

openSUSE

12.3

www.suse.com

2013/03/04

セキュリティガイド



セキュリティガイド

Copyright © 2006–2013 Novell, Inc. and contributors. All rights reserved.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or (at your option) version 1.3; with the Invariant Section being this copyright notice and license. A copy of the license version 1.2 is included in the section entitled 「GNU Free Documentation License」.

For Novell trademarks, see the Novell Trademark and Service Mark list <http://www.novell.com/company/legal/trademarks/tmlist.html>. All other third party trademarks are the property of their respective owners. A trademark symbol (® , # etc.) denotes a Novell trademark; an asterisk (*) denotes a third party trademark.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither Novell, Inc., SUSE LINUX Products GmbH, the authors, nor the translators shall be held liable for possible errors or the consequences thereof.

下記に上記の日本語翻訳を掲載します。日本語の翻訳は公式なものではないことに注意してください。

Copyright © 2006–2013 Novell, Inc. および貢献者が全権利を留保しています。

この文書を、フリーソフトウェア財団発行の GNU フリー文書利用許諾契約書 バージョン 1.2 または (希望すれば) 1.3 が定める条件の下で複製、頒布、あるいは 改変することを許可する。ただし、この著作権とライセンス表記については変更不可部分 とする。この利用許諾契約書の複製物は「GNU フリー文書利用許諾契約書」という章に含まれている。

Novell 社の商標については、Novell 社の商標とサービスマーカー覧 <http://www.novell.com/company/legal/trademarks/tmlist.html> をご覧ください。Linux は Linus Torvalds 氏による登録商標です。その他の商標は 各所有者の所有物です。商標シンボル (®, # など) は それぞれ Novell 社の商標であることを示しています。また、アスタリスク (*) は 第三者の商標を示しています。

この書籍内にある全ての情報は細部に至るまで最大限の注意を払って制作されていますが、完全に正確であることを保証するものではありません。Novell, Inc., SUSE LINUX Products GmbH, 著者, 翻訳者のいずれも、本書籍内の誤りとそこから生じる結果について、一切の保証はいたしません。

目次

このガイドについて	xi
1 利用可能な文書	xi
2 フィードバック	xii
3 文書規約	xiii
4 このマニュアルの作成について	xiii
5 ソースコード	xiv
6 謝辞	xiv
 1 セキュリティと機密保持	 1
1.1 ローカルセキュリティとネットワークセキュリティ	2
1.2 いくつかのセキュリティ関連のヒント	11
1.3 セキュリティ問題の報告先	14
 I 認証	 15
 2 PAM を利用した認証	 17
2.1 PAM とは	17
2.2 PAM の設定ファイルの構成	18
2.3 sshd の PAM 設定	21
2.4 PAM モジュールの設定	23
2.5 pam-config を利用した PAM の設定	25
2.6 手作業での PAM 設定	26
2.7 さらなる情報	27

3 NIS の使用	29
3.1 NIS サーバの設定	29
3.2 NIS クライアントの設定	36
4 ディレクトリサービス LDAP	39
4.1 LDAP と NIS	40
4.2 LDAP のディレクトリツリー構造	41
4.3 YaST を利用した LDAP サーバの設定	43
4.4 YaST を利用した LDAP クライアントの設定	51
4.5 YaST による LDAP ユーザとグループの設定	58
4.6 LDAP ディレクトリツリーの参照	59
4.7 LDAP サーバの手動設定	61
4.8 LDAP データの手作業での管理	62
4.9 さらなる情報	66
5 Active Directory への対応	69
5.1 Linux と AD 環境の統合	69
5.2 Linux における AD 対応の背景となる情報	70
5.3 Linux クライアントに対して Active Directory を設定する方法	75
5.4 AD ドメインへのログイン	78
5.5 パスワードの変更	79
6 Kerberos を利用したネットワーク認証	83
6.1 Kerberos で使用する用語	83
6.2 Kerberos の動作概要	85
6.3 Kerberos に対するユーザからの外観	88
6.4 Kerberos のインストールと管理	89
6.5 さらなる情報	110

7 指紋読み取り装置の使用 111

7.1 対応するアプリケーションと処理	111
7.2 YaST を利用した指紋の管理	112

II ローカルセキュリティ 115

8 YaST を利用したセキュリティ設定 117

8.1 セキュリティの概要	117
8.2 事前定義済みのセキュリティ設定	118
8.3 パスワード設定	119
8.4 起動設定	120
8.5 ログイン設定	120
8.6 ユーザ追加	121
8.7 その他の設定	121

9 Linux におけるアクセス制御リスト 123

9.1 従来のファイルパーミッション	123
9.2 ACL の利点	125
9.3 定義	125
9.4 ACL の処理	126
9.5 アプリケーションにおける ACL サポート	134
9.6 さらなる情報	135

10 パーティションとファイルの暗号化 137

10.1 YaST を利用した暗号化ファイルシステムの設定	138
10.2 暗号化されたホームディレクトリの使用	142
10.3 vi を使用した単一 ASCII テキストファイルの暗号化	143

11 AIDE を利用した侵入検知 145

11.1 AIDE を利用する理由	145
-------------------------	-----

11.2 AIDE データベースの設定	146
11.3 ローカルの AIDE チェック	148
11.4 システムに依存しないチェック	150
11.5 さらなる情報	151

III ネットワークセキュリティ 153

12 SSH: 機密を保護する通信 155

12.1 ssh—Secure Shell (機密を保持するシェル)	156
12.2 scp—機密保護機能付きのファイルコピー	157
12.3 sftp—機密保護機能付きのファイル転送	158
12.4 SSH デーモン (sshd)	158
12.5 SSH 認証の仕組み	160
12.6 ポート転送	163
12.7 YaST を利用した SSH デーモンの設定	164
12.8 さらなる情報	165

13 マスカレードとファイアウォール 167

13.1 iptables を利用したパケットフィルタ	167
13.2 マスカレードの基礎	169
13.3 ファイアウォールの基礎	170
13.4 SuSEfirewall2	171
13.5 さらなる情報	178

14 VPN サーバの設定 179

14.1 考え方の概要	179
14.2 最も簡単な VPN の作成例	181
14.3 証明機関を利用する VPN サーバの設定	183
14.4 VPN 内でのネームサーバの変更	189

14.5 クライアント向けの KDE および GNOME アプレット	190
14.6 さらなる情報	192

15 X.509 証明書の管理 **195**

15.1 デジタル証明書の仕組み	195
15.2 CA 管理用の YaST モジュール	200
15.3 さらなる情報	211

IV AppArmor を利用した権利制限 **213**

16 AppArmor の紹介 **215**

16.1 AppArmor プロファイリングの関連情報	216
-----------------------------------	-----

17 前置き **217**

17.1 AppArmor のインストール	217
17.2 AppArmor の有効化と無効化	218
17.3 プロファイルを行なうアプリケーションの選択	219
17.4 プロファイルの構築と修正	220
17.5 プロファイルの更新	222

18 プログラムに対する免疫付与 **223**

18.1 AppArmor フレームワークの紹介	224
18.2 免疫を与えるプログラムの判断	226
18.3 cron ジョブへの免疫付与	227
18.4 ネットワークアプリケーションへの免疫付与	228

19 プロファイルの構成と文法 **233**

19.1 AppArmor プロファイルの構成	234
19.2 プロファイルの種類	237
19.3 #include ステートメント	240

19.4 機能項目 (POSIX.1e)	241
19.5 ネットワークアクセス制御	241
19.6 パスとグロブ	242
19.7 ファイルのアクセス許可とアクセスモード	245
19.8 実行モード	248
19.9 リソース制限制御	253
19.10 監査ルール	255
20 AppArmor のプロファイルリポジトリ	257
20.1 ローカルリポジトリの使用	257
21 YaST を利用したプロファイルの構築と管理	259
21.1 ウィザードを使用したプロファイルの追加	261
21.2 手作業でのプロファイル追加	268
21.3 プロファイルの編集	269
21.4 プロファイルの削除	274
21.5 ログ項目からのプロファイル更新	275
21.6 AppArmor の管理	275
22 コマンドラインからのプロファイル構築	279
22.1 AppArmor のモジュール状態の確認	279
22.2 AppArmor プロファイルの構築	281
22.3 AppArmor プロファイルの追加と作成	282
22.4 AppArmor プロファイルの編集	282
22.5 AppArmor プロファイルの削除	282
22.6 プロファイルを作成する際の 2 つの方法	283
22.7 重要なファイル名とディレクトリ	305

23 ハット変更を利用した Web アプリケーションのプロファイル作成	307
23.1 Apache のハット変更	308
23.2 mod_apparmor 向けの Apache 設定	314
24 pam_apparmor によるユーザに対する制限	319
25 プロファイルを作成したアプリケーションの管理	321
25.1 セキュリティ拒否イベントへの対応	321
25.2 セキュリティプロファイルの管理	322
26 サポート	325
26.1 AppArmor のオンライン更新	325
26.2 マニュアルページの使用	325
26.3 さらなる情報	327
26.4 トラブルシューティング	327
26.5 AppArmor のバグ報告について	334
27 AppArmor 用語集	337
A GNU ライセンス	341
A.1 GNU General Public License	341
A.2 GNU 一般公衆利用許諾契約書 (日本語訳)	345
A.3 GNU Free Documentation License	349
A.4 GNU フリー文書利用許諾契約書	354

このガイドについて

このマニュアルでは、openSUSE におけるシステムセキュリティについての 基本的な考え方を紹介しています。大きく分類すると、Linux における認証機構 (たとえば NIS や LDAP など)、アクセス制御リストや暗号化、侵入検知などの ローカルセキュリティ、ファイアウォールやマスカレード、仮想プライベート ネットワークによるネットワークセキュリティをそれぞれ説明しています。また、このマニュアルでは AppArmor などの openSUSE 固有のセキュリティ ソフトウェア (このソフトウェアでは、プログラムが読み込み／書き込み／実行 できるファイルを個別に制限することができます) についても説明しています。

このマニュアルの多くの章では、追加の文書資源に対してリンクを示しています。これらはシステムに含まれているものを指す場合があるほか、インターネット上に 公開されている文書に対するリンクもあります。

お使いの製品に対して利用可能な文書の概要についてと、最新版の文書については、<http://www.suse.com/documentation> または下記の 章をお読みください。

1 利用可能な文書

HTML 版や PDF 版の各マニュアルは、それぞれ各種の言語に翻訳されています。この製品に対しては、それぞれ下記に示す ユーザ向けおよび管理者向けマニュアルが用意されています:

スタートアップ (↑スタートアップ)

DVD や ISO イメージから openSUSE のインストールを行ない、GNOME や KDE デスクトップの簡単な説明と、そこで動作する主なアプリケーションを紹介するまでの範囲を説明しています。また、LibreOffice の概要説明のほか、文書作成や表計算での 作業、およびグラフィックやプレゼンテーションの作成を行なうためのモジュールについても説明しています。

リファレンス (↑リファレンス)

openSUSE に関する一般的な理解を深め、より詳しいシステム管理作業を行なうための情報が書かれています。主にシステム管理者のほか、システム 管理知識のあるホームユーザに向けた文書です。また、複雑な配置シナリオやシステムの管理方法、主なシステムコンポーネントとのやりとりや openSUSE が提供するネットワークサービス、ファイルサービスに関する詳しい情報も 書かれています。

セキュリティガイド (i ページ)

ローカル環境やネットワークセキュリティを含めた、システムセキュリティに関する基本的な考え方が書かれています。AppArmor のようなセキュリティソフトウェア (プログラムが読み書きしたり実行したりするファイルをプログラム単位で指定できるもの) の一般的な使い方を示しているほか、セキュリティ関連のイベント情報を確実に収集するための監査システムの使い方も示しています。

システム分析とチューニングガイド (↑システム分析とチューニングガイド)

問題の検出や解決、最適化に対する管理者向けのガイドです。お使いのシステムに関して監視ツールを利用し点検と最適化を行なう方法や、効率的に資源を管理するための手順が記されています。また、一般的によくある問題やそれに対する解決方法、追加のヘルプや文書資源についても示しています。

KVM を利用した仮想化 (↑KVM を利用した仮想化)

このマニュアルでは、openSUSE で KVM (カーネルベースの仮想マシン) による仮想化を設定したり、管理したりするための手順を紹介しています。また、libvirt や QEMU を利用した VM ゲストの管理方法についても紹介しています。

ほとんどの製品マニュアルは HTML 版の形で、インストール済みシステムの `/usr/share/doc/manual` に置かれています。またデスクトップのヘルプセンターからもアクセスすることができます。最新の文書は、<http://www.suse.com/documentation> に置いています。ここからお使いの製品について、PDF 版と HTML 版をダウンロードすることができます。

2 フィードバック

いくつかの方法でフィードバックを送ることができます:

バグや機能追加リクエスト

製品のコンポーネントに対してバグの報告を行なったり、もしくは機能の追加リクエストを送信したりしたい場合は、<https://bugzilla.novell.com/> をご利用ください。文書内の間違いについては、各製品の *Documentation* コンポーネントに対してバグ報告をお願いいたします。

Bugzilla を初めてお使いになる場合は、下記の記事をお読みください:

- http://ja.opensuse.org/Submitting_bug_reports
- http://ja.opensuse.org/Bug_reporting_FAQ

ユーザコメント

このマニュアルに対するコメントや提案のほか、この製品に含まれる他のドキュメント 類に対するコメントを歓迎します。オンラインドキュメントの場合は、それぞれの ページ下部にあるコメント機能をご利用いただくか、もしくは <http://www.suse.com/documentation/feedback.html> から コメントをお送りください。

メール

この製品に対するフィードバックを送信するには、`doc-team@suse.de` 宛のメールもお使いいただけます。それぞれドキュメントのタイトルと製品バージョン、発行日時を添えてお送りください。また、間違いの報告や加筆に対する提案につきましては、その簡潔な説明と、セクション番号およびページ (または URL) をお送りください。

3 文書規約

このマニュアルでは、下記のルールで文書を記述しています:

- `/etc/passwd`: ディレクトリ名やファイル名を示しています
- *placeholder*: 置き換えを示しています *placeholder* を実際の値に置き換えます
- `PATH`: `PATH` という名前の環境変数を示しています
- `ls, --help`: コマンドやオプション、パラメータ を示しています
- `user`: ユーザまたはグループ
- `Alt, Alt + F1`: 入力するキーやキーの組み合わせを示しています; キーはキーボードに書かれている とおりに大文字で示されます
- `ファイル, ファイル > 名前を付けて保存`: メニュー項目やボタンなどを 示しています
- `ダンシングペンギン` (他のマニュアル内 `ペンギン` の章): 他のマニュアル内にある章を示しています

4 このマニュアルの作成について

この書籍は、DocBook (詳しくは <http://www.docbook.org> をご覧ください) のサブセットである Novdoc で書かれています。XML のソースファイルは `xmllint`

で検証された後に xsltproc で処理され、Norman Walsh 氏のスタイルシートのカスタマイズ版を利用して XSL-FO に変換されます。最終的な PDF ファイルは RenderX 提供の XEP で生成しています。また、この マニュアルを構築するために使用するオープンソースツールとその環境は、openSUSE と共に公開されている daps パッケージ内にあります。なお、daps の Web ページは <http://daps.sf.net/> です。

5 ソースコード

openSUSE のソースコードは、どなたにでもご利用いただけます。ダウンロードのリンクやその他の説明については、http://ja.opensuse.org/Source_code をお読みください。

6 謝辞

多数の無償貢献のお陰で、Linux 開発者はその開発にあたってグローバルな協力を行なうことができています。我々は彼らのそのような努力に感謝します— 彼らの貢献がなければ本ディストリビューションは存在していませんでした。また、Frank Zappa 氏と Pawar 氏にも感謝しています。もちろん Linus Torvalds 氏には特に感謝しています。

Have a lot of fun!

SUSE チームより

セキュリティと機密保持

Linux や UNIX システムにある特長の 1 つに、同時に複数のユーザに対する処理を実行できる (マルチユーザ環境) というものがあります。また、単一のコンピュータで、これらのユーザは複数のタスクを実行できる (マルチタスク) という特長もあります。さらに現状のオペレーティングシステムには、ネットワークを透過的に扱う仕組みが備わっていて、ユーザは自分自身が使用しているデータやアプリケーションが、ローカルにあるのかネットワーク上にあるのか、気にする必要がなくなっています。

マルチユーザの機能では、異なるユーザのデータを別々に保管しなければならないことを意味しているほか、セキュリティやプライバシーも保障されなければなりません。データに対するセキュリティは重要な問題で、コンピュータがネットワークに繋がるよりも前の段階から、議論が行なわれてきた問題です。もちろん現状であっても、データが失われた場合や障害が発生したような場合 (主にハードディスク) に、データを保持し続けられるということが最も重要ではありますが。

本章では機密保持に関する問題を中心に説明を行ない、ユーザのプライバシーを守るための方法を説明しています。ただし、広範囲にあたるセキュリティの考え方は、定期的に更新しなければならないものであるほか、実際に作業可能な範囲で、現場に即したやり方でなければならないことに注意してください。このマニュアルでは、ハードディスクに障害が発生した場合だけでなく、不正アクセスや不正なデータ破壊が発生した場合などの対応方法について、作業を効率よく進めるための説明を行なっています。

1.1 ローカルセキュリティとネットワークセキュリティ

データにアクセスする方法としては、下記のようなものがあります：

- 情報を必要としているユーザ同士で情報を交換したり、コンピュータにアクセスしたりする方法
- コンピュータのコンソールを利用した直接的な方法
- シリアルポートを介した方法
- ネットワークリンクを介した方法

いずれの場合でも、ユーザは必要な資源やデータにアクセスするにあたって、認証が行なわれるべきです。Web サーバは一般にこのようなデータに対する認証を実施しませんが、Web サーバは公開の場所であるため、機密を保持しなければならないデータはそもそも配置されるべきではありません。

上述の一覧において最初の方法は、最も多く人間が介在する方法です（たとえば銀行員とのやりとりで、口座の所有者であることを示すのに必要であったりする場合など）。このときは署名（押印）や暗証番号、もしくはパスワードなどの入力が必要とされ、本人確認が行なわれます。このような状況では、それらの個人情報を引き出して、本人に偽装できるようにする攻撃方法が考えられます。このような攻撃が発生した場合、一般にはさらなる個人情報を引き出すよう誘導されることが考えられます。これを **ソーシャルハッキング** と呼びます。このような攻撃を防ぐには、利用者に対して注意を喚起し、個人情報を迂闊に開示しないように教育する必要があります。コンピュータシステムに進入する前に、攻撃者はこのような行為を受け付け係や企業のサービス担当者のほか、場合によっては家族に対して実施する場合があります。多くの場合、このようなソーシャルハッキングは、ずっと時間が経過してから気づくことがあります。

データに不正アクセスを行ないたいと考えている攻撃者は、お使いのハードウェアに対して直接アクセスする従来の方法でデータを盗むことも考えます。そのため、そのようなマシンは取り外されたり入れ替えられたり、部品単位で交換されたりしないよう、物理的な盗難防止策を施すべきです。これはバックアップ媒体やネットワークケーブル、電源コードなどにも当てはまります。さらにはよく知られたキー入力での通常とは異なる処理を行なわせることができることから、起動処理についても保護を行なう必要があります。まずは BIOS とブートローダに対してパスワードを設定し、保護を行ってください。

シリアルポートに接続されているシリアル端末は、今なお様々な場所で使用されています。ネットワークインターフェイスとは異なり、ホスト間の通信にネットワークプロトコルが使用されておらず、シンプルなケーブルや赤外線通信で、相互に単純なデータ（多くは文字）がやりとりされています。このようなシステムではケーブル自身が最大の弱点で、たとえば古いプリンタが接続されているような環境があると、その中を流れるデータは簡単に記録することができてしまいます。これはプリンタだけでなく、その他のデバイスであっても同様です。

ローカルファイルの読み込みは、ネットワーク上離れた場所にあるサーバに対して接続する場合と比べて、追加のアクセスルールを必要とします。これがローカルセキュリティとネットワークセキュリティの違いで、これらは明確に区別して扱う必要があります。

1.1.1 ローカルセキュリティ

ローカルセキュリティは、まずコンピュータを配置する物理的な環境を考えることから始まります。コンピュータは必要なセキュリティ要件を満たす場所に配置されるべきで、お互いにユーザが切り離されていて、パーミッションや識別子が他のユーザときちんと区別できていることを目的とします。これは一般論ではありますが、特に root というシステム管理権限を持つユーザの場合は、特に明確な区別が必要です。root はその他のユーザに成り代わることができるばかりか、パスワードを尋ねられたりすることなくローカルに保存されている全てのファイルを読み込むことができます。

1.1.1.1 パスワード

Linux システムではパスワードはそのままの形では保管されておらず、入力したパスワードが保存したデータと単純に比較できるようなものでもありません。もしも単純に比較できるような仕組みであったとすると、そのファイルさえ読み込めれば、簡単にそれらのユーザを偽装することができてしまうためです。Linux システムではその代わりに、パスワードが暗号化された形で保管されていて、パスワードが入力されるたびにその暗号化を行ない、保管されている暗号データと、入力して暗号化したデータを比較することでパスワードを確認しています。これは暗号化されたパスワードを元の文字列に戻すことができない場合に有効な仕組みです。

このような仕組みは特殊なアルゴリズムを利用して実現するもので、一方向にしか動作しない仕組みから、*トラップドアアルゴリズム*とも呼ばれています。暗号化した文字列を取得した攻撃者が現われても、単純に同じアルゴリズムを適用するだけでは元の文字列には戻すことができません。その代わりに暗号化されたパスワード

と一致するまで、ありうるパスワードを端から試してみる、という作業が必要になります。パスワードが 8 文字程度の長さであれば、それを端から試してみるという作業が途方もないものになることは、想像に難くないでしょう。

1970 年代の時点で、この方法は 1 つのパスワードを暗号化するのに数秒程度しか必要としないのに、逆に暗号を解読するには極端に長い時間がかかるという点で、他の方法よりも機密を保持できることが示されてきました。現在と比較すると、PC の性能は数十万倍から数百万倍にまで高まってしまったことから、安全のため暗号化されたパスワード (/etc/shadow) も通常のユーザには読み込めないようになっています。もちろんのことながら、パスワードが簡単に推測できるものであってはなりません。万が一何らかのエラーでパスワードファイルを読み取られてしまった場合、よくあるパスワードではすぐに破られてしまうためです。そのため、パスワードを設定する際には何らかの「変換」を施すとよいでしょう。たとえば「tantalize」という単語をパスワードに使用したい場合は、「a」を似通った文字である「@」に、「e」を似通った数字である「3」置き換え、「t@nt@1lz3」のようにします。

単語内の特定の文字を似たような文字や数字に置き換える（「tantalize」を「t@nt@1lz3」に）だけでは不十分です。パスワードを解析するプログラムは辞書を利用して、置き換えるようなパターンにも対応しているためです。さらによい方法としては、汎用的な意味を持たない単語を利用し、自分自身にしか意味のない文字を作る方法があります。たとえば本のタイトルのうち、単語の最初の文字だけをとる方法があります。Umberto Eco 氏による「The Name of the Rose」という本であれば、単語の最初の文字を取って「TNNotRbUE9」のようなパスワードにします。逆に、「beerbuddy」とか「jasmine76」のように、少し調べればわかってしまうような文字列は、簡単に推測されてしまいます。

1.1.1.2 起動処理

まずはお使いのシステムを設定して、フロッピーディスクドライブや CD/DVD ドライブから起動できないように設定しておき、さらに BIOS に対してパスワードロックを設定して、ハードディスクだけを利用して起動できるように設定してください。また、通常の Linux システムは、ブートローダを利用して起動し、起動するカーネルに対してパラメータを設定できるようになっています。このようなパラメータ入力不正に行なわれることのないようにするため、ブートローダに対してパスワード保護を設定してください（詳しくは 第9章 ブートローダ GRUB (↑リファレンス) をお読みください)。これはカーネル自身が root で動作するだけでなく、システム起動時のスクリプトも root で動作することから、セキュリティを保護するにあたっては必須の設定です。

1.1.1.3 ファイルのパーミッション

一般的なルールとして、作業は最低限必要な権限だけを利用して行なうべきものです。たとえばメールを読んだり書いたりするのに、root の権限が必要となることはあり得ないでしょう。もしもメールプログラムにバグがあった場合は、攻撃者はこのバグを利用し、最大限の権限で不正なプログラムを起動できてしまうためです。上記のようなルールに従っていれば、被害をできるだけ小さくすることができます。

openSUSE に含まれている全ファイルに対するパーミッションは、注意して設定されています。追加のソフトウェアやその他のファイルをインストールする場合は、特にパーミッションビットを設定する際、管理を行なう方が注意をして設定を行ってください。経験者やセキュリティを意識するシステム管理者であれば、ls コマンドに -l オプションを設定し、長い形式でのファイル一覧を表示することで、誤ったファイルのパーミッションをチェックすることができます。ファイルの属性を誤って設定してしまうと、そのファイルが不用意に変更されたり削除されたりする場合があります。ほか、これらのファイルを root で実行することができてしまう場合があります。また、それが設定ファイルであるとすれば、root の権限で読み込んでしまう場合もあります。これにより、攻撃を受ける可能性が格段に高まってしまいます。このような攻撃は、プログラム (卵) は異なるユーザ (鳥) が実行する (暖めて孵化させる) ことから、「カッコウの卵 (cuckoo eggs)」と呼ばれます。これは、他の鳥に自分の卵を暖めて孵化させる (托卵) という、カッコウの習性から来た言葉です。

openSUSE® システムでは、/etc ディレクトリ内に permissions, permissions.easy, permissions.secure, permissions.paranoid という 4 種類のファイルが用意されています。これらのファイルは特殊なパーミッションを設定するために用意されていて、それぞれ全てのユーザに書き込みを許すディレクトリやファイル、setUID ビット (これが設定されたプログラムは、起動したユーザの ID ではなく、パーミッションに設定されたユーザの ID (多くの場合は root) で実行されるものです) のファイルなどを定義しています。このファイルに加え、/etc/permissions.local ファイルではそのシステム独自の設定を追加することができます。

openSUSE の設定プログラムで使用するファイルを選択し、パーミッションを設定するには、YaST から **セキュリティとユーザ** 内にある **ローカルセキュリティ** を利用して行ないます。詳しくは /etc/permissions ファイルに書かれているコメントか、もしくは chmod のマニュアルページ (man chmod) をお読みください。

1.1.1.4 バッファオーバーフローと書式文字列のバグ

ユーザ側で自由に変更のできるデータをプログラムが処理するにあたって、特別な配慮をしなければなりません。ただし、配慮を行なうのは一般の利用者ではなく、アプリケーションのプログラマです。プログラマは自分が作成しているプログラムが正

しくデータを解釈し、小さすぎることのないメモリ領域内に書き込んでいることを確認しなければなりません。またプログラマは、インターフェイスとして決めた方法で、一貫したデータ処理を実施する必要もあります。

もしもバッファ内にデータを書き込むにあたって、メモリバッファが十分なサイズでなかった場合、*バッファオーバーフロー* という状況が発生します。これは、利用可能なバッファサイズの領域よりもデータ (ユーザが提供したデータ) のサイズのほうが大きいことによって発生するもので、結果としてデータはバッファ領域の終端を超えて書き込まれてしまい、特定の状況下では、単にデータを処理するだけでなく、ユーザ側から提供されたデータがプログラムの動作に影響を与える結果になってしまいます。この種類のバグは、特にプログラムが特別な権限 (1.1.1.3項「ファイルのパーミッション」(4 ページ) を参照) で動作している場合に特に深刻な問題になります。

書式文字列のバグ は上記と少し異なる方法で行なわれるものですが、ユーザの入力がプログラムを混乱させるという点は同じです。多くの場合、これらのプログラミング上のエラーは、特別な権限 (setuid, setgid) で実行している場合に問題となるものであるため、それらのプログラムから特別な権限を取り除くことで、データやシステムに対する保護を行なうことができます。もちろんのことですが、最良の方法は最低限の権限だけを利用して実行するという点に違いはありません (詳しくは 1.1.1.3項「ファイルのパーミッション」(4 ページ) をお読みください)。

バッファオーバーフローや書式文字列のバグは、いずれもユーザデータの処理に関連したバグですが、ローカルから実行するタイプの攻撃だけを受けるというものではありません。この種類のバグとして報告されているものの多くは、ネットワークの接続を介して攻撃を受けています。そのため、バッファオーバーフローや書式文字列のバグは、ローカルセキュリティとネットワークセキュリティの両方に分類されることとなります。

1.1.1.5 ウイルス

一般的に言われていることとは異なり、Linux 上で動作するウイルスも存在します。ですが、ウイルスとして知られているものは、その作者が特定の技術や考え方が想定通りに動作することを示す、*コンセプト証明* の形で公開されているものです。今までに *本当の意味での* ウイルスが発見されたことはありません。

ウイルスは、生存し続けることのできる媒体無しでは生き残ることができません。この場合、媒体とはシステム内にあるプログラムや、重要と位置づけられているストレージ領域 (たとえばマスターブートレコード) で、ウイルスのプログラムコードから書き込めるものである必要があります。Linux ではマルチユーザの機能があるた

め、特定のファイルに対して書き込みを制限するような設定 (これは特にシステムファイルに対しては重要です) を行なうことができます。このことから、通常の作業を root の権限で実行してしまうと、システムをウイルスに感染させてしまう可能性が大きくなることになります。逆に、上述のとおりできる限り低い権限で実行するというルールを守っていれば、ウイルスに感染する可能性を低くすることができます。

また上記とは別に、詳しく知らないようなインターネットサイトから、ソフトウェアをダウンロードして実行してしまうようなことも避けるべきです。openSUSE の RPM パッケージには電子署名の技術が用いられていて、それらを構築するにあたって特別に注意して署名が付与される仕組みになっています。ウイルスは、管理者やユーザが必要なセキュリティへの注意を怠ったことを示す印であり、高度に安全性を高めたシステムでさえも脅かす可能性があります。

なお、ウイルスとワームはそれぞれ異なるものであることに注意してください。ワームはネットワーク世界にのみ存在するもので、それを広げるのに媒体を必要とするものではありません。

1.1.2 ネットワークセキュリティ

ネットワークに対するセキュリティは、外部のネットワークからの攻撃を防ぐのに重要です。ユーザ名とパスワードを入力する典型的なログイン認証処理は、ローカル側のセキュリティ問題ですが、ネットワークを介してログインするような場合では、2 種類のセキュリティ要件、つまり実際の認証が始まるまでのネットワークセキュリティの部分と、認証処理以降のローカルセキュリティの 2 つを考慮する必要があります。

1.1.2.1 X Window System と X 認証

冒頭でも記述しているとおり、ネットワークを透過的に扱うのが UNIX システムの特徴のうちの 1 つです。UNIX オペレーティングシステム向けのウインドウシステムである X Window System (以下 X) では、これをうまく利用しています。X ではリモートのホストからでも問題なくログインできるほか、ネットワークを介してグラフィカルなプログラムを起動し、表示内容を転送することもできます。

