allow any remote_id and any remote IP address.
  remote_id ""
  remote_addr 0.0.0.0/0

p2_pfs 14

p1_xform
{auth_method rsa_sig oakley_group 14 encr_alg aes auth_alg sha256 }
}
```

2. 各移動体システムにログインして、中央システムを見つけるように構成します。

a. /etc/hosts ファイルを構成します。

/etc/hosts ファイルは、移動体システムのアドレスを必要としませんが、提供は可能です。このファイルは、中央システムの公開 IP アドレスである *central* を含んでいる必要があります。

```
# /etc/hosts on mobile
central 192.xxx.xxx.x
```

b. ipsecinit.conf ファイルを構成します。

移動体システムは、公開 IP アドレスで中央システムを見つける必要があります。システムは同じ IPsec ポリシーで構成する必要があります。

```
# /etc/inet/ipsecinit.conf on mobile
# Find central
{raddr 192.xxx.xxx.x} ipsec {encr_algs aes encr_auth_algs sha256 sa shared}
```

c. IKEv1 構成ファイルを構成します。

識別子は IP アドレスであってはなりません。移動体システムに有効な識別子は次のとおりです。

- DN=*ldap-directory-name*
- DNS=*domain-name-server-address*
- email=*email-address*

証明書は、移動体システムである *mobile* の認証に使用されます。

```
## /etc/inet/ike/ike.config on mobile
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
#
# Use web proxy
proxy "http://somecache.domain:port/"
#
# Use LDAP server
ldap_server "ldap-server1.domain.org, ldap2.domain.org:port"
#
# List CA-signed certificates
cert_root "C=US, O=Domain Org, CN=Domain STATE"
#
# Self-signed certificates - trust me and enumerated others
#cert_trust "DNS=mobile.domain.org"
#cert_trust "DNS=central.domain.org"
#cert_trust "DN=CN=Domain Org STATE (CLASS), O=Domain Org"
#cert_trust "email=user1@domain.org"
#cert_trust "email=root@central.domain.org"
#
# Rule for off-site systems with root certificate
{
label "Off-site mobile with certificate"
local_id_type DNS
```

```
# NAT-T can translate local_addr into any public IP address
# central knows me by my DNS

local_id "mobile.domain.org"
local_addr 0.0.0.0/0

# Find central and trust the root certificate
remote_id "central.domain.org"
remote_addr 192.xxx.xxx.x

p2_pfs 14

p1_xform
{auth_method rsa_sig oakley_group 14 encr_alg aes auth_alg sha256 }
}
```

3. ike:default サービスを有効にします。

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

例 43 移動体システムから保護されたトラフィックを受信するために IKEv1 を使用する中央コンピュータを構成する

IKE は、NAT ボックス越しのネゴシエーションを開始できます。しかし、IKE の理想的な設定は NAT ボックスをはさまないことです。次の例では、CA の公開証明書は移動体システムと中央システムに格納されています。中央システムは NAT 越しのシステムからの IPsec ネゴシエーションを受け入れます。main1 は、遠隔地のシステムからの接続を受け入れることができる会社のシステムです。遠隔地のシステムを設定するには、[例44「NAT 越しのシステムを IPsec と IKEv1 で構成する」](#)を参照してください。

```
## /etc/hosts on main1
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on main1
# Keep everyone out unless they use this IPsec policy:
{} ipsec {encr_algs aes encr_auth_algs sha256 sa shared}

## /etc/inet/ike/ike.config on main1
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
#
# Use web proxy
proxy "http://cache1.domain.org:8080/"
#
# Use LDAP server
ldap_server "ldap1.domain.org,ldap2.domain.org:389"
#
# List CA-signed certificate
cert_root "C=US, O=ExampleCA Inc, OU=CA-Example, CN=Example CA"
#
# Rule for off-site systems with root certificate
{
label "Off-site system with root certificate"
local_id_type DNS
```

```

local_id "main1.domain.org"
local_addr 192.168.0.100

# CA's public certificate ensures trust,
# so allow any remote_id and any remote IP address.
remote_id ""
remote_addr 0.0.0.0/0

p2_pfs 14

p1_xform
{auth_method rsa_sig oakley_group 14 encr_alg aes auth_alg sha256}
p1_xform
{auth_method rsa_sig oakley_group 14 encr_alg aes auth_alg sha256}
p1_xform
{auth_method rsa_sig oakley_group 14 encr_alg aes auth_alg sha256}
p1_xform
{auth_method rsa_sig oakley_group 14 encr_alg aes auth_alg sha256}
}

```

例 44 NAT 越しのシステムを IPsec と IKEv1 で構成する

次の例では、CA の公開証明書は移動体システムと中央システムに格納されています。mobile1 は、家から会社の本社に接続しています。インターネットサービスプロバイダ (ISP) ネットワークは NAT ボックスを使用しているため、ISP は mobile1 に非公開アドレスを割り当てることができます。NAT ボックスは、非公開アドレスを公開 IP アドレスに変換します。この公開アドレスは、ほかの ISP ネットワークノードと共有されます。企業の本社は NAT を越えません。企業の本社のコンピュータの設定については、[例43「移動体システムから保護されたトラフィックを受信するために IKEv1 を使用する中央コンピュータを構成する」](#)を参照してください。

```

## /etc/hosts on mobile1
mobile1 10.1.3.3
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on mobile1
# Find main1
{raddr 192.168.0.100} ipsec {encr_algs aes encr_auth_algs sha256 sa shared}

## /etc/inet/ike/ike.config on mobile1
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
#
# Use web proxy
proxy "http://cache1.domain.org:8080/"
#
# Use LDAP server
ldap_server "ldap1.domain.org,ldap2.domain.org:389"
#
# List CA-signed certificate
cert_root "C=US, O=ExampleCA Inc, OU=CA-Example, CN=Example CA"
#
# Rule for off-site systems with root certificate
{
label "Off-site mobile1 with root certificate"
local_id_type DNS
local_id "mobile1.domain.org"
}

```

```

    local_addr 0.0.0.0/0

# Find main1 and trust the root certificate
remote_id "main1.domain.org"
remote_addr 192.168.0.100

p2_pfs 14

p1_xform
{auth_method rsa_sig oakley_group 14 encr_alg aes auth_alg sha256 }
}

```

例 45 移動体システムからの自己署名付き証明書の受け入れ

次の例では、自己署名付き証明書が発行されており、移動体システムと中央システムに格納されています。main1 は、遠隔地のシステムからの接続を受け入れることができる会社のシステムです。遠隔地のシステムを設定するには、[例46「自己署名付き証明書による中央システムとの接続」](#)を参照してください。

```

## /etc/hosts on main1
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on main1
# Keep everyone out unless they use this IPsec policy:
{} ipsec {encr_algs aes encr_auth_algs sha256 sa shared}

## /etc/inet/ike/ike.config on main1
# Global parameters
#
# Self-signed certificates - trust me and enumerated others
cert_trust "DNS=main1.domain.org"
cert_trust "jdoe@domain.org"
cert_trust "user2@domain.org"
cert_trust "user3@domain.org"
#
# Rule for off-site systems with trusted certificate
{
  label "Off-site systems with trusted certificates"
  local_id_type DNS
  local_id "main1.domain.org"
  local_addr 192.168.0.100

# Trust the self-signed certificates
# so allow any remote_id and any remote IP address.
remote_id ""
remote_addr 0.0.0.0/0

p2_pfs 14

p1_xform
{auth_method rsa_sig oakley_group 14 encr_alg aes auth_alg sha256 }
}

```

例 46 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には、[例 45 「移動体システムからの自己署名付き証明書の受け入れ」](#) を参照してください。

```
## /etc/hosts on mobile1
mobile1 10.1.3.3
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on mobile1
# Find main1
{raddr 192.168.0.100} ipsec {encr_algs aes encr_auth_algs sha256 sa shared}

## /etc/inet/ike/ike.config on mobile1
# Global parameters

# Self-signed certificates - trust me and the central system
cert_trust "jdoe@domain.org"
cert_trust "DNS=main1.domain.org"
#
# Rule for off-site systems with trusted certificate
{
  label "Off-site mobile1 with trusted certificate"
  local_id_type email
  local_id "jdoe@domain.org"
  local_addr 0.0.0.0/0

# Find main1 and trust the certificate
  remote_id "main1.domain.org"
  remote_addr 192.168.0.100

p2_pfs 14

p1_xform
{auth_method rsa_sig oakley_group 14 encr_alg aes auth_alg sha256 }
}
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#) を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#) を参照してください。

接続したハードウェアを検出するように IKEv1 を構成する

公開鍵証明書はまた、接続されたハードウェアに格納できます。Sun Crypto Accelerator 6000 ボードによってストレージが提供され、公開鍵の操作をシステムからこのボードにオフロードできます。

▼ Sun Crypto Accelerator 6000 ボードを検出するように IKEv1 を構成する方法

始める前に 次の手順では、Sun Crypto Accelerator 6000 ボードがシステムに接続されていると仮定します。さらに、ボードに必要なソフトウェアがすでにインストールされ、構成されているものとして扱います。手順については、[Sun Crypto Accelerator 6000 ボード製品のライブラリドキュメント \(http://docs.oracle.com/cd/E19321-01/index.html\)](http://docs.oracle.com/cd/E19321-01/index.html)を参照してください。

Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティ保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

リモートで管理する場合は、セキュアなリモートログイン手順について、[例27「ssh 接続を使用した IPsec ポリシーのリモート構成」](#)および『[Oracle Solaris 11.3 での Secure Shell アクセスの管理](#)』の「[Secure Shell を使用して ZFS をリモートで管理する方法](#)」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。

IKEv1 はライブラリのルーチンを使用して、Sun Crypto Accelerator 6000 ボード上で鍵の生成および鍵の格納を処理します。

```
$ ikeadm get stats
...
PKCS#11 library linked in from /usr/lib/libpkcs11.so
$
```

2. 接続された Sun Crypto Accelerator 6000 ボードのトークン ID を見つけます。

```
$ ikecert tokens
Available tokens with library "/usr/lib/libpkcs11.so":

"Sun Metaslot"
```

ライブラリは、32 文字のトークン ID (キーストア名とも呼ぶ) を戻します。この例では、ikecert コマンドに Sun Metaslot トークンを使用すると、IKEv1 鍵を格納および高速化できます。

トークンを使用する手順については、[217 ページの「ハードウェア上で IKEv1 の公開鍵証明書を生成および格納する方法」](#)を参照してください。

ikecert コマンドにより、後続スペースが自動的に付加されます。

例 47 メタスロットトークンの検索と使用

トークンは、ディスク、接続されたボード、または暗号化フレームワークが提供するソフトトークンキーストアに格納できます。次に、ソフトトークンキーストアのトークン ID の例を示します。

```
$ ikecert tokens
Available tokens with library "/usr/lib/libpkcs11.so":

"Sun Metaslot          "
```

ソフトトークンキーストアのパスフレーズを作成する方法については、[pktool\(1\)](#) のマニュアルページを参照してください。

次に、ソフトトークンキーストアに証明書を追加するコマンドの例を示します。Sun.Metaslot.cert は、CA 証明書を格納しているファイルです。

```
# ikecert certdb -a -T "Sun Metaslot" < Sun.Metaslot.cert
Enter PIN for PKCS#11 token:      Type user:passphrase
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。VPN を保護する IPsec ポリシーの例については、[146 ページの「IPsec による VPN の保護」](#)を参照してください。IPsec ポリシーのその他の例については、[139 ページの「IPsec によって 2 つのサーバー間でネットワークトラフィックをセキュリティー保護する方法」](#)を参照してください。

◆◆◆ 第 13 章

IPsec およびその鍵管理サービスのトラブルシューティング

この章では、IPsec とその鍵のトラブルシューティング方法、構成情報の表示方法、アクティブな IPsec、IKE、および手動鍵サービスに関する情報の表示方法について説明します。

この章では、次の内容について説明します。

- 233 ページの「IPsec およびその鍵管理構成のトラブルシューティング」
- 241 ページの「IPsec およびその鍵サービスに関する情報の表示」
- 246 ページの「IPsec およびその鍵サービスの管理」
- 248 ページの「実行中の IKE デーモンの管理」

IPsec およびその鍵管理構成のトラブルシューティング

トラブルシューティングが必要な問題の発生前または発生中に、システムのトラブルシューティングを設定できます。

トラブルシューティングの際は、Network IPsec Management 権利プロファイルを持つ管理者として、プロファイルシェルで多くのコマンドを実行できます。ただし、ログを読み取るには root 役割になる必要があります。

トラブルシューティングセクションのプロンプトは、コマンドの実行に権利が必要かどうかを示します。

- # プロンプト – 適切な管理権利またはそれらの権利がある役割を持つユーザーがコマンドを実行できます。
- % プロンプト – 通常のユーザーがコマンドを実行できます。

▼ IPsec および IKE システムのトラブルシューティングを準備する方法

IPsec およびその鍵管理サービスを有効にする前に、システムでトラブルシューティングに役立つログとツールを設定できます。

1. IPsec および IKEv2 サービスのログの場所を指定します。

-L オプションはログのフルパスを指定します。これらのログには情報メッセージとエラーメッセージが含まれます。

```
% svcs -L policy
/var/svc/log/network-ipsec-policy:default.log
```

```
% svcs -L ikev2
/var/svc/log/network-ipsec-ike:ikev2.log
```

2. IKEv2 のデバッグログファイルを構成します。

root 役割はこれらのログを読み取ることができます。

```
% svccfg -s ikev2 listprop | grep debug
config/debug_level          astring      op
config/debug_logfile        astring      /var/log/ikev2/in.ikev2d.log
```

デバッグのレベルについては、[ikeadm\(1M\)](#) のマニュアルページで説明されています。値 `verbose` および `all` は、トラブルシューティングの際に便利です。

3. (オプション) デバッグレベルを構成します。

次のコマンドはデバッグレベルを永続的に設定します。デバッグレベルを一時的に設定するには、[例50「実行中の IKE デーモンの新しいデバッグレベルの設定」](#) を参照してください。

```
# svccfg -s ikev2 setprop config/debug_level = all
```

ikev2 サービスが有効になっている場合、新しいデバッグレベルを使用するにはサービスをリフレッシュする必要があります。

```
# svcadm refresh ikev2
```

4. (オプション) wireshark パッケージをインストールします。

Wireshark アプリケーションは snoop 出力を読み取ることができます。

```
% pkg info -r wireshark
Name: diagnostic/wireshark
Summary: Graphical network protocol analyzer
Category: Applications/Internet
State: Not installed
Publisher: solaris
...
FMRI: pkg://solaris/diagnostic/wireshark@version
# pkg install diagnostic/wireshark
```

▼ IPsec と IKE を実行する前にシステムをトラブルシューティングする方法

サービス実行前に、IPsec 構成ファイルの構文、IPsec 鍵ファイル、およびキーストア内の証明書の有効性を確認できます。

1. IPsec 構成ファイルの構文を検証します。

```
# ipsecconf -c /etc/inet/ipsecinit.conf
ipsecconf: Invalid pattern on line 5: ukp
ipsecconf: form_ipsec_conf error
ipsecconf: Malformed command (fatal):
{ ukp 58 type 133-137 dir out} pass {}

ipsecconf: 1 policy rule(s) contained errors.
ipsecconf: Fatal error - exiting.
```

出力でエラーが表示される場合は、検証が成功するまでエラーを修正してコマンドを実行します。

2. ipseckey ファイルの構文を検証します。

```
# ipseckey -c /etc/inet/secret/ipseckey
Config file /etc/inet/secret/ipseckey has insecure permissions,
will be rejected in permanent config.
```

出力にエラーが表示されている場合は、エラーを修正してからサービスをリフレッシュします。

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

注記 - IKE 構成ファイルおよび IKE 事前共有鍵ファイルは、実行中の IKE デーモンによって検証されます。

3. 証明書の妥当性を検証します。

- IKEv2 の自己署名付き証明書の妥当性を検証するには、[185 ページの「自己署名付き公開鍵証明書により IKEv2 を構成する方法」](#)のステップ 4 を実行してください。
- IKEv2 で公開鍵証明書が失効していないことを確認するには、[193 ページの「IKEv2 で証明書検証ポリシーを設定する方法」](#)の手順に従ってください。
- IKEv1 の自己署名付き証明書の妥当性を検証するには、[207 ページの「自己署名付き公開鍵証明書により IKEv1 を構成する方法」](#)のステップ 4 を実行してください。
- IKEv1 で公開鍵証明書が失効していないことを確認するには、[221 ページの「IKEv1 で失効した証明書を処理する方法」](#)の手順に従ってください。

次の手順 IPsec およびその鍵サービスを有効にしても構成が機能しない場合は、サービスの実行中にトラブルシューティングする必要があります。

▼ IPsec 実行中にシステムをトラブルシューティングする方法

IKE を使用してパケットを交換している、または交換を試みている実行中のシステム上で、`ikeadm` コマンドを使用して統計、ルール、事前共有鍵などの項目を表示できます。ログファイルおよび Wireshark アプリケーションなどの一部のツールを使用することもできます。

1. 次の項目を調査します。

- ポリシーと適切な鍵管理サービスが有効になっていることを確認します。

次のテストシステムでは、鍵管理に `manual-key` サービスが使用されています。

```
% svcs -a | grep ipsec
online      Feb_04    svc:/network/ipsec/manual-key:default
online      Feb_04    svc:/network/ipsec/ipsecalgs:default
online      Feb_04    svc:/network/ipsec/policy:default
disabled    Feb_28    svc:/network/ipsec/ike:ikev2
disabled    Feb_28    svc:/network/ipsec/ike:default
```

このサービスが無効になっている場合は有効にします。

両方の IKE サービスを同時に使用できます。手動鍵と IKE を同時に使用することもできますが、この構成はトラブルシューティングが困難な不具合を引き起こす場合があります。

- IKEv2 サービスのログファイルの終わりを表示します。

```
# svcs -xL ikev2
svc:/network/ipsec/ike:ikev2 (IKEv2 daemon)
State: disabled since October 10, 2013 10:10:40 PM PDT
Reason: Disabled by an administrator.
See: http://support.oracle.com/msg/SMF-8000-05
See: in.ikev2d(1M)
See: /var/svc/log/network-ipsec-ike:ikev2.log
Impact: This service is not running.
Log:
Oct 01 13:20:20: (1) Property "debug_level" set to: "op"
Oct 01 13:20:20: (1) Errors and debug messages will be written to:
/var/log/ikev2/in.ikev2d.log
[ Oct 10 10:10:10 Method "start" exited with status 0. ]
[ Oct 10 10:10:40 Stopping because service disabled. ]
[ Oct 10 10:10:40 Executing stop method (:kill). ]
```

Use: 'svcs -Lv svc:/network/ipsec/ike:ikev2' to view the complete log.

- (オプション) 実行中のデーモンのデバッグレベルの一時的な値を設定できます。