X のクライアントがリモートの X サーバに対して何らかのデータを表示する必要がある場合、サーバ側は不正なアクセスが行なわれることの無いよう、ディスプレイ資源を管理する必要があります。もっと明確に言うと、クライアントプログラムに対して、必要なアクセス許可を設定しなければなりません。X のシステムでは 2 つの方法でのアクセス許可が用意されていて、それぞれホストベースのアクセス制御と、

Cookie ベースのアクセス制御と呼ばれます。ホストベースのアクセス 制御はクライアントの IP アドレスで制御を行なうもので、xhost というプログラム を利用して制御することができます。xhost は X サーバが動作するマシンに対し、そのデータベースに信頼するクライアントの IP アドレスを登録します。ただし、IP アドレスによる認証では安全とは言えません。たとえばクライアント側で複数の ユーザが利用していた場合では、全員に対して X サーバへのアクセスを許すことになってしまいます。これでは IP アドレスが盗まれた場合と同じ危険が発生することになってしまいます。このような欠点から、この認証方法についてここでは 詳しく説明しません。詳しく知りたい場合は、man xhost で表示されるマニュアル ページをお読みください。

Cookie ベースのアクセス制御では、X サーバと信頼されているユーザのみが 知る文字列が用意され、それがある種の ID カードとして機能します。Cookie はログイン時にユーザのホームディレクトリ内にある .Xauthority ファイルに保管され、X サーバを使用してウインドウを表示したい X クライアント に対して提供されます。 .Xauthority ファイルは xauth というツールを利用して操作することができます。誤って .Xauthority ファイルの名前を変更したり削除 したりしてしまった場合は、新しいウインドウや X クライアントを開くことが できなくなります。

また、SSH (Secure SHell) では X サーバとのネットワーク通信を暗号化し、その暗号化通信をユーザ側で気にすることなく、透過的に利用できるようになります。この機能は X 転送 (forwarding) と呼ばれていて、これは SSH のサーバ (X クライアント) 側で X サーバを擬似し、DISPLAY 環境変数を 介してリモートのホスト (X サーバ) に転送することによって実現しています。SSH について詳しくは 第12章 *SSH: 機密を保護する通信* (155 ページ) をお読みください。

警告

安全なホストにログインしていると判断できない場合は、X 転送機能を利用してはなりません。X 転送が有効化されている場合、そのホストに悪意のある ユーザが存在していると、SSH 接続を介してお使いのマシンにある X サーバに アクセスし、様々な攻撃 (キーボード入力を盗み見たりするなど) を行なう 可能性があります。

1.1.2.2 バッファオーバーフローと書式文字列のバグ

1.1.1.4項「バッファオーバーフローと書式文字列のバグ」(5 ページ) でも示しているとおり、バッファオーバーフローと書式文字列のバグは、ローカル セキュリティとネットワークセキュリティの両方に分類されます。ローカル 側での問題と同じように、ネットワークプログラムでのバッファオーバーフロー 攻撃は、root の権限を

奪取する 目的で行なわれます。対象のプログラムが root で動作していない場合でも、これらのバグを利用して特権を持たないアカウントへのアクセス手段を獲得し、さらなる脆弱性を利用してシステムへの侵入を試みる場合があります。

ネットワーク側から攻撃可能なバッファオーバーフローと書式文字列のバグは、リモートからの攻撃としては一般的にもっとも頻繁に発生しうるものです。新しく見つかったセキュリティホールを攻撃するプログラムは、セキュリティ 関連のメーリングリストによく投稿されます。これらはコードの詳細を知る ことなく脆弱性を利用できる仕組みではありますが、オペレーティングシステム の作成者に対して問題の修正を強制できることから、機密を高めるのに何年もの 間役立ってきた仕組みです。フリーソフトウェアでは誰もがソースコードに アクセスできる (openSUSE でも利用可能なすべてのソースコードを公開 しています) ため、誰もが脆弱性を発見することができ、誰もがそのバグを 修正することができるようになっています。

1.1.2.3 サービス拒否攻撃

サービス拒否攻撃 (Denial of Service; DoS) の目的は、特定のサーバ プログラムやサーバ全体を利用できなくすることにあります。これは様々な 方法で実現できてしまうもので、たとえばサーバに過大な負荷をかけたり、巨大なデータパケットを送りつけて本来の処理を妨害したり、リモートから バッファオーバーフローの攻撃を仕掛けたりして実現することができます。つまり、DoS 攻撃はサービスを提供できない状態にする目的で行なわれます。ただし、いったん特定のサービスが利用できない状態になると、*中間者攻撃* (盗聴、TCP コネクションの奪取や偽装) を受けやすい状態になってしまうほか、DNS ポイズニングの標的にもなってしまいます。

1.1.2.4 中間者攻撃: 盗聴, 乗っ取り, 偽装

一般的に、攻撃者自身が通信の仲介役となることで行なわれるリモートからの 攻撃を *中間者攻撃* (man-in-the-middle attack) と呼びます。一般的には、中間者攻撃が行なわれても被害者はその被害に気づく ことはありません。また、中間者攻撃には似たような攻撃方法が多数存在していて、たとえば攻撃者が接続要求を奪取し、本来の接続先を擬似するような マシンにその要求を転送したとすると、その被害者は異なるホストと通信を 行なうことになってしまいます。

中間者攻撃としてもっともシンプルなやり方は、*盗聴* (攻撃者は「単純に」ネットワークを流れる通信を聞き取るだけで何もしない) による方法です。もっと複雑な方法で中間者攻撃を行なうと すると、既に確立された通信を奪取する (ハイジャックする) という方法もあります。これを行なうには、まず攻撃者は接続されている TCP コネクションに 対して、そのシーケンス番号を予測するためにパケットの解析を行ない

ます。ただ、最終的に目的となるホストに対する接続を奪取したとしても、被害者側では接続がエラーになり強制終了してしまうため、TCP ではそれにすぐに気づくようになっています。ただし、その他のプロトコルではハイジャック 攻撃に対して対策を施していないもの (接続を開始したときに簡易な認証手順しか踏まないもの) があるのも事実で、このようなプロトコルを利用している 場合には攻撃が成立してしまいます。

偽装とは特定の packets を修正し、一般に IP アドレス の情報を偽造して packets を送りつける攻撃です。このような攻撃では主に、偽の packets を送りつけるようなことを行ないます (Linux マシンではこのような 攻撃を行なうのに、スーパーユーザ (root) の権限が必要です)。

中間者攻撃の多くは、DoS と組み合わせて用いられます。攻撃者が特定のホストに対して、たとえ短時間であっても、サービスを突然に落としたいと考えた場合、ホストは DoS によって、一時的に攻撃に対して抵抗することができなくなってしまう性質から、能動的な攻撃 (中間者攻撃) がやりやすくなるためです。

1.1.2.5 DNS ポイズニング

DNS ポイズニングとは、攻撃者が DNS サーバのキャッシュを間違った値にする 攻撃で、偽造された DNS の応答 packets を送信してそれを信用させることで、特定のサーバに対する通信を奪取しようとするものです。多くのサーバでは、他のホストとの信頼関係を IP アドレスやホスト名を基準にして構築しています。攻撃者はあらかじめ、このようなホスト同士の信頼関係について熟知していて、特定のホストに完全になりすますホストを用意しておく必要があります。通常、攻撃者はサーバから受信した packets を解析し、必要な情報を 取得します。また、攻撃者は同時にネームサーバに対して DoS 攻撃を行なう 必要がある場合もあります。このような攻撃を防ぐには、接続を暗号化して おき、接続するホスト間で認証情報を交換するように設計してください。

1.1.2.6 ワーム

ワームはウイルスとしばしば混同されますが、これらには明確な違いがあります。ウイルスとは異なり、ワームはプログラムを生存させるため、特定のホストに 感染するような必要はありません。その代わり、ネットワーク構造内にできるだけ素早く広がるように特化しています。Ramen や Lion, Adore のように過去 に存在していたワームでは、bind8 や lprNG のようなサーバプログラム内の 既知のセキュリティホールを利用して広がっていました。ワームを防ぐ方法は 比較的簡単なもので、セキュリティホールが見つかったからワームが広がる までにある程度の時間があ

ば、対象のプログラムの更新版が作成されている はずです。そのため、管理者がシステムに対して更新版をインストールすれば、解決することが可能です。

1.2 いくつかのセキュリティ関連のヒント

セキュリティの問題を十分に処理するため、いくつかの要件を監視しておくことが重要です。基本的なセキュリティ要件を満たすため、下記の一覧に 示されたルールをご確認ください：

- セキュリティアナウンスで推奨されている更新パッケージを、できる限り素早く取得してインストールすること。
- 最新のセキュリティ問題について常に最新情報を取得しておくこと (いずれも英語による情報提供であることに注意してください):
 - opensuse-security-announce@opensuse.org は SUSE によるセキュリティ通知のメーリングリストです。このメーリングリストは パッケージの更新を知らせる直接的な情報源であるほか、活動中の 貢献者と SUSE セキュリティチームが含まれています。このメーリング リストを購読するには、<http://ja.opensuse.org/Communicate/Mailinglists> をお読みください。
 - SUSE のセキュリティアドバイザリはニュースフィードの形式で公開されている情報です。http://www.novell.com/linux/security/suse_security.xml をお読みください。
 - bugtraq@securityfocus.com は世界中でもっともよく知られた セキュリティ関連のメーリングリストです。このメーリングリストには 1 日あたり 15 から 20 程度の投稿が行なわれます。詳しくは <http://www.securityfocus.com> をお読みください。
- 興味のあるセキュリティ関連の問題について、セキュリティ用のメーリング リスト opensuse-security@opensuse.org で議論を行なうこと。
- それぞれの作業に対して、最低限の権限で実施するルールに従うこと。特に 日々の作業を root では実施 しないこと。これにより、誤操作による「カッコウの卵 (cuckoo eggs)」やウイルスなどから身を守ることができます。
- 可能な限り、リモートのマシンで作業を行なうにあたっては通信の暗号化を行なうこと。telnet, ftp, rsh, rlogin などのコマンドを 利用する作業は、ssh (Secure SHell) を利用して 代替できます。

- IP アドレスだけをベースにした認証を避けること。
 - 最も重要なネットワーク関連のパッケージ (bind, postfix, ssh など) について、最新の状態を維持し、これらのプログラムに対する新バージョンのアナウンスを受けるため、関連するメーリングリストを購読すること。これはローカル セキュリティに関連するソフトウェアでも同様です。
 - /etc/permissions ファイルを修正し、ファイルに設定 されるパーミッションをセキュリティ用に最適化することで、システムの セキュリティを高めること。setuid ビットをプログラムから取り除く場合は、それに依存していた作業ができなくなることに注意してください。それでも setuid ビットを取り除くことで、潜在的なセキュリティリスクを抑えることに つながります。同様に誰にでも書き込むことのできるディレクトリやファイル についても注意してください。
 - お使いのサーバが正しく動作するのに絶対に必要なものを除き、すべての ネットワークサービスを停止すること。これによりお使いのシステムをより安全 にすることができます。開かれているポート (ソケットが LISTEN (受付) 状態 になっているもの) を調べるには、netstat というコマンドを 使用します。このコマンドにオプションを付けて、netstat -ap または netstat -anp として実行してください。-p オプションは、ポートを占有 しているプロセス名を表示するためのオプションです。
- netstat の結果は、同じことをホストの外側から行なうこと のできるポートスキャンの結果と比較してください。ポートスキャンは nmap というソフトウェアを利用することで行なうことが できます。これはお使いのマシンで開いているポートを調査 することができる だけでなく、そのポートの裏側にどのようなサービスが存在しているのかを 調べることもできます。しかしながら、ポートスキャンは攻撃的な行為 でもあることから、そのマシンを管理している責任者に対して明示的な許可を受けた 場合以外は、使用してはなりません。最後に、TCP ポートだけでなく UDP の ポートを調べることも重要であることに注意してください (それぞれ -sS と -sU のオプションで調べるこ とができます)。
- お使いのシステムでファイルの一貫性を信頼できる方法で監視する場合は、openSUSE に用意されている AIDE (Advanced Intrusion Detection Environment) というプログラムを利用すること。AIDE で作成したデータベースを暗号化しておき、データベースの改ざん を防止しておけば安全を保つことができます。また、データベースはお使いの マシンとは別の場所で保管し、ネットワーク接続などのない外部メディアに 置いておくことをお勧めします。
 - サードパーティ製のソフトウェアをインストールする場合は、特に注意して 取り扱うこと。これは悪意を持った攻撃者が、セキュリティソフトウェアの tar アーカイブ

などにトロイの木馬となるソフトウェアを混入させたりする場合も考えられるためです。その事例では素早く検出することができましたが、バイナリパッケージをインストールする場合は、ダウンロード元のサイトについて一切の疑念がないことを確認してください。

SUSE が提供する RPM パッケージには、gpg での署名が付けられています。SUSE で署名に使用する鍵は下記のとおりです:

```
ID:9C800ACA 2000-10-19 SUSE Package Signing Key <build@suse.de>
Key fingerprint = 79C1 79B2 E1C8 20C1 890F 9994 A84E DAE8 9C80 0ACA
```

`rpm --checksig package.rpm` コマンドを使用すると、インストールされていないパッケージに対してチェックサムと署名の確認を行なうことができます。また、上記の鍵は CD の 1 枚目に含まれているほか、世界中の鍵サーバにも登録されています。

- 定期的にユーザとシステムのファイルのバックアップを確認すること。バックアップが正しく書き戻して動作することを確認していない場合、そのバックアップは意味を 為さない場合があります。
- ログファイルを確認すること。可能であれば小さなスクリプトを作成して、怪しい項目を検索するようにすること。具体的にどのように作成して検索するのは、どの項目を怪しいと判断するのか、という経験に基づくものであるため、お使いの環境次第ではあります。
- `tcp_wrapper` を利用して、お使いのマシンで動作しているそれぞれのサービスに対し、アクセス制限を施すこと。これにより、どの IP アドレスからサービスに接続できるのかを制御することができます。`tcp_wrapper` について、詳しくは `tcpd` と `hosts_access` のマニュアルページ (`man 8 tcpd`, `man hosts_access`) をお読みください。
- `SuSEfirewall` を利用して、`tcpd` (`tcp_wrapper`) が提供するセキュリティをさらに拡張すること。
- セキュリティの要素に冗長性を持たせること。メッセージが何も表示されないよりは、二度表示されたほうがまだ適切であるためです。
- ディスクへのサスペンドを使用している場合は、`configure-suspend-encryption.sh` スクリプトを利用して、サスペンド時のイメージを暗号化するように設定すること。このプログラムは暗号 鍵を作成してそれを `/etc/suspend.key` にコピーするほか、サスペンド時のイメージを暗号化するよう `/etc/suspend.conf` を修正します。

1.3 セキュリティ問題の報告先

セキュリティ関連の問題が発見された場合は、まず更新パッケージが存在しないかどうかを確認してから、電子メールにて security@suse.de 宛に 英語 で連絡を取ってください。なお、このときには 問題の説明と、対象のパッケージのバージョン番号を明記してください。SUSE ではできる限り素早く応答するようにしております。また、pgp による暗号化を 施してメールを送信することをお勧めします。SUSE の暗号鍵は下記のとおりです:

```
ID:3D25D3D9 1999-03-06 SUSE Security Team <security@suse.de>  
Key fingerprint = 73 5F 2E 99 DF DB 94 C4 8F 5A A3 AE AF 22 F2 D5
```

この鍵は <http://www.suse.com/support/security/contact.html> からダウンロード可能です。

パート I. 認証

PAM を利用した認証

Linux では認証処理に PAM (Pluggable Authentication Modules) を利用し、ユーザのアプリケーションの間を仲介します。PAM モジュールは様々なシステム 向けに提供されているため、任意のアプリケーションから要求を送ることができるようになっています。本章では、モジュール型の PAM がどのように 動作するのかと、それをどのように設定すべきかについて述べています。

2.1 PAM とは

システム管理者やプログラマは、システムの特定箇所に対してアクセス制限を行なったり、アプリケーションの特定機能を制限したりしたいと考えます。PAM 無しでは、アプリケーションは LDAP や Samba, Kerberos など、新しい 認証方法が現われるたびにそれらに対応しなければなりません。このような 対応作業は時間がかかるばかりか、エラーを引き起こす可能性の高いものです。このような手間を克服する方法としては、認証を実施するアプリケーションを 独立させ、中央でモジュールを管理することで認証を委任する方法があります。新しく対応すべき認証方法が現われても、必要な *PAM モジュール* を作成して設定することで、プログラム側から利用できるようになります。

PAM で使用する用語は下記のとおりです:

- *PAM モジュール* とは共有モジュールの集合体で、それぞれ特定の認証方法に対応するためのものです。
- *モジュールスタック* とは、1 つ以上の PAM モジュール のことを指します。

- PAM 対応の サービス とは、モジュールスタックや PAM モジュールを使用して認証を行なうサービスです。通常、サービスは login や su など、関連するアプリケーションの名前を使用します。なお、サービス名で other (その他) とは、既定のルールを指定するための 予約語 です。
- モジュールのパラメータ とは、特定の PAM モジュール の実行の際に指定されるパラメータを指します。
- 結果 とは、特定の PAM モジュールの実行結果を指す ものです。結果が正の値であった場合は、次の PAM モジュールを実行します。結果が負の値であった場合は、設定内容によって「特に何もせず続行する」動作をしたり、「即時に終了する」動作をしたりします。

2.2 PAM の設定ファイルの構成

PAM は 2 種類の方法で設定することができます:

ファイルベースの設定 (/etc/pam.conf)

各サービスに対する設定を /etc/pam.conf 内に 保管します。この方法はメンテナンスと有用性の理由から、openSUSE では使用していません。

ディレクトリベースの設定 (/etc/pam.d/)

PAM の仕組みを使用するそれぞれのサービス (プログラム) に対して、/etc/pam.d/ ディレクトリ内に別々の設定ファイルを用意して使用します。たとえば sshd サービスに対する設定は、/etc/pam.d/sshd ファイルで設定します。

/etc/pam.d/ ディレクトリ内にあるファイルは、認証に使用する PAM モジュールを定義するためのものです。各ファイルは複数の 行から構成され、それらはそれぞれサービスを定義するためのものです。また、各行は最大で 4 列から構成されます:

```
種類
制御
モジュールパス
モジュールパラメータ
```

それぞれは下記のような意味を持ちます:

種類

サービスの種類を定義します。PAM モジュールはスタック型で処理され、異なるモジュールの種類はそれぞれ別々の用途で使用します。たとえば、1 つのモ

ジュールではパスワードのチェックを、もう 1 つのモジュール ではシステムのアクセス元のチェックを、そしてさらにもう 1 つはユーザ 固有の設定を読み込んだりします。PAM では下記の 4 種類のモジュール が存在しています:

auth

ユーザの正当性を確認するモジュールです。従来はパスワードの 問い合わせなどに使用していたものを指します。ただし、正当性の 確認は小型のチップや生体認証 (たとえば指紋や虹彩による認証) である場合もあります。

account

この種類のモジュールは、要求するサービスに対してユーザに一般的な 許可があるかどうかを確認します。たとえば期限の切れたアカウント では、誰もログインできないようにする仕組みなどがあります。

password

この種類のモジュールは、認証に使用する情報 (一般にはパスワード) を変更するためのものです。

session

この種類のモジュールは、ユーザセッションの管理と設定を行ないます。ログインのための認証が行なわれると、その最初と最後に起動され、ユーザ 固有の環境 (メールアカウント、ホームディレクトリ、システム 制限など) を設定します。

制御

ここでは PAM モジュールの振る舞いを設定します。それぞれの モジュールに対して、下記のような制御フラグを設定することができます:

required

このフラグが設定されているモジュールは、認証を続けるのに必ず成功しなければならないものを意味します。required フラグの設定されたモジュールが失敗すると、認証失敗を示すメッセージを ユーザに表示する前に、同一のフラグが設定されている全てのモジュールを 処理します。

requisite

このフラグが設定されているモジュールも、認証を続けるのに必ず成功しなければならないものを意味します。ここまでは required フラグと同じですが、このフラグが設定 されてモジュールが失敗すると、ユーザに対しては即時に応答が表示され、その他のモジュールは処理されなくなります。成功した場合は、required フラグと同じで、その他のモジュールが 処理され

ます。requisite フラグは、認証を正しく 行なうために必要な、特定条件のチェックなどに使用されます。

sufficient

このフラグが設定されているモジュールが成功すると、それまでに required フラグの設定されたモジュールが 失敗していない限り、要求元のアプリケーションには即時に成功のメッセージが 返却され、その他のモジュールは処理されなくなります。モジュールが 失敗したときには特に直接的な影響は発生しません。そのまま後続の モジュールを決められた順序で処理します。

optional

このフラグが設定されているモジュールは、成功した場合も失敗した場合も、特に直接的な影響は発生しなくなります。このフラグは、単純にメッセージを表示 (たとえばユーザに対してメールが到着している旨を示すメッセージを表示するなど) し、特に何もその後の処理に影響を与えないものなどに 使用します。

include

このフラグが設定されているモジュールは、パラメータとして指定されたファイルをこの場所に挿入して実行します。

モジュールパス

PAM モジュールの完全なファイル名を指定します。対象の PAM モジュールが既定のディレクトリである /lib/security (openSUSE® で対応している 64 ビットプラットフォームの場合は /lib64/security) ディレクトリ内にある場合は、ファイル名だけを指定することもできます。

モジュールパラメータ

スペース区切りの一覧形式で、PAM モジュールに与えるパラメータを指定します。たとえば debug (デバッグを有効化する) や nullok (パスワードに何も指定しないことを許可する) などがあります。

これらに加えて、/etc/security ディレクトリ以下には PAM モジュールに対するグローバルな設定ファイルも存在しています。これらは 対象のモジュールに対する正確な動作を指定するものです (たとえば pam_env.conf や time.conf などがあります)。PAM モジュールを利用するアプリケーションが実際に PAM の機能を呼び出す際、これらの設定ファイルを利用してアプリケーションに結果 を返却します。

また、PAM モジュールの作成とメンテナンスを簡略化するため、それぞれ auth, account, password, session の各モジュールの 種類に対して、既定の設定ファイ

ルが用意されています。これらはそれぞれのアプリケーションの PAM 設定を経由して読み出すもので、common-* 内でのグローバルな PAM 設定モジュールへの更新は、それぞれ個別の PAM 設定ファイルを更新することなく反映させることができます。

なお、グローバルな PAM の設定ファイルは、pam-config と呼ばれるツールを利用してメンテナンスします。このツールは設定に対して新しいモジュールを追加したり、既存のモジュールの設定を変更したり、モジュール (またはオプション) を設定から削除したりすることを自動で行なうことができます。これにより、PAM の設定に対して手作業で作業を行なうことが最小限に抑えられるほか、場合によっては全く行なう必要がなくなります。

注記: 64 ビットと 32 ビットの混在インストールについて

64 ビット版のオペレーティングシステムをお使いの場合は、32 ビットのアプリケーションに対しても PAM の機能を提供することもできます。この場合は、PAM モジュールについて両方のバージョンをインストールしてください。

2.3 sshd の PAM 設定

sshd に対して、下記のような PAM 設定を利用する場合を考えてみます:

例 2.1 sshd 向けの PAM 設定 (/etc/pam.d/sshd)

```
#%PAM-1.0
auth      requisite      pam_nologin.so ①
auth      include        common-auth ③
account   requisite      pam_nologin.so ②
account   include        common-account ③
password  include        common-password ③
session   required       pam_loginuid.so ④
session   include        common-session ③
```

- ① この設定ファイルが、PAM 1.0 のバージョン向けのものであることを宣言しています。これは単純に慣習上のもですが、将来バージョンをチェックする際に利用することができるものです。
- ② まずは /etc/nologin ファイルが存在しているかどうかをチェックします。存在した場合は、root 以外のユーザはログインできなくなります。
- ③ それぞれ 4 種類のモジュールに対する設定ファイルを参照します: common-auth, common-account, common-password, common-session。それぞれのファイルには、その種類に対する既定の設定が含まれています。

- ④ 認証を行なったプロセスに対して、ログイン UID のプロセス属性を 設定します。

それぞれの PAM モジュールに対して個別にモジュールを追加するのではなく、設定ファイルを取り込む方法で行なっていることにより、管理者が既定値を変更した場合でも自動的に更新版の PAM 設定を利用できるようになります。従来は PAM に対して何らかの変更があった場合や、新しいアプリケーション がインストールされた場合、全てのアプリケーションに対する全ての設定 ファイルを調整しなければなりませんでした。現在、PAM の設定は中央の 設定ファイルで管理され、全ての変更は各サービスの PAM 設定に自動的に 反映されます。

最初の取り込み指定 (common-auth) では、2 種類の auth モジュールを呼び出しています: pam_env.so, pam_unix2.so 詳しくは 例2.2「auth セクションの既定の設定 (common-auth)」(22 ページ) をお読みください。

例 2.2 auth セクションの既定の設定 (common-auth)

```
auth    required    pam_env.so      ①
auth    required    pam_unix2.so   ②
```

- ① pam_env.so は /etc/security/pam_env.conf ファイルを読み込み、このファイルに指定された環境変数を設定します。これは、pam_env ではログインの 実行元を認識していることから、DISPLAY 変数を 正しい値にするために使用しています。
- ② pam_unix2 はユーザのログイン 情報とパスワードを、/etc/passwd と /etc/shadow ファイルで確認します。

auth の全ての処理は、sshd がログインの成功可否に に関する情報を受け取る前に処理されます。また、required の制御フラグが設定された全てのモジュールは、sshd が成功メッセージを 受け取る前に処理が成功しなければなりません。いずれかのモジュールで 成功しなかった場合でも各モジュールはそれぞれ処理され、最終的に sshd に対して失敗が通知されます。

auth の全ての処理が成功すると、その他の include ステートメントが処理されます。この場合は 例2.3「account セクションの既定の設定 (common-account)」(22 ページ) に続くことになります。common-account には 1 つのモジュール pam_unix2 だけが書かれています。pam_unix2 がユーザが存在する旨の結果を返却すると、sshd に成功を通知して次のモジュール (password; 例 2.4「password セクションの既定の設定 (common-password)」(23 ページ)) に移動します。

例 2.3 account セクションの既定の設定 (common-account)

```
account required    pam_unix2.so
```

例 2.4 password セクションの既定の設定 (common-password)

password requisite	pam_pwcheck.so	nullok cracklib
password required	pam_unix2.so	nullok use_authtok

sshd の PAM 設定でも同様に、既定の設定では password モジュールの設定に対して、common-password を取り込む ように宣言しています。制御フラグが requisite と required に設定されているモジュールは、アプリケーションが 認証情報を変更する際に必ず成功しなければなりません。

なお、パスワードなどの認証情報を変更する際には、セキュリティチェックを 受ける必要があります。このようなチェック機能は pam_pwcheck モジュールが提供しています。pam_unix2 モジュールは pam_pwcheck による古いパスワードから新しいパスワード への移行作業が完了した後に使用されているため、ユーザ側ではパスワード を 変更した後に認証をやり直すことがないようにになっています。このような処理 により、pam_pwcheck が実施するチェックを妨げる ことがないようにしています。また、account や auth で期限切れのパスワードを警告するよう設定して いる場合 も、password のモジュールが使用されます。

例 2.5 session セクションの既定の設定 (common-session)

session required	pam_limits.so
session required	pam_unix2.so
session optional	pam_umask.so

最後のステップとして、session の種類のモジュール群 (common-session ファイル 内に書かれているもの) が 呼び出され、該当するユーザに対するセッション設定を 行ないます。まず pam_limits モジュールは /etc/security/limits.conf ファイル を読み込んで、特定のシステム資源の使用を制限します。また、ここでも pam_unix2 モジュールを利用しています。さらに pam_umask モジュールは、ファイル作成時の パーミッション のマスクを設定することができます。このモジュールには optional フ ラグが設定されているため、このモジュールの 実行が失敗してもセッションは中断 されず、そのまま続行されます。なお、session モジュールは、ユーザがログアウトす る 際にもう一度呼び出されます。

2.4 PAM モジュールの設定

PAM モジュールの中には設定可能なものがあります。設定ファイルは / etc/security 内に存在しています。この章では、sshd の例で使用している pam_env.conf と limits.conf について、設定ファイルを説明しています。

2.4.1 pam_env.conf

pam_env.conf はユーザに対して標準化した環境を提供するために利用するもので、pam_env モジュールが呼び出された際に読み込まれます。このファイルでは、環境変数は下記の書式で記述します:

変数 [DEFAULT=値] [OVERRIDE=値]

変数

設定する環境変数の名前を指定します。

[DEFAULT=<値>]

管理者が設定したい既定の 値 を指定します。

[OVERRIDE=<値>]

既に設定されている値を上書きするための設定で、pam_env が問い合わせ設定を行いません。

pam_env をどのように使用することができるのかの例として、DISPLAY 変数の設定を説明します。この変数は リモートログインを行なった際に変更しなければならないもので、例2.6「pam_env.conf」(24 ページ)で説明しています。

例 2.6 pam_env.conf

```
REMOTEHOST    DEFAULT=localhost OVERRIDE=@{PAM_RHOST}  
DISPLAY       DEFAULT=${REMOTEHOST}:0.0 OVERRIDE=${DISPLAY}
```

最初の行では REMOTEHOST という変数の値を設定しています。この値は pam_env がその他の値を設定できない場合に、既定値として localhost が設定されるようになっています。DISPLAY 変数では REMOTEHOST の値を利用して 設定するようになっています。詳しくは /etc/security/pam_env.conf 内のコメント (英語) をお読みください。

2.4.2 pam_mount.conf

pam_mount は、中央のファイルサーバで全ユーザの ホームディレクトリを提供しているような環境で利用するもので、ログイン処理 中に対象のユーザのホームディレクトリをマウントし、ログアウト処理中にマウント を解除します。このような方法をとること、全ユーザに対するホームディレクトリ を提供するの、/home ディレクトリ全体を

マウントしなくて済むようになります。その代わり、ログインしているユーザのホームディレクトリだけがマウントされます。

pam_mount をインストールすると、/etc/security ディレクトリ内に pam_mount.conf.xml という雛型ファイルが配置されます。それぞれの要素に対する説明は、man 5 pam_mount.conf で表示されるマニュアルページをお読みください。

この機能に対する基本的な設定は、YaST で行なうことができます。ファイルサーバを追加するには、YaST から ネットワークサービス > *Windows* ドメインメンバーシップ > *熟練者向け設定* を選択してください。詳しくは 項「クライアントの設定」(第19章 *Samba*, ↑リファレンス)をお読みください。

2.4.3 limits.conf

limits.conf はユーザやグループに対してシステム制限を設定するためのファイルで、pam_limits モジュールが読み込みを行ないます。このファイルでは、ハードリミットと呼ばれる超えることのできない値と、ソフトリミットと呼ばれる一時的な超過を許す値をそれぞれ設定します。文法とオプションについて、詳しくは /etc/security/limits.conf 内のコメント (英語)をお読みください。

2.5 pam-config を利用した PAM の設定

pam-config ツールは、PAM のグローバル設定ファイル (/etc/pam.d/common-*pc) の設定を支援するほか、いくつかの選択したアプリケーション設定を操作することのできるツールです。対応しているモジュールの一覧については、pam-config --list-modules コマンドを実行すると表示することができます。PAM の設定ファイルを管理する際には、pam-config コマンドを利用して、PAM の設定に新しいモジュールを追加したり、修正や削除を行なったりしてください。PAM のグローバル設定ファイルを変更しておくことで、個別のアプリケーションに対する設定を手作業で修正する必要がなくなります。

pam-config のシンプルな用途としては、下記のようなものがあります：

- 1 **Unix スタイルの設定を新規に自動生成する** pam-config を利用することで、後から設定することのできるシンプルな設定ファイルを作成することができます。

具体的には `pam-config --create` コマンドを利用して、シンプルな UNIX 認証の設定を作成することができます。既に設定ファイルが存在していて、`pam-config` で管理されていないものであった場合は、バックアップファイルが `*.pam-config-backup` として保存されたあと、設定が上書きされます。

- 2 **新しい認証方法を追加する** 新しい認証方法 (たとえば LDAP) を PAM モジュールに追加したい場合は、`pam-config --add --ldap` のようにして実行します。LDAP は全ての `common-*-pc` PAM 設定ファイル に対して追加されます。
- 3 **テスト用にデバッグを有効にする** 新しい認証方法が想定通りに動作していることを確認するには、PAM 関連の 操作に対してデバッグ機能を有効にする方法があります。`pam-config --add --ldap-debug` と実行すると、LDAP 関連の PAM 操作に対してデバッグ機能が有効に設定されます。デバッグ出力 は、`/var/log/messages` に出力されます。
- 4 **設定を確認する** 新しい PAM の設定を最終的に反映させる前に、追加する必要のあるオプション が全て含まれていることを確認することができます。`pam-config --query --モジュール` のように実行すると、指定した PAM モジュール に対する種類とオプションの 一覧が表示されます。
- 5 **デバッグ機能を無効にする** 全ての設定が完了したあとは、十分な性能を確保するため、最終的に設定から デバッグオプションを取り除きます。`pam-config --delete --ldap-debug` のように実行すると、LDAP 認証に対するデバッグ 機能を無効にすることができます。LDAP 以外のモジュールでデバッグ機能を 有効にしていた場合も、似たようなコマンドで無効にすることができます。

`pam-config` コマンドについて詳しい情報と、利用可能な オプションについては、`pam-config(8)` で表示される マニュアルページをお読みください。

2.6 手作業での PAM 設定

PAM の設定ファイルを手作業で作成したり管理したりしたい場合は、これらの ファイルに対して `pam-config` を無効化していることを 確認する必要があります。

`pam-config --create` コマンドを利用して、何もない状態 から PAM の設定ファイルを作成すると、`common-*` から `common-*-pc` に対するシンボリックリンクが作成されます。`pam-config` は `common-*-pc` の設定ファイルのみを修正 する仕組みであるため、これらのシンボリックリンクを削除すると、`pam-config` は `common-*-pc` ファイルのみを操作する仕組みであり、これらのファイルはシンボリックリンク無しでは効果の無いものであるため、`pam-config` を簡単に無効化することができます。

2.7 さらなる情報

pam-doc パッケージをインストールすることで、`/usr/share/doc/packages/pam` ディレクトリ内に 下記のような様々な文書が配置されます (いずれも英語です):

README

上記のディレクトリの直下には、それぞれの PAM モジュール向けの `modules` サブディレクトリが存在していて、その中には README ファイルが用意されています。

The Linux-PAM System Administrators' Guide

この文書には、システム管理者が PAM について知っておくべき全ての項目が含まれています。たとえば PAM の設定ファイルの文法から、セキュリティまわりの考慮事項など、幅の広い範囲が含まれます。

The Linux-PAM Module Writers' Manual

この文書には、管理者の視点からのトピックが含まれていて、たとえば標準に準拠した PAM モジュールの書き方などが記されています。

The Linux-PAM Application Developers' Guide

この文書には、PAM ライブラリを使用するアプリケーションの開発者に 対する 全ての内容が含まれています。

The PAM Manual Pages

一般的な PAM の情報や、全てのコンポーネントの機能概要を提供する マニュアルページなどが含まれています。

NIS の使用

UNIX システムを共有のリソースとしてネットワーク上に多数配置すると、ネットワーク内で全てのユーザとグループの識別子を一致させる必要が生じて きます。ネットワークはユーザに対して透過的であるべきもので、実際に使用 しているマシンがどれであるかに関わらず、ユーザに対しては同じ環境を提供 するのが理想的です。このような仕組みは、NIS や NFS のサービスを利用することで解決することができます。NFS はネットワーク上でファイルシステムを 共有するためのもので、こちらについては 第18章 *NFS* でのファイル共有 (↑リファレンス) で説明しています。

NIS (Network Information Service) はデータベースのようなサービスで、ネットワーク経由で `/etc/passwd`, `/etc/shadow`, `/etc/group` の各ファイルにアクセスする機能を提供します。NIS はそれ以外の用途 (`/etc/hosts` や `/etc/services` のようなファイルの中身を提供する用途) にも使用できますが、本章ではこれらの仕組みについては言及していません。また、NIS は *YP* と呼ばれる場合もあります。これはネットワーク に対する「イエローページ」(いわゆる電話帳) として機能 するためです。

3.1 NIS サーバの設定

NIS の情報をネットワーク経由で配布するには、単一のサーバ (マスター) から全てのクライアントに対してサービスを 提供するか、もしくは NIS のスレーブサーバを用意して、マスターサーバからの 情報を中継させてクライアントに提供する方法があります。

- 単一の NIS サーバをネットワーク内に構築するには、3.1.1項「NIS マスターサーバの設定」(30 ページ) の手順を実施します。

- NIS のマスターサーバからスレーブサーバにデータを公開する必要がある場合は、まず 3.1.1 項「NIS マスターサーバの設定」(30 ページ) の手順でマスターサーバを立ち上げてから、3.1.2 項「NIS スレーブサーバの設定」(35 ページ) の手順でスレーブサーバをサブネット内に立ち上げます。

3.1.1 NIS マスターサーバの設定

NIS マスターサーバをお使いのネットワーク内で立ち上げるには、下記の手順を実施します:

- 1 まずは YaST の NIS サーバ設定モジュールがインストール済みであるかどうかを確認します。YaST を起動して **ソフトウェア > ソフトウェア管理** を選択します。ソフトウェア管理モジュールを起動したら、`yast2-nis-server` パッケージを検索し、インストールされていなければインストールを行います。
- 2 *YaST > ネットワークサービス > NIS サーバ* を起動します。
- 3 お使いのネットワーク内で 1 台だけの NIS サーバを構築する場合、または NIS スレーブサーバに対してサービスを提供するマスターサーバとして構築する場合は、*NIS マスターサーバをインストールして設定する* を選択します。すると YaST は必要なパッケージをインストールします。

ヒント

NIS サーバソフトウェアが既にインストールされている場合は、*NIS マスターサーバを作成する* を選択し、NIS マスターサーバの作成を行ないます。

図 3.1 NIS サーバの設定



4 まずは NIS の基本的なオプションを設定します:

4a まずは NIS のドメイン名を入力します。

4b 次に NIS クライアントとしても使用するかどうか (ユーザに対して、NIS サーバが提供するログインとデータアクセスを許可するかどうか) を選択します。NIS クライアントとしても使用する場合は *このホストは NIS クライアントでもある* にチェックを入れます。

4c NIS サーバがマスターサーバとして動作し、他のサブネットの NIS スレーブサーバに情報を提供する場合は、*アクティブな NIS スレーブサーバが存在する* にチェックを入れます。

高速マップ配布 は、*アクティブな NIS スレーブサーバが存在する* にチェックを入れた場合のみ意味を持つ項目です。これを選択すると、スレーブに対するマップ配布の転送速度を上げることができます。

4d ネットワーク内のユーザ (ローカルユーザのほか、NIS サーバで管理されているユーザも含まれます) に対して、NIS サーバ上に登録されている自分自身のパスワードを変更できるようにする (`yppasswd` コマンドを使用して行ないます) には、*パスワードの変更を許可する* にチェックを入れます。これにチェックを入れると、*GECOS 項目の変更を許可する* と *ログイン*

ンシェルの変更を許可する が選択できるようになります。「GECOS」とは ypchfn コマンド を利用して名前や住所の情報を変更できる機能のことを指します。また「シェル」は、ypchsh コマンドで既定の シェルを変更 (たとえば bash から sh など) できる機能のことを指します。シェルについては、/etc/shells で定義されている シェル内のいずれかに変更することができます。

- 4e** YaST で NIS のサービスに合わせてファイアウォールの設定を調整するには、ファイアウォールでポートを開く を選択します。

図 3.2 マスターサーバの設定

マスターサーバのセットアップ

NIS ドメイン名 (D)

example.com

☐ このホストは NIS クライアントでもある (C)

☒ アクティブな NIS スレーブサーバが存在する (E)

☐ 高速マップ配布 (rpc.ypxfrd) (F)

パスワードの変更

☐ パスワードの変更を許可する (P)

☐ GECOS 項目の変更を許可する (G)

☐ ログインシェルの変更を許可する (S)

☐ ファイアウォールでポートを開く (F) ファイアウォールの詳細 (D)...

ファイアウォールで該当のポートは閉じられています

その他のグローバル設定 (G)...

ヘルプ 中止 (R) 戻る (B) 次へ (N)

- 4f** 次へ を押してダイアログを終了するか、もしくは *その他のグローバル設定* を押してさらなる設定を行ないます。

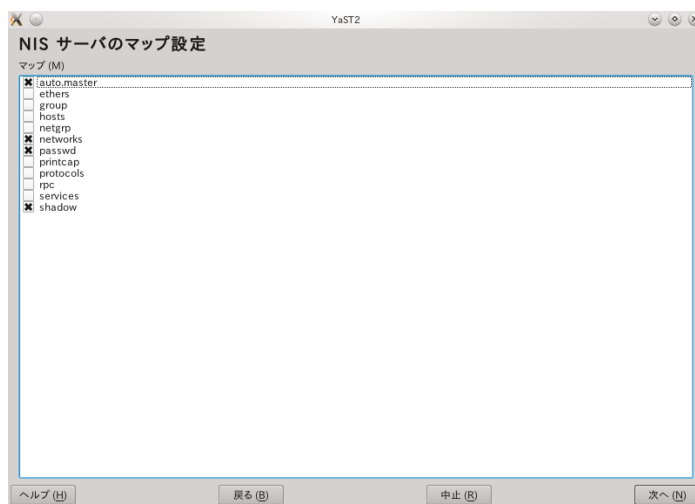
その他のグローバル設定 には、NIS サーバの情報源となるディレクトリ (既定では /etc) を変更することができるほか、パスワードの併合も設定することができます。この設定 は、システムの認証ファイル (/etc/passwd, /etc/shadow, /etc/group) からユーザデータベースを作成する場合は、選択を行なう必要のある項目 です。また、NIS で提供する最小のユーザ ID とグループ ID もここで 設定します。OK を押すと元の画面に戻ることができます。

図 3.3 NIS サーバ向けのディレクトリ変更とファイル同期



- 5 前の画面で **アクティブな NIS スレーブサーバが存在する** を選択していた場合は、ここでスレーブとして使用するホストのホスト名を 入力して **次へ** を押します。選択していない場合、スレーブを設定する画面は表示されません。
- 6 続けてデータベース設定のダイアログが表示されます。ここでは **NIS サーバマップ設定** が表示され、NIS サーバから クライアントに対して、転送すべきデータベースの箇所を指定します。通常は既定値のまま変更する必要はありません。**次へ** を押して進めてください。
- 7 なお、この画面ではマップとして提供するものを選択し、**次へ** を押してください。

図 3.4 NIS サーバのマップ設定

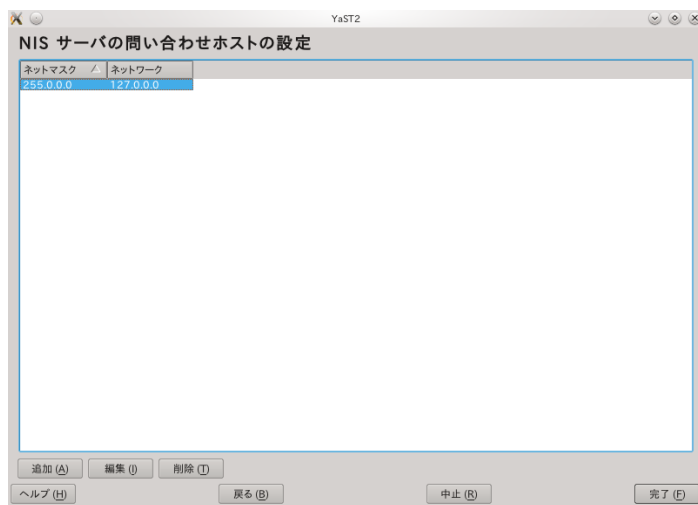


- 8** 次に NIS サーバへの問い合わせを許可するホストを設定します。それぞれ ボタンを押すことで、ホストを追加／編集／削除することができます。通常、ここでは内部ネットワークを設定します。既定では下記の 2 つの項目が設定されています：

255.0.0.0	127.0.0.0
0.0.0.0	0.0.0.0

最初の項目では、NIS サーバ自身からの接続を許可しています。2 つめの項目では、全てのホストからサーバに対して要求を送信できるようにしています。

図 3.5 NIS サーバに対するリクエスト許可の設定



9 最後に **完了** を押すと、設定を保存して終了することができます。

3.1.2 NIS スレーブサーバの設定

お使いのネットワーク内で NIS スレーブサーバを設定するには、下記の手順を実施します：

- 1 YaST > ネットワークサービス > NIS サーバ を起動します。
- 2 NIS スレーブサーバをインストールして設定する を選択し、次へ を押します。

ヒント

NIS サーバソフトウェアが既にインストールされている場合は、NIS スレーブサーバを作成する を選択し、NIS スレーブサーバの作成を行ないます。

- 3 まずは NIS スレーブサーバの基本的な設定を行ないます：

3a まずは NIS ドメインを入力します。

3b マスターサーバのホスト名または IP アドレスを入力します。

3c このサーバに対してユーザのログインを行なわせたい場合は、このホストは *NIS クライアントでもある* にチェックを入れます。

3d YaST で NIS のサービスに合わせてファイアウォールの設定を調整するには、*ファイアウォールでポートを開く* を選択します。

3e *次へ* を押します。

4 次に NIS サーバへの問い合わせを許可するホストを設定します。それぞれ ボタンを押すことで、ホストを追加／編集／削除することができます。全てのネットワークに適用するには、下記の設定を使用してください：

255. 0. 0. 0	127. 0. 0. 0
0. 0. 0. 0	0. 0. 0. 0

最初の項目では、NIS サーバ自身からの接続を許可しています。2 つめの 項目では、全てのホストからサーバに対して要求を送信できるようにしています。

5 最後に *完了* を押すと、設定を保存して終了することができます。

3.2 NIS クライアントの設定

ワークステーション上で NIS を使用するには、下記の手順を実施します：

1 *YaST > ネットワークサービス > NIS クライアント* を起動します。

2 *NIS を使用する* を選択します。

3 まずは NIS ドメインを入力します。これは管理者から提供されたドメイン 名か、DHCP で取得した固定の IP アドレスを入力します。DHCP について、詳しくは第16章 *DHCP* (↑リファレンス) をお読みください。

図 3.6 NIS サーバのドメインとアドレスの設定



- 4 次に NIS サーバを入力します。複数を入力する場合は半角スペースで区切ってください。NIS サーバのアドレスがわからない場合は、**検索** を押すことでメイン内の NIS サーバを検索することができます。これは お使いのネットワークの規模に依存しますが、時間のかかる処理であることに 注意してください。また、指定したサーバが応答しない場合にローカル ネットワークに対して NIS サーバを問い合わせる場合は、**ブロードキャスト** を選択してください。
- 5 インストール環境に合わせて automounter を使用することもできます。これを選択すると、追加のソフトウェアをインストールする場合があります。
- 6 お使いのホストに対して、他のホストからどのサーバを使用しているのかを 知らせたくない場合は、**熟練者設定** を押して **リモートホストに応答する** のチェックを外してください。また、**壊れたサーバ** を選択すると、**非特権ポート** を利用したサーバからの応答を受け取るようになります。詳しくは `man ypbind` をお読みください。
- 7 最後に **完了** を押すと、設定内容を保存して YaST コントロールセンターに戻ることができます。これでお使いのクライアントで NIS を使うことができますようになります。

ディレクトリサービス LDAP

Lightweight Directory Access Protocol (LDAP) は、ディレクトリ構造の情報にアクセスしたり、それらを管理したりするためのプロトコル集です。LDAP は ユーザやグループの管理に利用できるほか、システム設定の管理やアドレスの 管理などに利用できます。本章では、OpenLDAP の動作に関する基礎知識と、YaST を利用した場合の LDAP データの管理方法について述べています。

ネットワークを利用する環境では、重要な情報を構造的に管理し、素早く提供することが必須の要件となります。いわゆる電話帳のようなディレクトリサービスでは、情報をうまく分類し、読みやすく探しやすい形式で保存します。

理想的には中央のサーバのディレクトリ内にデータを保管し、すべてのクライアントに対して、うまく設計されたプロトコルを利用して配布する、という形式を取ることでしょう。また、データが構造化されていることにより、様々なアプリケーション から幅広くアクセスできるようになります。さらに、データの保管を中央に集めることで、管理にかかる手間を省くことができます。LDAP のようなオープンで標準化されたプロトコルを利用することで、数多くのクライアントアプリケーションからこれらの情報にアクセスできます。

本章でディレクトリとは、読み込みや検索の速度や、その効率に重点を置いた データベースのことを指しています：

- 複数のクライアントからできる限り同時並行で読み込むことができるようにするため、更新の同時並行数はとても低く設定されています。また、書き込み それ自身についても、管理者権限を持つごく少数のユーザに対してだけ許可 されています。通常のデータベースでは逆に、短い時間でできるだけ多くの データを受け入れることができるように設計されます。

- 固定のデータとして管理している場合、既存のデータに対する更新はほとんど発生しないことになります。逆に動的なデータで作業を行なっている場合、たとえば銀行口座の取り引きなどの場合は、データの一貫性は重要になります。たとえばある一定額を一方の口座から差し引いて、それを他方の口座に追加するような場合は、資金が途中で消えてしまったりすることがないようにするため、これを 1 つのトランザクションとして扱い、操作を両方とも実施しなければなりません。従来のリレーショナルデータベースではトランザクションの関係一貫性サポートなどの形で、データの一貫性についてとても強力な考慮がありますが、LDAP のディレクトリでは短期間の一貫性欠如は一般に受け入れられるものとして扱われます。LDAP のディレクトリでは、リレーショナルデータベースのように強力な一貫性の要件が存在していないためです。

LDAP のようなディレクトリサービスはその設計上、複雑な更新や問い合わせの仕組みを備えているわけではありません。すべてのアプリケーションからのアクセスが素早く簡単に完了するように作られています。

4.1 LDAP と NIS

Unix システムの管理者は従来、NIS (Network Information Service) を利用してネットワーク内での名前解決とデータ配布を行なっていました。/etc ディレクトリ内にある group, hosts, mail, netgroup, networks, passwd, printcap, protocols, rpc, services の各ファイルには設定データが含まれていて、クライアントに対してネットワーク経由で配布を行なっていました。これらのファイルはシンプルなテキストファイルであるため、特に労を要することなく管理することができていましたが、大規模な環境では構造が複雑化してしまい、難易度を大きく押し上げるようになっていました。また、NIS は Unix プラットフォームにしか対応しておらず、異なる OS が混在するようなネットワークでの集中型管理ツールとしては不適切でした。

NIS とは異なり、LDAP サービスは純粋な Unix ネットワークに制限されるようなことはありません。Windows サーバ (Windows 2000 以降) では LDAP をディレクトリサービスとして利用することができますし、上記のようなアプリケーションも非 Unix システムで対応するようになっていきます。

LDAP の考え方は、集中管理型のデータ構造であればどのようなものにも適用することができるものです。たとえば下記のような使い方が考えられます：

- NIS サービスの代替
- メールの経路制御 (postfix, sendmail)

- メールクライアント (Mozilla, Evolution, Outlook) 向けのアドレス帳
- BIND9 ネームサーバのゾーン定義の管理
- 異種混在型のネットワーク内で使用される Samba のユーザ認証

LDAP は NIS とは異なり拡張性に富んだ構造になっているため、上記に限らず 数多くの用途に使用することができます。また、わかりやすく設計された階層 構造型のデータ構造は、巨大なデータ量の管理を行ないやすくする効果があるばかりか、検索もより容易になります。

4.2 LDAP のディレクトリツリー構造

LDAP サーバがどのように動作するのかと、どのようにデータを保管するのかについて、背景となる知識を得るには、サーバ上でのデータ保持方法のほか、このようなデータ構造をどのように扱えば高速に検索できるのかを理解しておくことが必要不可欠です。LDAP の構築を正しく行なうには、基本的な LDAP 用語についても慣れておく必要があります。この章では、LDAP のディレクトリ構造に関する基本的な構造のほか、LDAP に関する基本的な用語説明を行なっています。既に LDAP に関する基礎知識をお持ちであり、openSUSE における LDAP 環境の構築方法について知りたい場合は、本章は読み飛ばしてかまいません。それぞれ 4.3 項「YaST を利用した LDAP サーバの設定」(43 ページ) や 4.7 項「LDAP サーバの手動設定」(61 ページ) をお読みください。

LDAP ディレクトリはツリー状の構造になっています。ディレクトリ内のすべての項目 (オブジェクトと呼びます) には、このツリー構造内の場所が定義されます。このような構造を *ディレクトリ情報ツリー* (DIT; Directory Information Tree) と呼びます。また、特定の項目に対するツリー構造 上の完全な位置 (略したり相対的に記述したりしないもの) を、*識別名* (DN; Distinguished Name) と呼びます。逆に特定の項目に対して単一の階層だけを示すものを、*相対識別名* (RDN; Relative Distinguished Name) と呼びます。

LDAP のディレクトリツリーにおける関係は、図4.1「LDAP ディレクトリの構造」(41 ページ) の例をご覧ください。

図 4.1 LDAP ディレクトリの構造

上記の図は架空の企業を例にしたディレクトリ構造を示しています。項目は 3 段階の階層から構成されていて、1 つの項目を 1 つの箱で示しています。たとえば Geeko Linux という 架空の従業員に対する、正しい *識別名* (DN) は、この例では

cn=Geeko Linux, ou=doc, dc=example, dc=com という書き方になります。これは階層構造の上位に対する識別名 ou=doc, dc=example, dc=com に対して、相対識別名 cn=Geeko Linux を前に追加したものになります。

DIT 内に保管できるオブジェクトの種類は、スキーマと呼ばれるものに従って決定されます。また、オブジェクトの種類は オブジェクトクラス で決定されます。これは、そのオブジェクトに対してどのような属性を割り当てなければならないのか、もしくは割り当てることができるのかを決めるものです。そのためスキーマは、特定の用途に適合するすべてのオブジェクトクラスや属性に関する定義を含んでいなければならないことになります。スキーマにはいくつか汎用的なもの (RFC 2252, 2256 をお読みください) もあります。また LDAP の RFC でも、いくつか一般的に使用されるスキーマが定義されています (RFC 4519 など)。それ以外にも、様々な用途に合わせたスキーマも存在している (たとえば Samba や NIS の代替など) ほか、独自のスキーマを作成したり、必要であれば複数のスキーマを相互に補完させる目的で使用したりすることもできます。

表4.1「よく使用されるオブジェクトクラスと属性」(42 ページ) には、core.schema と inetorgperson.schema で提供されているオブジェクト クラスについて、その概要と使用例、必要な属性と有効な属性値の一覧が書かれています。

表 4.1 よく使用されるオブジェクトクラスと属性

オブジェクトクラス	意味	項目例	必要な属性
dcObject	<i>domainComponent</i> (ドメインの名前部分)	example	dc
organizationalUnit	<i>organizationalUnit</i> (団体単位)	doc	ou
inetOrgPerson	<i>inetOrgPerson</i> (イントラネットやインターネットの個人関連データ)	Geeko Linux	sn および cn

また、例4.1「schema.core からの抜粋」(42 ページ) では、スキーマからの 抜粋と、その説明が書かれています。

例 4.1 schema.core からの抜粋

attributetype (2.5.4.11 NAME ('ou' 'organizationalUnitName') ❶


```

DESC 'RFC2256: organizational unit this object belongs to' ❷
SUP name ) ❸

...
objectclass ( 2.5.6.5 NAME 'organizationalUnit' ❹
    DESC 'RFC2256: an organizational unit' ❺
    SUP top STRUCTURAL ❻
    MUST ou ❼
    MAY (userPassword $ searchGuide $ seeAlso $ businessCategory ❽
        $ x121Address $ registeredAddress $ destinationIndicator
        $ preferredDeliveryMethod $ telexNumber
        $ teletexTerminalIdentifier $ telephoneNumber
        $ internationaliSDNNumber $ facsimileTelephoneNumber
        $ street $ postOfficeBox $ postalCode $ postalAddress
        $ physicalDeliveryOfficeName
        $ st $ l $ description) )
...

```

属性の種類 `organizationalUnitName` とそれに関連する オブジェクトクラス `organizationalUnit` は、この例の とおりに動作します。

- ❶ 属性の名称と属性に対するユニークな OID (オブジェクト 識別子) (数値)、そして属性の略称です。
- ❷ DESC には、属性に対する概要説明を記述します。ベースとしている RFC の番号などを記述します。
- ❸ SUP には、この属性が属する上位の属性名を記述します。
- ❹ ここから `organizationalUnit` のオブジェクトクラスが 始まります。属性の定義と同じで、OID とオブジェクトクラスの名称を 記述します。
- ❺ オブジェクトクラスに対する概要説明を記述します。
- ❻ SUP top と記述すると、このオブジェクトクラスには 上位の属性名が存在していないことを指定します。
- ❼ MUST の一覧には、`organizationalUnit` という種類を使用した場合に指定しなければならない属性をすべて記述します。
- ❽ MAY の一覧には、このオブジェクトクラスでの指定を許可 する属性をすべて記述します。

スキーマの使用方法について、わかりやすい説明は OpenLDAP のドキュメンテーション (`openldap2-doc`) 内に用意されています。インストール後、`/usr/share/doc/packages/openldap2/adminguide/guide.html` (英語) をお読みください。

4.3 YaST を利用した LDAP サーバの設定

LDAP サーバを設定するには YaST を使用します。LDAP サーバの一般的な 用途としては、ユーザアカウントのデータやメールサーバ／DNS サーバ／DHCP サーバの設定などがあります。

図 4.2 YaST LDAP サーバ設定

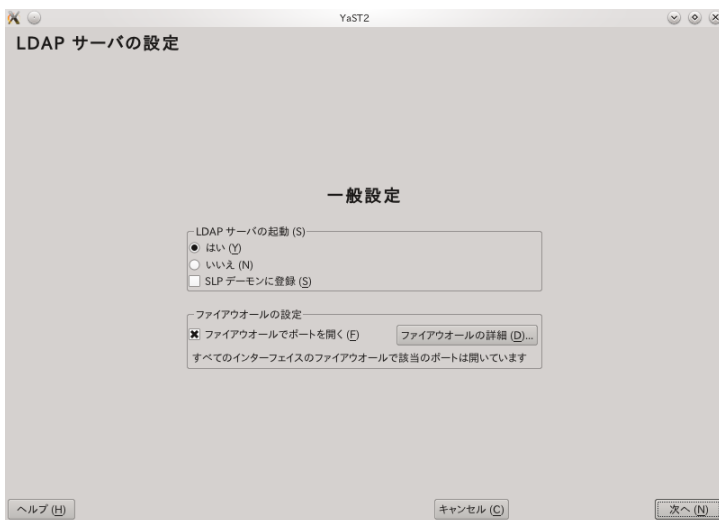


図 4.3 YaST LDAP サーバ新しいデータベース



ユーザアカウントデータを保管するために LDAP サーバを設定するには、まず `yast2-ldap-server` と `openldap2` の各パッケージがインストールされていることを確認してください。その後、下記の手順を実施します:

- 1 YaST を `root` で起動し、ネットワークサービス > *LDAP* サーバ を選択します。すると設定ウィザードが表示されます。
- 2 まずは LDAP サーバの *一般設定* を行ないます。図4.2「YaST LDAP サーバ設定」(44 ページ) をご覧ください:
 - 2a LDAP を起動するように設定します。
 - 2b LDAP サーバを SLP でアナウンスするように設定するには、*SLP デーモンに登録* にチェックを入れます。
 - 2c *ファイアウォールの設定* で必要な設定を行ないます。
 - 2d *次へ* を押します。
- 3 次にサーバの種類を選択します: 単独サーバ, レプリケーション設定 内でのマスターサーバ, レプリカ (スレーブ) サーバの中から選択します。
- 4 さらにセキュリティオプションを設定します (*TLS 設定*) 。

なお、*TLS を有効にする* を選択して、TLS を有効にしておくことを強くお勧めします。詳しくは ステップ 4 (47 ページ) をお読みください。

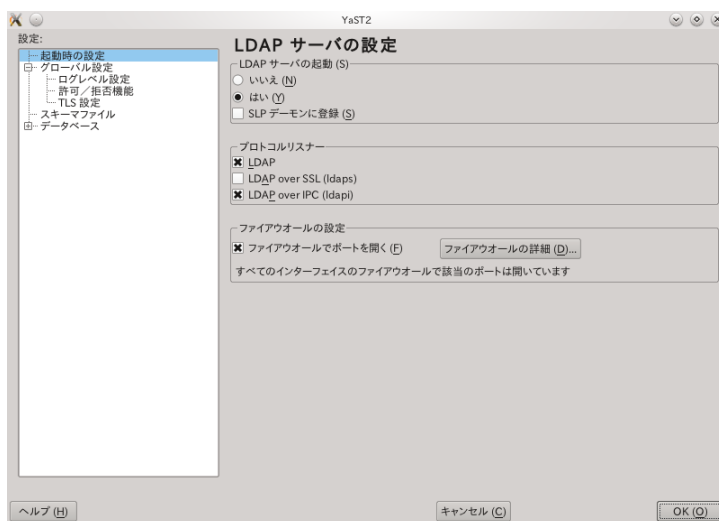
警告: パスワードの暗号化について

TLS による暗号化を行なうと、ネットワーク上にパスワードを流す際、それらを暗号化することになります。このオプションを有効にしない場合は、パスワードが暗号化されずに送信されてしまいます。