```
# ikeadm set debug verbose /var/log/ikev2/in.ikev2d.log
Successfully changed debug level from 0x80000000 to 0x6204
Debug categories enabled:
  Operational / Errors
  Config file processing
  Interaction with Audit
  Verbose Operational
```

2. **ipsecconf** コマンドの出力がポリシーファイルのコンテンツと一致していることを確認します。

```
# ipsecconf
#INDEX 14
...
{ laddr 10.133.66.222 raddr 10.133.64.77 }
  ipsec { encr_algs aes(256) encr_auth_algs sha512 sa shared }
...
{ laddr 10.134.66.122 raddr 10.132.55.55 }
  ipsec { encr_algs aes(256) encr_auth_algs sha512 sa shared }

# cat /etc/inet/ipsecinit.conf
...
{ laddr 10.133.66.222 raddr 10.133.64.77 }
  ipsec { encr_algs aes(256) encr_auth_algs sha512 sa shared }

{ laddr 10.134.66.122 raddr 10.132.55.55 }
  ipsec { encr_algs aes(256) encr_auth_algs sha512 sa shared }
```

注記 - ワイルドカードアドレスによって一致しているかどうかはわかりにくい場合があるため、`ipsecinit.conf` ファイル内の特定のアドレスが `ipsecconf` の出力のワイルドカードアドレスの範囲内に含まれていることを確認してください。

`ipsecconf` コマンドの出力が表示されない場合は、ポリシーサービスが有効になっていることを確認してサービスをリフレッシュしてください。

```
% svcs policy
STATE          STIME    FMRI
online         Apr_10   svc:/network/ipsec/policy:default
```

出力にエラーが表示される場合は、`ipsecinit.conf` ファイルを編集してエラーを修正してからサービスをリフレッシュしてください。

3. **IKEv2 構成を確認します。**

修正を必要とする可能性のある構成の出力については、[例48「無効な IKEv2 構成の修正」](#) および [例49「一致しないルールメッセージの修正」](#) を参照してください。次の例の出力は、構成が有効であることを示しています。

```
# /usr/lib/inet/in.ikev2d -c
Feb 04 12:08:25: (1) Reading service properties from smf(5) repository.
Feb 04 12:08:25: (1) Property "config_file" set to: "/etc/inet/ike/ikev2.config"
Feb 04 12:08:25: (1) Property "debug_level" set to: "all"
Feb 04 12:08:25: (1) Warning: debug output being written to stdout.
Feb 04 12:08:25: (1) Checking IKE rule #1: "Test 104 to 113"
Feb 04 12:08:25: (1) Configuration file /etc/inet/ike/ikev2.config is valid.
Feb 04 12:08:25: (1) Pre-shared key file /etc/inet/ike/ikev2.preshared is valid.
```

注記 - デバッグ出力に関する警告は、デバッグログファイルを指定したあとでも変更されません。debug_logfile サービスプロパティの値を指定する場合、この警告はデバッグ出力がそのファイルに渡されていることを意味します。それ以外の場合、デバッグ出力はコンソールに渡されます。

- Checking IKE rule 行で、IKE ルールに適切な IP アドレスが接続していることを確認します。たとえば、次のエントリは一致しています。ipsecinit.conf ファイルの laddr の値が ikev2.config ファイルの local_addr の値と一致し、リモートアドレス同士が一致しています。

```
{ laddr 10.134.64.104 raddr 10.134.66.113 }      /** ipsecinit.conf **/
    ipsec {encr_algs aes encr_auth_algs sha512 sa shared}

local_addr  10.134.64.104                        /** ikev2.config **/
remote_addr 10.134.66.113                        /** ikev2.config **/
```

エントリが対応していない場合は、構成を修正して正しい IP アドレスを識別します。

注記 - ルールには、アドレスの範囲を示す 10.134.0.0/16 のようなワイルドカードアドレスが含まれる場合があります。特定のアドレスに対して範囲を確認します。

- Pre-shared key file 行にファイルが有効でないことが示されている場合は、ファイルを修正します。

入力ミスを確認します。また、IKEv2 で、ikev2.config 内のルールのラベル値が ikev2.preshared ファイル内のラベル値と一致することを確認します。2つの鍵を使用している場合は、次に、1つのシステム上のローカル事前共有鍵がピアのリモート事前共有鍵と一致し、リモート鍵がピアのローカル鍵と一致することを確認します。

構成がまだ機能しない場合は、[240 ページの「IPsec および IKE の意味上の誤りのトラブルシューティング」](#)を参照してください。

例 48 無効な IKEv2 構成の修正

次の出力では、IKE SA のライフタイムが短すぎます。

```
# /usr/lib/inet/in.ikev2d -c
...
May 08 08:52:49: (1) WARNING: Problem in rule "Test 104 to 113"
May 08 08:52:49: (1) HARD lifetime too small (60 < 100)
May 08 08:52:49: (1) -> Using 100 seconds (minimum)
May 08 08:52:49: (1) Checking IKE rule #1: "config 10.134.13.113 to 10.134.13.104"
...
```

この値は `ikev2.config` ファイルに明示的に設定されています。警告を解除するには、ライフタイムの値を少なくとも `100` に変更してサービスをリフレッシュします。

```
# pfedit /etc/inet/ike/ikev2.config
...
## childsa_lifetime_secs 60
childsa_lifetime_secs 100
...
# /usr/lib/inet/in.ikev2d -c
...
# svcadm refresh ikev2
```

例 49 一致しないルールメッセージの修正

次の出力では、事前共有鍵は定義されていますがルールに使用されていません。

```
# /usr/lib/inet/in.ikev2d -c
Feb 4 12:58:31: (1) Reading service properties from smf(5) repository.
Feb 4 12:58:31: (1) Property "config_file" set to: "/etc/inet/ike/ikev2.config"
Feb 4 12:58:31: (1) Property "debug_level" set to: "op"
Feb 4 12:58:31: (1) Warning: debug output being written to stdout.
Feb 4 12:58:31: (1) Checking IKE rule #1: "Test 104 to 113"
Feb 4 12:58:31: (1) Configuration file /etc/inet/ike/ikev2.config is valid.
Feb 4 12:58:31: (1) No matching IKEv2 rule for pre-shared key ending on line 12
Feb 4 12:58:31: (1) Pre-shared key file /etc/inet/ike/ikev2.preshared is valid.
```

この出力は、ルールが1つだけ存在することを示しています。

- ルールが事前共有鍵を必要とする場合、事前共有鍵のラベルがルールのラベルと一致しません。 `ikev2.config` ルールのラベルと `ikev2.preshared` 鍵のラベルが一致するように修正します。
- ルールで証明書が使用されている場合は、 `ikev2.preshared` ファイルの12行目で終わる事前共有鍵を削除するかコメントを解除すると `No matching` メッセージを回避できます。

例 50 実行中のIKEデーモンの新しいデバッグレベルの設定

次の出力では、 `ikev2` サービスのデバッグ出力が `all` に設定されています。

```
# /usr/lib/inet/in.ikev2d -c
Feb 4 12:58:31: (1) Reading service properties from smf(5) repository.
...
Feb 4 12:58:31: (1) Property "debug_level" set to: "all"
...
```

235 ページの「[IPsec と IKE を実行する前にシステムをトラブルシューティングする方法](#)」の **ステップ 2** を実行してもデバッグ出力が `all` ではなく `op` の場合は、 `ikeadm` コマンドを使用して、実行中のIKEデーモンのデバッグレベルを設定してください。

```
# ikedadm set debug_level all
```

IPsec および IKE の意味上の誤りのトラブルシューティング

236 ページの「IPsec 実行中にシステムをトラブルシューティングする方法」の調査で問題に対処できない場合は、ファイルの構文やサービス構成ではなく、構成の意味に問題があると考えられます。

- `ike:default` と `ike:ikev2` 両方のサービスインスタンスが有効になっている場合は、IKEv2 と IKEv1 のルールが重複しないようにしてください。同じネットワークエンドポイントに適用されるルールによって IPsec SA が重複することがあり、これによって特定の状況で接続が失われる可能性があります。

IKE ルールを変更する場合は、ルールをカーネルに読み込みます。

```
# ikeadm -v[1|2] read rule
```

- IKEv1 を実行中の場合は、ルール内のアルゴリズムのメカニズムが接続先の IKEv1 システムで利用できることを確認してください。利用可能なアルゴリズムを表示するには、IKEv2 をサポートしていないシステム上で `ikeadm dump algorithms` コマンドを実行します。

```
# ikeadm dump groups      Available Diffie-Hellman groups
# ikeadm dump encralgs    All IKE encryption algorithms
# ikeadm dump authalgs    All IKE authentication algorithms
```

両方のシステムで利用可能なアルゴリズムを使用するように、IPsec と IKEv1 両方のポリシーファイルを修正します。その後、IKEv1 サービスを再起動して IPsec サービスをリフレッシュします。

```
# svcadm restart ike:default; svcadm refresh ipsec/policy
```

- IKEv1 で事前共有鍵を使用中にリモートの IKEv1 システムをリポートする場合は、ローカルシステム上で `ipseckey flush` コマンドを実行します。
- 自己署名付き証明書を使用している場合は、同じ DN を持つ証明書が再作成されていないこと、証明書のハッシュ値が一致していることを相手の管理者に確認します。確認の手順については、[185 ページの「自己署名付き公開鍵証明書により IKEv2 を構成する方法」のステップ 4](#) を参照してください。

証明書を更新する場合は、新しい証明書をインポートしたあと、IKEv2 サービスをリフレッシュして再起動してください。

- 現在の IKEv2 構成を表示するには `ikeadm -v2 dump | get` コマンドを使用します。使用方法のサマリーについては、[242 ページの「IKE 情報の表示」](#) を参照してください。
- IPsec 関連の統計を表示するには `kstat` コマンドを使用します。詳細は、[kstat\(1M\)](#) のマニュアルページを参照してください。

```
# kstat -m ipsecesp
# kstat -m ipsecah
```

```
# kstat -m ip
```

次の例の kstat 出力は、ipsecesp モジュールに問題がないことを示しています。

```
# kstat -m ipsecesp
module: ipsecesp           instance: 0
name:  esp_stat           class:  net
      acquire_requests    18
      bad_auth             0
      bad_decrypt         0
      bad_padding         0
      bytes_expired       0
      crtime              4.87974774
      crypto_async        0
      crypto_failures     0
      crypto_sync         172
      good_auth           86
      keysock_in          135
      num_aalgs           9
      num_ealgs           13
      out_discards        0
      out_requests        86
      replay_early_failures 0
      replay_failures     0
      sa_port_renumbers   0
      snaptime            5946769.7947628
```

- 保護されていないトラフィックを表示するには snoop コマンドを使用します。Wireshark アプリケーションは snoop 出力を読み取ることができます。snoop 出力の例は、[160 ページの「IPsec によってパケットが保護されていることを確認する方法」](#)を参照してください。

IPsec およびその鍵サービスに関する情報の表示

注記 - ほとんどのコマンドでは、Network IPsec Management 権利プロファイルが割り当てられている管理者になる必要があります。プロファイルシェルで入力する必要があります。詳細は、『[Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護](#)』の「[割り当てられている管理権利の使用](#)」を参照してください。

IPsec および手動鍵サービスプロパティの表示

IPsec ポリシーファイルと手動鍵を保持しているファイルの名前を表示できます。

- IPsec 構成ファイルの名前を表示するには:

```
% svccfg -s policy listprop config/config_file
config/config_file      astring      /etc/inet/ipsecinit.conf
```

- IPsec の手動鍵を保持しているファイルの名前を表示するには:

```
% svccfg -s manual-key listprop config/config_file
config/config_file      astring      /etc/inet/secret/ipseckeys
```

IKE 情報の表示

IKE サービスのプロパティ、IKE 状態および IKE デモンオブジェクトの要素、および証明書検証ポリシーを表示できます。両方の IKE サービスを実行している場合は、サービスごとの情報または両方のサービスの情報を表示できます。これらのコマンドは、テスト、トラブルシューティング、およびモニター中に役立ちます。

- IKE サービスインスタンスのプロパティの表示 – 出力には、構成ファイルの名前を含む IKEv2 サービスの構成可能なプロパティが表示されます。

注記 - IPsec、IKEv2、または IKEv1 サービスの config グループ内でプロパティを変更できること、または変更すべきことを確認するには、[ipseccnf\(1M\)](#)、[in.ikev2d\(1M\)](#)、および [in.iked\(1M\)](#) のマニュアルページを確認してください。たとえば、IKEv2 構成ファイルは特殊な権限で作成され、ikeuser によって所有されず。権限とファイルの所有者は決して変更しないでください。

```
% svccfg -s ipsec/ike:ikev2 listprop config
config          application
config/allow_keydump    boolean    false
config/config_file      astring    /etc/inet/ike/ikev2.config
config/ignore_errors    boolean    false
config/kmf_policy       astring    /etc/inet/ike/kmf-policy.xml
config/max_child_sas     integer    0
config/max_threads      integer    0
config/min_threads      integer    0
config/preshared_file   astring    /etc/inet/ike/ikev2.preshared
config/response_wait_time integer    30
```

```
config/value_authorization astring    solaris.smf.value.ipsec
config/debug_logfile      astring
config/debug_level        astring    op
```

次の例の出力には、IKEv1 サービスの構成可能なプロパティが表示されています。:default サービスインスタンスは指定しないでください。

```
% svccfg -s ipsec/ike listprop config
config                application
config/admin_privilege astring    base
config/config_file    astring    /etc/inet/ike/config
config/debug_level    astring    op
config/debug_logfile  astring    /var/log/in.iked.log
config/ignore_errors  boolean    false
config/value_authorization astring    solaris.smf.value.ipsec
```

- IKE デーモンの現在の状態の表示 – 次の例の出力には、ikeadm コマンドの引数が表示されています。これらの引数はデーモンの現在の状態を表示します。

注記 - ikeadm コマンドを使用するには、IKE デーモンを実行している必要があります。

```
% ikeadm help
...
get  debug|priv|stats|p1|ikesa|rule|preshared|defaults [identifier]
dump p1|ikesa|rule|preshared|certcache|groups|encrals|authalgs
read rule|preshared [filename]
help [get|set|add|del|dump|flush|read|write|token|help]
```

- ikeadm コマンドに対する特定の引数の構文の表示 – コマンド引数の構文を表示するには、help サブコマンドを使用します。例:

```
% ikeadm help read
This command reads a new configuration file into
in.iked, discarding the old configuration info.
```

```
Sets of data that may be read include:
rule          all phase 1/ikesa rules
preshared     all preshared keys
```

```
A filename may be provided to specify a source file
other than the default.
```

- 事前共有鍵の表示 – IKEv1 および IKEv2 の事前共有鍵を表示できます。

注記 - 1 つの IKE バージョンのみを実行している場合は、`-v` オプションを省略できます。

IKEv2 の場合:

```
# ikeadm -v2 dump preshared
```

IKEv1 の場合:

```
# ikeadm set priv keymat
# ikeadm -v1 dump preshared
```

```
PSKEY: Rule label: "Test PSK 197 to 56"
PSKEY: Local pre-shared key (80 bytes): 74206272696c6c696720...3/584
PSKEY: Remote pre-shared key (80 bytes): 74206272696c6c696720...3/584
```

Completed dump of preshared keys

- IKE SA の表示 – 出力には、SA、変換、ローカルシステムとリモートシステム、およびその他の詳細に関する情報が含まれます。通信が要求されていない場合は、SA が存在しないため、表示する情報がありません。

```
# ikeadm -v2 dump ikesa
IKESA: SPIs: Local 0xd3db95689459cca4 Remote 0xb5878717f5cfa877
...
XFORM: Encryption alg: aes-cbc(256..256); Authentication alg: hmac-sha512
...
LOCIP: AF_INET: port 500, 10.1.2.3 (example-3).
...
REMIP: AF_INET: port 500, 10.1.4.5 (ex-2).
...
LIFTM: SA expires in 11459 seconds (3.18 hours)
...
STATS: 0 IKE SA rekeys since initial AUTH.
LOCID: Initiator identity, type FQDN
...
CHILD: ESP Inbound SPI: 0x94841ca3, Outbound SPI 0x074ae1e5
...
Completed dump of IKE SA info
```

- アクティブな IKE ルールの表示 – リストされる IKE ルールは使用中でない場合がありますが、使用は可能です。

```
# ikeadm -v2 dump rule

GLOBL: Label 'Test Rule1 for PSK', key manager cookie 1
```

```

GLOBL: Local auth method=pre-shared key
GLOBL: Remote auth method=pre-shared key

GLOBL: childsa_pfs=false
GLOBL: authentication_lifetime=86400 seconds (1.00 day)
GLOBL: childsa_lifetime=120 seconds (2.00 minutes)
GLOBL: childsa_softlife=108 seconds (1.80 minute)
GLOBL: childsa_idletime=60 seconds
GLOBL: childsa_lifetime_kb=122880 kilobytes (120.00 MB)
GLOBL: childsa_softlife_kb=110592 kilobytes (108.00 MB)
LOCIP: IP address range(s):
LOCIP: 10.142.245.197
REMIP: IP address range(s):
REMIP: 10.134.64.56
LOCID: Identity descriptors:
LOCID: Includes:
LOCID:      fqdn="gloria@ms.mag"
REMIID: Identity descriptors:
REMIID: Includes:
REMIID:      fqdn="gloria@ms.mag"
XFRMS: Available Transforms:

XF 0: Encryption alg: aes-cbc(128..256); Authentication alg: hmac-sha512
XF 0: PRF: hmac-sha512 ; Diffie-Hellman Group: 2048-bit MODP (group 14)
XF 0: IKE SA lifetime before rekey: 14400 seconds (4.00 hours)

```

Completed dump of policy rules

- IKEv2 の証明書検証ポリシーの表示 – dbfile および policy の値を指定する必要があります。
- 動的にダウンロードされる CRL は、応答者タイムアウトの調整に管理者の介入が必要になる場合があります。
次の例の出力では、証明書に組み込まれた URI から CRL がダウンロードされたあと、リストがキャッシュされます。キャッシュに期限切れの CRL が含まれている場合は、新しい CRL がダウンロードされて古い CRL を上書きします。

```

# kmfcfg list dbfile=/etc/inet/ike/kmf-policy.xml policy=default
...
Validation Policy Information:
    Maximum Certificate Revocation Responder Timeout: 10
    Ignore Certificate Revocation Responder Timeout: true
...
CRL:
    Base filename: [not set]
    Directory: /var/user/ikeuser/crls

```