また、必要であれば LDAP over SSL と証明書の設定を行ないます。

- 5 さらに *基本的なデータベースの設定* に移動します。ここでは *LDAP 管理者パスワード* とその確認欄に入力を行ない、*次へ* を押します。詳しくは 図4.3「YaST LDAP サーバ新しいデータベース」(44 ページ) をご覧ください。
- 6 最後に *LDAP サーバの設定概要* に表示された内容を確認し、*完了* を押すと設定ウィザードを終了することができます。

図 4.4 YaST LDAP サーバ設定



設定の変更や追加を行なう場合は、LDAP サーバモジュールを再度起動し、左側のペインで **グローバル設定** を選択します。詳しくは 図4.4「YaST LDAP サーバ設定」(46 ページ) をご覧ください:

- 1 **ログレベルの設定** では、LDAP サーバのログ動作に 関する設定 (ログ出力の頻度に関わる設定) を行なうことができます。一覧に表示された項目から、ログに出力したいものを選んでください。選択を増やすほど、ログファイルが大きくなることに注意してください。
- 2 サーバ側でどのような接続を許すのかについては、**機能の許可／拒否** で設定を行ないます。下記の ものを選択することができます:

LDAPv2 バインド要求

このオプションを選択すると、LDAP の旧バージョン (LDAPv2) を使用するクライアントからの接続要求 (バインド要求) を受け付けるようになります。

認証情報が記入されている匿名バインド

通常 LDAP サーバは、認証情報 (DN またはパスワード) に何も記入されていない場合、認証を拒否するようになっています。このオプションを選択すると、DN のないパスワードのみの接続を受け付けるようになります。

DN が記入されていて認証情報のないバインド

このオプションを選択すると、DN は設定されているがパスワードのない (認証情報のない匿名の) 接続を受け付けるようになります。

未認証時の更新

このオプションを選択すると、認証情報のない匿名の更新操作を受け付けるようになります。アクセスは ACL やその他のルールで制限する必要があります。

3 機能の許可／拒否 では、下記のフラグを設定することも できます:

匿名バインド要求の受け入れを禁止する

このオプションを選択すると、匿名でのバインド要求を受け入れないように なります。ただし、これは匿名でのディレクトリアクセスを禁止するわけ ではない ことに注意してください。

簡易バインド認証を無効にする

簡易バインド認証を完全に無効化します。

StartTLS 操作を受け取った際、セッションに対して匿名状態の強要を禁止する

このオプションを選択すると、StartTLS の操作を受け取った際、サーバ側で 認証済みの状態を匿名の状態に戻さないようにします。

認証したあとは StartTLS を禁止する

このオプションを選択すると、既に認証が完了している接続に対しては StartTLS を許可しないようになります。

4 クライアントとサーバの間での暗号化接続を設定するには、*TLS 設定* で行ない ます:

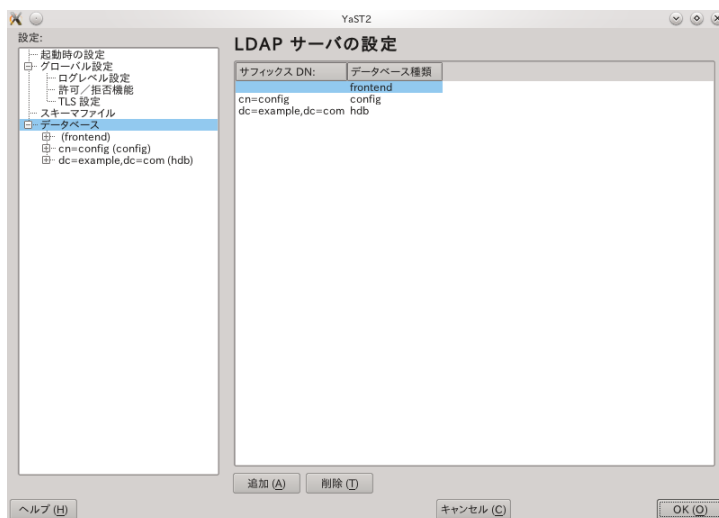
4a クライアントとサーバの間で TLS や SSL による暗号化を有効に するには、*TLS を有効にする* を選択します。

4b *証明書のインポート* で証明書の正確な場所を 指定するか、もしくは *共通サーバ証明書の使用* を選択します。今までに共通サーバ証明書を作成 していない場合は、左記のボタンを利用することができませんので、*証明 機関管理モジュールの起動* を押して証明書を作成してください。詳しく は 15.2 項「CA 管理用の YaST モジュール」(200 ページ)をお読みく ださい。

なお、ダイアログの左側のペインで **スキーマファイル** を選択することで、スキーマファイルをサーバの設定ファイルに含めるように 設定することができます。スキーマファイルの選択の既定値は、YaST の ユーザアカウントデータのソースを提供するように設定されています。

YaST では、スキーマファイルとして従来の形式 (通常は .schema で終わるファイル) のものを追加できるほか、OpenLDAP の LDIF スキーマ形式のものを追加することもできます。

図 4.5 YaST LDAP サーバデータベース設定



お使いの LDAP サーバで管理するデータベースを設定するには、下記の手順で行ないます:

- 1 ダイアログの左側のペインで、**データベース** の項目を 選択します。
- 2 **追加** を押してデータベースを追加します。
- 3 下記のとおり必要なデータを入力します:

ベース DN

LDAP サーバでのベース DN を入力します。

管理者 DN

サーバ管理者の DN を入力します。なお、ベース DN を追加を選択した場合は、管理者の cn 部分だけを指定するだけで、残りはシステム側で自動的に設定するようになります。

LDAP 管理者パスワード

データベース管理者のパスワードを入力します。

このデータベースを OpenLDAP クライアント向けの既定値とする必要であれば、利便性のためオプションを選択します。

- 4 次のダイアログではレプリケーションの設定を行ないます。
- 5 次のダイアログでは、パスワードポリシーの強制を設定し、LDAP サーバに 対するセキュリティの強化を設定することができます:
 - 5a パスワードポリシーを設定できるようにするには、パスワードポリシーを有効にするを選択します。
 - 5b 暗号化されていないパスワードに対して、それらをデータベースに追加したり修正したりする際、事前にハッシュ化する場合は、平文パスワードをハッシュするを選択します。
 - 5c ロック (施錠) されたアカウントに対してバインド要求を送信した際、その旨のエラーメッセージを提供する場合は、アカウントロック状態を通知するを選択します。

警告: セキュリティの厳しい環境におけるロック済みアカウントについて

お使いの環境がセキュリティに厳しい環境の場合は、アカウントロック状態を通知するを選択しては なりません。これは「ロックされている」ことを示すエラーメッセージを 通知してしまうと、潜在的な攻撃者に対してセキュリティ関連の情報を 公開することになってしまうためです。

- 5d 次に既定のポリシーオブジェクトに対する DN を指定します。YaST が 提案する値以外の DN を使用するには、その旨入力してください。それ以外の場合は、既定値のまま進めてください。
- 6 最後に 完了 を押すと、データベースの作成作業を 完了することができます。

パスワードポリシーを有効にするように選択していない場合、この時点でサーバが利用できるようになります。パスワードポリシーを有効にするように選択した場合は、ここからパスワードポリシーに関する詳細を設定します。また、存在していないパスワードポリシーのオブジェクトを指定した場合は、YaST でそれを作成します:

- 1 まずは LDAP サーバのパスワードを入力します。左側のペインから データベースを選び、作成したデータベースオブジェクト のツリーを開いていって、パスワードポリシーの設定 を選択します。
- 2 パスワードポリシーを有効にする が選択されていることを 確認し、ポリシーの編集 を押します。
- 3 まずはパスワードの変更ポリシーを設定します:

- 3a まずはパスワード履歴に保存するパスワード数を指定します。保存されたパスワードは、新しいパスワードとして設定することができなくなります。
- 3b また、各ユーザに対して自分自身のパスワードの変更を許可するか、および 管理者によるパスワードリセット時に、パスワード変更を求められるかどうか を設定します。さらに、パスワード変更時に古いパスワードの入力を必要とするかどうかを (任意で) 選択することもできます。
- 3c それ以外にも、パスワードに対して品質チェックを行なうかどうか、どの ように行なうのかを設定することもできます。まず最小のパスワード長は、パスワードの有効性が検証される前に確認される項目です。確認できないパスワードを受け付ける を選択すると、品質チェックが実施できない場合にでもパスワードを受け付けるよう になります。逆に 確認済みパスワードのみを受け付ける を選択すると、品質テストを通過したもののみを受け付けるようになります。

- 4 パスワードの時間制限に関するポリシーは、下記のように設定します:

- 4a まずはパスワード使用日数について、最小値と最大値を設定します (最小値の 設定は、パスワードをあまりに頻繁に変更させないためのものです)。
- 4b またパスワードの有効期限について、その警告日数と実際の期限切れを設定 します。
- 4c パスワードが恒久的な期限切れになる前に、期限の切れたパスワード の使用を許可する回数を指定します。

5 最後にロックアウトポリシーを設定します:

5a まずはパスワードロックを行なうかどうかを設定します。

5b 次に、パスワードロックを発動するバインド失敗回数を設定します。

5c パスワードロックの期間を設定します。

5d パスワードの失敗をキャッシュ内に保持する期間を設定します。

6 すべてを設定したら *OK* を押すと、パスワード ポリシーを保存することができます。

以前に作成したデータベースを編集するには、左側のペインのツリー表示から、ベース DN を選択します。選択すると右側のペインには、作成したときと似た表示形態でデータベースが表示されます (ただしベース DN の項目のみ、グレーで表示されて変更できなくなっています)。

完了 を押して LDAP のサーバ設定を終了すると、LDAP サーバに対して基本的な設定が完了します。この設定をさらに詳しく調整するには、OpenLDAP の動的な設定バックエンドをご利用ください。

また、OpenLDAP の動的な設定バックエンドでは、LDAP データベース内に設定情報を保管しています。データベースは `/etc/openldap/slapd.d` 内に存在し、複数の `.ldif` ファイルから構成されています。これらのファイルに直接アクセスする必要はありません。設定にアクセスしたい場合は、YaST の LDAP サーバモジュール (`yast2-ldap-server` パッケージ) か、`ldapmodify` や `ldapsearch` のような LDAP クライアントツールを利用して行ないます。OpenLDAP の動的な設定について、詳しくは OpenLDAP の管理ガイドをお読みください。

4.4 YaST を利用した LDAP クライアントの設定

YaST には LDAP ベースのユーザを管理するためのモジュールが用意されています。インストール時にこの機能を有効化していない場合は、ネットワークサービス > *LDAP クライアント* を選択して、モジュールを起動してください。YaST では LDAP を動作させるのに必要な PAM の有効化作業と NSS 関連の変更、および必要なファイルのインストールまでを自動的行ないます。後は単純にサーバに対する接

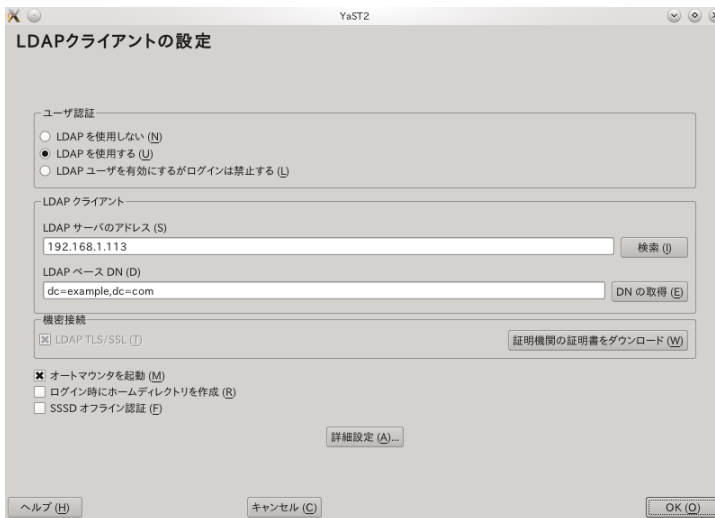
続設定を行なうことで、YaST はユーザを LDAP 経由で管理ようになります。基本的な設定方法は 4.4.1 項「基本的な設定」(52 ページ) で説明しています。

YaST の LDAP クライアント機能を利用することで、YaST のユーザとグループの設定モジュール側でも設定を行なうことができるようになります。これには新規に作成するユーザやグループに対する既定値の変更のほか、ユーザやグループに割り当てる属性値などを設定することができます。LDAP のユーザ管理では、通常のユーザやグループの管理機能よりも多くの種類の属性を、ユーザやグループに対して設定することができます。こちらについて、詳しくは 4.4.2 項「YaST ユーザとグループの管理モジュールの設定」(55 ページ) をお読みください。

4.4.1 基本的な設定

インストール時に LDAP によるユーザ管理を選択した場合や、インストール済みのシステムで YaST のコントロールセンターから、ネットワークサービス > LDAP クライアントを選択すると、LDAP クライアントの基本設定ダイアログ (図4.6「YaST: LDAP クライアントの設定」(52 ページ)) が表示されます。

図 4.6 YaST: LDAP クライアントの設定



お使いのマシンで、OpenLDAP サーバを利用したユーザ認証とユーザ管理を行なうには、下記の手順を実施します:

- 1 LDAP を使用するには、まず *LDAP を使用する* を選択 します。LDAP を認証には使用するものの、他のユーザがこのクライアントに ログインできないようにするには、*LDAP を使用するがログインは禁止する* を選択 します。
- 2 使用する LDAP サーバの IP アドレスを入力 します。
- 3 LDAP サーバ上での検索ベースを設定するには、*LDAP ベース DN* に入力 します。ベース DN を自動的に取得させたい場合は、*DN の取得* を押 します。このボタンを押すと、YaST は上記でアドレスを設定した LDAP サーバに対してデータベースを検索 します。あとは検索結果が表示 されますので、ここから適切なベース DN を選択 します。
- 4 サーバとの通信を TLS や SSL で保護する必要がある場合は、*LDAP TLS/SSL* を選択 します。特定の URL から PEM 形式で証明書をダウンロード したい場合は、*証明機関の証明書をダウンロード* を押 します。
- 5 リモートで管理されている /home が存在する ような環境の場合は、*automounter を起動* を選択 します。
- 6 初回のユーザログイン時に、自動的にユーザのホームディレクトリを 作成するには、*ログイン時にホームディレクトリを作成* を選択 します。
- 7 最後に *OK* を押 し、設定を適用 します。

管理者としてサーバ上のデータを修正するには、*詳細設定* を押 します。図 4.7「YaST: 詳細設定」(54 ページ) のダイアログは、2 つの タブから構成 されています。

図 4.7 YaST: 詳細設定



1 クライアント設定 のタブでは下記の設定を行なうことができます:

- 1a LDAP ベース DN で指定した DN とは異なる検索ベース をユーザやパスワード、グループに対して設定するには、それぞれ ユーザマップ、パスワードマップ、グループマップ で設定します。
- 1b 次にパスワードの変更プロトコルを設定します。パスワードを変更する場合の標準的なプロトコルは crypt です。これは crypt で生成されたパスワード ハッシュを利用する意味になります。詳細やその他のオプションについては、pam_ldap のマニュアルページをお読みください。
- 1c 使用する LDAP グループについては、グループメンバー属性 を利用します。既定値は member です。
- 1d 証明書確認に暗号化の接続が必要となる場合は、証明機関の証明書ファイル で PEM 形式のファイルを 指定するか、もしくは証明書のあるディレクトリを指定します。
- 1e お使いの LDAP サーバが現在も LDAPv2 を使用している場合は、LDAP バージョン 2 を選択して LDAPv2 を使用するようにします。

2 管理設定 では下記の設定を行なうことができます:

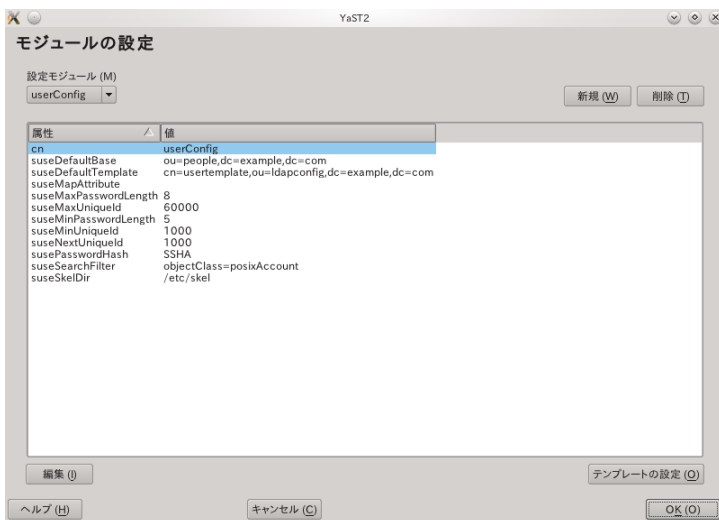
- 2a** まずはユーザ管理データを保存するベース DN を、*設定のベース DN* で指定します。
- 2b** 次に *管理者 DN* に適切な値を代入します。特定のユーザに対して LDAP サーバ内に保管されたデータを操作できるようにするため、`/etc/openldap/slapd.conf` 内の `rootdn` の値と同じでなければなりません。ここには 完全 DN (例: `cn=Administrator,dc=example,dc=com`) を入力することができるほか、`cn=Administrator` のように入力して *ベース DN* を *追加* を選択し、ベース DN を自動的に追加させることもできます。
- 2c** また、LDAP でユーザ管理を行なうため、基本的な設定オブジェクトを作成するには、*既定の設定オブジェクトを作成* を選択します。
- 2d** お使いのマシンが、ネットワーク経由でホームディレクトリを提供する ファイルサーバとして動作する場合は、*このマシン内にホームディレクトリを設置* を選択します。
- 2e** 使用するパスワードポリシーを追加したり削除したり、修正したりするには、*パスワードポリシー* のセクションを利用します。YaST でのパスワードポリシー設定は、LDAP サーバの設定の一部です。
- 2f** *詳細設定* を終了するには、*OK* を押します。すると元の画面に戻りますので、*完了* を押すと設定を適用することができます。

LDAP サーバ内の項目を編集したい場合は、*ユーザ管理の設定* を押します。サーバ上の設定モジュールへのアクセスは、サーバ側の ACL や ACI に従って許可されます。あとは 4.4.2 項「YaST ユーザとグループの管理モジュールの設定」(55 ページ) の手順に従って作業を行ってください。

4.4.2 YaST ユーザとグループの管理モジュールの設定

ユーザやグループの管理を行なうために YaST モジュールを設定するには、YaST の LDAP クライアントを使用します。ここではデータ登録を簡略化するため、それぞれの属性に対して既定値のあるテンプレートを作成することができます。ここで作成した既定値は、LDAP ディレクトリ内に LDAP オブジェクトとして保管されます。ユーザデータの登録は YaST のユーザとグループの管理モジュールから登録することもできます。こちらのデータについても、サーバ内で LDAP オブジェクトとして保管されます。

図 4.8 YaST: モジュール設定



モジュール設定のダイアログ (図4.8「YaST: モジュール設定」(56 ページ)) では、新しいモジュールの作成のほか、既存の設定モジュールの選択や修正、モジュール向けテンプレートの設計と修正を行なうことができます。

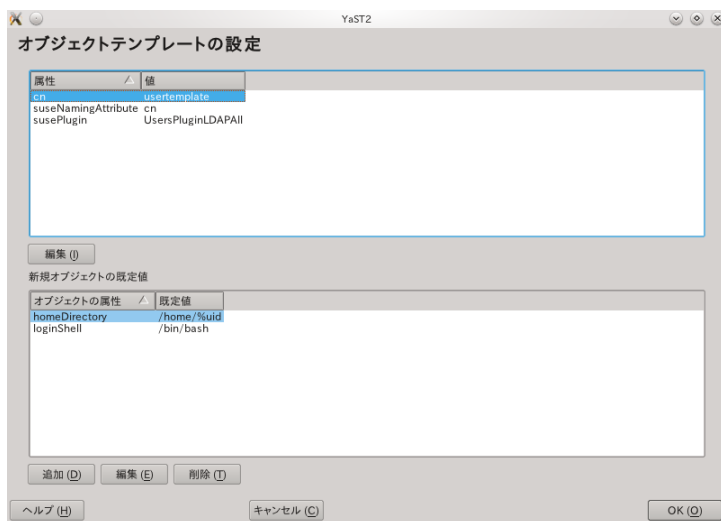
新しい設定モジュールを作成するには、下記の手順で行ないます:

- 1 **LDAP クライアント設定** で **詳細設定** を押し、**管理設定** のタブを開きます。ユーザ管理の設定を押して、LDAP サーバに対する 認証情報を入力します。
- 2 **新規** を押して作成するモジュールの種類を選択します。ユーザ設定モジュールの場合は **suseUserConfiguration** を、グループ設定モジュールの場合は **suseGroupConfiguration** を選択します。
- 3 新しいテンプレートに設定する名前を選択します (例: **userConfig**)。内容ビューでは、このモジュールで許可されているすべての属性と、それらに 割り当てられた値が表形式で一覧になっています。
- 4 事前に設定されている値をそのまま受け入れるか、もしくは属性値を選択してから **編集** を押し、ユーザやグループの既定値として設定する値を入力します。モジュールの名前を変更するには、モジュールの属性値内にある **cn** の値を変更します。**削除** を押すと、現在選択されている モジュールを削除することができます。
- 5 最後に **OK** を押すと、新規モジュールが選択メニュー 内に追加されます。

YaST のユーザとグループの管理モジュールには、既定値付きのテンプレート が用意されています。設定モジュールに関連づけられたテンプレートを編集 するには、オブジェクトテンプレートの設定を起動します (図4.9「YaST: オブジェクトのテンプレート設定」 (57 ページ))。

- 1 モジュールの設定 ダイアログで、テンプレートの設定 を押します。
- 2 このテンプレートに割り当てる一般的な属性値を設定するか、もしくは値を 入力しないままにしておきます。入力を行なわないと、LDAP サーバ内では 属性が削除されます。
- 3 新しいオブジェクト (LDAP ツリー内でのユーザやグループの設定オブジェクト) に対する新しい既定値を修正／削除／追加します。

図 4.9 YaST: オブジェクトのテンプレート設定



テンプレートを特定のモジュールに結びつけるには、モジュールの `susedefaulttemplate` の属性値に、結びつけるテンプレートの DN を指定します。

ヒント

属性の既定値は、絶対値の代わりに他の属性の値を指定することもできます。たとえば新しいユーザを作成する際、`cn=%sn %givenName` のように指定すると、それぞれ属性値の `sn` と `givenName` の値から自動生成されるようになります。

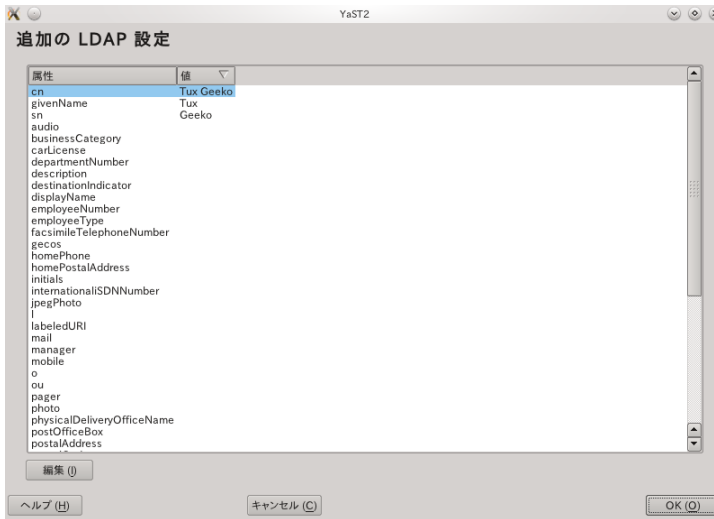
すべてのモジュールとテンプレートを設定し終わったら、通常の YaST での 作成方法を利用して新しいユーザ／グループを作成することができます。

4.5 YaST による LDAP ユーザとグループの設定

実際のユーザやグループの登録作業は、LDAP を使用していない場合と比べると少しだけ異なります。下記の手順ではユーザの管理について説明していますが、グループの管理もほぼ同様です。

- 1 YaST を起動して、**セキュリティとユーザ > ユーザとグループの管理** を選択し、ユーザ管理モジュールを起動します。
- 2 LDAP のユーザだけが表示されるようにするには、**フィルタの設定** を押し、ルート DN のパスワードを入力します。
- 3 **追加** を押してユーザ設定画面を表示させます。ダイアログは 4 つのタブから構成されています:
 - 3a **ユーザ情報** のタブでは、ユーザ名とログイン、パスワードをそれぞれ設定します。
 - 3b **詳細** タブではグループのメンバー設定のほか、ログイン シェルや新しいユーザに割り当てるホームディレクトリを設定します。必要であれば既定値を修正して使用することができます。既定値 (パスワードの設定 タブでも同様です) は、4.4.2項「YaST ユーザとグループの管理モジュールの設定」(55 ページ) に示されている 手順で設定することができます。
 - 3c **パスワードの設定** で既定値をそのまま受け入れるか、もしくは修正して設定します。
 - 3d **プラグイン** のタブでは LDAP プラグインを選択して **起動** を押すと、新しく作成するユーザに設定する 追加の LDAP 属性を設定することができます (詳しくは 図4.10「YaST: 追加の LDAP 設定」(59 ページ) をご覧ください)。
- 4 最後に **OK** を押すと、設定を保存してユーザ設定を 終了することができます。

図 4.10 YaST: 追加の LDAP 設定



ユーザ管理の最初の入力フォームでは、*LDAP オプション* の機能が用意されています。ここでは利用可能なユーザの中から、検索フィルタに適合するユーザを捜し出すことができるようになっています。これ以外にも、*LDAP ユーザとグループの設定* を選択することで、LDAP のユーザとグループを設定するモジュールを起動することもできます。

4.6 LDAP ディレクトリツリーの参照

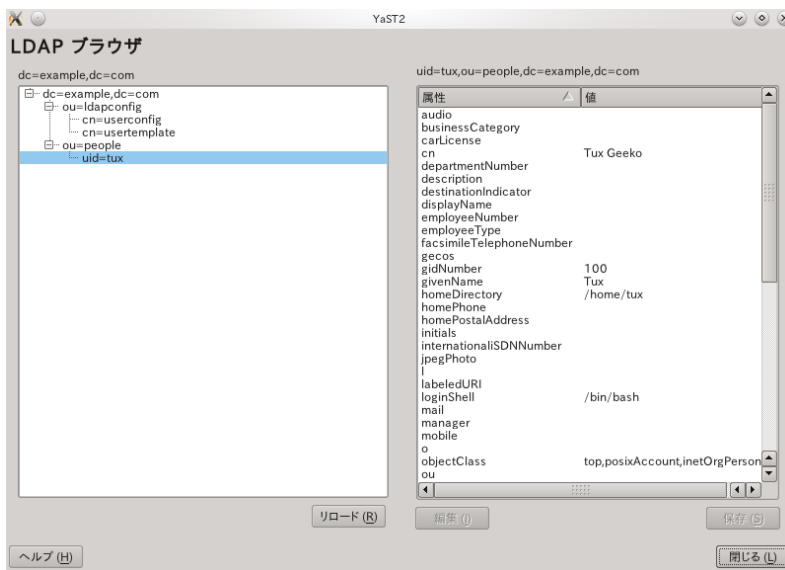
LDAP のディレクトリツリーとその項目を便利に参照するには、YaST の LDAP ブラウザを利用するのが便利です：

- 1 root でログインします。
- 2 YaST > ネットワークサービス > LDAP ブラウザを選択します。
- 3 まずは LDAP サーバのアドレスと管理者 DN、そして LDAP サーバのパスワード (サーバ上に保管されているデータに対して、読み込みと書き込みの両方を必要とする場合) をそれぞれ入力します。

上記以外にも、パスワードを入力せずに *匿名アクセス* を押し、ディレクトリに対する読み込みアクセスだけを行なう方法もあります。

LDAP ツリー のタブには、接続先のマシンでの LDAP ツリーの内容が表示されます。それぞれのツリーを押すことで、項目を展開 することができます。

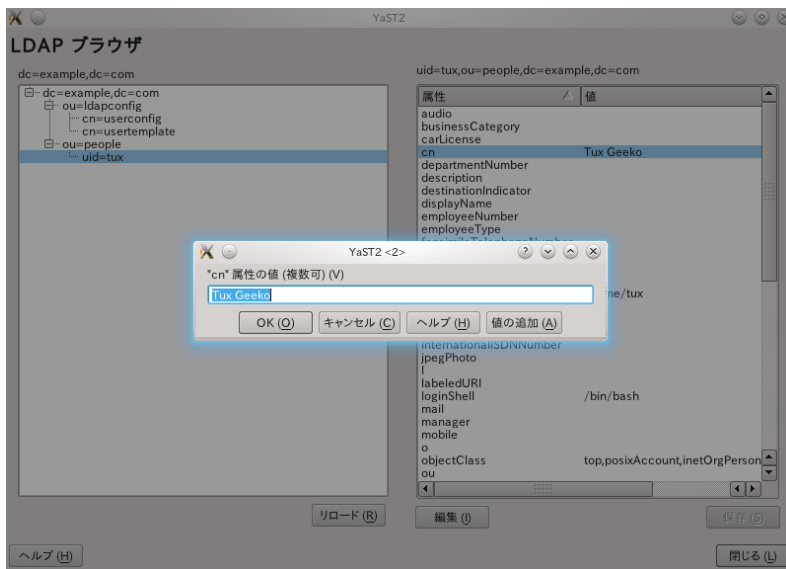
図 4.11 LDAP ディレクトリツリーの参照



- 4 特定の項目について詳細を読むには、LDAP ツリー のビューで項目を選んで、項目データ 他部を開きます。

なお、項目に対するすべての属性とその値が表示されます。

図 4.12 項目データの参照



- 5 設定されている値を変更するには、対象の属性を選んで **編集** を押します。その後、新しい値を入力して **保存** を押します。場合によってはルート DN のパスワード入力を求められる場合があります。
- 6 **閉じる** を押すと LDAP ブラウザを終了することができます。

4.7 LDAP サーバの手動設定

YaST では OpenLDAP の動的な設定データベース (back-config) を利用して、LDAP サーバの設定を保管しています。動的な設定バックエンドについて、詳しくは `slapd-config(5)` のマニュアルページ、または `openldap2` パッケージをインストールすることで配置される、`/usr/share/doc/packages/openldap2/guide/admin/guide.html` にある OpenLDAP ソフトウェア 2.4 管理者ガイドをお読みください。

ヒント: 古い OpenLDAP 環境のアップグレードについて

YaST では `/etc/openldap/slapd.conf` に配置されている OpenLDAP の設定ファイルは、もはや読み込まないようになっています。システムアップグレードの際

は、`/etc/openldap/slapd.conf` にある元の設定ファイルは、`/etc/openldap/slapd.conf.YaSTsave` としてコピーされます。

設定バックエンドにアクセスしやすくするため、SASL 外部認証をご利用になることをお勧めします。たとえば下記の `ldapsearch` コマンドを `root` で実行すると、`slapd` の設定すべてを表示させることができます：

```
ldapsearch -Y external -H ldapi:/// -b cn=config
```

4.7.1 サーバの起動と停止

LDAP サーバをすべて設定し、必要な全項目を 4.8 項「LDAP データの手作業での管理」(62 ページ) にあるパターンに従って作成したあとは、`root` で `rcldap start` を実行し、LDAP サーバを起動します。サーバを手作業で停止するには、同様に `rcldap stop` を実行します。また、サーバの状態を問い合わせるには、`rcldap status` を実行します。

4.8 LDAP データの手作業での管理

OpenLDAP では LDAP のディレクトリ内のデータを管理するのに、各種のツールが用意されています。本章では、それぞれ追加や削除、検索や修正を行なうための 4 種類のツールについて紹介しています。

4.8.1 LDAP ディレクトリへのデータ挿入

LDAP サーバの設定を行なったあと (それぞれ `suffix`, `directory`, `rootdn`, `rootpw`, `index` に対して 必要な設定を行なったあと) は、レコードの追加を行なうことができるようになります。OpenLDAP では `ldapadd` コマンドでこれを行なうことができます。また、可能であればオブジェクトを一括で追加してください (実際の用途でも一括で登録します)。LDAP はシンプルな LDIF 形式 (LDAP データ交換フォーマット) のファイルで処理を行ないます。LDIF ファイルはシンプルなテキストファイルで、任意の数の属性と値の対を記述することができます。図 4.1「LDAP ディレクトリの構造」(41 ページ) にある例のようなレコードを作成する際の LDIF ファイルは、例 4.2「LDIF ファイル」(63 ページ) のような形で記述します。

重要: LDIF ファイルのエンコーディング

LDAP は UTF-8 (Unicode) で動作しています。また、発音記号などは正しくエンコードされていなければなりません。正しくエンコードされていない場合は発

音記号や特殊記号類を削除するか、もしくは iconv コマンドで入力ファイルを UTF-8 に エンコードしてください。

例 4.2 LDIF ファイル

```
# The Organization
dn: dc=example,dc=com
objectClass: dcObject
objectClass: organization
o: Example dc: example

# The organizational unit development (devel)
dn: ou=devel,dc=example,dc=com
objectClass: organizationalUnit
ou: devel

# The organizational unit documentation (doc)
dn: ou=doc,dc=example,dc=com
objectClass: organizationalUnit
ou: doc

# The organizational unit internal IT (it)
dn: ou=it,dc=example,dc=com
objectClass: organizationalUnit
ou: it
```

.ldif という拡張子でファイルを保存したあと、下記の コマンドでサーバに渡します:

```
ldapadd -x -D 管理者の DN -W -f file.ldif
```

-x はこの場合、SASL による認証を無効化する設定です。また、-D は操作を呼び出すユーザを指定するものです。ここには slapd.conf で設定した、正しい管理者の DN を入力します。上記の例では cn=Administrator,dc=example,dc=com という DN になっています。また、-W はコマンドライン 内に (暗号化されない形で) パスワードを入力しないようにするもので、これを指定すると実行後にパスワードプロンプトが表示されるようになります。-f オプションにはファイル名を指定します。ldapadd を実行する際の詳細は、例 4.3「example.ldif を利用した場合の ldapadd」(63 ページ)をお読みください。

例 4.3 example.ldif を利用した場合の ldapadd

```
ldapadd -x -D cn=Administrator,dc=example,dc=com -W -f example.ldif
```

```
Enter LDAP password:
adding new entry "dc=example,dc=com"
adding new entry "ou=devel,dc=example,dc=com"
```

```
adding new entry "ou=doc,dc=example,dc=com"
adding new entry "ou=it,dc=example,dc=com"
```

各個人のユーザデータは、別々の LDIF ファイルとして用意することができます。たとえば 例4.4「Tux 向けの LDIF データ」(64 ページ) では、Tux という新しい LDAP 項目を追加しています。

例 4.4 *Tux 向けの LDIF データ*

```
# coworker Tux
dn: cn=Tux Linux,ou=devel,dc=example,dc=com
objectClass: inetOrgPerson
cn: Tux Linux
givenName: Tux
sn: Linux
mail: tux@example.com
uid: tux
telephoneNumber: +49 1234 567-8
```

LDIF ファイルには任意の数のオブジェクトを含めることができます。また、サーバにある複数のディレクトリの項目を一括で追加することもできますし、例に示しているとおり個別に追加することもできます。ただし、複数のデータに関連性があるもの場合は、それらはまとめて追加することをお勧めします。

4.8.2 LDAP ディレクトリ内のデータ修正

ldapmodify というツールで保管されているデータの修正を行なうことができます。最も簡単な方法として、修正する LDIF ファイルを用意して、それらを LDAP サーバに送信する方法があります。たとえば Tux という従業員の電話番号を、+49 1234 567-8 から +49 1234 567-10 に変更するには、LDIF ファイルを 例4.5「修正済みの LDIF ファイル tux.ldif」(64 ページ) のように修正します。

例 4.5 *修正済みの LDIF ファイル tux.ldif*

```
# coworker Tux
dn: cn=Tux Linux,ou=devel,dc=example,dc=com
changetype: modify
replace: telephoneNumber
telephoneNumber: +49 1234 567-10
```

LDAP ディレクトリ内の項目をファイルから修正するには、下記のコマンドを入力します:

```
ldapmodify -x -D cn=Administrator,dc=example,dc=com -W -f tux.ldif
```

上記の方法以外にも、`ldapmodify` コマンドに対して変更する属性を直接指定する方法もあります:

- 1 `ldapmodify` コマンドを下記のように入力し、パスワードを入力します:

```
ldapmodify -x -D cn=Administrator,dc=example,dc=com -W
Enter LDAP password:
```

- 2 文法と順序に注意しながら、下記のように変更内容を入力します:

```
dn: cn=Tux Linux,ou=devel,dc=example,dc=com
changetype: modify
replace: telephoneNumber
telephoneNumber: +49 1234 567-10
```

`ldapmodify` コマンドとその文法について、詳しくは `ldapmodify` のマニュアルページをお読みください。

4.8.3 LDAP ディレクトリの検索と読み込み

OpenLDAP では `ldapssearch` コマンドを利用することで、LDAP ディレクトリ内にあるデータを検索することができるほか、それらの データを表示することができます。最も簡単な問い合わせ方法は下記のとおりです:

```
ldapssearch -x -b dc=example,dc=com "(objectClass=*)"
```

`-b` オプションは検索ベース (検索を開始するツリー) を指定 するためのものです。上記の例では `dc=example, dc=com` を 指定しています。たとえば LDAP ディレクトリ内の特定のサブツリー内を検索 するような場合は、そのツリーを `-b` で指定します。`-x` は簡易認証を有効にする設定です。また、`(objectClass=*)` はディレクトリ内に含まれるすべての オブジェクトを読み込むための指定です。このコマンドは、新しいディレクトリ ツリーを作成した後に使用し、すべての項目が正しく作成されていて、サーバが 正しく応答するかどうかを確認する際にも使用できます。`ldapssearch` の使用方法について、詳しくは `ldapssearch(1)` のマニュアルページをお読みください。

4.8.4 LDAP ディレクトリ内のデータ削除

不要な項目は `ldapdelete` で削除することができます。コマンドの文法は他のコマンドとほぼ同様です。たとえば Tux Linux の項目を削除するには、下記のコマンドを入力します:

```
ldapdelete -x -D cn=Administrator,dc=example,dc=com -W cn=Tux ¥
```

4.9 さらなる情報

良く複雑な設定方法 (たとえば SASL の設定や複数のスレーブにデータを配置するレプリケーション付き LDAP サーバなど) については、本章では省略しています。これらの詳細について、詳しくは「OpenLDAP 2.4 Administrator's Guide」(66 ページ) または「OpenLDAP 2.3 管理者ガイド」(66 ページ) に書かれている *OpenLDAP 2.4 Administrator's Guide* をお読みください。

OpenLDAP プロジェクトの Web サイトでは、下記のように様々な初心者向け／上級者向けのドキュメントが用意されています：

OpenLDAP Faq-O-Matic

OpenLDAP のインストールや設定、使用方法などについて、よくある質問やその回答が記されています。<http://www.openldap.org/faq/data/cache/1.html> (英語) をお読みください。

Quick Start Guide

LDAP サーバをはじめてインストールする際の、手順を追った概要説明が記されています。<http://www.openldap.org/doc/admin24/quickstart.html> または、インストール後のシステムで `/usr/share/doc/packages/openldap2/guide/admin/guide.html` ファイルをお読みください (いずれも英語です)。

OpenLDAP 2.4 Administrator's Guide

LDAP の設定について、重要なすべての要素が書かれた詳細な紹介です。アクセス制御や暗号化などにも言及しています。<http://www.openldap.org/doc/admin24/> または、インストール後のシステムで `/usr/share/doc/packages/openldap2/guide/admin/guide.html` をお読みください (いずれも英語です)。

OpenLDAP 2.3 管理者ガイド

少しバージョンの古い情報ですが、LDAP の管理者に向けた文書です。<http://www5f.biglobe.ne.jp/~inachi/openldap/admin23/index-ja.html> をお読みください (日本語です)。

Understanding LDAP

LDAP における基本的な考え方を紹介している、一般向けの情報です：
<http://www.redbooks.ibm.com/redbooks/pdfs/sg244986.pdf>.

LDAP に関する書籍もあります:

- *LDAP System Administration* by Gerald Carter (ISBN 1-56592-491-6) (英語)
- *Understanding and Deploying LDAP Directory Services* by Howes, Smith, and Good (ISBN 0-672-32316-8)
- *LDAP - 設定・管理・プログラミング-* (Gerald Carter, でびあんぐる) (ISBN-13 978-4274065507)
- *入門 LDAP/OpenLDAP - ディレクトリサービス導入・運用ガイド* (デージーネット) (ISBN-13 978-4798018003)

LDAP に関する巨大なリファレンスとしては、関連する RFC (Request For Comments) である 2251 から 2256 もあります。

Active Directory への対応

Active Directory* (以下 AD) は LDAP と Kerberosなどをベースにし、リソースやサービス、ユーザアカウント (以下オブジェクト)などを管理するための、Microsoft Windows で使用されるサービスです。MS Windows のネットワークでは、AD はそれらの オブジェクトに関する情報を提供するほか、それらへのアクセス制限やポリシーの 強制などを提供します。openSUSE® では、既存の AD ドメインに参加し、Linux マシンと Windows 環境を統合することができるようになっています。

5.1 Linux と AD 環境の統合

既存の Active Directory ドメインに参加させた Linux クライアントでは、設定していない場合と比較して、下記のような様々な機能を利用できるようになります:

SMB を利用した共有ファイルや共有フォルダへの参照

Nautilus (GNOME ファイルマネージャ), Dolphin, Konqueror (KDE ファイルマネージャ) では、SMB を利用して共有リソースにアクセスすることができます。

SMB を利用したファイルやフォルダの共有

Nautilus (GNOME ファイルマネージャ), Dolphin, Konqueror (KDE ファイルマネージャ) では、SMB 経由で Windows にファイル共有を行なうことができます。

Windows サーバ上にあるユーザデータへのアクセスや操作

Nautilus と Konqueror を利用することで、ユーザは自分自身の Windows ユーザデータにアクセスすることができるほか、Windows サーバ上にある ファ

イルやフォルダなどを編集したり、作成／削除したりすることもできます。ユーザー側でパスワードを何回も入力する必要はありません。

オフライン認証

ユーザは Linux マシン上に対して、何らかの理由で AD サーバが利用できない場合でも、ログインを行なってローカルデータにアクセスできるようになります。

Windows のパスワード変更

Linux における AD 対応の仕組みでは、Active Directory 内に保存されているパスワードポリシーの強制機能が備わっています。ディスプレイマネージャやコンソールでは、パスワード変更メッセージや入力の受け付け機能も存在しているほか、passwd コマンドで Windows のパスワードを変更することもできます。

Kerberos 対応のアプリケーションによるシングルサインオン

GNOME, KDE のいずれのデスクトップとも、多数のアプリケーションが Kerberos に対応しています。これにより、Web サーバやプロキシ、グループウェアアプリケーションなどを利用する場合でも、それぞれでパスワードを入力する必要がなくなります。

これらの機能に関する技術情報の概要は、下記の章で説明しています。

5.2 Linux における AD 対応の背景となる情報

Linux クライアントを既存の Windows での Active Directory ドメインに統合するには、多数のシステムコンポーネントを完全に動作させる必要があります。図 5.1「Active Directory 認証スキーマ」(70 ページ) では、もっとも重要な点を説明しています。下記の章では、AD サーバとクライアントの動作について、主なイベントの裏側で動作する処理について説明しています。

5.1 Active Directory 認証スキーマ

ディレクトリサービスとの通信するには、クライアントとサーバは 少なくとも下記の 2 つのプロトコルを利用する必要があります:

LDAP

LDAP はディレクトリ情報を管理するために最適化されたプロトコルです。Windows の AD ドメインコントローラでは、クライアントとのディレクトリ 情報の交換に際して、LDAP のプロトコルを使用しています。LDAP について 詳しい情報や、LDAP のオープンソース実装である OpenLDAP について、詳しくは 第4章 ディレクトリサービス LDAP (39 ページ) をお読みください。

Kerberos

Kerberos はサードパーティ製の信頼できる認証サービスです。全ての Kerberos クライアントは、他のクライアントの Kerberos 認証情報を信頼する仕組みになっていて、これによって Kerberos のシングルサインオン (SSO) を構成しています。Windows にも Kerberos 実装が存在しているため、これによって Linux クライアントが参加できるようになっています。

下記のクライアントコンポーネントでは、アカウントと認証データを取り扱います:

Winbind

Active Directory のクライアント機能を利用するにあたって、最も重要な位置を 占めるのが winbind デーモンです。これは Samba プロジェクトから提供されているもので、AD サーバとの通信を全て取り扱います。

NSS (*Name Service Switch*)

NSS ルーチンはネームサービスの情報を提供します。ユーザとグループに対する ネームサービスは、nss_winbind が提供します。この モジュールは、winbind デーモンと直接通信して処理を行ないます。

PAM (*Pluggable Authentication Modules*)

AD ユーザに対する認証は、pam_winbind モジュールが 行ないます。また、AD ユーザに対する Linux クライアント側のホーム ディレクトリの作成は、pam_mkhomedir が行ないます。さらに、pam_winbind は winbind と直接通信を行なって 処理を行ないます。一般的な PAM の情報については、第2章 *PAM を利用した認証* (17 ページ) をお読みください。

PAM に対応したアプリケーション、たとえばログインルーチンや GNOME/KDE のディスプレイマネージャでは、Windows サーバに対する認証を行なうのに PAM や NSS レイヤと情報をやりとりします。また、Kerberos 認証に対応したアプリケーション (たとえばファイルマネージャや Web ブラウザ、電子メールクライアントなど) では、ユーザの Kerberos チケットにアクセスするのに Kerberos の認証キャッシュ を使用し、SSO として動作するようになっています。

5.2.1 ドメインへの参加

ドメインへの参加にあたっては、サーバとクライアントの間の通信には機密を 保持する必要があります。クライアント側では、既存の Windows ドメインコントローラ が提供する LDAP および Kerberos SSO 環境に参加するにあたって、下記の作業を実施する必要があります。ドメインへの参加作業は、全て YaST のドメイン メンバーシップモジュールを利用して行ないます。これはインストール中に実施 することができますほか、インストール済みのシステムでも行なうことができます。具体的には下記の作業を行なう必要があります：

- 1 LDAP と KDC (鍵配布センター) の各サービスを提供する Windows ドメイン コントローラを用意すること。
- 2 参加するクライアントのマシンアカウントが、ディレクトリサービス内に 作成すること。
- 3 クライアントに対する初期のチケット許可チケット(TGT) を取得し、Kerberos の認証情報キャッシュ内に保管すること。クライアントはこの TGT を利用して、LDAP 機能を提供するディレクトリサーバなど、他のサービスと通信を行なうための チケットを取得します。
- 4 NSS と PAM の設定を調整し、クライアントからドメインコントローラに対して 認証を行なうようにすること。

クライアントの起動時、winbind デーモンが起動され、マシンアカウントに対応 する初期の Kerberos チケットを取得します。その後、winbindd はそのマシンの チケットが有効であり続けるよう、チケットを更新し続けます。また、最新の アカウントポリシーを取得するため、winbindd はドメインコントローラに対して 定期的に問い合わせを行ないます。

5.2.2 ドメインへのログインとユーザのホームディレクトリ

GNOME や KDE のログインマネージャ (GDM や KDM) では、AD ドメインへのログイン を処理するために機能拡張が行なわれています。ユーザ側ではそのマシンが参加した ドメインにログインすることができるほか、そのドメインと信頼関係が設定されている他のドメインにログインすることもできます。

ユーザ認証は 5.2項「Linux における AD 対応の背景となる情報」(70 ページ) で示されているとおり、多数の PAM モジュールが処理するよ

うになっています。このうち Active Directory や NT ドメイン に対する認証機能を提供する pam_winbind モジュールでは、ユーザのログイン時に発生する可能性がある Windows エラーについて、全てのものに対応できる仕組みが備わっています。ログイン時に発生した Windows エラーのエラーコードは対応するテキストに変換され、それぞれ対応する方法 (GDM, KDM, コンソール, SSH) で表示します:

パスワードの有効期限が切れた場合

パスワードの有効期限が切れていて、変更しなければならない旨を表示します。システムは新しいパスワードを入力するように促すほか、新しいパスワードが パスワードポリシーに適合しているか (たとえばパスワードが短すぎないか、単純すぎないか、既に履歴内に存在していないか) を確認し、ポリシーに適合していない場合にはメッセージを表示します。パスワード変更が失敗した場合は、その理由が表示されて新しいパスワードを再度入力するように促します。

アカウントが無効化されていた場合

アカウントが無効化されていて、システム管理者に連絡を取るよう促す エラーメッセージが表示されます。

アカウントはロック (施錠) されていた場合

アカウントがロック (施錠) されていて、システム管理者に連絡を取るよう 促すエラーメッセージが表示されます。

パスワードを変更する必要がある場合

ユーザは問題なくログインできますが、パスワードを近いうちに変更する必要 が発生している旨の警告メッセージが表示されます。この警告メッセージは、パスワードの有効期限が切れる 3 日前に送信されます。パスワードの期限が 切れると、ログインできなくなります。

不正なワークステーションであった場合

ユーザが特定のワークステーションからの接続のみに制限されていて、そのワークステーションに openSUSE のマシンが含まれていない場合、このワークステーションからはログインできない旨のメッセージが表示されます。

不正なログイン時間帯であった場合

ユーザが特定の時間帯にのみログインできるように設定されていて、その 時間外にログインしようとしている場合は、その時間帯にはユーザはログイン できない旨のメッセージが表示されます。

アカウントの有効期限が切れた場合

管理者側では、特定のユーザアカウントに対して有効期限を設定することが できます。有効期限が切れた後にユーザがログインしようとすると、ユーザ 側には

アカウントの有効期限が切れていて、ログインできない旨のメッセージが表示されます。

認証が成功した場合、pam_winbind はチケット許可チケット (TGT) を Active Directory の Kerberos サーバから取得し、ユーザの認証キャッシュ内に保管します。その後、TGT の更新は裏側で自動的に行なわれるため、ユーザ側で操作を行なう必要はありません。

openSUSE では、AD ユーザに対してローカルのホームディレクトリを提供する機能を備えています。YaST を利用して 5.3 項「Linux クライアントに対して Active Directory を設定する方法」(75 ページ) の手順で設定を行なった場合、ユーザのホームディレクトリは、Windows (AD) ユーザが Linux クライアントにはじめてログインした際に作成されるようになります。ここで作成されたホームディレクトリは、通常の Linux ユーザ向けホームディレクトリとほぼ同じもので、AD のドメインコントローラとは独立して動作します。また、ローカルのユーザ向けホームディレクトリを使用した場合、Linux クライアント側でオフライン認証の設定を行なっている場合は、AD サーバとの通信が切れてもこれらのデータにアクセスできるようになります。

5.2.3 オフラインサービスとポリシーへの対応

企業環境内のユーザに対しては、ローミング機能 (たとえばネットワークを切り替えたり、一時的にネットワークを切断したりしても、認証を継続できる機能) に対応させなければなりません。ネットワーク接続の切れたマシンから、ユーザがログインできるようにするため、winbind デーモンには大規模なキャッシュ機能が備わっています。winbind デーモンは、オフライン状態でもパスワードポリシーの強制を実施するようになっていて、ログインの失敗回数を追跡するほか、Active Directory 内に設定されたポリシーへの対応も行なうことができます。既定ではオフラインへの対応は無効に設定されているため、YaST のドメインメンバーシップモジュールで明示的に有効化する必要があります。

ドメインコントローラが利用不可能な状態になった場合でも、ユーザは接続を失う前に獲得した Kerberos のチケットを利用して、ネットワーク上のリソースにアクセスし続けることができます (ただし AD サーバそれ自身を除きます。また、この機能は Windows クライアントにも存在するものです)。ただし、パスワードの変更はドメインコントローラへの接続が失われている場合には、実施することができません。また、AD サーバとの接続が切れている間は、このサーバに保管されているデータにアクセスすることはできません。さらに、そのマシンがネットワークから完全に切り離されていて、しばらく経ってから接続を回復させた場合、openSUSE ではユーザが

デスクトップを ロック／アンロック (施錠または解錠) したタイミング (スクリーンセーバを 利用した場合などを含みます) で新しい Kerberos チケットを獲得します。

5.3 Linux クライアントに対して Active Directory を設定する方法

お使いのクライアントを AD ドメインに追加する前に、クライアントとサーバが うまく動作するようにするため、お使いのネットワーク設定をいくつか調整 しなければなりません。

DNS

お買いのクライアントマシンに設定されている DNS サーバの設定を変更し、AD の DNS サーバにリクエストを転送できる DNS サーバを利用するように 設定する必要があります。もちろん AD の DNS サーバそれ自身を DNS サーバとして設定してもかまいません。

NTP

Kerberos 認証が正しく動作するようにするため、クライアント側の時刻を 正確に設定しておく必要があります。この目的を達成するために、NTP 時刻 サーバによる時刻同期の設定を強くお勧めします (お使いの Active Directory のドメインコントローラで動作している NTP サーバでもかまいません)。お使いの Linux ホストとドメインコントローラとの間で、一定以上のズレが あると Kerberos の認証は失敗してしまい、クライアントはより弱い認証方法である NTLM (NT LAN Manager) でログインするようになってしまいます。Active Directory を利用した時刻同期について、詳しくは 手順5.1「AD ドメインへの参加」(76 ページ) をお読みください。

DHCP

お使いのクライアントのネットワーク設定を DHCP で行なっている場合は、DHCP サーバ側を設定して、クライアントに常に固定のアドレスとホスト名を 割り当てるように設定する必要があります。また、可能であれば固定の IP アドレスを設定します。

ファイアウォール

ネットワーク内にあるコンピュータを参照できるようにするには、ファイアウォールを完全に無効化するか、もしくは参照に利用する ネットワークインターフェイスを内部ゾーンに設定する必要があります。

お使いのクライアントでファイアウォールの設定を変更するには、root でログインしたあと、YaST のファイアウォールモジュールを起動します。モジュールが起動したら **インターフェイス** を選択し、参照に利用するインターフェイスを選んで **変更** を押します。ここから **内部ゾーン** を選択し、**OK** を押して設定を適用します。あとは **次へ > 完了** を押せば設定を終えることができます。ファイアウォールを無効化したい場合は、**サービスの開始** の欄で **ファイアウォールを自動で起動しない** を選んでから、**次へ > 完了** と押していけば設定は完了です。

AD のアカウント

AD の管理者から、そのドメインに対する有効なアカウント情報を提供してもらわない限り、AD ドメインにログインすることができません。お使いの Linux クライアントから AD ドメインにログインするにあたっては、AD のユーザ名とパスワードをご用意ください。

AD ドメインへの参加は、インストール時に行なうことができるほか、インストール済みのシステムでも YaST を利用して SMB ユーザ認証を有効にすることで、後から参加させることもできます。

注記

現時点では、ドメインの管理者アカウント (たとえば Administrator) を利用しないと、openSUSE を Active Directory に参加させることができません。

インストール済みのシステムで AD ドメインへの参加を設定するには、下記の手順で行ないます:

手順 5.1 AD ドメインへの参加

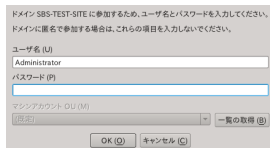
- 1 root でログインし、YaST を起動します。
- 2 ネットワークサービス > *Windows ドメインメンバーシップ* を選択します。
- 3 まずは *Windows ドメインメンバーシップ* の画面で、**ドメイン／ワークグループ** の欄にドメイン名を入力します (図5.2「Windows ドメインメンバーシップの設定」(77 ページ) をご覧ください)。なお、お使いのホストでの DNS 設定が Windows の DNS サーバと正しく統合できている場合は、AD のドメイン名を DNS 形式 (例: mydomain.mycompany.com) で入力します。ドメイン名を短い形式 (Windows 2000 のドメイン名としても知られている形式です) で入力した場合、YaST は DNS ではなく NetBIOS の名前解決によってドメインコントローラを検出するようになります。

図 5.2 Windows ドメインメンバーシップの設定

- 4 Linux 側の認証に SMB の認証を利用する場合は、*Linux の認証に SMB の情報を使用する* にチェックを入れます。
- 5 AD ユーザが Linux マシンにログインした際、ローカルのホームディレクトリを自動的に作成するには、*ログイン時にホームディレクトリを作成する* にチェックを入れます。
- 6 また、AD サーバが一時的に利用不可能な場合でも、ドメインユーザがログインできるように設定したい場合は、*オフライン認証* にチェックを入れます。
- 7 Samba のユーザやグループに対して設定する UID や GID を変更したい場合は、*熟練者向け設定* を選択します。なお、DHCP に対して WINS サーバの情報を取得する設定は、必要な場合にのみ行なってください。この設定は、WINS のシステムで名前解決を行なう場合にのみ利用します。
- 8 また、お使いの AD 環境で NTP による時刻同期を設定するには、*NTP の設定* を押し、サーバのホスト名または IP アドレスを入力します。ただし、YaST の NTP モジュールで既に設定済みの場合は、ここで設定する必要はありません。
- 9 設定が終わったら *OK* を押してください。ドメインへの参加確認メッセージが表示されます。

- 次に AD サーバの Windows 管理者アカウントのパスワードを入力し、OK を押します (図5.3「管理者の認証情報設定」(78 ページ) をご覧ください)。

図 5.3 管理者の認証情報設定



AD ドメインへの参加が完了すると、お使いのデスクトップのディスプレイ マネージャやコンソールから AD のアカウントでログインできるようになります。

5.4 AD ドメインへのログイン

お使いのマシンを Active Directory で認証するように設定し、必要なユーザ アカウントに関する情報を取得すれば、その情報を元にログインを行なうことができます。ログインはいずれのデスクトップ環境 (GNOME, KDE) にも対応しているほか、コンソールや SSH など、PAM に対応した アプリケーションでも利用可能です。

重要: オフライン認証

openSUSE ではオフライン認証に対応しているため、クライアント側の マシンがネットワークから切り離された状態でも、そのマシンでは AD の ユーザによるログインを行なうことができます。

5.4.1 GDM と KDM

GNOME のクライアントマシンで AD サーバに対して認証を行なうには、下記の手順で行ないます：

- まずはドメインを選択します。
- AD のユーザ名を入力して Enter を押します。
- AD のパスワードを入力して Enter を押します。

KDE のクライアントマシンで AD サーバに対して認証を行なうには、下記の 手順で行ないます:

- 1 まずドメインを選択します。
- 2 AD のユーザ名を入力します。
- 3 AD のパスワードを入力して Enter を押します。

openSUSE で AD のユーザがはじめてログインした際、ホームディレクトリを 作成するように設定していた場合は、ホームディレクトリが作成されます。これにより、Linux として完全に動作するマシンでありながら、openSUSE の AD サポートによるメリットを生かすことができるようになります。

5.4.2 コンソールログイン

AD のクライアントマシンでは、グラフィカルなフロントエンドでログイン するだけでなく、テキストベースのコンソールログインや SSH によるログイン を行なうこともできます。

コンソールから AD のアカウントでログインしたい場合は、login: プロンプトに `ドメイン¥ユーザ名` の形式で入力を行ない、そのユーザのパスワードを入力します。

SSH 経由で AD のアカウントでログインしたい場合は、下記のように 実施します:

- 1 コマンドは下記のように入力します:

```
ssh ドメイン¥ユーザ@ホスト名
```

ドメインとユーザ名の区切り文字である ¥ は、¥ を 2 度入力することでシェルの エスケープ処理を行なっています。

- 2 あとはそのユーザに対するパスワードを入力します。

5.5 パスワードの変更

openSUSE では、企業などで適用されているパスワードポリシーに適合 するパスワードを、ユーザに設定させるような機能が備わっています。PAM モジュールは、現在のパスワードポリシーをドメインコントローラから取得 し、メッセージにより表示でパ

スワードの品質設定などを通知することができます。Windows 側での動きと同様に、openSUSE では下記に関する メッセージを表示します:

- パスワードの履歴設定
- 最小のパスワード長の要件
- 最小のパスワード経過日数
- パスワードの複雑さ

パスワードの変更処理は、すべての要件が満たされている場合にのみ実施することができます。また、パスワードの状態に関する情報は、ディスプレイ マネージャやコンソールに表示されます。

GDM や KDM では、パスワードの有効期限に関する情報のほか、新しいパスワードを対話モードで設定するプロンプトが用意されています。ディスプレイ マネージャでパスワードを変更するには、問い合わせが表示されたときに パスワードを入力するだけです。

AD 側のパスワードを変更する場合でも、わざわざサーバ側で変更したりする必要はなく、通常の Linux ユーティリティである `passwd` コマンドで変更することができます。AD のパスワードを変更するには、下記の手順で行います:

- 1 コンソールにログインします。
- 2 `passwd` と入力します。
- 3 現在のパスワードについて問い合わせがあった場合は、現在のパスワードを入力します。
- 4 新しく設定するパスワードを入力します。
- 5 確認のため、再度新しく設定するパスワードを入力します。Windows サーバ側のパスワードポリシーを満たしていないパスワードを入力してしまった場合は、このフィードバックが表示され、他のパスワードを入力するように求められます。