```

Download and cache CRL: true
CRL specific proxy override: www-proxy.cagate.example.com:80
Ignore CRL signature: false
Ignore CRL validity date: false
IPsec policy bypass on outgoing connections: true

```

- 静的にダウンロードされる CRL は、管理者が頻繁に確認する必要があります。管理者が次の値に CRL エントリを設定すると、その管理者は CRL の手動ダウンロード、ディレクトリの入力、および現在の CRL の維持に責任を負います。

```

...
Directory: /var/user/ikeuser/crls
Download and cache CRL: false
Proxy: [not set]
...

```

IPsec およびその鍵サービスの管理

IPsec ポリシーはデフォルトで有効になっていますが、構成情報は含まれていません。

鍵の管理はデフォルトでは有効ではありません。IKE または手動鍵管理、あるいはその両方を構成できます。各 IKE ルールは、どの鍵管理サービスが使用されているかを示しています。ikeadm コマンドは実行中の IKE デーモンを変更できます。

IPsec およびその鍵サービスの構成および管理

- IPsec の構成およびリフレッシュのあとにポリシーを表示:

```

# pfedit /etc/inet/ipsecinit.conf
# ipsecconf -c /etc/inet/ipsecinit.conf
# svcadm refresh ipsec/policy
# ipsecconf -Ln

```

- IPsec の手動鍵の構成および有効化:

```

# pfedit -s /etc/inet/secret/ipseckeys
# svcadm enable ipsec/manual-key

```

- IKEv2 の構成および有効化:

```

# pfedit /etc/inet/ike/ikev2.config
# /usr/lib/inet/in.ikev2d -c
# svcadm enable ipsec/ike:ikev2

```

- IKEv1 の構成および有効化:

```
# pfedit /etc/inet/ike/config
# /usr/lib/inet/in.iked -c
# svcadm enable ipsec/ike:default
```

- IPsec および IKE が有効になっているシステム上でそれらのサービスが構成されていることを確認:

```
# ipsecconf -Ln
# ikeadm -v2 dump rule
# ikeadm set priv keymat
# ikeadm -v1 dump rule
```

- 鍵管理の変更:

IKEv2 の場合:

```
# pfedit /etc/inet/ike/ikev2.config
# /usr/lib/inet/in.ikev2d -c
# svcadm restart ipsec/ike:ikev2
```

IKEv1 の場合:

```
# pfedit /etc/inet/ike/config
# /usr/lib/inet/in.iked -c
# svcadm restart ipsec/ike:default
```

手動鍵管理の場合:

```
# pfedit -s /etc/inet/secret/ipseckeys
# ipseckey -c /etc/inet/secret/ipseckeys
# svcadm refresh ipsec/manual-key
```

- IPsec および IKE の構成可能なプロパティの変更:

IPsec サービス:

```
# svccfg -s ipsec/policy setprop config/property = value
# svcadm refresh ipsec/policy; svcadm restart ipsec/policy
```

機密の鍵情報のための IKEv2 サービス:

```
# svccfg -s ike:ikev2 editprop
# svcadm refresh ipsec/ike:ikev2; svcadm restart ipsec/ike:ikev2
```

その他のプロパティのための IKEv2 サービス:

```
# svccfg -s ike:ikev2 setprop config/property = value
# svcadm refresh ipsec/ike:ikev2; svcadm restart ipsec/ike:ikev2
```

IKEv1 サービス:

```
# svccfg -s ipsec/ike setprop config/property = value
# svcadm refresh ipsec/ike:default; svcadm restart ipsec/ike:default
```

手動鍵サービス:

```
# svccfg -s ipsec/manual-key setprop config/property = value
# svcadm refresh ipsec/manual-key; svcadm restart ipsec/manual-key
```

- IKEv2 の事前共有鍵の構成。

```
# pfedit -s /etc/inet/ike/ikev2.preshared
# /usr/lib/inet/in.ikev2d -c
# svcadm restart ikev2
```

- IKEv1 の事前共有鍵の構成:

```
# pfedit -s /etc/inet/secret/ike.preshared
# svcadm restart ike
```

実行中の IKE デーモンの管理

詳細は、[ikeadm\(1M\)](#) のマニュアルページを確認してください。このセクションのコマンドは、IKEv2 または IKEv1 デーモンの実行中のみ利用できます。

- 実行中の IKE デーモンの変更:

次の出力は、現在のデーモンの状態を変更できる `ikeadm` コマンドの引数を示しています。一部の引数は IKEv2 または IKEv1 デーモンに固有です。

```
% ikeadm help
...
    set  priv level
    set  debug level [filename]
    add  rule|preshared {definition}|filename
    del  p1|ikesa|rule|preshared identifier
    flush p1|ikesa|certcache
    write rule|preshared filename
    token login|logout PKCS#11-Token-Object
```

- `ikeadm` コマンド固有の引数の構文を表示:

```
% ikeadm help add
This command adds items to in.iked's tables.

Objects that may be set include:
    rule          a phase 1 or IKE SA policy rule
    preshared     a preshared key
```

Objects may be entered on the command-line, as a series of keywords and tokens contained in curly

braces ('{', '}'); or the name of a file containing the object definition may be provided.

For security purposes, preshared keys may only be entered on the command-line if `ikeadm` is running in interactive mode.

■ `ikeadm` コマンドで IKEv2 デーモンを変更:

```
# ikeadm add rule | preshared {definition} | filename
# ikeadm flush ikesa
# ikeadm del ikesa | rule | preshared identifier
# ikeadm set debug level
# ikeadm token login | logout PKCS#11-Token-Object
# ikeadm write rule | preshared filename
```

■ `ikeadm` コマンドで IKEv1 デーモンを変更:

```
# ikeadm set debug level
# ikeadm set privlevel
# ikeadm add rule | preshared {definition} | filename
# ikeadm del p1 | rule | preshared identifier
# ikeadm flush p1 | certcache
# ikeadm del rule | preshared id
# ikeadm write rule | preshared filename
```


◆◆◆ 第 14 章

IPsec および鍵管理のリファレンス

この章には、IPsec、IKEv2、および IKEv1 に関する参照情報が記載されています。

- 251 ページの「IPsec リファレンス」
- 257 ページの「IKEv2 リファレンス」
- 261 ページの「IKEv1 リファレンス」

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9章「IPsec の構成」](#)を参照してください。IPsec の概要については、[第8章「IP セキュリティーアーキテクチャーについて」](#)を参照してください。

IKE の実装の手順については、[第11章「IKEv2 の構成」](#)を参照してください。概要については、[第10章「インターネット鍵交換について」](#)を参照してください。

IPsec リファレンス

IPsec のサービス、ファイル、およびコマンド

このセクションには、IPsec サービス、一部の IPsec RFC、および IPsec に関連するファイルとコマンドがリストされます。

IPsec サービス

サービス管理機能 (SMF) は、次の IPsec サービスを提供します。

- `svc:/network/ipsec/policy` サービス – IPsec ポリシーを管理します。デフォルトでは、このサービスは有効になっています。`config_file` プロパティの値によって `ipsecinit.conf` ファイルの場所が決まります。DefaultFixed ネットワーク構成プロファイルを実行しているシステムの初期値は `/etc/inet/ipsecinit.conf` です。このプロファイルを実行していないシステムでは、プロパティ値が空です。

- `svc:/network/ipsec/ipsecalgs` サービス – IPsec で使用できるアルゴリズムを管理します。デフォルトでは、このサービスは有効になっています。
- `svc:/network/ipsec/manual-key` サービス – 手動鍵管理をアクティブにします。デフォルトでは、このサービスは無効になっています。`config_file` プロパティの値によって `ipseckey` 構成ファイルの場所が決まります。初期値は `/etc/inet/secret/ipseckey` 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については、[258 ページの「IKEv2 サービス」](#) および [262 ページの「IKEv1 サービス」](#) を参照してください。

SMF については、『[Oracle Solaris 11.3 でのシステムサービスの管理](#)』の第 1 章、「[サービス管理機能の概要](#)」を参照してください。[smf\(5\)](#)、[svcadm\(1M\)](#)、および [svccfg\(1M\)](#) のマニュアルページも参照してください。

ipsecconf コマンド

ホストの IPsec ポリシーを構成するには、`ipsecconf` コマンドを使用します。このコマンドを実行してポリシーを構成すると、IPsec ポリシーのエントリがカーネル内に作成されます。システムは、これらのエントリを使用して、インバウンドおよびアウトバウンドの IP パケットすべてがポリシーに沿っているかどうかを検査します。トンネリングされずに転送されるパケットは、このコマンドを使用して追加されるポリシー検査の対象になりません。また、`ipsecconf` コマンドは[セキュリティポリシーデータベース \(SPD\)](#) 内の IPsec エントリも管理します。IPsec ポリシーオプションについては、[ipsecconf\(1M\)](#) のマニュアルページを参照してください。

`ipsecconf` コマンドを呼び出すには、`root` 役割になる必要があります。このコマンドは、両方向のトラフィックを保護するエントリを構成できます。このコマンドは、片方向だけのトラフィックを保護するエントリも構成できます。

ローカルアドレスとリモートアドレスというパターンのポリシーエントリは、1つのポリシーエントリで両方向のトラフィックを保護します。たとえば、指定されたホストに対して方向が指定されていない場合、`laddr host1` と `raddr host2` というパターンを含むエントリは、両方向のトラフィックを保護します。そのため、各ホストにポリシーエントリを 1つだけ設定すれば済みます。

`ipsecconf` コマンドで追加されたポリシーエントリには持続性がなく、システムのリブート時に失われます。システムのブート時に IPsec ポリシーが確実にアクティブになるようにするには、`/etc/inet/ipsecinit.conf` ファイルにポリシーエントリを追加したあと、`policy` サービスをリフレッシュするか有効化します。例については、[137 ページの「IPsec によるネットワークトラフィックの保護」](#) を参照してください。

ipsecinit.conf 構成ファイル

Oracle Solaris を起動したときに IPsec セキュリティポリシーを有効化するには、特定の IPsec ポリシーエントリを使用して構成ファイルを作成し IPsec を初期化します。このファイルのデフォルトの名前は `/etc/inet/ipsecinit.conf` です。ポリシーエントリとその形式の詳細については、[ipsecconf\(1M\)](#) のマニュアルページを参照してください。ポリシーの構成が完了したら、`svcadm refresh ipsec/policy` コマンドでポリシーをリフレッシュできます。

サンプルの ipsecinit.conf ファイル

Oracle Solaris ソフトウェアには、サンプルの IPsec ポリシーファイル `ipsecinit.sample` が含まれます。このファイルをテンプレートとして独自の `ipsecinit.conf` ファイルを作成できます。`ipsecinit.sample` ファイルには、次のエントリが含まれています。

```
...
# In the following simple example, outbound network traffic between the local
# host and a remote host will be encrypted. Inbound network traffic between
# these addresses is required to be encrypted as well.
#
# This example assumes that 10.0.0.1 is the IPv4 address of this host (laddr)
# and 10.0.0.2 is the IPv4 address of the remote host (raddr).
#
{laddr 10.0.0.1 raddr 10.0.0.2} ipsec
  {encr_algs aes encr_auth_algs sha256 sa shared}

# The policy syntax supports IPv4 and IPv6 addresses as well as symbolic names.
# Refer to the ipsecconf(1M) man page for warnings on using symbolic names and
# many more examples, configuration options and supported algorithms.
#
# This example assumes that 10.0.0.1 is the IPv4 address of this host (laddr)
# and 10.0.0.2 is the IPv4 address of the remote host (raddr).
#
# The remote host will also need an IPsec (and IKE) configuration that mirrors
# this one.
#
# The following line will allow ssh(1) traffic to pass without IPsec protection:

{lport 22 dir both} bypass {}

#
# {laddr 10.0.0.1 dir in} drop {}
#
# Uncommenting the above line will drop all network traffic to this host unless
# it matches the rules above. Leaving this rule commented out will allow
# network packets that do not match the above rules to pass up the IP
# network stack. , , ,
```

ipsecinit.conf と ipsecconf のセキュリティーについて

確立された接続の IPsec ポリシーを変更することはできません。ポリシーの変更ができないソケットを、「ラッチされたソケット」と呼びます。新しいポリシーエントリは、すでにラッチされたソケットを保護しません。詳細については、[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。詳細は、[ipsecconf\(1M\)](#) のマニュアルページの「セキュリティー」セクションを参照してください。

ipsecalgs コマンド

暗号化フレームワークは、認証と暗号化のアルゴリズムを IPsec に提供します。ipsecalgs コマンドを使用すると、各 IPsec プロトコルでサポートされているアルゴリズムを一覧表示できます。ipsecalgs の構成は /etc/inet/ipsecalgs ファイルに保存されます。通常、このファイルを変更する必要はなく、直接編集してはいけません。ただし、ファイルを変更する必要がある場合は、ipsecalgs コマンドを使用します。サポートされるアルゴリズムは、システムのブート時に svc:/network/ipsec/ipsecalgs:default サービスによってカーネルと同期されます。

有効な IPsec プロトコルおよびアルゴリズムは、RFC 2407 に記載されている ISAKMP 解釈ドメイン (DOI) によって記述されます。特に、ISAKMP DOI は、有効な IPsec アルゴリズム PROTO_IPSEC_AH とそのプロトコルの PROTO_IPSEC_ESP の命名規約や番号付け規約を定義します。1つのアルゴリズムは1つのプロトコルだけに関連します。このような ISAKMP DOI 定義は、/etc/inet/ipsecalgs ファイルにあります。アルゴリズム番号とプロトコル番号は、Internet Assigned Numbers Authority (IANA) によって定義されます。ipsecalgs コマンドは、IPsec アルゴリズムのリストを拡張します。

アルゴリズムの詳細については、[ipsecalgs\(1M\)](#) のマニュアルページを参照してください。暗号化フレームワークの詳細については、『[Oracle Solaris 11.3 での暗号化と証明書管理](#)』の第1章、「暗号化フレームワーク」を参照してください。

ipseckey コマンド

ipseckey コマンドは、各種オプションとともに IPsec の鍵を手動で管理します。ipseckey コマンドについては、[ipseckey\(1M\)](#) のマニュアルページを参照してください。

ipseckey におけるセキュリティーについて

ipseckey コマンドを使用すると、Network Security または Network IPsec Management 権利プロファイルを持つ役割は、暗号鍵に関する機密情報を入力できます。場合によっては、不正にこの情報にアクセスして IPsec トラフィックのセキュリティーを損なうことも可能です。

注記 - 可能な場合は、手動の鍵処理ではなく IKE を使用します。

詳細は、[ipseckey\(1M\)](#) のマニュアルページの「セキュリティー」セクションを参照してください。

kstat コマンド

kstat コマンドは、ESP、AH、およびその他の IPsec データの統計を表示できます。IPsec 関連のオプションは、[240 ページの「IPsec および IKE の意味上の誤りのトラブルシューティング」](#) にリストされています。[kstat\(1M\)](#) のマニュアルページも参照してください。

snoop コマンドと IPsec

snoop コマンドは、AH ヘッダーと ESP ヘッダーを構文解析できます。ESP はそのデータを暗号化するため、ESP で暗号化および保護されたヘッダーは snoop コマンドでは読み取ることができません。AH はデータを暗号化しないため、AH で保護されたトラフィックは snoop コマンドで検査できます。このコマンドに `-v` オプションを指定すると、いつ AH がパケットに使用されているかを表示できます。詳細は、[snoop\(1M\)](#) のマニュアルページを参照してください。

保護されたパケットに snoop コマンドを実行した場合の詳細な出力については、[160 ページの「IPsec によってパケットが保護されていることを確認する方法」](#) を参照してください。

このリリースにバンドルされている無料のオープンソースソフトウェア [Wireshark \(http://www.wireshark.org/about.html\)](#) のように、サードパーティーのネットワークアナライザも利用できます。

IPsec RFC

インターネットエンジニアリングタスクフォース (IETF) は、IP 層のセキュリティーアーキテクチャーを説明するいくつかの RFC (Requests for Comments) を公表していま

す。RFC へのリンクについては、<http://www.ietf.org/>を参照してください。次の RFC リストは、比較的一般的な IP セキュリティーの参考文献です。

- RFC 2411、『IP Security Document Roadmap』、1998 年 11 月
- RFC 2401、『Security Architecture for the Internet Protocol』、1998 年 11 月
- RFC 2402、『IP Authentication Header』、1998 年 11 月
- RFC 2406、『IP Encapsulating Security Payload (ESP)』、1998 年 11 月
- RFC 2408、『Internet Security Association and Key Management Protocol (ISAKMP)』、1998 年 11 月
- RFC 2407、『The Internet IP Security Domain of Interpretation for ISAKMP』、1998 年 11 月
- RFC 2409、『The Internet Key Exchange (IKEv1)』、1998 年 11 月
- RFC 5996、『Internet Key Exchange Protocol Version 2 (IKEv2)』、2010 年 9 月
- RFC 3554、『On the Use of Stream Control Transmission Protocol (SCTP) with IPsec』、2003 年 7 月

IPsec のセキュリティーアソシエーションデータベース

IPsec セキュリティーサービスの鍵情報は、セキュリティーアソシエーションデータベース (SADB) に保存されます。セキュリティーアソシエーション (SA) は、インバウンドパケットとアウトバウンドパケットを保護します。

`in.iked` デーモンと `ipseckey` コマンドは `PF_KEY` ソケットインタフェースを使用して SADB を保守します。SADB が要求やメッセージを処理する方法の詳細については、[pf_key\(7P\)](#) のマニュアルページを参照してください。

IPsec での鍵管理

インターネット鍵交換 (IKE) プロトコルにより、IPsec の鍵管理が自動的に行われます。IPsec SA は `ipseckey` コマンドによって手動で管理することもできますが、IKE が推奨されます。詳細は、[124 ページの「IPsec セキュリティーアソシエーションの鍵管理」](#)を参照してください。

Oracle Solaris のサービス管理機能 (SMF) 機能は、次の IPsec 用鍵管理サービスを提供します。

- `svc:/network/ipsec/ike` サービス – 自動鍵管理のための SMF サービスです。ike サービスには 2 つのインスタンスがあります。ike:ikev2 サービスイ

インスタンスは `in.ikev2d` デーモン (IKEv2) を実行して自動鍵管理を提供します。ike:default サービスは `in.iked` デーモン (IKEv1) を実行します。IKE については、[第10章「インターネット鍵交換について」](#)を参照してください。デーモンの詳細は、[in.ikev2d\(1M\)](#) および [in.iked\(1M\)](#) のマニュアルページを参照してください。

- `svc:/network/ipsec/manual-key:default` サービス – 手動鍵管理のための SMF サービスです。manual-key サービスは `ipseckey` コマンドを各種オプションで実行して、鍵を手動で管理します。`ipseckey` コマンドについては、[ipseckey\(1M\)](#) のマニュアルページを参照してください。

IKEv2 リファレンス

IKEv2 は IKEv1 の後継です。比較については、[168 ページの「IKEv2 と IKEv1 の比較」](#)を参照してください。

IKEv2 のユーティリティーおよびファイル

次の表は、IKEv2 ポリシーの構成ファイル、IKEv2 鍵の保管場所、および IKEv2 を実装する各種コマンドとサービスについてまとめたものです。サービスの詳細については、『[Oracle Solaris 11.3 でのシステムサービスの管理](#)』の [第 1 章、「サービス管理機能の概要」](#)を参照してください。

表 17 IKEv2 のサービス名、コマンド、構成と鍵の保管場所、およびハードウェアデバイス

ファイル、場所、コマンド、またはサービス	説明	マニュアルページ
<code>svc:/network/ipsec/ike:ikev2</code>	IKEv2 を管理する SMF サービス。	smf(5)
<code>/usr/lib/inet/in.ikev2d</code>	インターネット鍵交換 (IKE) のデーモン。ike:ikev2 サービスが使用可能なときに自動鍵管理をアクティブ化します。	in.ikev2d(1M)
<code>/usr/sbin/ikeadm [-v 2]</code>	IKEv2 ポリシーの表示および一時的な変更用の IKE 管理コマンド。使用可能な Diffie-Hellman グループなどの IKEv2 管理オブジェクトを表示できます。	ikeadm(1M)
<code>/usr/sbin/ikev2cert</code>	構成の所有者 <code>ikeuser</code> として公開鍵証明書を作成および格納するための証明書データベース管理コマンド。pktool コマンドを呼び出します。	ikev2cert(1M) pktool(1)
<code>/etc/inet/ike/ikev2.config</code>	IKEv2 ポリシーのデフォルト構成ファイル。インバウンド IKEv2 要求のマッチングとアウトバウンド IKEv2 要求の準備に関するサイトのルールが含まれています。 このファイルが存在する場合、ike:ikev2 サービスが有効になると in.ikev2d デーモンが起動します。このファイルの場所は <code>svccfg</code> コマンドを使用して変更できます。	ikev2.config(4)

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
/etc/inet/ike/ ikev2.preshared	証明書ベースの認証を使用していない2つのIKEv2インスタンスが互いを認証するために使用できる秘密鍵が含まれています。	ikev2.preshared(4)
ソフトトークンキーストア	ikeuser が所有する IKEv2 の非公開鍵および公開鍵証明書が含まれています。	pkcs11-softtoken(5)

IKEv2 サービス

サービス管理機能 (SMF) は、IKEv2 を管理するための `svc:/network/ipsec/ike:ikev2` サービスインスタンスを提供します。デフォルトでは、このサービスは無効になっています。このサービスを有効にする前に、`/etc/inet/ike/ikev2.config` ファイル内に有効な IKEv2 構成を作成する必要があります。

次の `ike:ikev2` サービスプロパティは構成可能です。

- **config_file プロパティ** – IKEv2 構成ファイルの場所を指定します。初期値は `/etc/inet/ike/ikev2.config` です。このファイルは特殊な権限を持ち、`ikeuser` に所有されている必要があります。別のファイルを使用しないでください。
- **debug_level プロパティ** – `in.ikev2d` デーモンのデバッグレベルを設定します。初期値は `op` (動作) です。指定可能な値については、[ikeadm\(1M\)](#) のマニュアルページで、オブジェクトタイプの下にあるデバッグレベルの表を参照してください。
- **debug_logfile プロパティ** – IKEv2 をデバッグするためのログファイルの場所を指定します。初期値は `/var/log/ikev2/in.ikev2d.log` です。
- **kmf_policy プロパティ** – 証明書ポリシーのためのログファイルの場所を設定します。デフォルト値は `/etc/inet/ike/kmf-policy.xml` です。このファイルは特殊な権限を持ち、`ikeuser` に所有されている必要があります。別のファイルを使用しないでください。
- **pkcs11_token/pin プロパティ** – IKEv2 デーモンの起動時にキーストアへのログインに使用する PIN を設定します。この値は、`ikev2cert setpin` コマンドでトークンに設定した値と一致する必要があります。
- **pkcs11_token/uri プロパティ** – キーストアへの PKCS #11 URI を設定します。暗号化アクセラレータカード上のハードウェアストレージを使用するには、この値を指定する必要があります。

SMF については、『[Oracle Solaris 11.3 でのシステムサービスの管理](#)』の第1章、「[サービス管理機能の概要](#)」を参照してください。[smf\(5\)](#)、[svcadm\(1M\)](#)、および [svccfg\(1M\)](#) のマニュアルページも参照してください。

IKEv2 デーモン

`in.ikev2d` デーモンは、Oracle Solaris システム上で IPsec の暗号化鍵の管理を自動化します。また、同じプロトコルを実行するリモートシステムとのネゴシエーションを行い、認証された鍵情報が、保護された方法でセキュリティーアソシエーション (SA) に提供されます。このデーモンは、IKEv2 プロトコルを使用して通信を保護するために IPsec の使用を予定しているすべてのシステム上で実行されている必要があります。

デフォルトでは、`svc:/network/ipsec/ike:ikev2` サービスは有効になっていません。`/etc/inet/ike/ikev2.config` ファイルを構成して `ike:ikev2` サービスインスタンスを有効にすると、システムブート時に SMF が `in.ikev2d` デーモンを起動します。

IKEv2 デーモンを実行すると、システムはそのピア IKEv2 エンティティーに対して自分自身を認証し、セッション鍵を確立します。構成ファイルで指定した間隔で、IKE 鍵が自動的に置き換えられます。`in.ikev2d` デーモンは、ネットワークからの着信 IKE 要求と `PF_KEY` ソケット経由のアウトバウンドトラフィックの要求を待機します。詳細は、[pf_key\(7P\)](#) のマニュアルページを参照してください。

2 つのコマンドが IKEv2 デーモンをサポートします。`ikeadm` コマンドを使用して IKE ポリシーを表示できます。詳細は、[260 ページの「IKEv2 の `ikeadm` コマンド](#)」を参照してください。`ikev2cert` コマンドでは、公開鍵証明書と非公開鍵証明書を表示および管理できます。詳細は、[260 ページの「IKEv2 `ikev2cert` コマンド](#)」を参照してください。

IKEv2 構成ファイル

IKEv2 構成ファイル `/etc/inet/ike/ikev2.config` は、IPsec ポリシーファイル `/etc/inet/ipsecinit.conf` で保護されている、指定されたネットワークエンドポイントの鍵をネゴシエートするのに使用されるルールを管理します。

IKE での鍵管理には、ルールとグローバルパラメータが関係します。IKE ルールは、その鍵情報で保護するシステムやネットワークを識別します。さらに、ルールは認証方式も指定します。グローバルパラメータには、IKEv2 SA の鍵が再生成されるまでのデフォルトの時間 `ikesa_lifetime_secs` などの項目が含まれます。IKEv2 構成ファイルの例については、[176 ページの「事前共有鍵による IKEv2 の構成](#)」を参照してください。IKEv2 ポリシーエントリの例と説明については、[ikev2.config\(4\)](#) のマニュアルページを参照してください。

IKEv2 がサポートする IPsec SA は、IPsec 構成ファイル `/etc/inet/ipsecinit.conf` のポリシーに従って IP パケットを保護します。

ike/ikev2.config ファイルのセキュリティーに関する注意点は、ipsecinit.conf ファイルの注意点と同様です。詳細は、[254 ページの「ipsecinit.conf と ipsecconf のセキュリティーについて」](#)を参照してください。

IKEv2 の ikeadm コマンド

in.ikev2d デーモンの実行中は、ikeadm [-v2] コマンドを使用して次を実行できます。

- IKEv2 状態の要素の表示。
- ポリシー規則、事前共有鍵、使用可能な Diffie-Hellman グループ、暗号化アルゴリズムと認証アルゴリズム、および既存のアクティブな IKEv2 SA などの IKEv2 デーモンオブジェクトの表示。

このコマンドのオプションの例と詳しい説明については、[ikeadm\(1M\)](#)のマニュアルページを参照してください。

ikeadm コマンドのセキュリティーについては、ipseckey コマンドのセキュリティーと同様です。詳細は、[255 ページの「ipseckey におけるセキュリティーについて」](#)を参照してください。

IKEv2 事前共有鍵ファイル

/etc/inet/ike/ikev2.preshared には、IKEv2 サービスによって使用される事前共有鍵が含まれます。このファイルは ikeuser によって所有され、0600 で保護されます。

ike/ikev2.config ファイル内で事前共有鍵を要求するルールを構成する場合は、デフォルトの ikev2.preshared ファイルをカスタマイズする必要があります。IKEv2 はこれらの事前共有鍵を使用して IKEv2 ピアを認証するため、このファイルは in.ikev2d デーモンが事前共有鍵を要求するルールを読み込む前に有効になっている必要があります。

IKEv2 ikev2cert コマンド

ikev2cert コマンドは、公開および非公開の鍵および証明書を生成、格納、および管理するのに使用されます。このコマンドは、ike/ikev2.config ファイルが公開鍵証明書を要求するときに使用します。IKEv2 はこれらの証明書を使用して IKEv2 ピアを認証するため、証明書は in.ikev2d デーモンがその証明書を要求するルールを読み込む前に適用されている必要があります。

ikev2cert コマンドは pktool コマンドを ikeuser として呼び出します。

次の ikev2cert コマンドは IKEv2 の証明書を管理します。このコマンドは ikeuser アカウントによって実行される必要があります。結果は PKCS #11 ソフトトークン キーストアに格納されます。

- ikev2cert setpin-ikeuser ユーザーの PIN を生成します。この PIN は証明書を使用する際に必要です。
- ikev2cert gencert-自己署名付き証明書を生成します。
- ikev2cert gencsr-証明書署名要求 (CSR) を生成します。
- ikev2cert list-キーストア内の証明書を一覧表示します。
- ikev2cert export-エクスポート用ファイルに証明書をエクスポートします。
- ikev2cert import-証明書または CRL をインポートします。

ikev2cert サブコマンドの構文については、[pktool\(1\)](#) のマニュアルページを参照してください。例については、[ikev2cert\(1M\)](#) のマニュアルページを参照してください。ソフトトークンキーストアについては、[cryptoadm\(1M\)](#) のマニュアルページを参照してください。

IKEv1 リファレンス

次のセクションでは、IKEv1 の参照情報を示します。IKEv1 は、より高速な自動鍵管理を提供する IKEv2 に置き換わっています。IKEv2 の詳細については、[257 ページの「IKEv2 リファレンス」](#)を参照してください。比較については、[168 ページの「IKEv2 と IKEv1 の比較」](#)を参照してください。

IKEv1 のユーティリティーおよびファイル

次の表は、IKEv1 ポリシーの構成ファイル、IKEv1 鍵の保管場所、および IKEv1 を実装する各種コマンドとサービスについてまとめたものです。サービスの詳細については、『[Oracle Solaris 11.3 でのシステムサービスの管理](#)』の第 1 章、「[サービス管理機能の概要](#)」を参照してください。

表 18 IKEv1 のサービス名、コマンド、構成と鍵の保管場所、およびハードウェアデバイス

サービス、コマンド、ファイル、またはデバイス	説明	マニュアルページ
svc:/network/ipsec/ike:default	IKEv1 を管理する SMF サービス。	smf(5)
/usr/lib/inet/in.iked	インターネット鍵交換 (IKEv1) デーモン。ike サービスが使用可能などに自動鍵管理をアクティブ化します。	in.iked(1M)

サービス、コマンド、ファイル、またはデバイス	説明	マニュアルページ
/usr/sbin/ikeadm [-v1]	IKE ポリシーの表示および一時的な変更用の IKE 管理コマンド。フェーズ 1 アルゴリズムや使用可能な Diffie-Hellman グループなどの IKE 管理オブジェクトを表示できます。	ikeadm(1M)
/usr/sbin/ikecert	公開鍵証明書が格納されているローカルデータベースを操作する証明書データベース管理コマンド。データベースは、接続されたハードウェアにも格納できます。	ikecert(1M)
/etc/inet/ike/config	デフォルトの IKEv1 ポリシー構成ファイル。インバウンド IKEv1 要求のマッチングとアウトバウンド IKEv1 要求の準備に関するサイトのルールが含まれています。 このファイルが存在する場合、ike サービスが有効になると in.iked デーモンが起動します。このファイルの場所は svccfg コマンドを使用して変更できます。	ike.config(4)
ike.preshared	/etc/inet/secret ディレクトリにある事前共有鍵ファイル。フェーズ 1 交換での認証の秘密鍵が含まれます。事前共有鍵で IKEv1 を構成するときに使用されます。	ike.preshared(4)
ike.privatekeys	/etc/inet/secret ディレクトリにある非公開鍵ディレクトリ。公開鍵と非公開鍵のペアの非公開部分が含まれています。	ikecert(1M)
publickeys ディレクトリ	公開鍵および証明書ファイルが格納されている /etc/inet/ike ディレクトリ内のディレクトリ。公開鍵と非公開鍵のペアの公開部分が含まれています。	ikecert(1M)
cr1s ディレクトリ	公開鍵および証明書ファイルの失効リストが格納されている /etc/inet/ike ディレクトリ内のディレクトリ。	ikecert(1M)
Sun Crypto Accelerator 6000 ボード	オペレーティングシステムの処理をオフロードすることで公開鍵の処理を高速化するハードウェア。公開鍵、非公開鍵、および公開鍵証明書も格納します。Sun Crypto Accelerator 6000 ボードはレベル 3 の FIPS 140-2 認定デバイスです。	ikecert(1M)

IKEv1 サービス

サービス管理機能 (SMF) は、IKEv1 を管理するための `svc:/network/ipsec/ike:default` サービスを提供します。デフォルトでは、このサービスは無効になっています。このサービスを有効にする前に、IKEv1 構成ファイル `/etc/inet/ike/config` を作成する必要があります。

次の `ike` サービスのプロパティは構成可能です。

- `config_file` プロパティ – IKEv1 構成ファイルの場所を設定します。初期値は `/etc/inet/ike/config` です。
- `debug_level` プロパティ – `in.iked` デーモンのデバッグレベルを設定します。初期値は `op` (動作) です。指定可能な値については、[ikeadm\(1M\)](#) のマニュアルページで、オブジェクトタイプの下にあるデバッグレベルの表を参照してください。

- `admin_privilege` プロパティ – `in.iked` デーモンの特権レベルを設定します。初期値は `base` です。ほかの値は `modkeys` と `keymat` です。詳細は、264 ページの「IKEv1 `ikeadm` コマンド」を参照してください。

SMF については、『Oracle Solaris 11.3 でのシステムサービスの管理』の第 1 章、「サービス管理機能の概要」を参照してください。[smf\(5\)](#)、[svcadm\(1M\)](#)、および [svccfg\(1M\)](#) のマニュアルページも参照してください。

IKEv1 デーモン

`in.iked` デーモンは IPsec SA の管理を自動化し、これには IPsec を使用するパケットを保護する暗号化鍵が含まれます。このデーモンは、IKEv1 プロトコルを実行しているピアシステムと、ISAKMP SA および IPsec SA をセキュアにネゴシエートします。

デフォルトでは、`svc:/network/ipsec/ike:default` サービスは有効になっていません。`/etc/inet/ike/config` ファイルを構成して `ike:default` サービスを有効にすると、システムブート時に SMF が `in.iked` デーモンを起動します。`/etc/inet/ike/config` ファイルに加え、その他の構成がほかのファイルおよびデータベースに、または SMF プロパティとして格納されています。詳細は、261 ページの「IKEv1 のユーティリティおよびファイル」、および [ike.preshared\(4\)](#)、[ikecert\(1M\)](#)、[in.iked\(1M\)](#) のマニュアルページを参照してください。

`ike:default` サービスを有効にすると、`in.iked` デーモンが構成ファイルを読み込んで、SA に対する IKE ピアからの外部要求および IPsec からの内部要求を待機します。

IKEv1 ピアからの外部要求の場合、`ike:default` サービスの構成によってデーモンの応答方法が決まります。内部要求は `PF_KEY` インタフェースを介してルーティングされます。このインタフェースは、IPsec SA を格納してパケットの暗号化および復号化を実行する IPsec のカーネル部分と、ユーザーランドで実行する鍵管理デーモン `in.iked` との間の通信を処理します。カーネルは、パケットの保護に SA を必要とする場合、`PF_KEY` インタフェースを介して `in.iked` デーモンにメッセージを送信します。詳細は、[pf_key\(7P\)](#) のマニュアルページを参照してください。

2 つのコマンドが IKEv1 デーモンをサポートします。`ikeadm` コマンドは、実行中のデーモンにコマンド行インタフェースを提供します。`ikecert` コマンドは、ディスクおよびハードウェア上の証明書データベース `ike.privatekeys` および `publickeys` を管理します。

これらのコマンドの詳細については、[in.iked\(1M\)](#)、[ikeadm\(1M\)](#)、および [ikecert\(1M\)](#) のマニュアルページを参照してください。

IKEv1 構成ファイル

IKEv1 構成ファイル `/etc/inet/ike/config` は、IPsec 構成ファイル `/etc/inet/ipsecinit.conf` 内のポリシーに従って、IPsec の保護を必要とするネットワークパケットのための SA を管理します。

IKE での鍵管理には、ルールとグローバルパラメータが関係します。IKEv1 ルールは、別の IKEv1 デーモンを実行中のシステムを識別します。さらに、ルールは認証方式も指定します。グローバルパラメータには、接続されたハードウェアアクセラレータへのパスなどがあります。IKEv1 ポリシーファイルの例については、[176 ページの「事前共有鍵による IKEv2 の構成」](#)を参照してください。IKEv1 ポリシーエントリの例と説明については、[ike.config\(4\)](#) のマニュアルページを参照してください。

`/etc/inet/ike/config` ファイルには、RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki) 規格に従って実装されているライブラリへのパスが含まれることがあります。IKEv1 はこの PKCS #11 ライブラリを使用して、鍵アクセラレーションおよび鍵の格納のためにハードウェアにアクセスします。

`ike/config` ファイルのセキュリティーに関する注意点は、`ipsecinit.conf` ファイルの注意点と同様です。詳細は、[254 ページの「ipsecinit.conf と ipsecconf のセキュリティーについて」](#)を参照してください。

IKEv1 `ikeadm` コマンド

`ikeadm` コマンドを実行すると、次のことができます。

- IKE 状態の要素の表示
- IKE デーモンのプロパティの変更
- フェーズ 1 交換時の SA 作成に関する統計の表示
- IKE プロトコル交換のデバッグ
- すべてのフェーズ 1 SA、ポリシー規則、事前共有鍵、使用可能な Diffie-Hellman グループ、フェーズ 1 暗号化および認証アルゴリズム、証明書キャッシュなどの IKE デーモンオブジェクトの表示

このコマンドのオプションの例と詳しい説明については、[ikeadm\(1M\)](#) のマニュアルページを参照してください。

実行する IKE デーモンの特権レベルにより、表示および変更可能な IKE デーモンの要素が決まります。使用可能な特権レベルは 3 つあります。

base レベル	鍵を表示したり変更したりすることはできません。base レベルはデフォルトの特権レベルです。
keymat レベル	ikeadm コマンドで実際の鍵を表示できます。
modkeys レベル	事前共有鍵の削除、変更、追加ができます。

特権の一時的な変更には、ikeadm コマンドを使用できます。永続的に変更する場合は、ike サービスの admin_privilege プロパティを変更します。特権の一時的な変更については、[248 ページの「実行中の IKE デーモンの管理」](#)を参照してください。

ikeadm コマンドのセキュリティーについては、ipseckey コマンドのセキュリティーと同様です。[255 ページの「ipseckey におけるセキュリティーについて」](#)を参照してください。ikeadm コマンドに固有の詳細については、[ikeadm\(1M\)](#)のマニュアルページを参照してください。

IKEv1 事前共有鍵ファイル

事前共有鍵を手動で作成すると、鍵は、/etc/inet/secret ディレクトリのファイルに格納されます。ike/config 内に事前共有鍵を使用するルールを構成すると、ike.preshared ファイルにフェーズ 1 交換用の事前共有鍵が含まれます。ipseckey ファイルには、IP パケットの保護に使用される事前共有鍵が含まれます。これらのファイルは 0600 で保護されます。secret ディレクトリは 0700 で保護されます。

フェーズ 1 交換の認証に事前共有鍵を使用するため、このファイルを in.iked デーモンの開始前に有効にする必要があります。

手動による IPsec 鍵の管理の例については、[154 ページの「IPsec の鍵を手動で作成する方法」](#)を参照してください。

IKEv1 公開鍵のデータベースおよびコマンド

ikecert コマンドは、ローカルシステムの公開鍵/非公開鍵、公開証明書、および静的 CRL のデータベースを管理します。このコマンドは、IKEv1 構成ファイルが公開鍵証明書を要求するときに使用します。IKEv1 ではこれらのデータベースを使用してフェーズ 1 交換を認証するため、in.iked デーモンを起動する前に、それらのデータベースに必要な情報が含まれていなければなりません。3つのサブコマンド certlocal、certdb、および certrldb は、3つの各データベースを処理します。

システムに Sun Crypto Accelerator 6000 ボードが接続されている場合、ikecert コマンドは PKCS #11 ライブラリを使用してハードウェア鍵と証明書のストレージにアクセスします。

詳細は、[ikecert\(1M\)](#) のマニュアルページを参照してください。メタスロットとソフトトークンキースタについては、[cryptoadm\(1M\)](#) のマニュアルページを参照してください。

IKEv1 `ikecert tokens` コマンド

`tokens` 引数は、使用可能なトークン ID を一覧表示します。トークン ID により、`ikecert certlocal` コマンドと `ikecert certdb` コマンドは、公開鍵証明書と CSR を生成できます。鍵および証明書も接続された Sun Crypto Accelerator 6000 ボードに格納できます。`ikecert` コマンドは、PKCS #11 ライブラリを使用してハードウェアキースタにアクセスします。

IKEv1 `ikecert certlocal` コマンド

`certlocal` サブコマンドは、非公開鍵データベースを管理します。このサブコマンドを選択すると、非公開鍵の追加、表示、および削除を行うことができます。このサブコマンドは、自己署名付き証明書または CSR も作成します。`-ks` オプションを選択すると、自己署名付き証明書が作成されます。`-kc` オプションは CSR を作成します。鍵はシステムの `/etc/inet/secret/ike.privatekeys` ディレクトリに格納されるか、`-T` オプションを指定した場合は、接続されたハードウェアに格納されます。

非公開鍵を作成する場合は、`ikecert certlocal` コマンドのオプションに関連するエントリが `ike/config` ファイルに存在しなければなりません。`ikecert` オプションと `ike/config` エントリの対応を次の表に示します。

表 19 IKEv1 の `ikecert` オプションと `ike/config` エントリの対応

ikecert オプション	ike/config エントリ	説明
<code>-A subject-alternate-name</code>	<code>cert_trust subject-alternate-name</code>	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。
<code>-D X.509-distinguished-name</code>	<code>X.509-distinguished-name</code>	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。
<code>-t dsa-sha1 dsa-sha256</code>	<code>auth_method dsa_sig</code>	RSA よりもわずかに遅い認証方式。
<code>-t rsa-md5</code> および	<code>auth_method rsa_sig</code>	DSA よりもわずかに速い認証方式。
<code>-t rsa-sha1 rsa-sha256 rsa-sha384 rsa-sha512</code>		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。
<code>-t rsa-md5</code> および	<code>auth_method rsa_encrypt</code>	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。
<code>-t rsa-sha1 rsa-sha256 rsa-sha384 rsa-sha512</code>		

`ikecert certlocal -kc` コマンドで CSR を発行する場合、コマンドの出力を認証局 (CA) に送信します。会社が独自の公開鍵インフラストラクチャー (PKI) を運用している場合は、PKI の管理者に出力を送信します。その後、CA または PKI の管理者が証明書を作成します。返された証明書は `certdb` サブコマンドに渡されます。CA から返された証明書失効リスト (CRL) は、`certrldb` サブコマンドに渡されます。

IKEv1 `ikecert certdb` コマンド

`certdb` サブコマンドは、公開鍵データベースを管理します。そのサブコマンドを選択すると、公開鍵と証明書を追加、表示、および削除できます。このコマンドは、リモートシステムで `ikecert certlocal -ks` コマンドによって生成された証明書を入力として受け入れます。手順については、[207 ページの「自己署名付き公開鍵証明書により IKEv1 を構成する方法」](#)を参照してください。このコマンドは、CA から受信する証明書も入力として受け入れます。手順については、[212 ページの「CA の署名付き証明書で IKEv1 を構成する方法」](#)を参照してください。

証明書と公開鍵は、システムの `/etc/inet/ike/publickeys` ディレクトリに格納されます。`-T` オプションを指定した場合、証明書、非公開鍵、公開鍵は、システムに接続されたハードウェアに格納されます。

IKEv1 `ikecert certrldb` コマンド

`certrldb` サブコマンドは、証明書失効リスト (CRL) データベース `/etc/inet/ike/crls` を管理します。CRL データベースには、公開鍵の失効リストが保存されています。よって、このリストには、すでに有効でない証明書が明記されます。CA から CRL が提供される場合、その CRL を `ikecert certrldb` コマンドで CRL データベースにインストールできます。手順については、[221 ページの「IKEv1 で失効した証明書を処理する方法」](#)を参照してください。

IKEv1 `/etc/inet/ike/publickeys` ディレクトリ

`/etc/inet/ike/publickeys` ディレクトリのファイルまたはスロットには、公開鍵と非公開鍵のペアの公開部分とその証明書が含まれています。このディレクトリは 0755 で保護されています。`ikecert certdb` コマンドを使用して、そのディレクトリを読み込みます。`-T` オプションは、鍵を `publickeys` ディレクトリではなく Sun Crypto Accelerator 6000 ボード上に格納します。

スロットには、別のシステムで生成された証明書の X.509 識別名が、エンコードされた形式で含まれます。自己署名付き証明書を使用する場合、そのコマンドへの入力として、リモートシステムの管理者から受信する証明書を使用します。CA からの証明書を使用する場合、CA から受け取る 2 つの署名付き証明書をこのデータベースに格

納します。CA に送信した CSR に基づいた証明書をインストールします。また、CA の証明書も格納します。

IKEv1 /etc/inet/secret/ike.privatekeys ディレクトリ

/etc/inet/secret/ike.privatekeys ディレクトリには、公開鍵/非公開鍵ペアの一部である非公開鍵ファイルが格納されています。このディレクトリは 0700 で保護されています。ikecert certlocal コマンドを実行して、ike.privatekeys ディレクトリを読み込みます。非公開鍵は、ペアとなる公開鍵、自己署名付き証明書や CA が格納されてから有効になります。ペアとなる公開鍵は、/etc/inet/ike/publickeys ディレクトリか、サポートされるハードウェアに格納されます。

IKEv1 /etc/inet/ike/crls ディレクトリ

/etc/inet/ike/crls ディレクトリには、証明書失効リスト (CRL) ファイルが含まれています。各ファイルは、/etc/inet/ike/publickeys ディレクトリにある公開証明書ファイルに対応しています。CA は証明書の CRL を提供します。ikecert certrldb コマンドを使用して、そのデータベースを読み込むことができます。

ネットワークセキュリティ用語集

3DES	Triple-DES を参照してください。
インターネット トプロトコル (IP)	インターネットを介してデータがあるコンピュータから別のコンピュータに送信するための方法またはプロトコル。
解釈ドメイン (DOI)	データ形式や、ネットワークトラフィック交換タイプ、セキュリティ関連情報の命名規約を定義します。セキュリティ関連情報の例としては、セキュリティポリシーや、暗号化アルゴリズム、暗号化モードなどがあります。
鍵管理	セキュリティアソシエーション (SA) を管理する方法。
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)	パケットに対して完全性と機密性を提供する拡張ヘッダー。ESPは、IPセキュリティアーキテクチャー (IPsec) の5つのコンポーネントの1つです。
キーストア名	管理者がストレージ領域 (つまり、キーストア) に与える、 ネットワークインタフェースカード (NIC) 上の名前。キーストア名は、「トークン」、「トークン ID」とも呼ばれます。
公開鍵暗号化	2つの鍵を使用する暗号化システム。公開鍵はだれでも知ることができます。非公開鍵は、メッセージの受信者だけが知っています。IKEにより、IPsecの公開鍵が提供されます。
識別名 (DN)	通常の文字列を使用して共有情報を表すための標準化された方法。識別名は、LDAP証明書や X.509 証明書などのテクノロジーで使用されています。詳細は、 A String Representation of Distinguished Names (http://www.ietf.org/rfc/rfc1779.txt) を参照してください。

証明書失効リスト (CRL)	CA が無効とした公開鍵証明書のリスト。CRL は、IKE を使用して管理される CRL データベースに格納されます。
ステートフルパケットフィルタ	アクティブな接続の状態をモニターし、そこから得た情報を使って パケットフィルタ を通過させるネットワークパケットを決める ファイアウォール 。要求と応答を追跡、照合することによって、ステートフルパケットフィルタは、要求と一致しない応答を選別できます。
ストリーム制御転送プロトコル (SCTP)	TCP と似た方法で接続指向の通信を行う転送層プロトコル。さらに、このプロトコルは、接続のエンドポイントの 1 つが複数の IP アドレスをもつことができる複数ホーム機能をサポートします。
セキュリティーアソシエーション (SA)	1 つのホストから 2 つめのホストにセキュリティー属性を指定するアソシエーション。
セキュリティーパラメータインデックス (SPI)	受信したパケットを復号化するために使用する、SADB (セキュリティーアソシエーションデータベース) 内の行を特定する整数値。
セキュリティーポリシーデータベース (SPD)	パケットにどのレベルの保護を適用するかを指定するデータベース。SPD は、IP トラフィックをフィルタして、パケットを破棄すべきか、検証済みとして通過させるべきか、IPsec で保護すべきかを決めます。
双方向 SA	両方向のパケットを保護する ISAKMP SA。IPsec SA は単方向です。
対称鍵暗号化	メッセージの送信側と受信側が 1 つの共通鍵を共有する暗号化システム。この共通鍵は、メッセージを暗号化および復号化するために使用されます。対称鍵は、IPsec での大量データ転送の暗号化に使用します。 AES は対称鍵の 1 つの例です。
デジタル署名	送信側を一意に識別する、電子的に転送されたメッセージに添付されるデジタルコード。
盗聴	コンピュータネットワーク上で盗聴すること。普通のテキストによるパスワードなどの情報をネットワークから自動的に選別するプログラムの一部としてしばしば使用されます。
トラストアンカー	X.509 証明書では、認証局からのルート証明書。ルート証明書から目的の証明書までの証明書によって、トラストチェーンが確立されます。
トラストチェーン	X.509 証明書では、 トラストアンカー からユーザーの証明書までの証明書が、一続きの認証チェーンを提供しているという認証局からの保証。

トンネル	<p>パケットがカプセル化される間に通過するパス。カプセル化を参照してください。</p> <p>IPsec では、構成されたトンネルはポイントツーポイントインタフェースです。トンネルによって、IP パケットを別の IP パケット内にカプセル化できます。</p>
なりすまし	<p>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。</p>
認証局 (CA)	<p>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。</p>
認証ヘッダー	<p>IP パケットに対して認証と完全性を提供する拡張ヘッダーですが、機密性は提供されません。</p>
ネットワークアドレス変換 (NAT)	<p>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。</p>
ネットワークインタフェースカード (NIC)	<p>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。</p>
ネットワークポリシー	<p>ネットワークトラフィックを保護するためにネットワークユーティリティーで行われる設定。</p>
パケット	<p>IP パケットを参照してください。</p>
パケット	<p>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。</p>
パケットフィルタ	<p>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。</p>
パケットヘッダー	<p>IP ヘッダーを参照してください。</p>
ハッシュ値	<p>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。SHA-1 は、1 方向のハッシュ関数の 1 つの例です。</p>
非対称鍵暗号化	<p>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ンネルを</p>

作成します。[Diffie-Hellman アルゴリズム](#)は、非対称鍵プロトコルの例です。[対称鍵暗号化](#)と比較してください。

ファイアウォール	組織のプライベートネットワークやイントラネットをインターネットから切り離し、外部からの進入を防止するためのデバイスまたはソフトウェア。ファイアウォールには、フィルタリングや、プロキシサーバー、NAT (ネットワークアドレス変換)などを組み込むことができます。
物理インタフェース	リンクへのシステムの接続。この接続は通常、デバイスドライバとネットワークインタフェースカード (NIC) として実装されます。NIC によっては、igb のように複数の接続点を持つものもあります。
ブロードキャストアドレス	アドレスのホスト部分のビットがすべてゼロ (10.50.0.0) か 1 (10.50.255.255) である IPv4 ネットワークアドレス。ローカルネットワーク上のシステム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システムに配信されます。
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。
ポリシー	一般には、意思やアクションに影響を与えたり、これらを決定したりする計画や手続き。コンピュータシステムでは、多くの場合セキュリティポリシーを指します。たとえば、IPsec セキュリティポリシーには最小の長さが 512 バイトの対称鍵が必要である可能性があり、証明書ポリシーでは証明書の期限が 1 年以内に切れることが必要である可能性があります。
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 と同様の機能があります。
マルチホームホスト	複数の物理インタフェースをもち、パケット転送を行わないシステム。マルチホームホストではルーティングプロトコルを実行できます。
メッセージ認証コード (MAC)	データの整合性を保証し、データの出所を明らかにするコード。MAC は盗聴行為には対応できません。
ラベル	<ol style="list-style-type: none"> 1. IKEv2 ルールのキーワードで、その値は、<code>auth_method</code> が <code>preshared</code> の場合には共有鍵ファイルの <code>label</code> キーワードの値に一致する必要があります。 2. IKEv2 証明書の作成時に使用されるキーワード。この値は、キーストア内の証明書のあらゆる部分 (非公開鍵、公開鍵、および公開鍵証明書) を特定する際に便利です。 3. オブジェクトまたはプロセスの機密性レベルの強制アクセス制御 (MAC) の表示。Confidential および Top Secret はラベルの例です。ラベル付きネットワーク送信には MAC ラベルが含まれます。

4. IKEv1 ルールのキーワードで、その値はルールの取得に使用されます。

リプレー攻撃	IPsec では、パケットが侵入者によって捕捉されるような攻撃のこと。格納されたパケットは、あとで元のパケットを置き換えるか繰り返します。そのような攻撃を防止するために、パケットを保護している秘密鍵が存在している間、値が増加を続けるフィールドをパケットに含めることができます。
リンク層	IPv4/IPv6 のすぐ下の層。
リンクローカルアドレス	IPv6 では、自動アドレス構成などのために、単一リンク上でアドレスを指定するために使用することを表します。デフォルトでは、リンク - ローカル・アドレスはシステムの MAC アドレスから作成されます。
ルーター	複数のインタフェースを通常もち、ルーティングプロトコルを実行し、パケットを転送するシステム。システムが PPP リンクのエンドポイントである場合は、ルーターとしてのインタフェースを 1 つだけもつようなシステムを構成できます。
ルーター広告	自身の存在を、リンクやインターネットのさまざまなパラメータとともに通知するルーターのプロセス。
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。
AES	Advanced Encryption Standard。対称ブロックのデータ暗号技術。2000 年の 10 月、米国政府は暗号化標準としてこのアルゴリズムの Rijndael 方式を採用しました。AES は DES に代わる米国政府の標準として採用されています。
DES	データ暗号化規格。1975 年に開発され、1981 年に ANSI X.3.92 として ANSI で標準化された対称鍵の暗号化方式。DES では 56 ビットの鍵を使用します。
Diffie-Hellman アルゴリズム	「公開鍵」暗号化としても知られています。1976 年に Diffie 氏と Hellman 氏が開発した非対称暗号鍵協定プロトコルです。このプロトコルを使用すると、セキュアでない伝達手段で、事前の秘密情報がなくても 2 人のユーザーが秘密鍵を交換できます。Diffie-Hellman は、IKE プロトコルで使用されます。
DoS 攻撃	サービス拒否 (Denial of Service) 攻撃。システムまたはネットワークをパケットで溢れさせ、それによって適正なパケットの配信を妨害したり、遅くさせたりする攻撃。DDos は、複数の場所から仕掛けられる攻撃である、分散型サービス拒否 (Distributed Denial of Service) 攻撃です。
DSA	デジタル署名アルゴリズム。512 ビットから 4096 ビットまでの可変長鍵の公開鍵アルゴリズム。米国政府標準である DSS は最大 1024 ビットです。この場合、DSA では入力に SHA-1 を使用します。
ECDSA	Elliptic Curve Digital Signature Algorithm。楕円曲線数学に基づく公開鍵アルゴリズム。ECDSA 鍵サイズは、同じ長さの署名の生成に必要な DSA 公開鍵のサイズより大幅に小さくなります。

FIPS 140-2	機密情報ではあるが、機密扱いを受けていない情報を処理する多くの規制産業および米国政府機関にとっての要件である、米国連邦情報処理標準。FIPS 140-2 の目的は、そのシステムが暗号化を正しく実装しているという一定程度の保証を提供することにあります。
HMAC	メッセージ認証を行うための鍵付きハッシュ方法。HMAC は秘密鍵認証アルゴリズムの 1 つです。HMAC は、 SHA-1 などの繰り返し暗号化のハッシュ関数で、秘密共有鍵と組み合わせて使用されます。HMAC の暗号の強さは、基になるハッシュ関数のプロパティによって異なります。
ICMP エコー要求パケット	応答を促すためにインターネット上のシステムに送信されるメッセージ。そのようなパケットは一般に “ping” パケットといわれています。
IKE	インターネット鍵交換。IPsec セキュリティーアソシエーション (SA) 用の認証された鍵情報の供給を自動化します。
IP	インターネットプロトコル (IP) 、 IPv4 、 IPv6 を参照してください。
IP スタック	TCP/IP はしばしば「スタック」と呼ばれます。データ交換のクライアントエンドとサーバーエンドですべてのデータが通過する層 (TCP、IP、場合によってはそのほかを含む) のことを意味します。
IP 内 IP カプセル化	IP パケット内で IP パケットをトンネリングするためのメカニズム。
IP パケット	IP 経由で転送される情報パケット。IP パケットはヘッダーとデータを含みます。ヘッダーにはパケットのソースと宛先のアドレスが含まれます。ヘッダーのその他のフィールドは、データと付随するパケットを宛先で識別し、再結合する際に役立ちます。
IP ヘッダー	インターネットパケットを固有に識別する 20 バイトのデータ。ヘッダーには、パケットの送信元と送信先のアドレスが含まれています。さらに、ヘッダー内のオプションによって、新しいバイトを追加できます。
IP リンク	リンク層でノード間の通信に使用される通信設備や通信メディア。リンク層とは IPv4 および IPv6 のすぐ下の層です。例としては、Ethernet (ブリッジされたものも含む) や ATM ネットワークなどがあります。1 つまたは複数の IPv4 サブネット番号またはネットワーク接頭辞が IP リンクに割り当てられます。同じサブネット番号またはネットワーク接頭辞を複数の IP リンクに割り当てることはできません。ATM LANE では、IP リンクは 1 つのエミュレートされた LAN です。ARP を使用する場合、ARP プロトコルの有効範囲は単一の IP リンクです。
IPsec	IP セキュリティー。IP パケットを保護するためのセキュリティーアーキテクチャー。
IPv4	インターネットプロトコルのバージョン 4。単に IP と呼ばれることもあります。このバージョンは 32 ビットのアドレス空間をサポートしています。

IPv6	インターネットプロトコルのバージョン 6。128 ビットのアドレス空間をサポートしています。
NAT	ネットワークアドレス変換 (NAT) を参照してください。
Perfect Forward Secrecy (PFS)	<p>PFS では、データ伝送を保護するために使用される鍵が、追加の鍵を導き出すために使用されることはありません。さらに、データ伝送を保護するために使用される鍵のソースが、追加の鍵を導き出すために使用されることはありません。したがって、PFS は以前に記録されたトラフィックの復号化を回避できます。</p> <p>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。</p>
PKI	公開鍵インフラストラクチャー。インターネットトランザクションに関係する各関係者の有効性を確認および承認する、デジタル署名、認証局、ほかの登録機関のシステム。
RSA	デジタル署名と公開鍵暗号化システムを取得するための方法。その開発者である Rivest 氏、Shamir 氏、Adleman 氏によって 1978 年に最初に公開されました。
SADB	セキュリティーアソシエーションデータベース。暗号化鍵と暗号化アルゴリズムを指定するテーブル。鍵とアルゴリズムは、安全なデータ転送で使用されます。
SHA-1	セキュアなハッシュアルゴリズム。メッセージ要約を作成するために 2^{64} 文字以下の長さを入力するときに操作します。SHA-1 アルゴリズムは DSA に入力されます。
smurf 攻撃	リモートロケーションから IP ブロードキャストアドレス または複数のブロードキャストアドレスに向けられた ICMP echo request パケットを使用して、深刻なネットワークの輻輳や中断を引き起こすこと。
Triple-DES	Triple-Data Encryption Standard。対称鍵暗号化システムの 1 つ。Triple-DES では鍵の長さとして 168 ビットが必要です。Triple-DES を「3DES」と表記することもあります。

索引

あ

- アクション
 - パケットフィルタ (PF) 規則でのオプション, 57
 - パケットフィルタ (PF) での規則セット, 55
- アクティブな規則セット 参照 IP フィルタ
- アドレスプール
 - IP フィルタでの, 85
 - IP フィルタでの構成, 85
 - IP フィルタでの構成ファイル, 85
 - 削除, 105
 - 追加, 106
 - 統計の表示, 110
 - 表示, 105
- 暗号 参照 暗号化アルゴリズム
- 暗号化アルゴリズム
 - SSL カーネルプロキシ, 36
- 暗号化フレームワーク
 - IPsec, 254
- 移行
 - IKEv1 から IKEv2 へ, 168
 - IP フィルタ からパケットフィルタ (PF), 46
- 一覧表示
 - 証明書, 187, 210
- 一致パラメータ
 - パケットフィルタ (PF) での規則セット, 56, 64
- 移動体システム
 - 用の IKEv1 の構成, 223
- 移動体システム用の IKEv1 の構成 (タスクマップ), 223
- インストール
 - firewall パッケージ, 70
 - パケットフィルタパッケージ, 70
- エクスポート
 - IKEv2 の証明書, 186
- オプション
 - パケットフィルタ (PF) でのアクション, 57

か

- カーネル
 - SSL パケットの高速化, 35
 - Web サーバー用の SSL カーネルプロキシ, 35
- 鍵
 - ike.privatekeys データベース, 268
 - ike/publickeys データベース, 267
 - IPsec SA 向けの作成, 154
 - IPsec での手動管理, 124, 154
 - IPsec の管理, 256
 - 格納 (IKEv1)
 - 公開鍵, 267
 - 証明書, 267
 - 非公開, 266
 - 事前共有 (IKE), 165
 - 事前共有 (IKEv1), 173
 - 自動管理, 169, 172
- 鍵管理
 - ike:default サービス, 256
 - IKEv1, 172
 - IKEv2, 169
 - ikev2 サービス, 258
 - IPsec, 256
 - ipseckey コマンド, 254
 - manual-key サービス, 257
 - 自動, 169, 169, 172
 - 手動, 124
 - ゾーンと, 137
- 鍵の格納
 - IKEv1
 - ISAKMP SA, 265
 - ソフトトークンキーストア, 230, 265

- メタスロットのトークン ID, 230
- IKEv2
 - ソフトトークンキーストア, 258, 260
- IPsec SA, 136
- SSL カーネルプロキシ, 38
- 確認
 - hostmodel 値, 30
 - ipseccinit.conf 構文, 141, 152, 152
 - ipseckeykeys 構文, 156
 - パケットの保護, 160
 - 無効にされているルーティングデーモン, 28
 - リンク保護, 23
- 格納
 - IKEv1 鍵をディスクに, 267
 - 鍵をハードウェアで, 230
 - 証明書をディスクに, 186
 - ディスク上の IKEv1 鍵, 267
 - ディスク上の鍵, 213
 - ハードウェア上の証明書, 196
- 仮想プライベートネットワーク (VPN)
 - IPsec で構築, 132
 - IPsec による保護, 149
 - IPv4 の例, 149
 - routeadm コマンドでの構成, 150
 - routeadm コマンドによる構成, 150
 - トンネルモードと, 146
- 仮想マシン
 - IPsec と, 135
- カプセル化セキュリティーペイロード (ESP)
 - AH との比較, 125
 - IPsec 保護プロトコル, 125
 - IP パケットの保護, 119
 - セキュリティーについて, 127
 - 説明, 126
- キーストア
 - IKEv2 証明書の格納, 185
 - IKEv2 の作成, 182
 - IKEv2 の初期化, 181
 - IKE での使用, 167
- キーストア名 参照 トークン ID
- 規則セット, 75
 - 参照 IP フィルタ
 - パケットフィルタでの表示, 71
 - パケットフィルタリング, 80
- 規則の構文 参照 パケットフィルタ (PF) で設定される規則
- 計算
 - ハードウェアによる IKEv1 の高速化, 230
- 現在の規則セットの更新後のリロード
 - パケットフィルタリング, 97
- 検証
 - ikev2.config 構文, 177
 - IKE 証明書, 166
 - IKE 証明書をそのフィンガープリントで, 190
 - 自己署名付き証明書の妥当性, 187
 - 証明書の妥当性 (IKEv2), 194
- 権利プロファイル
 - Network IPsec Management, 157
 - Network Management, 157
 - Network Security, 39
- 公開鍵
 - 格納 (IKEv1), 267
- 公開鍵証明書 参照 証明書
- 公開鍵証明書による IKEv1 の構成 (タスクマップ), 207
- 公開鍵証明書による IKEv2 の構成 (タスクマップ), 184
- 更新
 - パケットフィルタでの規則, 70
- 構成
 - Apache 2.2 Web サーバーと SSL カーネルプロキシ, 37
 - Apache 2.2 Web サーバーと SSL 保護, 43
 - Apache 2.2 Web サーバーとフォールバック SSL, 41
 - IKEv1
 - CA 証明書, 212
 - 移動体システム, 223
 - 公開鍵証明書, 207
 - 自己署名付き証明書, 207
 - ハードウェア上の証明書, 217
 - IKEv2
 - CA 証明書, 190
 - 公開鍵証明書, 184
 - 公開証明書用キーストア, 181
 - 自己署名付き証明書, 185
 - 事前共有鍵, 176
 - 証明書検証ポリシー, 193
 - ハードウェア上の証明書, 196

- IPsec, 137
- ipsecinit.conf ファイル, 253
- IPsec で保護された VPN, 149
- IP フィルタでの NAT 規則, 84
- IP フィルタでのアドレスプール, 85
- Oracle iPlanet Web Server と SSL カーネルプロキシ, 39
- Web サーバーと SSL カーネルプロキシ, 35
- ネットワークセキュリティ、役割, 157
- パケットフィルタ, 69
- パケットフィルタ (PF), 67
- パケットフィルタ (PF) での規則, 54
- パケットフィルタリング規則, 81
- ファイアウォール, 67
- リンクの保護, 27
- リンク保護, 21
- 構成ファイル
 - /etc/firewall/pf.conf, 54
 - /etc/inet/secret/ike.preshared, 173, 203, 206
 - /etc/inet/secret/ipseckeys, 125, 155, 252
 - ike.preshared, 248
 - ike/config ファイル, 262, 264
 - ike/ikev2.config ファイル, 257, 259
 - ike/ikev2.preshared ファイル, 258
- IP フィルタ, 80
- IP フィルタサンプル, 114
- パケットフィルタのサンプル, 72
- 高速化
 - IKEv1 計算, 229
 - IP フィルタでのルール処理, 80
 - Web サーバー通信, 35
- コマンド
 - IKEv1
 - ikeadm コマンド, 263, 264
 - ikecert コマンド, 262, 263, 265
 - in.iked デーモン, 263
 - 説明, 265
 - IKEv2
 - ikeadm コマンド, 257, 259, 260, 262
 - ikev2cert コマンド, 257, 259, 260
 - in.ikev2d デーモン, 259
 - 説明, 260
 - IPsec
 - in.iked コマンド, 256
 - ipsecalgs コマンド, 254
 - ipseconf コマンド, 136, 252
 - ipseckey コマンド, 124, 136, 254
 - kstat コマンド, 255
 - snoop コマンド, 255
 - セキュリティについて, 255
 - リスト, 135
 - パケットフィルタ (PF)
 - pfconf, 70
 - pfctl, 67, 71
- さ
 - サービス 参照 Service Management Facility (SMF)
 - サービス管理機能 (SMF)
 - Apache Web サーバーサービス, 39
 - IKEv1 サービス
 - ike サービス, 261
 - 構成可能なプロパティ、262
 - 説明, 262
 - 有効化, 226, 263
 - IKEv2 サービス
 - ike:ikev2 サービス, 257
 - 構成可能なプロパティ、258
 - 説明, 258
 - 有効化, 141, 259
 - リフレッシュ, 141
 - IKE サービス, 256
 - IPsec サービス, 251
 - ipsecalgs サービス, 254
 - manual-key サービス, 136
 - manual-key の使用, 156, 156
 - manual-key の説明, 257
 - policy サービス, 136, 141
 - リスト, 135
 - IP フィルタサービス
 - 構成, 91
 - チェック, 90
 - SSL カーネルプロキシ サービス, 38
 - system-log サービス, 111
 - パケットフィルタ (PF) サービス
 - firewall, 54
 - ファイアウォール, 48
 - パケットフィルタサービス
 - firewall, 54

- チェック, 71
- 作成, 119
 - 参照 追加
 - IKEv2 キーストア, 182
 - IPsec SA, 140, 154
 - ipsecinit.conf ファイル, 140
 - IP フィルタの構成ファイル, 91
 - 自己署名付き証明書 (IKEv1), 208
 - 自己署名付き証明書 (IKEv2), 185
 - 証明書署名要求 (CSR), 191, 212
 - セキュリティ関連の役割, 157
- 識別名 (DN)
 - 使用, 267
 - 定義, 206
 - 例, 167, 208
- 自己署名付き証明書
 - IKEv1 での構成, 207
 - IKEv2 での構成, 185
 - IKE の概要, 166
- システム
 - Web サーバーの保護, 35
 - 通信の保護, 139, 139
 - ネットワークのチューニング可能な設定, 27
 - ファイアウォールの使用, 67, 89
 - リンクレベルの保護, 19
- 事前共有鍵の置き換え, 178, 204
- 事前共有鍵 (IKE), 165
- 事前共有鍵 (IKEv1)
 - 置き換え, 204
 - 格納, 265
 - サンプル, 206
 - 使用, 203
 - 説明, 173
 - 定義, 173
- 事前共有鍵 (IKEv2)
 - 置き換え, 178
 - 格納, 260
 - 構成, 176
 - ルールとのマッチング, 238
- 失効した証明書 参照 CRL、OCSP
- 手動鍵管理
 - IPsec, 125, 155, 252
 - 作成, 154
- 状態テーブル
 - IP フィルタでの表示, 108
- 状態統計
 - IP フィルタでの表示, 108
- 証明書
 - IKEv1
 - CA から, 213
 - CA から要求, 212
 - CRL を無視, 215
 - ike/config ファイル内, 219
 - 一覧表示, 210
 - 確認, 210
 - 格納, 267
 - 検証, 210
 - コンピュータに格納, 206
 - 自己署名付きの作成, 208
 - 失効済み, 221
 - データベースへの追加, 213
 - ハードウェア上で要求, 218
 - ハードウェアで CA, 220
 - ハードウェアに格納, 229
 - IKEv2
 - CA から, 192
 - CA に要求, 191
 - ikev2.config ファイル内, 198
 - 一覧表示, 187
 - インポート, 192
 - エクスポート, 186
 - 格納, 184
 - キーストアに追加, 192
 - 検証, 187, 187
 - 構成, 193
 - 自己署名付きの作成, 185
 - 失効済み, 194
 - 証明書の検証ポリシー, 193
 - ハードウェア上で要求, 197
 - ハードウェア上に格納, 196
 - ポリシー, 170
 - IKE での検証, 235
 - IKE での使用, 167
 - IKE でのトラブルシューティング, 235
 - IKE での取り消し, 167
 - IKE の概要, 166
 - SSL に使用, 37
 - 失効済みかどうかの判定 (IKEv2), 194
 - 失効済みの動的な取得, 195
 - 静的 CRL, 195

- 説明, 191
 - 証明書検証ポリシー
 - IKEv2 での構成, 193
 - 証明書失効リスト 参照 CRL
 - 証明書署名要求 参照 CSR
 - 証明書のデジタル署名, 266
 - スロット
 - ハードウェア内, 267
 - セキュリティー
 - IKEv1, 263
 - IKEv2, 259
 - IPsec, 119
 - セキュリティーアソシエーションデータベース (SADB), 120, 256
 - セキュリティーアソシエーション (SA)
 - IKEv1, 263
 - IKEv2, 259
 - IPsec, 124, 140, 151
 - IPsec データベース, 256
 - IPsec の追加, 140, 151
 - ISAKMP, 172
 - 手動作成, 154
 - 定義, 120
 - 乱数発生, 169, 173
 - セキュリティー上の考慮事項
 - 事前共有鍵, 166
 - セキュリティーについて
 - AH と ESP の比較, 125
 - ike/config ファイル, 264
 - ike/ikev2.config ファイル, 259
 - ipseccomf コマンド, 254
 - ipseccinit.conf ファイル, 254
 - ipseckey ファイル, 156
 - ipseckey コマンド, 255
 - カプセル化セキュリティーペイロード (ESP), 127
 - セキュリティープロトコル, 127
 - 認証ヘッダー (AH), 127
 - ラッチされたソケット, 254
 - セキュリティーパラメータインデックス (SPI), 124
 - セキュリティープロトコル
 - IPsec 保護プロトコル, 125
 - Secure Sockets Layer (SSL), 35
 - 概要, 120
 - カプセル化セキュリティーペイロード (ESP), 126
 - セキュリティーについて, 127
 - 認証ヘッダー (AH), 126
 - セキュリティーポリシー
 - ike/config ファイル, 136
 - ike/ikev2.config ファイル, 136
 - IPsec, 128
 - ipseccinit.conf ファイル, 253
 - kmp-policy.xml ファイル, 245
 - pf.conf ファイル, 67
 - セキュリティーポリシーデータベース (SPD), 120, 252
 - ゾーン
 - IPsec と, 135, 137
 - IPsec の静的 IP アドレス, 135
 - SSL 保護による Apache Web サーバーの構成, 43
 - 鍵管理と, 137
 - パケットフィルタ (PF) および, 45
 - ソケット
 - IPsec のセキュリティー, 254
 - ソフトトークンキーストア
 - IKEv2 鍵の格納, 260
 - メタスロットでの鍵の格納, 230, 265
- た**
- タスクマップ
 - IPsec によるネットワークトラフィックの保護 (タスクマップ), 138
 - 移動体システム用の IKEv1 の構成 (タスクマップ), 223
 - 公開鍵証明書による IKEv1 の構成 (タスクマップ), 207
 - 公開鍵証明書による IKEv2 の構成 (タスクマップ), 184
 - チェック
 - パケットフィルタ (PF) での規則の構文, 68
 - パケットフィルタでの規則の構文, 72
 - チューニング可能パラメータ
 - IP フィルタ, 109
 - 追加
 - CA 証明書 (IKEv1), 212
 - CA 証明書 (IKEv2), 190

- IPsec SA, 140, 154
- 鍵を手動で (IPsec), 154
- 公開鍵証明書 (IKEv1), 212
- 公開鍵証明書 (IKEv2), 190
- 公開鍵証明書 (SSL), 41
- 自己署名付き証明書 (IKEv1), 208
- 自己署名付き証明書 (IKEv2), 185
- 事前共有鍵 (IKEv1), 205
- 事前共有鍵 (IKEv2), 179
- ネットワーク管理の役割, 158
- パケットフィルタファイアウォール, 70
- ファイアウォール, 45
- ディレクトリ
 - /etc/apache2/2.2, 42
 - /etc/firewall, 54
 - /etc/inet/ike, 257, 258, 262
 - /etc/inet/publickeys, 267
 - /etc/inet/secret/ike.privatekeys, 266
 - /etc/inet/secret, 262
 - /etc/inet, 262
 - /var/user/ikeuser, 181
- 公開鍵 (IKEv1), 267
- 事前共有鍵, 260, 265
- 証明書 (IKEv1), 267
- 非公開鍵 (IKEv1), 266
- ディレクトリ名 (DN)
 - CRL にアクセスするための, 221
- データベース
 - ike.privatekeys データベース, 266, 268
 - ike/crls データベース, 267, 268
 - ike/publickeys データベース, 267, 267
 - IKEv1, 265
 - kmfcfg コマンドの dbfile 引数, 170
 - セキュリティアソシエーションデータベース (SADB), 256
 - セキュリティポリシーデータベース (SPD), 120
- デーモン
 - in.iked デーモン, 169, 172, 261, 263
 - in.ikev2d デーモン, 177, 182, 257, 259
 - in.routed デーモン, 28
 - webservd デーモン, 41
- デバッグ 参照 [トラブルシューティング](#)
- デフォルトの表示
 - IP フィルタ, 90
 - デフォルト CA ポリシー
 - kmf-policy.xml ファイル, 193
- トークン 引数
 - ikecert コマンド, 266
- トークン ID
 - ハードウェア内, 267
- トラブルシューティング
 - IKEv1 ペイロード, 217
 - IPsec および IKE システムの実行, 236
 - IPsec および IKE で必要な権利, 233
 - IPsec および IKE の意味上の誤り, 240
 - IPsec および IKE の準備, 234
 - IPsec およびその鍵管理, 233
 - IP フィルタ規則セット, 99, 101
 - 現在の CRL を維持, 246
 - システム実行前の IPsec および IKE, 235
 - パケットフィルタ (PF) 規則, 68
- トランスポートモード
 - ESP で保護されるデータ, 131
 - IPsec, 129
- トンネル
 - IPsec, 132
 - IPsec での tunnel キーワード, 147
 - IPsec の tunnel キーワード, 130, 151
 - IPsec のトンネルモード, 129
 - IPsec のモード, 129
 - 内側の IP パケット全体の保護, 131
 - トランスポートモード, 130
 - パケットの保護, 132
 - を使用した VPN の保護, 149
- な
 - なりすまし
 - リンクの保護, 20
 - 認証アルゴリズム
 - IKEv1 証明書, 266
 - IKEv2 証明書, 188
 - 認証局 (CA), 119
 - 参照 証明書、CSR
 - IKE 証明書, 166
 - 認証ヘッダー (AH)
 - ESP との比較, 125, 125
 - IPsec 保護プロトコル, 125
 - IP パケットの保護, 119, 126

- セキュリティーについて, 127
- ネットワーク/ファイアウォール サービス, 48
- ネットワークアドレス変換 (NAT) 参照 NAT
- ネットワーク全体の管理の役割, 158
- ネットワークプロトコル
 - DefaultFixed
 - IPsec, 137, 175, 201
 - 自動, 137, 175, 201
- は
- ハードウェア
 - IKEv1 鍵の格納, 230
 - IKEv1 計算の高速化, 229
 - IKEv2 鍵の格納, 196
 - 公開鍵証明書, 217
 - 接続されたものを検出, 229
 - 接続を探す, 196
- バイパス
 - IPsec ポリシー, 129
 - LAN 上の IPsec, 151
- パケット
 - IP, 119
 - IP フィルタでの再構築の無効化, 93
 - アウトバウンドプロセスのフローチャート, 123
 - インバウンドプロセスのフローチャート, 122
 - パケットフィルタ (PF) での処理順序, 48
 - パケットフィルタ (PF) でのフィルタリング, 45
 - パケットフィルタ (PF) でのフロー, 51
 - 保護
 - IKEv1 による, 172
 - IPsec による, 121, 125
 - アウトバウンドパケット, 121
 - インバウンドパケット, 121
 - 保護の確認, 160
 - パケットフィルタリング 参照 パケットフィルタ (PF)
 - IP フィルタ, 75
 - 規則セット間の切り替え, 101
 - 現在の規則セットの更新後のリロード, 97
 - 構成, 81
 - 削除
 - アクティブなルールセット, 99
 - 非アクティブな規則セット, 102
 - 追加
 - アクティブなセットへの規則, 99
 - パケットフィルタでの規則の更新, 70
 - 別の規則セットのアクティブ化, 97
 - ルールセットの管理, 96
 - パケットフィルタ (PF)
 - IP フィルタ 構成からの構成例, 60, 72
 - IP フィルタ との比較, 46, 67
 - match および pass を使用した規則の同等, 64
 - NAT および, 48
 - NAT 規則の例, 59, 63
 - OpenBSD PF との比較, 47
 - Oracle Solaris でのバージョン, 45
 - Oracle Solaris にはない OpenBSD 機能, 45
 - spam テーブルの例, 60
 - 一致パラメータ, 56
 - インストール, 70
 - 概要, 45
 - 規則セット
 - パケットフィルタでの更新, 70
 - 規則セットの表示, 71
 - 規則のアクション, 55
 - 規則のオプション, 57
 - 規則の構文, 54
 - 規則の構文の支援, 59
 - 規則の処理, 65
 - 構成, 67
 - 構成タスク, 69
 - 構成の準備, 68
 - 構成ファイル, 54
 - サンプルの構成ファイル, 72
 - 状態の一致規則の構文, 61
 - 使用のガイドライン, 48
 - ゾーンおよび, 45
 - タスクのモニタリング, 71
 - デフォルトの構成ファイル, 68
 - パケット処理の順序, 48
 - パケットのフロー, 51
 - ファイアウォール サービス, 48
 - マニュアルページのサマリー, 66
 - 無効化, 71
 - 有効化, 70
 - リダイレクトの例, 63, 64
 - リファレンス, 66
 - ループバックフィルタリングの選択, 60
 - パケットフィルタ (PF) で設定される規則

- PF および IP フィルタ の比較, 46
- パケットフィルタ (PF) での規則セット
 - IP フィルタ からパケットフィルタ (PF) への変換, 67
 - IP フィルタ との違い, 60, 72
 - match および pass を使用した同等, 64
 - NAT の例, 59
 - spam テーブルの例, 60
 - アクション, 55
 - アクションに対するオプション, 57
 - 一致パラメータ, 56
 - 構文, 54
 - 処理, 65
 - テスト, 67
 - 読みやすさ, 59
 - パケットフィルタ (PF) でのテーブル
 - spam の例, 60
 - 概要, 59
 - パケットフィルタ (PF) でのマクロ, 59
 - パッケージ
 - firewall, 70
- ピア
 - IKEv2 構成の作成, 176
 - IKEv2 構成への追加, 179
- 非アクティブな規則セット 参照 IP フィルタ
- 非アクティブなセットへの規則
 - IP フィルタでの追加, 100
- 比較
 - IPF および PF ファイアウォールでのループバック規則セット, 60
 - IP フィルタ およびパケットフィルタ (PF), 46, 67
 - IP フィルタ およびパケットフィルタ (PF) の規則セット, 60, 72
 - Oracle Solaris および OpenBSD PF, 47
 - PF および IP フィルタ での状態の一致規則, 61
- 非公開鍵
 - 格納 (IKEv1), 266
- 表示
 - IKE SA, 244
 - IKE 事前共有鍵, 243
 - IKE 情報, 242
 - IKE デーモンの状態, 243
 - IKE プロパティ値, 242
 - IPsec 構成, 253
 - IPsec 情報, 241
 - IPsec 情報の手動鍵, 241
 - IP フィルタでの NAT 統計, 109
 - IP フィルタでのアドレスプール, 105
 - IP フィルタでのアドレスプール統計, 110
 - IP フィルタでの状態テーブル, 108
 - IP フィルタでの状態統計, 108
 - IP フィルタでのチューニング可能パラメータ, 109
 - IP フィルタログファイル, 112
 - アクティブな IKE ルール, 244
 - 証明書検証ポリシー, 245
 - パケットフィルタ (PF) での DNS 逆引き検索, 68
 - パケットフィルタ (PF) でのカーネル情報, 68
 - パケットフィルタ (PF) での規則セット, 67
 - パケットフィルタ (PF) での詳細出力, 68
 - ファイアウォール 参照 パケットフィルタ (PF)
 - ファイアウォール サービス, 48
 - ファイル
 - httpd.conf, 42
 - IKEv1
 - cr1s ディレクトリ, 262, 268
 - ike.preshared ファイル, 262, 265
 - ike.privatekeys ディレクトリ, 262, 268
 - ike/config ファイル, 136, 173, 262, 264
 - publickeys ディレクトリ, 262, 267
 - IKEv2
 - ike/ikev2.config ファイル, 136, 170, 257, 259
 - ike/ikev2.preshared ファイル, 258, 260
 - IPsec
 - ipsecinit.conf ファイル, 136, 136, 253
 - ipseckey ファイル, 136
 - IP フィルタ 構成からの PF 構成, 60, 72
 - kmf-policy.xml, 170, 193
 - rsyslog.conf, 111
 - ssl.conf, 41
 - syslog.conf, 111
 - パケットフィルタ (PF)
 - pf.conf ファイル, 54
 - パケットフィルタ (PF) のデフォルト構成, 68
 - フラッシュ 参照 削除
 - 別の規則セットのアクティブ化
 - パケットフィルタリング, 97

変換

- IP フィルタ から PF への構成, 60, 72
- IP フィルタ からパケットフィルタ (PF) の規則セット, 67
- IP フィルタ からパケットフィルタ (PF) への規則セット, 60, 72

変更

- 実行中の IKE デーモン, 248

保護

- 2つのシステム間のパケット, 139
- IPsec で Web サーバーを, 144
- IPsec でネットワークトラフィックを, 137
- IPsec トラフィック, 119
- IPsec による移動体システム, 223
- IPsec のトンネルモードで VPN を, 149

保護プロトコル

- IPsec, 125

ホスト構成

- IP フィルタ ポリシーからの, 60, 72
- パケットフィルタ (PF) および, 72

ポリシー

- IPsec, 128
- 証明書検証, 170, 193, 245

ポリシーファイル

- ike/config ファイル, 136
- ike/ikev2.config ファイル, 136
- ipsecinit.conf ファイル, 253
- kmf-policy.xml, 170
- pf.conf, 54
- セキュリティーについて, 254

ま

無効化

- firewall サービス, 71
- パケットフィルタ (PF), 71

メタスロット

- 鍵の格納, 230

モニタリング

- パケットフィルタ, 71

や

役割

- ネットワーク管理の役割, 158

- ネットワークセキュリティーの役割の作成, 157

有効化

- firewall サービス, 70
- パケットフィルタ (PF), 70

ユーザー

- IPsec の管理と構成, 158

ら

リスト

- CRL, 195
- CRL (IKEv1), 221
- IKE デーモンの情報, 243
- アルゴリズム (IPsec), 127
- 証明書, 195, 221
- ハードウェア (IKEv1), 230
- ハードウェアトークン, 197, 197, 230, 230

リフレッシュ

- ikev2 サービス, 183
- policy サービス, 152
- system-log サービス, 111
- 事前共有鍵, 178, 204

リモート事前共有鍵, 238

リンクの保護

- 構成, 27
- リンク保護, 19
 - dladm コマンド, 21
 - 概要, 20
 - 確認, 23
 - 構成, 21

リンク保護のタイプ

- 説明, 20
- なりすましに対する, 20

ループバックフィルタリング

- IP フィルタでの有効化, 94
- パケットフィルタ (PF) および, 60

ルールセット

- IP フィルタ, 95
- IP フィルタでの NAT, 84

ローカル事前共有鍵, 238

ローカルファイルネームサービス

- /etc/inet/hosts ファイル, 140

ロギングされたパケット

- ファイルへの保存, 114

ログバッファ

- IP フィルタでのフラッシュ, 113
- ログファイル
 - IP フィルタ, 110
 - IP フィルタ用に作成, 111
 - IP フィルタ用の表示, 112
- 論理ドメイン 参照 仮想マシン

A

- AH 参照 認証ヘッダー (AH)
- Apache Web サーバー
 - SSL カーネルプロキシ と, 37
 - SSL カーネルプロキシ とフォールバック, 41
 - SSL カーネルプロキシ による構成, 37
 - SSL パケットの高速化, 35
 - ゾーン内で SSL 保護によって構成する, 43
 - フォールバック SSL 保護, 41

B

- BPDU 保護
 - リンク保護, 20
- bypass オプション
 - IPsec 構成, 129

C

- cert_root キーワード
 - IKEv1 構成ファイル, 214, 219
- cert_trust キーワード
 - ikecert コマンドと, 266
 - IKEv1 構成ファイル, 211, 219
- config_file プロパティ
 - IKEv2, 258
- CRL (証明書失効リスト)
 - ike/crls データベース, 268
 - ikecert certrldb コマンド, 267
 - IKEv2 での構成, 193
 - 説明, 167
 - 中央位置からアクセス, 221
 - 無視, 215
 - リスト, 195, 221
- CRL への HTTP アクセス
 - use_http キーワード, 222
- CSR (証明書署名要求)

- IKEv1
 - CA から, 212
 - 使用, 267
 - 送信, 213
 - ハードウェア上, 218
- IKEv2
 - CA から, 191
 - ハードウェア上, 197
 - SSL に使用, 41

D

- debug_level プロパティ
 - IKEv2, 234, 258
- debug_logfile プロパティ
 - IKEv2, 258
- DefaultFixed ネットワークプロトコル
 - IPsec, 137, 175, 201
- dhcp-nospoof
 - リンク保護のタイプ, 21
- DHCP 保護
 - リンク保護, 20
- dladm コマンド
 - IPsec トンネル保護, 149
 - リンク保護, 21
- DSS 認証アルゴリズム, 266

E

- /etc/firewall/pf.conf 参照 pf.conf ファイル
- /etc/inet/hosts ファイル, 140
- /etc/inet/ike/config ファイル
 - cert_root キーワード, 214, 219
 - cert_trust キーワード, 211, 219
 - ignore_crls キーワード, 215
 - ikecert コマンドと, 266
 - ldap-list キーワード, 222
 - PKCS #11 ライブラリエントリ, 265
 - pkcs11_path キーワード, 217, 265
 - proxy キーワード, 222
 - use_http キーワード, 222
- 公開鍵証明書, 214, 219
- サマリー, 262
- サンプル, 202
- 自己署名付き証明書, 211

- 事前共有鍵, 202
 - セキュリティについて, 264
 - 説明, 173, 264
 - ハードウェアに証明書を格納, 219
 - /etc/inet/ike/cr1s ディレクトリ, 268
 - /etc/inet/ike/ikev2.config ファイル
 - サマリー, 257
 - 自己署名付き証明書, 185
 - 事前共有鍵, 176
 - セキュリティについて, 260
 - 説明, 170, 259
 - ハードウェアに証明書を格納, 198
 - /etc/inet/ike/ikev2.preshared ファイル
 - サマリー, 258
 - サンプル, 180
 - 使用, 177, 178
 - 説明, 260
 - トラブルシューティング, 239
 - /etc/inet/ike/kmf-policy.xml ファイル
 - 使用, 193, 245
 - 定義, 170
 - デフォルト CA ポリシー, 193
 - /etc/inet/ike/publickeys ディレクトリ, 267
 - /etc/inet/ipsecinit.conf ファイル, 253
 - IKE バージョンの指定, 143
 - LAN のバイパス, 151
 - or pass オプションの指定, 143
 - Web サーバーの保護, 144
 - 構文の確認, 141, 152
 - サンプル, 253
 - セキュリティについて, 254
 - 説明, 136
 - トンネル構文, 146
 - 場所と範囲, 135
 - 目的, 129
 - /etc/inet/secret/ike.preshared ファイル
 - サンプル, 206
 - 使用, 203, 248
 - 定義, 173
 - /etc/inet/secret/ike.privatekeys ディレクトリ, 268
 - /etc/inet/secret/ipseckeys ファイル
 - IPsec 鍵の格納, 136
 - 構文の確認, 156
 - 使用, 155, 246
 - 定義, 125
 - デフォルトパス, 252
 - /etc/inet/secret/ ファイル, 265
 - ESP 参照 カプセル化セキュリティペイロード (ESP)
 - export サブコマンド
 - ikev2cert コマンド, 186
- ## F
- FIPS 140-2
 - IKE, 20, 164, 169
 - IKEv2 構成および, 171
 - IPsec および, 137
 - IPsec 構成と, 133
 - Sun Crypto Accelerator 6000 ボード, 262
 - Web サーバーの 2048 ビットの鍵と, 41
 - firewall:default サービスのデフォルト, 71
 - firewall/rules プロパティ
 - パケットフィルタ, 71
 - firewall サービス, 54, 54
 - firewall パッケージ, 70
- ## G
- gencert サブコマンド
 - ikev2cert コマンド, 197
 - gencsr サブコマンド
 - ikev2cert コマンド, 191
- ## H
- hosts ファイル, 140
 - httpd.conf ファイル, 42
- ## I
- ignore_cr1s キーワード
 - IKEv1 構成ファイル, 215
 - ike.preshared ファイル 参照 /etc/inet/secret/ike.preshared ファイル
 - ike.privatekeys データベース, 268
 - ike/config ファイル 参照 /etc/inet/ike/config ファイル

- ike/ikev2.config ファイル 参照 /etc/inet/
- ike/ikev2.config ファイル
- IKE, 119
 - 参照 IKEv1、IKEv2
 - FIPS 140-2 モード, 20, 164, 169
 - IKEv2 への移行, 168
 - IKE 情報の表示, 242
 - NAT と, 228
 - RFC, 255
 - 事前共有鍵, 165
 - 証明書, 166
 - プロトコルバージョン, 163
 - リファレンス, 251
- ike サービス
 - 説明, 252, 256
- IKE バージョン
 - 使用するものを 1 つ選択, 168
- ikeadm コマンド
 - 使用方法のサマリー, 243, 248
 - 説明, 259, 260, 263, 264
- ikecert certlocal コマンド
 - kc オプション, 212
 - ks オプション, 208
- ikecert コマンド
 - a オプション, 217
 - A オプション, 266
 - certdb サブコマンド, 209, 213
 - certrldb サブコマンド, 222
 - tokens サブコマンド, 230
 - t オプション, 266
 - 説明, 259, 263, 265
 - ハードウェア上で使用, 217
- ikeuser アカウント, 181
- ikeuser ディレクトリ, 181
- IKEv1
 - cr1s データベース, 268
 - CSR の生成, 212
 - ike.preshared ファイル, 265
 - ike.privatekeys データベース, 268
 - ikeadm コマンド, 264
 - ikecert certdb コマンド, 213
 - ikecert certrldb コマンド, 222
 - ikecert コマンド, 230, 265
 - in.iked デーモン, 263
 - ISAKMP SA , 172
 - NAT と, 226
 - Oracle Solaris システムでの IKEv2 との比較, 168
 - Perfect Forward Secrecy (PFS), 172
 - publickeys データベース, 267
 - SMF からのサービス, 262
 - SMF サービスの説明, 261
 - Sun Crypto Accelerator 6000 ボードの使用, 230
 - Sun Crypto Accelerator ボードの使用, 266, 267
 - 移動体システムと, 223
 - 鍵管理, 172
 - 鍵の保管場所, 261
 - 構成
 - CA 証明書による, 212
 - 移動体システム用の, 223
 - 概要, 201
 - 公開鍵証明書による, 207
 - 事前共有鍵による, 202
 - ハードウェア上, 229
 - 構成ファイル, 261
 - コマンドの説明, 261
 - 自己署名付き証明書の作成, 208
 - 自己署名付き証明書の追加, 208
 - 事前共有鍵, 173, 173, 203, 206
 - 実装, 201
 - セキュリティーアソシエーション, 263
 - データベース, 265
 - デーモン, 263
 - 特権レベル
 - 説明, 264
 - 変更, 265
 - 特権レベルの変更, 265
 - フェーズ 1 交換, 172
 - フェーズ 2 交換, 173
 - 有効な構成かどうかのチェック, 203
- ikev2.preshared ファイル 参照 /etc/inet/
- ike/ikev2.preshared ファイル
- ikev2cert gencert コマンド
 - ハードウェア上で使用, 197
- ikev2cert import コマンド
 - CA 証明書, 192
 - キーストアへの鍵の追加, 187
 - 証明書の追加, 192
 - ラベルの適用, 188
- ikev2cert list コマンド

- 使用, 195
- ikev2cert tokens コマンド, 183
- ikev2cert コマンド
 - gencert サブコマンド, 197
 - gencsr サブコマンド, 191
 - import サブコマンド, 187
 - list サブコマンド, 186, 190
 - setpin サブコマンド, 182
 - 説明, 260
- IKEv2
 - FIPS 140-2 および, 171
 - ikeadm コマンド, 260
 - IKEv1 からの移行, 168
 - IKEv1 の代わりに選択, 168
 - ikev2.preshared ファイル, 260
 - ikev2cert コマンド
 - tokens サブコマンド, 197
 - 自己署名付き証明書の作成, 185
 - 証明書のインポート, 192
 - 説明, 260
 - ハードウェア上で使用, 196, 197
 - in.ikev2d デーモン, 259
 - IPsec 接続に使用されるプロトコルのみ, 143
 - ISAKMP SA, 172
 - Oracle Solaris システムでの IKEv1 との比較, 168
 - SMF サービスの説明, 257, 258
 - Sun Crypto Accelerator 6000 ボードの使用, 196
 - 鍵管理, 169
 - 鍵交換, 169
 - 鍵の格納, 260
 - 鍵の保管場所, 257
 - 公開鍵証明書の格納, 184
 - 公開証明書のポリシー, 193
 - 構成
 - CA 証明書, 190
 - 概要, 175
 - 公開鍵証明書による, 184
 - 公開証明書用キーストア, 181
 - 事前共有鍵による, 176
 - 構成の確認, 237
 - 構成ファイル, 257
 - コマンドの説明, 257
 - 自己署名付き証明書の作成, 185
 - 自己署名付き証明書の追加, 185
 - 実装, 176
 - 証明書署名要求の生成, 191
 - セキュリティーアソシエーション, 259
 - デーモン, 259
 - ハードウェア PIN の確認, 183
 - ハードウェアトークンのリスト, 197
 - 有効な構成かどうかのチェック, 177
- ikev2 サービス
 - ikeuser アカウント, 181
 - 使用, 141
 - import サブコマンド
 - ikev2cert コマンド, 187
 - in.iked デーモン
 - c オプション, 203
 - f オプション, 203
 - アクティブ化, 263
 - 説明, 172
 - in.ikev2d デーモン
 - c オプション, 177
 - f オプション, 177
 - アクティブ化, 259
 - 説明, 169
 - in.routed デーモン, 28
 - Internet Security Association and Key Management Protocol (ISAKMP) SA
 - 説明, 172
 - 保管場所, 260, 265
 - ip-nospoof
 - リンク保護のタイプ, 21
 - IP セキュリティーアーキテクチャー 参照 IPsec
 - IP 転送
 - IPv4 VPN での, 150
 - VPN 内, 132
 - IP パケット, 119
 - 参照 パケット
 - IPsec による保護, 119
 - IP フィルタ
 - ipfilter サービス, 79
 - ipfstat コマンド
 - 6 オプション, 86
 - ipf コマンド
 - 6 オプション, 86
 - ipmon コマンド
 - IPv6 および, 86
 - ippool コマンド, 105

- IPv6 および, 86
- IPv6, 86
- IPv6 構成ファイル, 86
- NAT および, 83
- NAT 規則
 - 追加, 104
 - 表示, 103
- NAT 構成ファイル, 83
- アドレスプール
 - 管理, 105
 - 削除, 105
 - 追加, 106
 - 表示, 105
- アドレスプールおよび, 85
- アドレスプール構成ファイル, 85
- 概要, 75
- 規則セット
 - アクティブへの追加, 99
 - 概要, 80
 - 切り替え, 101
 - 非アクティブ, 97
 - 非アクティブの削除, 102
 - 非アクティブへの追加, 100, 100
 - 別のもののアクティブ化, 97
- 構成タスク, 89
- 構成ファイル, 80
- 構成ファイルの作成, 91
- 削除
 - NAT 規則, 103
- 作成
 - ログファイル, 111
- サンプル構成ファイル, 114
- 使用のガイドライン, 79
- デフォルトの表示, 90
- 統計, 107
- 統計の表示, 107
- パケット再構築の無効化, 93
- パケット処理の順序, 76
- パケットフィルタ (PF) との比較, 46, 67
- パケットフィルタリングの概要, 80
- パケットフィルタリングのルールセットの管理, 96
- 表示
 - アドレスプール統計, 110
 - 状態テーブル, 108
 - 状態統計, 108
 - チューニング可能パラメータ, 109
 - ログファイル, 112
 - マニュアルページの概要, 87
 - 無効化, 95
 - 有効化, 93
 - ループバックフィルタリング, 94
 - ルールセット
 - アクティブ, 97
 - 削除, 99
 - ルールセットの操作, 95
 - ロギングされたパケットをファイルに保存, 114
 - ログバッファのフラッシュ, 113
 - ログファイル, 110
- IP フィルタサービス
 - デフォルト, 90
- IP フィルタでの IPv6 構成ファイル, 86
- IP 保護
 - パケットフィルタ (PF) を使用したファイアウォール, 45
 - リンク保護, 20
- ipadm コマンド
 - hostmodel パラメータ, 150
 - 厳密なマルチホーム, 150
- ipf コマンド, 75
 - 参照 IP フィルタ
 - 6 オプション, 86
 - F オプション, 99
 - f オプション, 100
 - I オプション, 100
 - オプション, 97
 - コマンド行からの規則の追加, 99
- ipfilter:default サービス, 90
- ipfilter サービス, 79
- ipfstat コマンド, 75, 108
 - 参照 IP フィルタ
 - 6 オプション, 86
 - i オプション, 97
 - o オプション, 97
 - オプション, 97
- ipmon コマンド
 - IPv6 および, 86
 - IP フィルタログの表示, 112
- ipnat コマンド, 75

- 参照 IP フィルタ
 - 1 オプション, 103
 - コマンド行からの規則の追加, 104
- ippool コマンド, 75
 - 参照 IP フィルタ
 - F オプション, 105
 - IPv6 および, 86
 - 1 オプション, 105
 - コマンド行からの規則の追加, 106
- IPsec
 - /etc/hosts ファイル, 140
 - bypass オプション, 129
 - FIPS 140-2 および, 137
 - FIPS 140-2 承認アルゴリズムを使用した実行, 142
 - FIPS 140-2 と, 133
 - IKEv2 のみ使用, 143
 - IKE バージョンの指定, 143
 - in.iked デーモン, 256
 - in.ikev2d デーモン, 256
 - ipsecalgs コマンド, 254
 - ipseccconf コマンド, 129, 252
 - ipseccinit.conf ファイル
 - LAN のバイパス, 151
 - Web サーバーの保護, 144
 - 構成, 140
 - 説明, 253
 - トンネル構文の例, 146
 - ポリシーファイル, 129
 - ipseckey コマンド, 124, 254
 - IPsec 情報の表示, 242
 - IPv4 VPN と, 149
 - kstat コマンド, 255
 - NAT と, 133
 - or pass オプション, 129
 - or pass オプションの指定, 143
 - RBAC と, 138
 - RFC, 255
 - route コマンド, 153
 - SA の手動作成, 154
 - SCTP プロトコルと, 134, 138
 - snoop コマンド, 255
 - Trusted Extensions ラベルと, 138
 - VPN の保護, 146
 - アウトバウンドパケットプロセス, 121
 - アクティブ化, 136
 - アルゴリズムのソース, 254
 - 暗号化フレームワーク, 254
 - インバウンドパケットプロセス, 121
 - 概要, 119
 - 鍵管理
 - IKEv1, 172
 - IKEv2, 169
 - ipseckey コマンド, 124
 - リファレンス, 256
 - 仮想プライベートネットワーク (VPN), 132, 149
 - 仮想マシンと, 135
 - カプセル化セキュリティーペイロード (ESP), 125, 126
 - 構成, 252
 - 構成ファイル, 135
 - コマンド、リスト, 135
 - コンポーネント, 120
 - サービス
 - ipsecalgs, 136
 - manual-key, 136
 - policy, 136
 - サマリー, 251
 - リスト, 135
 - 実装, 138
 - 手動鍵, 125, 155
 - 手動鍵管理, 252
 - 手動鍵コマンド, 254
 - 信頼できるユーザーによる構成, 158
 - セキュアなりモートログインのための ssh の使用, 142
 - セキュリティーアソシエーション (SA), 120, 124
 - セキュリティーアソシエーション (SA) の追加, 140, 151
 - セキュリティーアソシエーションデータベース (SADB), 120, 256
 - セキュリティー上の役割, 157
 - セキュリティートラフィック, 139
 - セキュリティーパラメータインデックス (SPI), 124
 - セキュリティープロトコル, 120, 124
 - セキュリティーポリシーデータベース (SPD), 120, 252
 - ゾーンと, 135, 137

- データの 캡セル화, 126
 - 統計コマンド, 255
 - トランスポートモード, 129
 - トンネル, 132
 - トンネルモード, 129
 - バイパス, 129, 144
 - パケット保護の確認, 160
 - フローチャート, 121
 - 保護
 - VPN, 149
 - Web サーバー, 144
 - 移動体システム, 223
 - パケット, 119
 - 保護プロトコル, 125
 - 保護ポリシー, 128
 - ポリシーコマンド
 - ipseccconf, 252
 - ポリシーの設定
 - 一時的に, 252
 - 永続的に, 253
 - ポリシーファイル, 253
 - ユーティリティに対する拡張機能
 - snoop コマンド, 255
 - ラベル付きパケットと, 138
 - ルールの適用, 128
 - IPsec で使用する or pass オプション, 129
 - IPsec での ike_version オプションの使用, 143
 - IPsec での or pass オプションの使用, 143
 - IPsec での route コマンドの使用, 153
 - IPsec によるネットワークトラフィックの保護 (タスクマップ), 138
 - ipseccalg サービス, 252
 - ipseccconf コマンド
 - IPsec ポリシーの構成, 252
 - IPsec ポリシーの表示, 144, 253
 - セキュリティについて, 254
 - 説明, 136
 - トンネルの設定, 130
 - 目的, 129
 - ipseccinit.conf ファイル 参照 /etc/inet/
 - ipseccinit.conf ファイル
 - ipseckey コマンド
 - セキュリティについて, 255
 - 説明, 124, 136
 - 目的, 254
 - ipseckey ファイル 参照 /etc/inet/secret/
 - ipseckey ファイル
 - IPv6
 - および IP フィルタ, 86
- ## K
- kmf_policy プロパティ
 - IKEv2, 258
 - kmf-policy.xml ファイル 参照 /etc/inet/ike/
 - kmf-policy.xml ファイル
 - kmfcfg コマンド, 193
 - ksslcfg コマンド, 37, 41
 - kstat コマンド, 43
 - および IPsec, 255
- ## L
- L2 フレーム保護
 - リンク保護, 20
 - label キーワード
 - ikev2.config ファイル, 176
 - ikev2.preshared ファイル, 178
 - ikev2cert gencert コマンド, 185, 189
 - ikev2cert import コマンド, 188, 192
 - ikev2cert list コマンド, 195
 - IKEv2 のルールと事前共有鍵をマッチング, 238, 238
 - ldap-list キーワード
 - IKEv1 構成ファイル, 222
 - LDOM 参照 仮想マシン
 - list サブコマンド
 - ikev2cert コマンド, 186, 190
- ## M
- mac-nospoof
 - リンク保護のタイプ, 20
 - MAC 保護
 - リンク保護, 20
 - manual-key サービス
 - 使用, 156

説明, 252, 257

N

NAT

IPsec と IKE の使用, 226, 228

IPsec の制限, 133

IP フィルタ規則の構成, 84

IP フィルタでの概要, 83

NAT 規則

追加, 104

表示, 103

NAT 規則の削除, 103

RFC, 134

構成ファイル, 83

統計の表示, 109

パケットフィルタ (PF) での概要, 48

パケットフィルタ (PF) での規則の例, 59, 63

network/firewall サービス, 54

Network Firewall Management 権利プロファイル, 48

Network IPsec Management 権利プロファイル, 157

Network Management 権利プロファイル, 157

Network Security 権利プロファイル, 157

O

OCSP

説明, 167

ポリシー, 193, 222

OpenBSD パケットフィルタ, 45 参照 パケット
フィルタ (PF)

参照 パケットフィルタ (PF)

Oracle Solaris PF との比較, 47

openss1 コマンド, 41

Oracle iPlanet Web Server

SSL カーネルプロキシ と, 39

SSL パケットの高速化, 35

SSL 保護による構成, 39

P

pass オプション

IPsec 構成, 143

Perfect Forward Secrecy (PFS), 172

PF_KEY ソケットインタフェース, 124, 136

pf.conf ファイル

インストール, 69

説明, 66

デフォルトの規則セット, 68

パケットフィルタ (PF) 構成ファイル, 54

pf.os ファイル

インストール, 69

説明, 66

pfconf スクリプト, 54, 54, 70

pfctl コマンド

-s rules オプション, 71

規則をテストするためのオプション, 67

説明, 66

PFS 参照 Perfect Forward Secrecy (PFS)

PKCS #11 ライブラリ

ike/config ファイル内, 265

pkcs11_path キーワード

使用, 217

説明, 265

pkcs11_token/pin プロパティ

使用, 182, 198

定義, 258

リスト, 183

pkcs11_token/uri プロパティ

使用, 198

定義, 258

PKI 参照 認証局 (CA)

policy サービス

使用, 141, 152

説明, 251

proxy キーワード

IKEv1 構成ファイル, 222

publickeys データベース, 267

R

RBAC

IPsec と, 138

Requests for Comments (RFC)

IPv6 ジャンボグラム, 86

restricted

リンク保護のタイプ, 21

routeadm コマンド
 IP 転送, 150, 150
RSA 暗号化アルゴリズム, 266
rsyslog.conf エントリ
 IP フィルタ用に作成, 111
rules プロパティー
 パケットフィルタ, 71

S
SA 参照 セキュリティーアソシエーション (SA)
SADB 参照 セキュリティーアソシエーションデー
タベース (SADB)
SCA6000 board 参照 Sun Crypto Accelerator 6000
ボード
SCTP プロトコル
 IPsec と, 138
 IPsec の制限事項, 134
Secure Sockets Layer (SSL) 参照 SSL プロトコル
setpin サブコマンド
 ikev2cert コマンド, 182
snoop コマンド
 パケット保護の確認, 160
 保護されたパケットの表示, 255
spam
 パケットフィルタ (PF) でのテーブルの例, 60
ssl.conf ファイル, 41
SSL カーネルプロキシ
 Apache Web サーバーと, 37, 41
 Apache Web サーバーへのフォールバック, 41
 Oracle iPlanet Web Server の保護, 39
 鍵の格納, 41
 ゾーン内で Apache Web サーバーを保護する,
 43
 パズフレーズファイル, 41
SSL プロトコル, 35
 参照 SSL カーネルプロキシ
 SMF による管理, 39
 Web サーバーの高速化, 35
Sun Crypto Accelerator 6000 ボード
 FIPS 140-2 検証済み, 262
 IKEv1 で使用, 217, 230
 IKEv2 で使用, 196
syslog.conf エントリ
 IP フィルタ用に作成, 111

system-log サービス, 111

T

TCP/IP ネットワーク
 ESP による保護, 126
tokens サブコマンド
 ikecert コマンド, 230
 ikev2cert コマンド, 197
Trusted Extensions
 IPsec と, 138

U

Uniform Resource Indicator (URI)
 失効済証明書リストにアクセスするための, 221
use_http キーワード
 IKEv1 構成ファイル, 222

V

/var/user/ikeuser, 181
VPN 参照 仮想プライベートネットワーク (VPN)

W

Web サーバー
 SSL カーネルプロキシ の使用, 35
 SSL パケットの高速化, 35
 バックエンド通信の保護, 144
webservd デーモン, 41
Wireshark アプリケーション
 snoop コマンドで使用, 161
 URL, 255
 インストール, 234
 使用, 236