AD 側のパスワードを GNOME デスクトップで変更するには、下記の手順で 行います:

- 1 パネルの左側の端にある **コンピュータ** を押します。

- 2 *コントロールセンター* を選択します。
- 3 *個人設定* のセクションから、*自分について* > *パスワードの変更* を選択します。
- 4 現在のパスワードを入力します。
- 5 新しいパスワードを二度入力します。
- 6 設定を反映するには、*閉じる* ボタンを押します。

AD 側のパスワードを KDE デスクトップで変更するには、下記の手順で 行ないます:

- 1 メインメニューから *デスクトップを設定* を選択します。
- 2 *About Me* を選択します。
- 3 *パスワード & ユーザアカウント* を選択します。
- 4 *パスワードを変更* を押します。
- 5 現在のパスワードを入力します。
- 6 新しいパスワードを二度入力し、*OK* を押して 設定を反映します。
- 7 あとは *with ファイル* > *終了* を選択し、設定ダイアログを終了します。

Kerberos を利用したネットワーク認証

オープンなネットワークでは、通常のパスワードによる認証を除き、ワークステーションがユーザを確認できる方法がありません。そのため、一般的なインストール環境の場合、ネットワーク内にあるサービスにアクセスする際には、毎回パスワードの入力を行わなければなりません。Kerberos はこのような問題に対応するための仕組みで、ユーザを一カ所で登録しておいて、これに対する認証を他のマシンでも信頼する方法を提供しています。ネットワークの機密を保持するには、下記の要件が満たされていなければなりません：

- すべてのユーザが必要なサービスに対する識別情報を持ち、他人がその識別情報を偽装できないこと。
- それぞれのネットワークサーバも識別情報を持っていること。識別情報が存在しないと、攻撃者がサーバを装うことができしまい、サーバに送信されるはずの機密情報を取得できてしまいます。サーバはクライアントを、クライアントはサーバを認証する、という考え方を *相互認証* と呼びます。

Kerberos はこれらの要件を、強い暗号化による認証で実現しています。ここでは Kerberos に関する基本的な動作原理を説明していますが、より詳しい技術的な説明については、Kerberos のドキュメンテーションをお読みください。

6.1 Kerberos で使用する用語

下記の用語集では、いくつかの Kerberos 用語を定義しています。

資格

ユーザやクライアントは、対象のサービスに要求を送信するため、何らかの 資格情報を送信する必要があります。Kerberos では、「チケット」と「認証情報」の 2 つの資格情報を利用します。

チケット

チケットとはクライアントが使用するサーバ単位の証明書で、サービスを利用するサーバを認証する際に使用します。ここにはサーバの名前と クライアントの名前、クライアントのインターネットアドレスとタイム スタンプ、有効期間と乱数から生成されるセッション鍵が含まれます。これらすべてのデータは、サーバ側の鍵を利用して暗号化されます。

認証情報

認証情報はチケットと組み合わせることで、チケットを提示したクライアントが正当なものであることを確認することができます。認証情報はクライアントの 名前とワークステーションの IP アドレス、現在のワークステーションの 時刻とクライアントと関連するサーバだけが知っている暗号化された セッション鍵から構成されています。認証情報はチケットとは異なり、一度だけしか使用することができます。また、クライアントは 認証情報それ自身を作成することができます。

プリンシパル

Kerberos でのプリンシパルとは、チケットを割り当てることのできる識別 可能な存在 (ユーザやサービス) を指します。プリンシパルには下記から 構成されています:

- **プライマリ** プリンシパルの最初の部分で、ユーザの場合はユーザ名と同じです。
- **インスタンス** プライマリを細分化する任意情報です。この文字列とプライマリとの 間は / で区切られます。
- **領域**— ここでは Kerberos の領域を指定します。通常、この領域には大文字でのドメイン名を指定します。

相互認証

Kerberos はクライアントとサーバとの間で、一方が他方の識別情報を 確認します。ここではセッション鍵が使用され、これによって通信の 機密を維持しています。

セッション鍵

セッション鍵とは Kerberos が生成した一時的な機密鍵のことを指します。セッション鍵はクライアント側に伝えられ、チケットの送受信を行なう際の クライアントとサーバの間での暗号鍵として使用します。

リプレイ

ネットワークでは、ほぼすべてのメッセージを傍受や盗難、そして再送信することができてしまいます。Kerberos の世界では、これは攻撃者がチケットと 認証情報を含むサービス向けの要求を取得できてしまうことを意味するため、もっとも危険なものとなります。攻撃者はそれを再送信 (リプレイ) することで、本来の送信者を偽装しようとしていくことができます。ただし、Kerberos では複数の仕組みを利用して、そのような偽装を防ぐ ことができるようになっています。

サーバ／サービス

サービス とは、ここでは実行すべき特定の処理を指しています。この処理を受け持つプロセスは、サーバと言います。

6.2 Kerberos の動作概要

Kerberos はしばしば、サードパーティ製の信頼できる認証サービスと呼ばれます。これは Kerberos の全クライアントが他のクライアントの認証情報を判断する際、Kerberos の判断結果を信頼することからそのように呼ばれます。Kerberos はすべてのユーザとそのユーザの機密鍵を管理します。

Kerberos が正しく動作するようにするため、認証サーバとチケット発行サーバは専用のマシンで動作させることをお勧めします。また、このマシンには管理者だけが物理的にアクセスできるようにするだけでなく、ネットワーク経由でも管理者しかアクセスできないようにしておいてください。また、ネットワークサービスについても、最小限のサービスのみを動作させるようにしてください。特に sshd などは動作させてはなりません。

6.2.1 最初の通信

Kerberos との最初の通信は、通常のネットワークシステムにおけるログイン処理ととても似ていて、ユーザ名を入力します。この情報とチケット発行サービスの名称が認証サーバ (Kerberos) に送信されます。認証サーバがそのユーザ名を知っていた場合は、今後使用するセッション鍵を乱数から生成し、クライアントとチケット発行サーバに送信します。あとは認証サーバが、チケット発行サーバのチケットを 用意し

ます。このチケットは、認証サーバとチケット発行サーバだけが知る セッション鍵ですべて暗号化され、下記の情報が含まれています：

- クライアントとチケット発行サーバの名称
- 現在の日時
- このチケットに設定された有効期限
- クライアントの IP アドレス
- 新しく生成したセッション鍵

このチケットはセッション鍵とともにクライアントに送り返されますが、この時点ではクライアントからの機密鍵を利用します。この機密鍵は Kerberos とクライアントだけが知るもので、これはユーザのパスワードから生成されるものです。クライアントがその応答を受け取ると、パスワードを尋ねられます。このパスワードは鍵に変換されたあと、認証サーバが送信したパケットを復号するのに使用され、ようやく暗号という「包装を解く」ことができることになります。また、パスワードはクライアントのメモリから消去されます。チケットに書かれた有効期限が切れるまでの間に、そのチケットの鍵を利用して次のチケットを取得することで、クライアントはその正当性を維持できることになります。

6.2.2 サービスの要求

ネットワーク内にあるサーバのサービスに対して要求を送信する際は、まずクライアントアプリケーションがサーバに対して自身の識別情報を証明する必要があります。そのため、アプリケーションは認証情報を生成します。認証情報には下記の情報が含まれています：

- クライアント側のプリンシパル
- クライアントの IP アドレス
- 現在の時刻
- チェックサム (クライアント側で選択)

これらの情報は、このサーバ用にクライアントが受信したセッション鍵で、すべて暗号化されます。その後、認証情報とチケットはサーバ側に送信されます。サーバ側ではセッション鍵のコピーを利用して認証情報の解読を行いません。この解読によってクライアントが必要としているサービスについて必要な情報をすべて得ることが

できるほか、チケット内に含まれている情報とも比較を行なうことができます。サーバ側ではチケットと認証情報が、いずれも同じクライアントから発信されたものであることを確認します。

サーバ側で何らかのセキュリティ対応が行なわれていない場合、処理の中でもこの段階がリプレイ攻撃として格好の標的になりかねません。誰かがネットワーク越しに盗聴した情報を元に、要求を再送信できてしまいます。これを排除するため、サーバ側では以前に受け付けたタイムスタンプおよびチケットを持つ要求を受け付けないようにしています。これに加えて、要求を行なった時点とは大きく異なる要求についても、無視されるようになっています。

6.2.3 相互認証

Kerberos の認証は双方向に利用することができます。クライアントがサーバに対して自身の正当性を主張するだけでなく、サーバがクライアントに対して正当性を主張することができます。そのため、サーバ側でも認証情報を送信します。クライアントの認証情報として受け取った中にあるチェックサムに対し、これを追加してサーバとクライアント間で共有されているセッション鍵で暗号化します。これにより、クライアントはサーバの正当性を応答の形で確認することができるため、これで相互の認証が完成したことになります。

6.2.4 チケットの発行—すべてのサーバとの通信

チケットは同時に 1 つのサーバでのみ使用されるように設計されています。これは、クライアント側で他のサービスを利用するような場合には、新しいチケットを取得しなければならないことを意味しています。Kerberos ではサーバごとにチケットを取得する仕組みが備わっていて、これを「チケット発行サービス」と呼んでいます。チケット発行サービスはその名の通りサービス (他のサービスと同様の位置づけ) で、他のサービスと同じアクセスプロトコルを利用します。アプリケーションがある特定のサービスにアクセスする必要がある場合は、まずチケット発行サーバに対して通信を行ないます。この要求には下記のものが含まれています：

- 要求を行なっているプリンシパル
- チケット許可チケット
- 認証情報

他のサーバと同様に、チケット発行サーバもチケット許可チケットと認証情報を確認します。これらが正しいものであると判断された場合は、チケット発行サーバは元々のクライアントとサーバとの間で使用する新しいセッション鍵を生成します。あとは新しいサーバに対するチケットを構築します。このチケットには下記のものが含まれます:

- クライアント側のプリンシパル
- サーバ側のプリンシパル
- 現在の時刻
- クライアントの IP アドレス
- 新しく生成されたセッション鍵

新しいチケットには有効期間が設定されていて、それぞれチケット許可チケットの残り有効期限か、サービスに対する有効期限のいずれか小さいほうが割り当てられます。クライアントは、チケット発行サービスが送信したチケットとセッション鍵を受け取りますが、この時点での応答は元々のセッション許可チケットから生成されるセッション鍵で暗号化されています。クライアントは新しいサービスに通信する際、ユーザのパスワード無しでこの応答を解読することができます。これにより、Kerberos はユーザ側に問い合わせを行なうことなく、チケットを継続的に取得できるようになっています。

6.2.5 Windows 2000 との互換性について

Windows 2000 には Kerberos 5 の Microsoft 版の実装が含まれています。openSUSE® では Kerberos 5 の MIT 版の実装を使用していますので、詳しい情報やガイドについては、MIT のドキュメンテーション 6.5 項「さらなる情報」(110 ページ)をお読みください。

6.3 Kerberos に対するユーザからの外観

理想的にはユーザと Kerberos との通信は、ログイン時に一度だけ発生するものです。ログイン処理にはチケット許可チケットの取得が含まれます。また、ログアウト時にはユーザの Kerberos チケットは自動的に破壊されます。これにより他のユーザがそのユーザになりすますことを難しくしています。なお、チケットの自動廃

棄の仕組みにより、ユーザのログインセッションが チケット許可チケットに設定された最大の有効期限 (妥当な値は 10 時間程度です) よりも長く持続した場合には、若干やっかいなことになります。しかしながら、ユーザは新しいチケット許可チケットを `kinit` の実行で 取得できるため、パスワードを再入力することで、追加の認証を行わずに Kerberos 側で必要なサービスにアクセスできるようになります。Kerberos で暗黙のうちに取得したチケットを一覧表示するには、`klist` を実行します。

下記には Kerberos 認証を使用するアプリケーションのうち、いくつかを紹介しています。これらのアプリケーションは、`krb5-apps-clients` パッケージをインストール後、`/usr/lib/mit/bin` または `/usr/lib/mit/sbin` に配置されます。これらはいずれも 汎用的な UNIX/Linux サービスを提供するほか、追加で Kerberos による透過的な 認証機能が備わっています：

- `telnet, telnetd`
- `rlogin`
- `rsh, rcp, rshd`
- `ftp, ftpd`
- `ksu`

これらのアプリケーションを利用するにあたっては、Kerberos が既に利用者の 正当性を確認しているため、パスワードを入力する必要はありません。また、`ssh` を Kerberos 対応付きでコンパイルしている場合も、一方のワークステーションで獲得したすべてのチケットを他方のワークステーションに転送 することができます。`ssh` を利用して他のワークステーションにログイン すると、`ssh` はチケットの暗号化された部分が新しい状況に合わせて調整 します。単純にワークステーション間でチケットをコピーするだけでは、チケット内にワークステーション固有の情報 (IP アドレス) が含まれるため 不十分です。XDM, GDM, KDM でもそれぞれ Kerberos に対応しています。Kerberos ネットワークアプリケーションについて、詳しくは <http://web.mit.edu/kerberos> にある *Kerberos V5 UNIX User's Guide* をお読みください。

6.4 Kerberos のインストールと管理

Kerberos の環境は複数のコンポーネントから構成されています。鍵配布センター (KDC) はすべての Kerberos 関連データを保持する中央データベースで、すべて

のクライアントは、ネットワーク経由での認証を正しく動作させるため、この KDC に依存しています。KDC とクライアントは、お使いの環境に合わせて 設定する必要があります:

一般的な準備

まずはネットワークの設定を確認し、6.4.1 項「Kerberos のネットワーク構造」(90 ページ) に書かれている最小限の要件を満たしていることを確認します。また、Kerberos の設定を行なうにあたって、適切な領域を選択します。詳しくは 6.4.2 項「Kerberos の領域 (realm) 選択」(91 ページ) をお読みください。さらに、KDC として動作させるマシンを注意して設定し、強固なセキュリティ 設定を行ないます。こちらについて、詳しくは 6.4.3 項「KDC ハードウェアの設定」(92 ページ) をお読みください。最後に、ネットワーク内で信頼性の高い時刻提供源を用意し、すべてのチケット に正しいタイムスタンプが付与されるようにします。こちらについては 6.4.4 項「時刻同期の設定」(93 ページ) をお読みください。

基本設定

まずは KDC とクライアントの設定を行ないます。詳しくは 6.4.5 項「KDC の設定」(94 ページ) と 6.4.6 項「Kerberos クライアントの設定」(97 ページ) をお読みください。また、Kerberos サービスに対してリモートからの管理を有効にし、KDC の起動しているマシンに対して物理的なアクセスを不要にします。詳しくは 6.4.7 項「リモートからの Kerberos 管理の設定」(102 ページ) をお読みください。最後に、お使いの領域内の各サービスに対して、プリンシパルを作成します。詳しくは 6.4.8 項「Kerberos サービスプリンシパルの作成」(104 ページ) をお読みください。

Kerberos 認証の有効化

お使いのネットワーク内にある様々なサービスを Kerberos 対応にすることができます。PAM を使用するアプリケーションに対して Kerberos の パスワードチェック機能を設定するには、6.4.9 項「PAM の Kerberos 対応の有効化」(105 ページ) の手順を行なってください。また、SSH や LDAP に対して Kerberos の認証を設定するには、6.4.10 項「SSH に対する Kerberos 認証の設定」(106 ページ) や 6.4.11 項「LDAP と Kerberos」(107 ページ) の手順を行なってください。

6.4.1 Kerberos のネットワーク構造

Kerberos を完全に機能させるためには、Kerberos を利用する環境が下記の 要件を満たしていなければなりません:

- お使いのネットワーク内に DNS サーバが存在し、クライアントとサーバが 相互に 名前を解決できるようになっていること。DNS の設定について、詳しくは 第15 章 ドメインネームシステム (↑リファレンス) をお読みください。
- お使いのネットワーク内にタイムサーバが存在していること。Kerberos の 仕組みを正しく動作させるには、Kerberos のチケットに書かれたタイム スタンプが正しくなければならないため、時刻の同期は必須条件です。NTP の設定方法について、詳しくは 第17章 *NTP を利用した時刻同期* (↑リファレンス) をお読みください。
- 鍵配布センター (KDC) を Kerberos 構造の中心要素として用意すること。ここには Kerberos のデータベースが保持されます。このマシンに対する セキュリティはもっとも厳重なものとし、一切の攻撃を受けないように すべきものです。
- クライアント側で Kerberos 認証を使用するように設定すること。

下記の図では、Kerberos の仕組みを構築するにあたって最小限のものを 揃えた 場合の例を示しています。ネットワークの規模や配置環境に合わせて、下記の設計を調整すべきものです。

図 6.1 Kerberos のネットワーク構造

ヒント: サブネットルーティングの設定

図6.1「Kerberos のネットワーク構造」(91 ページ) のような構成で構築する際は、2 つの サブネット (192.168.1.0/24 と 192.168.2.0/24) の間でルーティングを 設定してください。YaST でのルーティング設定方法について、詳しくは 項「ルーティング (経路制御) の設定」(第13章 ネットワークの基礎, ↑リファレンス) をお読みください。

6.4.2 Kerberos の領域 (realm) 選択

Kerberos による認証の範囲のことを、領域 (realm) と呼びます。これは名前によって識別するもので、たとえば EXAMPLE.COM のような 形式をとる場合があるほか、単に ACCOUNTING のような 名前にする場合があります。Kerberos は大文字と小文字を区別する仕組みであるため、example.com と EXAMPLE.COM は別々の領域を意味することになります。大文字と小文字のどちらを使用しても かまいませんが、一般的には領域名に大文字を使用します。

DNS のドメイン名 (または ACCOUNTING.EXAMPLE.COM のような サブドメイン) を使用するのも良い考えです。下記に示しているとおり、Kerberos のクライアントで DNS を設定し、KDC やその他の Kerberos サービスを発見するように設定すると、設定を単純化することができます。これを行なうには、領域の 名称を DNS のサブドメインとして設定するのがよいでしょう。

なお、DNS の名前領域とは異なり、Kerberos は階層構造をとることができます。たとえば EXAMPLE.COM という領域があるとすると、DEVELOPMENT や ACCOUNTING のような「副領域」をその下に作成し、それらを EXAMPLE.COM にあるプリンシパルの副次的な領域とすることはできません。その代わり、これらを 3 種類の個別の領域とし、ユーザに対して領域をまたがる認証を設定し、一方の領域のユーザやサーバを他方のユーザやサーバと協調的に動作させることができます。

単純化のため、下記の説明では企業全体で 1 つの領域を設定するものと仮定します。この章の残りの部分では、EXAMPLE.COM という領域を使用するものとして説明します。

6.4.3 KDC ハードウェアの設定

Kerberos を使用するために最初に用意しなければならないことは、鍵配布センター (KDC と略します) として動作するマシンです。このマシンは Kerberos のユーザとパスワードなど、すべての情報に関するデータベースを保持します。

KDC はセキュリティ面で最も重要な部分です。もしも誰かがそれに侵入できると、Kerberos で保護しているすべてのユーザアカウントとデータ構造を取得できてしまいます。Kerberos のデータベースにアクセスできてしまえば、そのデータベースに書かれている任意のプリンシパルに偽装することもできてしまいます。そのため、このマシンに対するセキュリティはできるだけ厳しく設定する必要があります：

- 1 サーバマシンは物理的に機密を保つことのできる場所、たとえば限られたごく少数の人間しか入ることのできない、施錠されたサーバ室などに配置してください。
- 2 KDC を除くネットワークアプリケーションは一切動作させないでください。これはサーバ／クライアントの両方を含みます。たとえば KDC では NFS 経由でのファイルシステムのインポートや、DHCP によるネットワーク設定の取得などを行なうべきではありません。
- 3 まずは最小限のシステム構成でインストールを行ない、インストール済みのパッケージ一覧を取得して、不要なパッケージはすべてアンインストールしてください。これには inetd, portmap, cups のほか、X ベースのサーバ類をすべて含

みます。SSH サーバについても、潜在的なセキュリティリスクとして インストールすべきではありません。

- 4 X サーバを動作させ、グラフィカルなログインを提供してはなりません。これも潜在的なセキュリティリスクになりうるためです。Kerberos 側で 自身の管理インターフェイスを用意しています。
- 5 /etc/nsswitch.conf ファイル内に、ユーザや グループの参照先がローカルファイルだけとなるように設定してください。具体的には、passwd と group は下記のようになります：

```
passwd:      files
group:       files
```

/etc ディレクトリ内にある passwd, group, shadow の各ファイルを 編集し、行頭が + であるもの (これらは NIS を参照する 項目です) をすべて削除してください。

- 6 /etc/shadow ファイルを編集し、root を除くすべてのアカウントを無効化してください。具体的には、パスワード欄を * または ! の 1 文字に置き換えます。

6.4.4 時刻同期の設定

Kerberos が正しく動作するようにするため、ネットワーク内の全サーバについて、システム時刻を特定の時刻発信源で同期できていることを確認してください。これは Kerberos で認証情報のリプレイを防ぐために、重要な項目です。攻撃者がネットワーク上を流れる Kerberos の認証情報を盗聴すると、それを 利用してそのユーザを偽装することができてしまうためです。Kerberos ではこれを防ぐために複数の防御を備えています。その防御のうちの 1 つがタイム スタンプです。これをチケット内に書き込んでおくことで、チケットを受信する サーバが現在時刻とは異なるチケットを受け取った場合、これを無視することで リプレイを防いでいます。

また、Kerberos ではタイムスタンプの比較にあたって、一定量の余裕を持たせています。しかしながら、コンピュータの時刻はその正確性を維持するのに不十分な作りになっています。場合によっては 1 週間で 30 分も狂うような場合さえあります。このような理由から、ネットワーク内に存在するすべてのホストでは 特定の時刻発信源による同期を設定してください。

これを実現するのに最も簡単な方法は、いずれか 1 台のサーバに NTP 時刻サーバを インストールすることです。また、残りのマシンではクライアントモードで NTP デーモンを動作させるか、もしくは 1 日 1 回程度 ntpdate を動作させるか (こちら

の方法は少数のクライアントしか存在していない場合に 限られます) を設定します。KDC それ自身についても、同じ時刻発信源に対して 同期を設定する必要があります。NTP デモンをそのマシンで動作させるのは セキュリティリスクとなりうるものであるため、cron を利用して 1 日 1 回 程度の同期を設定するのがよいでしょう。NTP クライアントの設定について、詳しくは 項「YaST を利用した NTP クライアントの設定」(第17章 *NTP を利用した時刻同期*, ↑リファレンス) に書かれています。

また、別の方法として、専用の NTP サーバを用意してそこにハードウェアの 時刻参照源を設定し、ここから NTP デモンによる時刻同期を設定する方法が あります。

また、Kerberos に対してタイムスタンプの比較の際、時刻の最大誤差を調整 する方法もあります。この値 (*clock skew* (時刻誤差) と呼びます) は `krb5.conf` ファイルから設定することができます。詳しくは「時計の誤差の調整」(102 ページ) をお読みください。

6.4.5 KDC の設定

この章では、KDC のインストールと初期設定、および管理者プリンシパルの 作成方法について述べています。この作業は複数の手順から構成されています：

- 1 RPM のインストール** KDC として動作させるマシンに対しては、下記のソフトウェアパッケージを インストールします: `krb5`, `krb5-server`, `krb5-client`
- 2 設定ファイルの調整** `/etc/krb5.conf` と `/var/lib/kerberos/krb5kdc/kdc.conf` の各ファイルに 対して、お使いの環境に合わせた調整を行ないます。これらのファイルには KDC 上でのすべての情報が含まれています。
- 3 Kerberos データベースの作成** Kerberos はすべてのプリンシパルに対する識別情報と、認証の際に必要な すべての機密鍵を保持します。詳しくは 6.4.5.1 項「データベースの設定」(95 ページ) をお読みください。
- 4 ACL ファイルの調整: 管理者の追加** KDC 上にある Kerberos のデータベースは、リモートから管理することができます。データベースに対して不正なアクセスが無いようにするため、Kerberos ではアクセス制御リストを使用します。管理者のプリンシパルに 対しては、明示的にリモートアクセスを許可してデータベースを管理できるようにしなければなりません。Kerberos の ACL ファイルは、`/var/lib/kerberos/krb5kdc/kadm5.acl` にあります。詳しくは 6.4.7 項「リモートからの Kerberos 管理の設定」(102 ページ) をお読みください。
- 5 Kerberos データベースの調整: 管理者の追加** Kerberos を実行するには、少なくとも 1 人以上の管理者プリンシパルを設定 する必要があります。プリンシパ

ルは KDC を起動する前に追加しなければなりません。詳しくは 6.4.5.2 項「プリンシパルの作成」(96 ページ) をお読みください。

6 Kerberos デーモンの起動 KDC のソフトウェアをインストールして必要な設定を完了したら、Kerberos のデーモンを起動し、領域内に Kerberos のサービスを提供します。詳しくは 6.4.5.3 項「KDC の起動」(96 ページ) をお読みください。

7 自分自身のプリンシパルの作成 最後に自分自身のプリンシパルを追加する必要があります。詳しくは 6.4.5.2 項「プリンシパルの作成」(96 ページ) をお読みください。

6.4.5.1 データベースの設定

次の作業は、Kerberos がプリンシパルに関するすべての情報を保持するデータベースを作成することです。データベースにはマスターキーと呼ばれる、データベースを保護するため (特にテープなどへのバックアップ時) の鍵を設定します。マスターキーはパスワードから構成されるもので、stash ファイルと呼ばれる場所に保管します。これにより、KDC の起動時に パスワードを入力しなくても起動できるようになっています。なお、パスワードは辞書や本などをランダムに開いて書いてあった文字列を利用する など、適切なものを設定してください。

Kerberos データベース (/var/lib/kerberos/krb5kdc/principal) をテープなどにバックアップする場合は、stash ファイル (/var/lib/kerberos/krb5kdc/.k5.EXAMPLE.COM にあります) はバックアップしてはなりません。そうでないと、テープを読み取るだけでデータベースを解読できることになってしまいます。そのため、パスワードは安全な場所にコピーを作成しておくか、もしくはその他の安全な方法で保管してください。これはパスワードについても何らかの方法で保管を行っておかないと、忘れたり失われたりした場合に復元できなくなってしまうためです。

stash ファイルとデータベースを作成するには、下記のように実行します:

```
kdb5_util create -r EXAMPLE.COM -s
```

すると、下記のように出力が行なわれます:

```
Initializing database '/var/lib/kerberos/krb5kdc/principal' for realm 'EXAMPLE.COM',
master key name 'K/M@EXAMPLE.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key: ❶
Re-enter KDC database master key to verify: ❷
```

- ❶ マスターパスワードを入力します。
- ❷ パスワードを再入力します。

確認を行なうには、下記の一覧コマンドを入力します:

```
kadmin.local
```

```
kadmin> listprincs
```

データベース内には、Kerberos が内部的に使用する複数のプリンシパルが存在しています:

```
K/M@EXAMPLE.COM  
kadmin/admin@EXAMPLE.COM  
kadmin/changepw@EXAMPLE.COM  
krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

6.4.5.2 プリンシパルの作成

次に自分自身に対する Kerberos のプリンシパルを 2 つ作成します。1 つは 日々の作業用、もう 1 つは Kerberos 関連の管理作業を行なうためのものです。ここではログイン名を `geeko` と仮定します。下記の手順で実施します:

```
kadmin.local
```

```
kadmin> ank geeko
```

下記のような出力が表示されます:

```
geeko@EXAMPLE.COM's Password: ❶  
Verifying password: ❷
```

- ❶ `geeko` に対して設定するパスワードを入力します。
- ❷ `geeko` に対して設定するパスワードを再入力します。

次に `kadmin` のプロンプトから `ank geeko/admin` と入力し、`geeko/admin` という名前のプリンシパルを作成します。ユーザ名に `admin` という接尾語が付属していますが、これは *ロール* と呼ばれるものです。このロールは Kerberos のデータベース管理の際に使用します。ユーザにはそれぞれの 目的に合わせて、複数のロールを設定します。ロールは似たような 名前ですが、異なるアカウントになります。

6.4.5.3 KDC の起動

KDC デーモンと `kadmin` デーモンを起動します。デーモンを手動で起動するには、それぞれ `rckrb5kdc start` および `rckadmind start` と入力します。なお、システ

ムが 起動したときに KDC と kadmind を起動させたい場合は、`insserv krb5kdc` および `insserv kadmind` と入力する か、もしくは YaST のランレベルエディタを使用します。

6.4.6 Kerberos クライアントの設定

DNS や NTP などのインフラが整っていて、かつ KDC の設定と起動が完了して いれば、ようやくクライアントマシン側の設定を行なうことができます。YaST を利用し て Kerberos クライアントを設定するか、もしくは下記に示す 2 つの手作業で実施 することができます。

6.4.6.1 YaST を利用した Kerberos クライアントの設定

Kerberos クライアントとして動作させるにあたって、関連する全ての設定を 手作業 で変更するよりも、YaST にその作業を行なわせたほうが便利です。Kerberos クラ イアントは、お使いのマシンをインストールする際に設定する ことができるほか、イン ストール済みのマシンでも設定することができます。具体的には下記の手順で行な います:

- 1 root でログインし、ネットワークサービス > Kerberos クライアント を選択します (図6.2「YaST: Kerberos クライアントの基本設定」(98 ページ))。
- 2 Kerberos を使用する を選択します。
- 3 DNS ベースの Kerberos クライアントを設定するには、下記の手順で行ないま す:

3a DNS ベースの固定の Kerberos クライアントを設定します。

注記: DNS サポートの使用

DNS サーバが Kerberos 関連のデータを提供していない場合は、*実行時に DNS から設定データを取得する* のオプションを 選択することはできません。

3b チケット関連の問題や OpenSSH への対応、時刻同期や拡張 PAM 設定 などを 行なう場合は、*詳細設定* を押します。

- 4 固定の Kerberos クライアントを設定するには、下記の手順で行ないます:

4a まずはお使いの環境に合わせて、既定のドメイン、既定の領域、KDC サーバアドレス をそれぞれ設定します。

4b チケット関連の問題や OpenSSH への対応、時刻同期や拡張 PAM 設定 などを 行なう場合は、詳細設定 を押します。

図 6.2 YaST: Kerberos クライアントの基本設定

Kerberos クライアントの設定

☐ Kerberos を使用しない (N)
☒ Kerberos を使用する (Y)
☒ 実行時に DNS から設定データを取得する (S)

基本的な Kerberos の設定

既定のドメイン (D)	既定の領域 (M)
example.com	EXAMPLE.COM
KDC サーバアドレス (K)	
kdc.example.com	

詳細設定 (V)...

ヘルプ (H) キャンセル (C) OK (O)

詳細設定 ダイアログでチケット関連のオプションを 設定する (図6.3「YaST: Kerberos クライアントの高度な設定」(99 ページ)) には、下記のいずれかを 設定します:

- まずは 既定のライフタイム と 既定の更新可能なライフタイム について、日／時／分 単位で設定します (それぞれ数字の後に空白を開けずに d, h, m を入力すると、それぞれ日／時／分単位での設定になります)。
- 完全な識別情報を転送できるようにする (チケットを他のホストで使用する 場合) には、転送可能フラグ を選択します。
- また、特定のチケットを転送できるようにするには、代理可能フラグ を選択します。
- お使いの OpenSSH に対して、Kerberos 認証への対応を行なわせたい場合は、関連するチェックボックスに印を入れます。これにより、クライアントは SSH サーバに対し、Kerberos チケットを利用するようになります。

- Kerberos 認証について特定のユーザアカウントの範囲を除外したい場合は、**最小ユーザ ID** の値を指定します。この値は、たとえば システム管理者 (root) を ログインさせたくない場合などに利用します。
- タイムスタンプとお使いのホスト時刻との間で、許される時刻差を設定するには、**クロックのズレ** の値を設定します。
- NTP サーバを利用したシステム時刻の同期を行ないたい場合は、**NTP の設定** ボタンを押します。このボタンを押すと、項「YaST を利用した NTP クライアントの設定」(第17章 *NTP を利用した時刻同期*, ↑リファレンス) で説明している YaST NTP クライアントダイアログが表示されます。設定が完了すると、YaST は全ての必要な設定を行なった後、Kerberos クライアントが利用できるようになります。

図 6.3 YaST: Kerberos クライアントの高度な設定

Kerberos クライアントの詳細設定

PAM 設定 熟練者向け PAM 設定 PAM サービス

チケットの属性

既定のライフタイム (D)
1d

既定の更新可能なライフタイム (F)
1d

転送可能フラグ (W) 代理可能フラグ (P)
すべてのサービス すべてのサービスを除外

☐ OpenSSH クライアント用の Kerberos サポート (S)
☒ 不明なユーザを無視 (I)

最小ユーザ ID (U)
1

クロックのズレ (L)
300 NTP の設定 (N)...

ユーザデータの設定 (O) ▼

ヘルプ (H) キャンセル (C) OK (O)

熟練者向け PAM 設定 と PAM サービス のタブでの設定方法について、詳しくは公式のドキュメンテーション (6.5項「さらなる情報」(110 ページ) に参照先が書かれています) と `man 5 krb5.conf` のマニュアルページ (`krb5-doc` パッケージに含まれています) を お読みください。

6.4.6.2 Kerberos クライアントの手動設定

Kerberos を手作業で設定するにあたっては、2 種類の方法で設定することができます。1 つは `/etc/krb5.conf` ファイル内で 固定の設定を記述する方法、もう 1 つ

は DNS による動的な設定を行なう 方法です。DNS の設定の場合は、Kerberos アプリケーションが DNS レコードを利用して、KDC サービスの場所を特定します。固定の設定の 場合は、krb5.conf 内に KDC サーバのホスト名を 指定します (KDC のサーバ名が変更になったり、領域などが変わったりした 場合には、適宜更新が必要になります)。

DNS ベースの設定のほうがより柔軟な仕組みであり、設定にかかる手間も大きく 省くことができます。しかしながら、領域 (realm) 名がお使いの DNS ドメイン か、もしくはサブドメインと一致している必要があります。また、DNS での Kerberos 設定は、セキュリティ面で若干の問題があります。攻撃者が DNS サービスを攻撃したような場合 (ネームサーバのサービスを止めてしまったり、DNS レコードを偽造してしまったり)、Kerberos のインフラストラクチャが 破壊されてしまいます。ただし、このような場合でも、最悪のケースを想定 してもサービスが停止するだけの被害範囲になります。また、固定の設定でも、krb5.conf 内に IP アドレスではなくホスト名で指定 した場合には、似たような問題が発生する可能性があります。

固定の設定

Kerberos を設定するための唯一の方法は、/etc/krb5.conf を編集することです。このファイルは既定でインストールされるもので、様々な設定例が書かれています。設定を行なうにあたっては、これらを 忘れずに消去してください。また、krb5.conf は複数のセクションから構成されていて、それぞれの章の名前は大括弧で 括られています (例: [this])。

Kerberos クライアントの設定を行なうには、下記のようなセクションを krb5.conf に追加します (ここで、kdc.example.com は KDC のホスト名とします):

```
[libdefaults]
    default_realm = EXAMPLE.COM

[realms]
    EXAMPLE.COM = {
        kdc = kdc.example.com
        admin_server = kdc.example.com
    }
```

default_realm の行では、Kerberos アプリケーション での既定の領域を設定します。複数の領域を設定したい場合は、[realms] セクション内で追加します。

このファイルでは、アプリケーション側でホスト名と領域をどのように 対応づけるのかについても設定をします。たとえばリモートのホストに 接続するような場合、Kerberos のライブラリはそのホストにどの領域を 使用すべきかを知っておく必要があるためです。これらの設定は [domain_realms] セクションで行ないます:

```
[domain_realm]
```

```
.example.com = EXAMPLE.COM  
www.foobar.com = EXAMPLE.COM
```

上記の設定では、example.com の DNS ドメインに対しては EXAMPLE.COM の Kerberos 領域を利用する意味になります。また、www.foobar.com という外部のホストは、EXAMPLE.COM の領域に対するメンバーであるものと判断させる意味になります。

DNS ベースの設定

DNS ベースの Kerberos 設定では、DNS の SRV レコードを頻繁に使用する形になります。詳しくは <http://www.ietf.org> にある (RFC2052) *A DNS RR for specifying the location of services* (英語) をお読みください。日本語訳については、<http://www5d.biglobe.ne.jp/~stssk/rfc/rfc3597j.html> などの参考資料をお読みください。

Kerberos においては、SRV レコードの名称は常に `_service._proto.realm` という形式で記述します。ここで realm の部分には Kerberos の領域が入ります。また、DNS におけるドメイン名は大文字と小文字を区別しませんが、この設定方法を使用した場合は、大文字と小文字を区別する Kerberos の領域の仕組みが適用されることに注意してください。また、`_service` にはサービス名 (KDC やパスワードサービスなどに通信する際には、異なる名前を使用します) が入ります。さらに `_proto` は `_udp` または `_tcp` のいずれかですが、全てのサービスが両方のプロトコルに対応しているわけではないことに注意してください。

SRV レコードのデータ部は、優先順位の値とウェイト値、ポート番号とホスト名から構成されています。優先順位は、複数のホストの中からいずれかを選択する際、どれを優先的に選択するのかを指定します (低い値ほど高い優先順位を意味します)。また、ウェイト値は同じ優先順位の複数のサーバに対して、ある種の負荷分散を行なうための値です。設定を必要としない場合は、0 を指定します。

現時点では、MIT Kerberos はサービスを検出する際に下記の名前を検索します:

`_kerberos`

この名前は KDC デーモンの場所を探す際に利用します (認証およびチケット発行サーバ)。たとえば下記ようになります:

```
_kerberos._udp.EXAMPLE.COM. IN SRV 0 0 88 kdc.example.com.  
_kerberos._tcp.EXAMPLE.COM. IN SRV 0 0 88 kdc.example.com.
```

`_kerberos-adm`

この名前はリモート管理サービスの場所を探す際に利用します。たとえば下記ようになります:

```
_kerberos-adm._tcp.EXAMPLE.COM. IN SRV 0 0 749 kdc.example.com.
```

kadmind は UDP に対応していないため、_udp のレコードは設定しません。

固定の設定ファイルでは、クライアントに対して特定のホストが example.com の DNS ドメイン内に存在しない場合でも、EXAMPLE.COM の領域にあることを知らせる方法があります。これは TXT レコードで設定するもので、_kerberos.hostname のような形式で指定します。たとえば下記のようになります：

```
_kerberos.www.foobar.com. IN TXT "EXAMPLE.COM"
```

時計の誤差の調整

時刻誤差 とはチケットを受け入れる際、ホスト側の システム時刻とチケットのタイムスタンプとの間で、どれだけのズレを許容 するのかを指定する値です。通常は時刻誤差として 300 秒 (5 分) を設定 します。これは、チケットに書かれた時刻とサーバの時刻との間で、5 分間のズレを許容し、5 分前のものから 5 分後のものまで受け入れる意味に なります。

NTP を利用して全てのホストの時刻同期を行なっている場合、この値を 1 分 程度など、小さく設定することもできます。時刻誤差の値は /etc/krb5.conf 内で下記のように設定します：

```
[libdefaults]
    clockskew = 60
```

6.4.7 リモートからの Kerberos 管理の設定

Kerberos のデータベースに対して、KDC のコンソールに直接アクセスする以外の方法でプリンシパルを追加したり削除したりできるようにするには、/var/lib/kerberos/krb5kdc/kadm5.acl ファイルを編集して Kerberos 管理サーバを設定します。ACL (アクセス制御リスト) ファイルでは、権限を設定することで正確な制御を行なうことができます。詳しくは man 8 kadmind で表示されるマニュアル ページをお読みください。

ここでは自分自身に対して権限を追加し、データベースを管理できるようにするものとします。下記の行をファイルに追加します：

```
geeko/admin *
```

ここで geeko は自分自身のユーザ名に置き換えてください。設定を反映させるには、kadmind を再起動する必要があります。

これで kadmin ツールをリモートから実行することで、Kerberos の管理作業を行なうことができるようになりました。まずは管理者ロールのためのチケットを取得し、そのチケットを利用して kadmin サーバに接続します:

```
kadmin -p geeko/admin
Authenticating as principal geeko/admin@EXAMPLE.COM with password.
Password for geeko/admin@EXAMPLE.COM:
kadmin: getprivs
current privileges: GET ADD MODIFY DELETE
kadmin:
```

getprivs コマンドを利用することで、その時点で持っている権限を確認することができます。上記の例では、全ての権限を持っていることを示しています。

ここではたとえば、geeko のプリンシパルを修正するとします:

```
kadmin -p geeko/admin
Authenticating as principal geeko/admin@EXAMPLE.COM with password.
Password for geeko/admin@EXAMPLE.COM:

kadmin: getprinc geeko
Principal: geeko@EXAMPLE.COM
Expiration date: [never]
Last password change: Wed Jan 12 17:28:46 CET 2005
Password expiration date: [none]
Maximum ticket life: 0 days 10:00:00
Maximum renewable life: 7 days 00:00:00
Last modified: Wed Jan 12 17:47:17 CET 2005 (admin/admin@EXAMPLE.COM)
Last successful authentication: [never]
Last failed authentication: [never]
Failed password attempts: 0
Number of keys: 2
Key: vno 1, Triple DES cbc mode with HMAC/sha1, no salt
Key: vno 1, DES cbc mode with CRC-32, no salt
Attributes:
Policy: [none]

kadmin: modify_principal -maxlife "8 hours" geeko
Principal "geeko@EXAMPLE.COM" modified.
kadmin: getprinc joe
Principal: geeko@EXAMPLE.COM
Expiration date: [never]
Last password change: Wed Jan 12 17:28:46 CET 2005
Password expiration date: [none]
Maximum ticket life: 0 days 08:00:00
Maximum renewable life: 7 days 00:00:00
Last modified: Wed Jan 12 17:59:49 CET 2005 (geeko/admin@EXAMPLE.COM)
Last successful authentication: [never]
Last failed authentication: [never]
Failed password attempts: 0
Number of keys: 2
Key: vno 1, Triple DES cbc mode with HMAC/sha1, no salt
```

```
Key: vno 1, DES cbc mode with CRC-32, no salt
Attributes:
Policy: [none]
kadmin:
```

上記の例では、チケットの有効期限の最大値を 8 時間に設定しています。kadmin コマンドと利用可能なオプションの一覧について、詳しくは krb5-doc パッケージ内をご覧ください。または、<http://web.mit.edu/kerberos/www/krb5-1.8/krb5-1.8.3/doc/krb5-admin.html#Kadmin%20Options> や man8 kadmin のマニュアルページにも 記述があります。

6.4.8 Kerberos サービスプリンシパルの作成

ここまでの章では、ユーザの認証情報のみを取り扱ってきました。Kerberos 互換の サービスでは、サービスそれ自身がクライアントに対して認証を行なう必要もあります。そのため、サービス側のプリンシパルについても、提供範囲の領域にある Kerberos データベースに登録しておかなければなりません。たとえば ldap.example.com で LDAP サービスを提供している場合、サービスプリンシパルとして ldap/ldap.example.com@EXAMPLE.COM を登録し、全クライアントに 対して認証できるようにする必要があります。

サービスプリンシパルの表記は、サービス/ホスト名@領域 のように記述します。ここで *hostname* には、ホストの 完全修飾ドメイン名を記述します。

サービス記述子として利用可能なものは下記のとおりです：

サービス記述子	サービス
host	Telnet, RSH, SSH
nfs	NFSv4 (Kerberos 対応のもの)
HTTP	HTTP (Kerberos 対応のもの)
imap	IMAP
pop	POP3
ldap	LDAP

サービスプリンシパルはユーザプリンシパルに似ていますが、大きく異なる箇所があります。ユーザプリンシパルの鍵はパスワードで保護されていて、ユーザが KDC からチケット許可チケットを受け取ると、チケットを復号化するためにパスワードの入力が必要になります。このような仕組みはサービスプリンシパルには非常に不便なものになってしまい、このままであったとすると、たとえば SSH デーモンを動作させている場合、8 時間ごとにパスワードの入力が必要になってしまいます。

サービスプリンシパルでは、初期のチケットを復号化するのに必要な鍵は管理者が KDC から一度だけ取り出すものとし、*keytab* と呼ばれるローカルファイルに保管します。SSH デーモンのようなサービスであれば、この鍵を読み込んで使用し、必要であれば新しいチケットを自動的に取得します。*keytab* ファイルは、既定では `/etc/krb5.keytab` に配置されています。

`jupiter.example.com` に対してホストのサービスプリンシパルを作成するには、`kadmin` セッション内で下記のコマンドを入力します：

```
kadmin -p geeko/admin
Authenticating as principal geeko/admin@EXAMPLE.COM with password.
Password for geeko/admin@EXAMPLE.COM:
kadmin: addprinc -randkey host/jupiter.example.com
WARNING: no policy specified for host/jupiter.example.com@EXAMPLE.COM;
defaulting to no policy
Principal "host/jupiter.example.com@EXAMPLE.COM" created.
```

新しいプリンシパルに対してパスワードを設定する代わりに `-randkey` フラグを設定し、`kadmin` に対してランダムな鍵を生成するように指示しています。これはこのプリンシパルに対して、一切のユーザ操作を無くしたい意図があることによります。つまり、これはマシンに対するサーバアカウントということになります。

最後に鍵を取り出し、ローカルの *keytab* ファイル `/etc/krb5.keytab` 内に保管します。このファイルはスーパーユーザが所有しているべきものであるため、下記のコマンドを実行するには `kadmin` シェルを `root` で起動しなければなりません：

```
kadmin: ktadd host/jupiter.example.com
Entry for principal host/jupiter.example.com with kvno 3, encryption type Triple
DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/jupiter.example.com with kvno 3, encryption type DES
cbc mode with CRC-32 added to keytab WRFILE:/etc/krb5.keytab.
kadmin:
```

上記の処理が完了したら、忘れずに `kdestroy` コマンドを利用し、`kinit` で取得した管理チケットを破棄してください。

6.4.9 PAM の Kerberos 対応の有効化

openSUSE® では pam_krb5 という PAM モジュールが 同梱されていて、これを利用して Kerberos のログインとパスワード更新を行なうことができるようになっています。このモジュールは、コンソールでの ログインや su, KDM などのグラフィカルなログインアプリケーション (ユーザがパスワードを提供し、認証対象のアプリケーションに対して初期の Kerberos チケットを取得させたい場合) で利用することができます。Kerberos に対応するよう PAM を設定するには、下記のコマンドを入力します:

```
pam-config --add --krb5
```

上記のコマンドは既存の PAM 設定ファイルに対して pam_krb5 モジュールを追加し、正しい順序で呼び出されるように設定するものです。pam_krb5 の使用方法についてより細かい調整を行なう には、/etc/krb5.conf ファイルを編集し、既定のアプリケーションを pam に追加してください。詳しくは man 5 pam_krb5 で表示されるマニュアルページをお読みください。

pam_krb5 モジュールは、ユーザ認証の一部として Kerberos チケットを受け取るようなネットワークサービスに対して、特に設計 されているわけではありません。これは全く異なる問題で、下記の章で詳しく 説明しています。

6.4.10 SSH に対する Kerberos 認証の設定

OpenSSH はプロトコルバージョン 1 と 2 の両方で Kerberos 認証に対応しています。バージョン 1 では、Kerberos チケットを送信するのに特殊なプロトコル メッセージを利用します。バージョン 2 では Kerberos を直接利用するような 仕組みにはなっており、その代わりに GSSAPI (General Security Services API; 汎用セキュリティサービス API) を利用するようになっています。これは Kerberos 固有のプログラミングインターフェイスというわけではなく、Kerberos や SPKM のような公開鍵認証システムなどのような、認証システムの特異性を隠蔽するための仕組みです。ただし、同梱の GSSAPI では Kerberos にのみ対応しています。

sshd で Kerberos 認証を利用するには、/etc/ssh/sshd_config ファイルを編集して下記のオプションを設定します:

```
# プロトコルバージョン 1 向けの設定
#
# KerberosAuthentication yes
# KerberosTicketCleanup yes

# プロトコルバージョン 2 向けの設定 - できればこちらを利用してください
GSSAPIAuthentication yes
GSSAPICleanupCredentials yes
```

設定を変更したら、rcsshd restart と入力して SSH デーモンを再起動します。

プロトコルバージョン 2 で Kerberos 認証を利用する場合は、クライアント側 でも同様に有効化のための設定を行なってください。設定にはシステム全体の 設定ファイルである `/etc/ssh/ssh_config` ファイルを 編集する方法があるほか、ユーザーレベルでも `~/.ssh/config` ファイルを編集して対応する方法があります。いずれのファイルとも、対象の ファイルに `GSSAPIAuthentication yes` というオプションを追記します。

これで Kerberos 認証を利用して接続を行なうことができるようになりました。`klist` コマンドを利用することで、有効なチケットを取得 できているかどうかを確認でき、SSH サーバに接続することができます。SSH プロトコルのバージョン 1 を使用したい場合は、コマンドラインで `-1` というオプションを指定してください。

ヒント: 追加情報

`/usr/share/doc/packages/openssh/README.kerberos` ファイルでは、OpenSSH と Kerberos の作用についてより詳しい説明が 書かれています。

6.4.11 LDAP と Kerberos

Kerberos を使用する場合、ユーザの情報 (ユーザ ID, グループ, ホーム ディレクトリなど) をネットワーク内に配布するには、LDAP を利用するのが 唯一の方法です。LDAP を利用するにあたっては、パケットの盗聴やその他の 攻撃から防御するため、強度な暗号化メカニズムを設定する必要があります。1 つの解決策として、LDAP の通信そのものに対しても Kerberos を利用する 方法があります。

OpenLDAP では多くの認証手順を SASL 経由で実装しています。SASL とは Simple Authentication Session Layer の略で、シンプルな認証機能を提供するものです。SASL は基本的には認証に使用するよう設計されたネットワーク プロトコルですが、SASL の実装は `cyrus-sasl` と呼ばれるもので、数多くの 認証方法に対応した実装になっています。なお、Kerberos の認証は GSSAPI (General Security Services API) 経由で行ないます。既定では GSSAPI の SASL プラグインはインストールされていないため、LDAP を利用する際には YaST から `cyrus-sasl-gssapi` パッケージを インストールしてください。

Kerberos に対して OpenLDAP サーバへの接続を有効にするには、`ldap/ldap.example.com` という名称のプリンシパルを作成し、これを `keytab` 内に追加します。

既定では、LDAP サーバ `slapd` は `ldap` のユーザおよびグループで 動作し、`keytab` ファイルは `root` でのみ読み込むことができます。そ

のため、LDAP 側の設定を変更して root で動作するように設定するか、もしくは keytab ファイルを ldap グループから読み込むことができるように設定する必要があります。後者は /etc/sysconfig/openldap ファイル内で OPENLDAP_KRB5_KEYTAB の変数に対し、keytab ファイルの場所を指定し、かつ OPENLDAP_CHOWN_DIRS の変数に対して yes を設定した場合、OpenLDAP の起動スクリプト (/etc/init.d/ldap) が自動的に設定します。これらの設定はあらかじめ設定済みのため、特に変更を行なう必要はありません。また、OPENLDAP_KRB5_KEYTAB に何も指定しない場合は、/etc/krb5.keytab 以下にある既定の keytab が使用され、下記のように権限を手作業で設定しなければなりません。

slapd を root で起動するには、/etc/sysconfig/openldap ファイルを編集します。それぞれ OPENLDAP_USER と OPENLDAP_GROUP の値に対して、行頭にコメント (#) マークを入れてコメントアウトしてください。

グループ LDAP に対して keytab ファイルの読み込みができるようにするには、下記のように実行します:

```
chgrp ldap /etc/krb5.keytab
chmod 640 /etc/krb5.keytab
```

もう一つの (そしておそらく最適であると思われる) 方法として、OpenLDAP に対して独自の keytab ファイルを用意する方法があります。これを行なうには kadmin を起動し、ldap/ldap.example.com のプリンシパルを追加した後、下記のようなコマンドを入力します:

```
ktadd -k /etc/openldap/ldap.keytab ldap/ldap.example.com@EXAMPLE.COM
```

その後、下記のように実行します:

```
chown ldap.ldap /etc/openldap/ldap.keytab
chmod 600 /etc/openldap/ldap.keytab
```

さらに、異なる keytab ファイルを使用するように OpenLDAP 側を設定するには、/etc/sysconfig/openldap 内にある下記の変数を変更します:

```
OPENLDAP_KRB5_KEYTAB="/etc/openldap/ldap.keytab"
```

最後に rcldap restart のコマンドを実行し、LDAP サーバを再起動すれば完了です。

6.4.11.1 LDAP を利用した Kerberos 認証の使用

ここまでの手順で、ldapsearch のようなツールで Kerberos 認証を利用できるようになりました。

```
ldapsearch -b ou=people,dc=example,dc=com '(uid=geeko)'
```

```
SASL/GSSAPI authentication started
```

```
SASL SSF: 56
```

```
SASL installing layers
```

```
[...]
```

```
# geeko, people, example.com
```

```
dn: uid=geeko,ou=people,dc=example,dc=com
```

```
uid: geeko
```

```
cn: Olaf Kirch
```

```
[...]
```

上記のとおり、`ldapsearch` を起動すると GSSAPI 認証のメッセージが表示されます。次のメッセージは意味不明なメッセージですが、*セキュリティの強度因数* (略して SSF) が 56 であることを示しています (56 は任意の数字で、DES の暗号鍵のビット数であるため、よく使用されます)。これは GSSAPI の認証が正しく動作していて、LDAP 接続でのデータ保護と機密保持のために暗号化を利用していることを示しています。

Kerberos では認証は双方向に動作します。これはユーザ自身が LDAP サーバに対して認証するだけでなく、LDAP サーバもユーザに対して認証する必要があります。特に、これは攻撃者が提供している不正な LDAP サーバではなく、正しい LDAP サーバに対して通信していることを確認するために必要な仕組みです。

6.4.11.2 Kerberos 認証と LDAP アクセス制御

ここまでの作業で、各ユーザに対して LDAP ユーザレコード内にある ログインシェルの属性を修正できるようにすることができます。たとえば `joe` に対する LDAP の項目が、`uid=joe,ou=people,dc=example,dc=com` にあるとすると、`/etc/openldap/slapd.conf` のアクセス制御は下記のように記述します:

```
# 全てのものをうまく動作させるために必要な設定です
access to dn,base="" by * read
# 各ユーザに対して、自身のログインシェルを変更できるようにする設定
access to dn="*,ou=people,dc=example,dc=com" attrs=loginShell
    by self write
# 各ユーザが全てのものを読み込めるようにする設定
access to *
    by users read
```

2 つめの構文では、認証済みのユーザに対して、自分自身の LDAP 項目内にある `loginShell` への書き込み権限を付与しています。3 つめの構文では、全ての認証済みユーザに対して、LDAP の階層構造全体への読み込み権限を設定しています。

ここまでの設定では 1 つだけ、見逃しやすい問題に対する回答が欠けています。それは、LDAP サーバが Kerberos 側のユーザ joe@EXAMPLE.COM と LDAP の識別名 uid=joe, ou=people, dc=example, dc=com をどのようにして結びつけるか、です。この割り当ては、`saslExpr` というディレクティブを利用して設定します。たとえばこの例では、`slapd.conf` に対して下記の行を追加します:

```
authz-regexp
    uid=(.*), cn=GSSAPI, cn=auth
    uid=$1, ou=people, dc=example, dc=com
```

上記がどのように動作するのか知るには、まず SASL がユーザを認証する際、OpenLDAP は SASL 側で提供された名前 (たとえば joe) と SASL の設定 (ここでは GSSAPI) から識別名を構成 することを知っておく必要があります。この例では、たとえば uid=joe, cn=GSSAPI, cn=auth のようになります。

`authz-regexp` が設定されている場合は、最初のパラメータとして正規表現を使用することで、SASL の情報から生成された DN をチェックします。正規表現がマッチした場合は、名前は `authz-regexp` の 2 つめのパラメータで置き換えられます。そのうち、`$1` の部分は、`(.*)` の正規表現で マッチした文字列に置き換えられます。

さらに複雑な表現を利用することもできます。より複雑なディレクトリ構造であった場合や、ユーザ名が DN の一部になっていないような場合は、SASL の DN からユーザ DN を検索するような設定を行なうこともできます。

6.5 さらになる情報

MIT Kerberos の公式サイトは <http://web.mit.edu/kerberos> (英語) にあります。ここには Kerberos のインストールや管理ガイドなど、Kerberos 関連のリソースが用意されています。

<ftp://athena-dist.mit.edu/pub/kerberos/doc/usenix.PS> (英語) の論文には、難しい用語を使わない形で Kerberos の基本的な考え方の見解が書かれています。また、Kerberos についてさらなる調査や知識を得るための 参照リンクが書かれています。

公式の Kerberos FAQ は <http://www.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html> (英語) で提供されています。また、*Kerberos—A Network Authentication System* (Brian Tung 氏; (ISBN 0-201-37924-4)) では、広範囲にわたる情報が記されています。

指紋読み取り装置の使用

お使いのシステムに指紋読み取り装置があれば、ユーザ ID とパスワードによる 認証に加えて、生体認証を利用することができるようになります。あらかじめ ユーザの指紋を登録しておけば、指紋読み取り装置で指紋を読み取らせるか、もしくはパスワードを入力するかのどちらかで、システムにログインできるようになります。openSUSE® では、数多くの読み取り装置に対応しています。対応デバイスの一覧について、詳しくは <http://www.freedesktop.org/wiki/Software/fprint/libfprint> をお読みください。

ハードウェアのチェックの際、お使いのシステムに指紋読み取り装置が内蔵 (または 接続) されていることを検出すると、libfprint, pam_fp, yast2-fingerprint-reader の各パッケージが自動的に インストールされます。

現時点では、1 ユーザあたり 1 つまでの指紋を登録できます。ユーザの指紋 データは、`/home/ユーザ名/.fprint/` 以下に保存されます。

7.1 対応するアプリケーションと処理

PAM モジュール `pam_fp` では、下記のアプリケーションや 処理の指紋認証に対応しています (すべての場合で指紋を読み取るような メッセージが表示されるとは限りません):

- GDM/KDM またはシェルでのログイン
- GNOME/KDE のデスクトップのアンロック (解錠)
- YaST と YaST モジュールの起動

- root の権限が必要なアプリケーションの起動 (sudo または gnomesu を使用した場合)
- su や su - ユーザ名 を利用して、異なる ユーザに切り替える処理

注記: 指紋読み取り装置とホームディレクトリの暗号化について

指紋読み取り装置を使用する場合は、ホームディレクトリの暗号化 (詳しくは 第 10 章 *YaST を利用したユーザ管理* (↑ スタートアップ) をお読みください) を利用してはなりません。ホームディレクトリを暗号化してしまうと、指紋読み取り 装置から指紋データを読み取ることができなくなってしまうため、ログインが 失敗するようになります。

7.2 YaST を利用した指紋の管理

手順 7.1 指紋認証の有効化

生体認証を利用するには、PAM を設定する必要があります。パッケージの インストール時にハードウェア側で対応する指紋読み取り装置が検出されれば、それらの設定は自動的に行なわれますが、手作業で行ないたい場合は下記の 手順で行ないます:

- 1 YaST を起動し、ハードウェア > *指紋読み取り装置* を選択します。
- 2 設定ダイアログが表示されたら、*指紋読み取り装置を利用する* を選択し、*完了* を押します。これで設定が保存され、ダイアログが閉じます。

あとはユーザの指紋を登録します。

手順 7.2 指紋の登録

- 1 YaST を起動し、*セキュリティとユーザ* > *ユーザとグループの管理* を選択して、*ユーザとグループの管理* ダイアログを表示します。システム内に登録されているユーザやグループの一覧が表示されます。
- 2 指紋を登録したいユーザを選択し、*編集* を押します。
- 3 *プラグイン* のタブを選択し、指紋の項目を選んで *起動* を押します。すると *指紋認証の設定* ダイアログが表示されます。

- 4 YaST はユーザに対して 3 種類の指紋情報を正しく読み取れるまで、繰り返し指紋の読み取りを求めます。



- 5 指紋情報を正しく読み取ったあとは、**了解** を押して *指紋認証の設定* ダイアログを閉じ、ユーザ設定の *ダイアログ* も閉じてください。

- 6 YaST や YaST のモジュールを起動するために指紋認証を使用したい 場合は、**root** の指紋についても登録を行なう必要があります。

これを行なうには、*ユーザとグループの管理* ダイアログで *フィルタの設定* から *システムユーザ* を選び、**root** の項目を選んで 指紋を登録します。

- 7 設定したいユーザに対して指紋の登録を終えたら、最後に **完了** を押して管理ダイアログを閉じ、設定を保存します。

ユーザに対する指紋情報の登録を行なったら、その直後から指紋とパスワードの 両方による認証を利用して、7.1項「対応するアプリケーションと処理」(111 ページ) に書かれている処理やアプリケーションを起動することができます。

現時点では YaST には、指紋情報の検証や削除の機能がありません。指紋情報を削除したい場合は、`/home/ユーザ名/.fprint` のディレクトリを直接削除してください。

技術的な詳細については、<http://www.freedesktop.org/wiki/Software/fprint> をお読みください。

パート II. ローカルセキュリティ

YaST を利用したセキュリティ設定

YaST のモジュール *セキュリティセンターとセキュリティの強化* では、openSUSE におけるセキュリティ設定を集中管理することができます。ログイン手順やパスワードの作成、起動許可やユーザの作成、既定のファイル アクセス権など、様々なセキュリティ関連設定を行なうことができます。この モジュールを起動するには、YaST のコントロールセンターから *セキュリティとユーザ > セキュリティセンターとセキュリティの強化* を選択してください。セキュリティセンター のダイアログが表示されると、*セキュリティの概要* 画面が表示されますが、その他の設定 ダイアログについても左側のペインで選択することができます。

8.1 セキュリティの概要

セキュリティの概要 には、お使いのシステムで最も重要な セキュリティ設定について、概要が表示されます。それぞれの項目にセキュリティ 状態が明示され、緑色のチェックマークはセキュリティに問題がないことを、赤いバツ印は問題があることを示します。また、各項目の ヘルプ ボタンを押すと設定の説明を表示することができます。ようになっていて、どのように すれば問題を解決できるのかわかるようになっています。設定を変更するには、それぞれの *状態* 列にあるリンクを押してください。設定 項目によって、下記のような状態があります：

有効/無効

対象の項目を押すと、有効と無効の間で状態を切り替えることができます。

設定

対象の項目を押すと、対応する YaST モジュールを起動することができます。モジュールを終了させればセキュリティの概要に戻ることができます。

不明

関連するサービスがインストールされていない場合、状態は不明として表示されます。この状態になっていた場合でも、潜在的なセキュリティリスクを示すものではないことに注意してください。

図 8.1 YaST セキュリティセンターとセキュリティの強化 - 概要



8.2 事前定義済みのセキュリティ設定

openSUSE では 3 種類の **事前定義済みのセキュリティ設定** を用意しています。この設定は **セキュリティセンター** モジュール全体に対して影響のあるもので、それぞれの設定は左側の一覧から選択して設定することで、細かい調整を行なうことができます。下記のような選択をすることができます：

ホームワークステーション

この設定は、ネットワーク接続（インターネット接続を含む）を全く持たないコンピュータ向けの設定です。事前定義済みのセキュリティ設定の中では、最低限のセキュリティ設定だけを行なうものです。

ネットワーク接続ステーション

ある種のネットワーク接続（インターネット接続を含む）を行なっているワークステーション向けの設定です。

ネットワークサーバ

Web サーバやファイルサーバ、ネームサーバなど、何らかのネットワーク サービスを提供するマシン向けのセキュリティ設定です。事前定義済みの セキュリティ設定の中では、最大限のセキュリティ設定を行ないます。

カスタム設定

事前定義済みのセキュリティ設定 ダイアログを開いた とき、あらかじめ選択されているのが **カスタム設定** です。これは事前定義済みの設定から何らかの細かい調整を行なったことを 示していて、これを選択しても設定は何も変わりません。

8.3 パスワード設定

パスワードはセキュリティ問題の中でもっとも注力しやすい項目です。パスワード設定 のダイアログでは、より安全なパスワード だけを使わせるような設定を行なうことができます。

新しいパスワードのチェック

このオプションを選択すると、新しいパスワードが辞書内に載っているものであったり、固有名詞であったりした場合に警告メッセージを表示するようになります。パスワードの最低長についてもチェックを行ないたい場合は、**新しいパスワードのチェック** にチェックを行なったあと、**最小パスワード文字数** の欄で指定を行なってください。

最小パスワード文字数

ユーザがここで指定した長さよりも短いパスワードを入力すると、警告が 表示されます。

記録するパスワードの数

パスワードの有効期限が設定されていた場合 (**パスワード有効日数** の欄)、ここで指定した値だけ過去のパスワードを保存するようになります。新しいパスワードが過去のものと同じした場合は、変更が拒否されます。

パスワードの暗号化方法

パスワードの暗号化方法を指定します。通常、ここは既定値 (Blowfish) のままでかまいません。

パスワード有効日数

それぞれ最小値と最大値を指定することで、パスワードの有効期限を設定することができます。最小値を 0 日より大きい値に設定すると、ユーザが過度にパ

パスワードを変更することを防止することができます (また、これによりパスワードの有効期限設定を回避することもできなくなります)。パスワードの期限が切れないように設定したい場合は、それぞれ 0 と 99999 の値を設定してください。

パスワード失効警告日数

パスワードの有効期限が切れる場合、ユーザに対しては事前に警告メッセージを表示します。ここでは実際にパスワードの有効期限が切れる際、どれだけ前もってメッセージを表示するかを日数で指定します。

8.4 起動設定

このダイアログでは、グラフィカルなログインマネージャからマシンをシャットダウンする場合、どのユーザに対してこれを許可するかを設定します。またこれ以外にも、Ctrl + Alt + Del をどう解釈するかについても設定することができます。

8.5 ログイン設定

このダイアログでは、下記に示すセキュリティ関連のログイン設定を変更することができます：

ログイン失敗後の待機時間

繰り返しログインを試すことでユーザのパスワードを当てられることのないようにするため、ログイン失敗時にはログインプロンプトの表示を遅らせる設定をしておくことをお勧めします。ここでは秒単位の値を設定します。ただし、ユーザが自分のパスワードの入力を間違える可能性もあるため、あまりに長い時間にならないように設定しておくことをお勧めします。

成功したログインを記録

このオプションを選択すると、もっとも新しいログイン試行が /var/log/lastlog ファイル内に記録されるようになり、この値がログイン時に表示されるようになります。このデータは finger コマンドでも使用されます。

注記

このオプションを設定しても、/var/log/wtmp には影響が及ばないことに注意してください。上記のファイルはログイン日時と再起動の日時を収集するものです。/var/log/wtmp の内容は last で表示することができます。

リモートのグラフィカルログインを許可

これを選択すると、グラフィカルなログインマネージャ (gdm や kdm など) に対して、ネットワーク経由でアクセスできるようになります。この許可を 設定すると、潜在的なセキュリティリスクとなることに注意してください。

8.6 ユーザ追加

ここではユーザ ID とグループ ID に割り当てる ID について、それぞれ最小値と 最大値を設定します。この設定を変更する必要はほとんどありません。

8.7 その他の設定

上記の分類に該当しない、その他のセキュリティ設定がここにまとめられています:

ファイルのアクセス権

openSUSE では、システムファイルに対するアクセス権設定について、3 種類の事前定義が用意されています。これらの許可セットは、それぞれログ ファイルを一般のユーザが参照できるかどうかと、特定のプログラムを起動できる かどうかを設定するためのものです。**簡易** アクセス権は 一人で利用するタイプのマシンに適切な選択で、この設定ではたとえば、多くの システムファイルに対して一般ユーザからのアクセスを許可しています。詳しい 設定については、`/etc/permissions.easy` ファイルをお読みください。**厳格** アクセス権はネットワークアクセス のあるマルチユーザのマシンに適切な選択で、詳しい設定は `/etc/permissions.secure` ファイル内に書かれています。**偏執** は最大限に制限をかけたアクセス権設定で、注意して 使用する必要があります。詳しくは `/etc/permissions.paranoid` をお読みください。

`updatedb` を実行するユーザ

`updatedb` プログラムはシステム内を走査し、後から `locate` コマンドで検索できるようにするため、すべての ファイルの場所についてデータベースを作成します。`updatedb` を `nobody` ユーザで実行すると、誰にでも読めるファイルに対してのみデータベース 内に投入されることになります。`root` で実行すると、ほぼすべてのファイル (ただし `root` が読み込みを許可されていないファイルを除く) に対して、データベース内に投入されます。

`root` の `PATH` 環境変数にカレントディレクトリを含める / 通常ユーザの `PATH` 環境変数にカレントディレクトリを含める

プログラムを起動する際、パスを指定せずに実行ファイルを指定した場合、システムはユーザの検索パス (`PATH` 環境変数) 内に実行ファイル があるものとして

検索を行いません。既定では検索パス内に今現在位置している (カレント) ディレクトリは含まれていません。この設定を有効にしても、たとえば `ls` と入力した場合、`/bin/ls` が存在していれば `/カレントディレクトリ/ls` よりも優先して動作します。この設定を有効にしない場合、一般的には `./` を頭に付けてプログラム名を指定します。このオプションを有効にした場合は、検索パスの最後にカレントディレクトリ (`.`) が追加されます。この設定は変更しないことを お勧めします。

マジック *SysRq* キーを有効にする

マジック *SysRq* キーとは、システムがクラッシュしたりしたような場合に、システムに対していくつかの制御を行なうための機能です。詳しい説明は `/usr/src/linux/Documentation/sysrq.txt` (ただし `kernel-source` パッケージをインストール する必要があります)。また、上記のドキュメントは英語で書かれていることに注意してください) をお読みください。

Linux におけるアクセス制御リスト

POSIX ACL (アクセス制御リスト) は、ファイルシステムの要素 (ファイルや ディレクトリなど) に設定する従来のパーミッションを拡張するために利用 することができるものです。ACL を利用すると、従来のパーミッションよりも より柔軟な構成を取ることができます。

POSIX ACL という用語は正規の POSIX (*Portable Operating System Interface*) 内の標準として 定められていたものです。ACL を規定している POSIX 1003.1e と POSIX 1003.2c のドラフト標準は諸般の理由により取り消されていますが、ACL (UNIX 系に属する多くの システムで利用されているもの) はこれらのドラフト標準に従って作られていて、ファイルシステムの ACL (本章での記述範囲) 実装も、これら 2 つの標準に従って 作られています。

9.1 従来のファイルパーミッション

従来のファイルパーミッションに関する詳細な情報は、GNU coreutils の info パッケージ内 *File permissions* ノードに詳しく 書かれています (info coreutils "File permissions" で表示することができます)。それらをさらに拡張したものとして、それぞれ *setuid*, *setgid*, *sticky* の各ビットが用意されています。

9.1.1 *setuid* ビット

特定の状況下において、アクセス許可は過度の制限をもたらしてしまいます。そのため、Linux では特定の作業を行なうにあたって、現在のユーザと グループを一時的に変更できる設定が用意されています。たとえば *passwd* プログラムでは、通

常 `/etc/passwd` にアクセスするため、`root` の権限が 必要になります。それは、このファイルにはいくつかの重要な情報、たとえば ユーザのホームディレクトリの情報や、ユーザ／グループの ID などがあります。そのため、通常のユーザに対しては `passwd` ファイルの 修正が許可されることはありません。もしも全てのユーザに対して書き込みを 許可してしまうと、情報を自由に書き換えることができてしまい、非常に 危険であるためです。この問題を解決するための方法の 1 つが *setuid* の仕組みです。*setuid* (*set user ID*; ユーザ ID の設定) ビットは、システムに対して指定したユーザ ID での実行を行なわせる ための特殊なファイル属性で、たとえば `passwd` コマンドは 下記のような属性になっています:

```
-rwsr-xr-x 1 root shadow 80036 2004-10-02 11:08 /usr/bin/passwd
```

上記の表示のうち `s` が、*setuid* ビットの設定されている 印です。*setuid* ビットが設定されていることから、`passwd` コマンドを実行する全てのユーザは、そのコマンドを `root` として実行します。

9.1.2 *setgid* ビット

setuid ビットはユーザに対して適用されるものですが、グループに対しても同様のことができます。それが *setgid* ビットです。このビットが 設定されたプログラムは、どのユーザから起動された場合であっても、そのファイル のグループ ID で動作させたものとして扱われます。そのため、*setgid* ビットが 設定されたディレクトリ内で新しくファイルやサブディレクトリを作成すると、そのディレクトリに設定されたグループに属するものとして作成されます。たとえば下記のようなディレクトリがあると仮定します:

```
drwxrws--- 2 tux archive 48 Nov 19 17:12 backup
```

グループパーミッションの欄に `s` が書かれていて、これによって *setgid* ビットが設定されていることを表わしています。また、このディレクトリの 所有者と、`archive` グループに属する ユーザは、このディレクトリにアクセスすることができます。このグループに属して いないユーザは、関連するグループに「マップ」されます。この ディレクトリに書き込まれた全てのファイルは、`archive` グループが実効グループ ID となります。たとえば `archive` のグループ ID を持つバックアッププログラムがあったとすると、このプログラムは `root` の権限無しでこのディレクトリにアクセスできるようになります。

9.1.3 *sticky* ビット

sticky ビット は、実行可能なプログラムやディレクトリに 設定した場合、それぞれ異なる動作をします。このビットをプログラムに対して 設定すると、そのファイルはハー

ドディスクから毎回読み込まれることがなくなり、RAM 内に保持されるようになります。この属性は、今となってはハードディスクが十分に高速化してしまったことから、ほとんど使われることがありません。このビットをディレクトリに対して設定すると、ユーザ間でお互いのファイルを削除できないようになります。典型例が /tmp や /var/tmp のディレクトリです：

```
drwxrwxrwt 2 root root 1160 2002-11-19 17:15 /tmp
```

9.2 ACL の利点

従来、Linux システムではそれぞれのファイルオブジェクトに対して 3 種類のパーミッションセットを設定していました。各セットには読み込み (r)、書き込み (w)、実行 (x) の各パーミッションが存在し、それを 3 種類のタイプ、ファイルの所有者、グループ、その他のユーザに対してそれぞれ設定していました。これに加えて *set user id (setuid)*、*set group id (setgid)*、*sticky* の各ビットが存在していました。このようなシンプルな仕組みは実際の使用形態でも十分な仕組みでしたが、より複雑なシナリオや高度なアプリケーションの場合、システム管理者はこのようなシンプルな仕組みをうまく使うため、数多くの回避策を利用しなければならませんでした。

ACL は従来のパーミッションに対する拡張として使用することができます。また、ACL は所有者以外の個別のユーザや、個別のグループであっても設定を行なうことができます。アクセス制御リストは Linux カーネルに搭載されている機能で、現時点では ReiserFS、Ext2、Ext3、Ext4、JFS、XFS に対応しています。ACL を利用することで、アプリケーション側で複雑なアクセス許可モデルを構築したりすることなく、複雑なシナリオを利用できるようになります。

ACL の利点は、Windows サーバを Linux サーバで置き換えたいような場合に明確になります。サーバ側の移行後も、接続されたワークステーションで Windows を利用することが想定されますが、このような場合でも Linux システムは Samba を利用して、Windows クライアントに対するファイル／印刷サービスの提供を行ないます。Samba も ACL に対応しているため、アクセス許可を Windows のグラフィカルユーザインターフェイスから設定（ただし Windows NT またはそれ以降のバージョンである必要があります）し、それを Linux サーバ側に保存することができます。また Samba スイートの一部である winbindd を利用すると、Windows ドメインにのみ存在し、Linux サーバ上には存在していないユーザに対しても、アクセス許可を設定することができます。

9.3 定義

ユーザクラス

通常のファイルパーミッションでは、ファイルシステム内でパーミッションを設定する際に、ユーザを 3 種類の クラス に区分して扱います。それはそれぞれ、所有者とグループ、その他のユーザです。また、それぞれのユーザクラスに対して、3 種類のビットでアクセス許可を設定することができます。それぞれ読み込み (r)、書き込み (w)、実行 (x) です。

ACL

全ての種類のファイルシステム要素 (ファイルとディレクトリ) に対して、ユーザとグループに対するアクセス許可を設定するものです。

デフォルト ACL

デフォルト ACL はディレクトリに対してのみ設定するものです。これらは、そのディレクトリ内に何らかの要素 (ファイルやディレクトリ) が作成された 際に設定される ACL の既定値を設定します。

ACL 項目

それぞれの ACL には ACL 項目のセットが存在しています。ACL 項目には、そのタイプとユーザまたはグループ、そしてアクセス許可セットが含まれます。項目の種類によっては、ユーザまたはグループが未定義になる場合もあります。

9.4 ACL の処理

表9.1「ACL 項目タイプ」(127 ページ) には、ACL 項目として存在しうる 6 種類のタイプが示されています。それぞれはユーザまたはユーザのグループに対して許可を設定するものです。ここで *所有者* の項目は、ファイルやディレクトリの所有者に対する許可を設定します。また、*所有グループ* の項目は、ファイルを所有するグループに対して設定する許可を指します。なお、スーパーユーザであれば `chown` や `chgrp` コマンドで所有者や 所有グループを変更することができますが、この場合これらの項目は、新しい 所有者や所有グループに対する設定として解釈されるようになります。また、*指名ユーザ* の項目は、項目内に書かれたユーザに対して 設定する許可のことを指します。さらに *指名グループ* の項目では、同様に項目内に書かれたグループに対して、アクセス許可を設定 するものを指します。なお、指名ユーザと指名グループの項目にのみ、それぞれ ユーザまたはグループ欄に記載があります。最後に *その他* の項目には、その他のユーザに対するアクセス許可を設定します。

また、マスクの項目は、指名ユーザや指名グループ、所有グループに対して許可の制限をかけることができるもので、許可の項目の うち、どれを有効としてどれを

マスクするかを設定します。指名ユーザや指名 グループ、所有グループに対して設定した許可がマスクとしても存在する場合、その許可は有効なものとして扱われます。逆にマスクとしてしか存在していない 場合や、実際の項目内にしか存在していない項目は、それらは無効なものとなり、許可は取り消されます。ただし、所有者 に対する許可は常に有効になります。表9.2「アクセスパーミッションのマスク」(127 ページ) では、例をもとに仕組みを説明しています。

ACL には 2 種類の基本分類があります: 最小 ACL には、所有者と所有グループ、およびその他のユーザに対する項目だけが含まれていて、これらはファイルやディレクトリに対して設定する、従来のアクセス許可ビットの 仕組みと同じものです。拡張 ACL はそこから拡張した もので、マスク項目が含まれていなければならないほか、指名ユーザや指名 グループの種類の項目を含む場合があるものです。

表 9.1 ACL 項目タイプ

種類	テキストでの書式
所有者	user::rwx
指名ユーザ	user:name:rwx
所有グループ	group::rwx
指名グループ	group:name:rwx
マスク	mask::rwx
その他	other::rwx

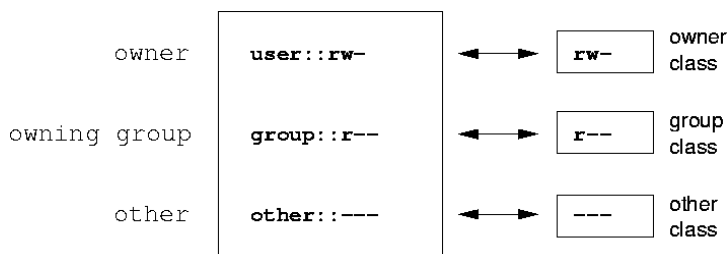
表 9.2 アクセスパーミッションのマスク

項目の種類	テキストでの書式	許可
指名ユーザ	user:geeko:r-x	r-x
マスク	mask::rw-	rw-
	有効となる許可:	r--

9.4.1 ACL の項目とファイルモードパーミッションビット

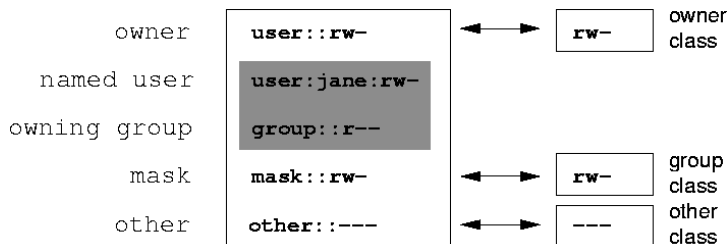
図9.1「最小 ACL: パーミッションビットと ACL 項目の比較」(128 ページ) と 図 9.2「拡張 ACL: パーミッションビットと ACL 項目の比較」(128 ページ) では、それぞれ最小 ACL と拡張 ACL に関する説明を図示しています。図には 3 つのブロックがあります。左側のブロックでは ACL のタイプ説明を、中央のブロックでは ACL の例を、右側では対応する従来のファイルパーミッション (たとえば `ls -l` で表示できるもの) を示しています。いずれの場合とも、所有者クラス (*owner class*) のパーミッションは ACL の所有者 (owner) の項目に当てはまるほか、その他のクラス (*other class*) のパーミッションは「その他 (other)」の ACL 項目に当てはまります。ただし、グループクラス (*group class*) のパーミッションは、2 つの場合で異なっていることに注意してください。

図 9.1 最小 ACL: パーミッションビットと ACL 項目の比較



最小 ACL (マスク無し) では、グループクラスのパーミッションは所有グループの ACL 項目に対応します (参照: 図9.1「最小 ACL: パーミッションビットと ACL 項目の比較」(128 ページ))。拡張 ACL (マスクあり) では、グループクラスのパーミッションはマスク項目に対応します (参照: 図9.2「拡張 ACL: パーミッションビットと ACL 項目の比較」(128 ページ))。

図 9.2 拡張 ACL: パーミッションビットと ACL 項目の比較



このような対応付けにより、アプリケーション側が ACL に対応しているかどうかに関係なく、円滑な動作を行なうことができるようになっています。つまり、従来のファイルパーミッションでは大まかな設定を、ACL では「細かい調整」を行なう形です。また、ファイルパーミッションに対する変更は ACL 側に自動的に反映されますし、逆も同様です。

9.4.2 ACL の設定されたディレクトリ

`getfacl` と `setfacl` のコマンドラインツールを利用することで、ACL にアクセスすることができるようになっています。これらのコマンドの使用方法について下記で説明します。

ディレクトリを作成する前に、まずは `umask` コマンドを利用して、ファイルなどの要素を作成する際にマスクされるべきファイルパーミッションを設定します。`umask 027` のように実行すると、所有者に対しては既定ですべての範囲のパーミッション (0) を、所有グループに対しては書き込みアクセスだけを禁止するパーミッション (2) を、最後にその他のユーザに対してはすべてのパーミッションを拒否 (7) する意味になります。つまり `umask` では、マスクして OFF に設定するパーミッションビットを 1 として指定することで設定を行ないます。詳しくは `umask` のマニュアルページをお読みください。

`mkdir mydir` を実行すると、`umask` で設定した既定のファイルパーミッション設定で `mydir` ディレクトリを作成します。作成後は、`ls -dl mydir` と実行して、すべてのパーミッションが正しく設定されたかどうかを確認してください。正しく設定されると、下記のような出力になります：

```
drwxr-x--- ... tux project3 ... mydir
```

ここから `getfacl mydir` と実行すると、ACL の初期状態を確認することができます。下記のように出力されるはずです：

```
# file: mydir
# owner: tux
# group: project3
user::rwx
group::r-x
other::---
```

出力のうち、最初の 3 行にはディレクトリの名前と所有者、そして所有グループが表示されています。次の 3 行には 3 種類の ACL 項目が表示され、それぞれ所有者、所有グループ、その他の項目を表わしています。このように最小 ACL の場合は、`getfacl` で表示できる項目と `ls` で表示できる項目に差はありません。

追加のユーザ `geeko` とグループ `mascots` に対して、読み込みと書き込み、そして実行の権限を設定するよう ACL を修正するには、下記のように実行します:

```
setfacl -m user:geeko:rxw,group:mascots:rxw mydir
```

`-m` オプションは、`setfacl` に対して 既存の ACL を修正するように指定するオプションです。続く文字列には修正 対象の ACL 項目 (複数の項目がある場合はカンマで区切ります) が書かれていて、最後は変更を適用するディレクトリ名を指定しています。上記を実行すると、ACL は下記のようになります:

```
# file: mydir
# owner: tux
# group: project3
user::rxw
user:geeko:rxw
group::r-x
group:mascots:rxw
mask::rxw
other::---
```

ユーザ `geeko` とグループ `mascots` に対して設定した項目に加え、マスク項目も生成されています。マスク項目は すべてのパーミッションに対して効果があるようにするため自動的に作成されます。`setfacl` では、`-n` を利用して 明示的に無効化しない限り、修正内容に合わせて自動的にマスク項目を設定します。マスク項目は、効力のあるアクセス許可の最大値を表示するものです。これには 指名ユーザと指名グループ、そして所有グループに対するアクセス許可が 当てはまります。この マスク 項目は `ls -dl mydir` コマンドでグループクラスとして表示することができません。

```
drwxrwx---+ ... tux project3 ... mydir
```

出力の最初の列に `+` 記号が書かれていますが、これは その項目に 拡張 ACL が存在することを示しています。

`ls` コマンドの出力によると、マスク項目に書き込みアクセス が含まれていることを示しています。従来はこのような許可ビットが設定されていた場合、所有グループ (ここでは `project3` グループ) は、`mydir` ディレクトリへの書き込み権限があることを示す ものでした。しかしながら、ここでは所有グループに対するアクセス許可は、所有グループに対して設定された ACL の値が有効となります。この例では、`r-x` (詳しくは 表9.2「アクセスパーミッションのマスク」(127 ページ) をお読みください) となります。この例で示されているとおり、ACL の項目に対して 何らかの追加を行っても、ファイルパーミッション側には変化がないことに 注意してください。

マスク項目の修正は `setfacl` または `chmod` で行ないます。たとえば `chmod g-w mydir` のように実行すると、`ls -dl mydir` は下記のような出力になります:

```
drwxr-x---+ ... tux project3 ... mydir
```


getfacl mydir では下記のような出力に なります:

```
# file: mydir
# owner: tux
# group: project3
user::rwx
user:geeko:rwx      # effective: r-x
group::r-x
group:mascots:rwx   # effective: r-x
mask::r-x
other::---
```

chmod コマンドを実行すると、グループクラスのアクセス 許可から書き込みの権限が外されます。ls コマンドの出力は マスクビットが変更できたかどうかを確認するには十分な仕組みで、表示のとおり mydir の所有グループに対する書き込みを制限するようになっていました。getfacl でもこの設定を確認することができます。この出力では、元々のアクセス許可設定と実際に適用されるアクセス 許可について、差異がある箇所にコメントが記されています。これはマスク 項目の効力によるものです。元のアクセス許可に戻すには、chmod g+w mydir のように実行します。

9.4.3 デフォルト ACL が設定されたディレクトリ

ディレクトリに対してはデフォルト ACL を設定することができます。これは特殊 用途の ACL で、そのディレクトリ内に何らかの要素 (ファイルやサブディレクトリ など) を作成する際、継承されるべきアクセス許可を規定するものです。

9.4.3.1 デフォルト ACL の効果

ディレクトリのデフォルト ACL がファイルやサブディレクトリに渡される 仕組みとしては、下記の 2 つのものがあります:

- サブディレクトリは、親ディレクトリのデフォルト ACL を、自身のデフォルト ACL と通常の ACL として引き継ぎます。
- ファイルは、デフォルト ACL を自身の ACL として引き継ぎます。

ファイルシステム内に何らかの要素を作成するすべてのシステムコールでは、mode というパラメータを使用し、新しく作成するものに対して設定するアクセス許可を指定します。親ディレクトリにデフォルト ACL が設定されていない場合は、mode パラメータで渡された アクセス許可から umask の値を差し引いたあと、新しく 作成した要素に設定が行なわれます。親ディレクトリにデフォルト ACL が存在 する場合は、デフォルト ACL に mode パラメータで渡された 設定を被せて設定します。この場合、umask の設定値は 無視されます。

9.4.3.2 デフォルト ACL の適用例

下記の 3 つの例で、ディレクトリやそのデフォルト ACL に対する主な操作を説明します:

1. 既存のディレクトリ `mydir` に対して、デフォルト ACL を追加するには、下記のように実行します:

```
setfacl -d -m group:mascots:r-x mydir
```

`setfacl` の `-d` オプションは、`setfacl` に対して 下記の修正 (`-m` オプション) をデフォルト ACL に対して適用するよう指定するオプションです。

下記のコマンド出力をお読みください:

```
getfacl mydir

# file: mydir
# owner: tux
# group: project3
user::rwx
user:geeko:rwx
group::r-x
group:mascots:rwx
mask::rwx
other:---
default:user::rwx
default:group::r-x
default:group:mascots:r-x
default:mask::r-x
default:other:---
```

`getfacl` は通常の ACL とデフォルト ACL の両方を表示します。デフォルト ACL は `default` で始まる 行で記されています。単純に `setfacl` コマンドをデフォルト ACL の `mascots` グループに対して実行した場合でも、`setfacl` は自動的に通常の ACL 内に存在する項目をデフォルト ACL としてコピーします。なお、デフォルト ACL はアクセス権に対して即時に効力のあるような仕組みでは無く、新しくファイルシステムの要素 (ファイルやディレクトリなど) を作成した際にはじめて効力が現われます。なお新しく作成した要素は、自身が属する親ディレクトリのデフォルト ACL のみを引き継ぎます。

2. 次の例では、`mkdir` コマンドを利用して `mydir` 以下にサブディレクトリを作成し、デフォルト ACL を引き継ぐ仕組みを示しています。

```
mkdir mydir/mysubdir
```

```
getfacl mydir/mysubdir
```

```
# file: mydir/mysubdir
# owner: tux
# group: project3
user::rwx
group::r-x
group:mascots:r-x
mask::r-x
other:---
default:user::rwx
default:group::r-x
default:group:mascots:r-x
default:mask::r-x
default:other:---
```

期待の通り、新しく作成されたサブディレクトリ `mysubdir` には、親ディレクトリのデフォルト ACL がそのまま適用されています。つまり、`mysubdir` の ACL は `mydir` のデフォルト ACL を即時に反映させたものと言えます。このディレクトリのデフォルト ACL は、通常の ACL と同じになっています。

3. さらに `touch` コマンドを利用すると、`mydir` ディレクトリ内にファイルを作成することができます。たとえば `touch mydir/myfile` のように実行すると、`ls -l mydir/myfile` は下記のような表示になります:

```
-rw-r-----+ ... tux project3 ... mydir/myfile
```

また、`getfacl mydir/myfile` の出力は 下記のとおりです:

```
# file: mydir/myfile
# owner: tux
# group: project3
user::rw-
group::r-x      # effective:r--
group:mascots:r-x # effective:r--
mask::r--
other:---
```

`touch` コマンドは、新しくファイルを作成するにあたって `mode` の設定値を `0666` に設定します。これは `umask` やデフォルト ACL で制限を行なわない限り、すべてのユーザクラスに対して読み書きのアクセスを許可する意味になります (詳しくは 9.4.3.1 項「デフォルト ACL の効果」(131 ページ) をお読みください)。実際には、`mode` の設定値に含まれていないアクセス許可は関連する ACL 項目から取り除かれます。グループクラス内の ACL 項目から削除されることはありませんが、その代わりにマスク項目が `mode` での設定値をマスクするために修正されています。

このような仕組みにより、ACL はアプリケーション (たとえばコンパイラ) との 柔軟な対応を実現しています。たとえば制限されたアクセス許可でファイルを作成して、後からそれらを実行可能なものとしてマークするようなことができます。つまり mask の仕組みは、正当なユーザと グループが必要に応じて実行できることを保証する仕組みです。

9.4.4 ACL チェックアルゴリズム

チェックアルゴリズムは、ACL で保護されたファイルシステムの要素に対して、プロセスやアプリケーションがアクセスしようとしたときに呼び出されます。基本的なルールとして、ACL の項目は所有者、指名ユーザ、所有グループまたは 指名グループ、その他の順序で解釈されます。また、アクセスはその処理に もっとも当てはまる項目に対して処理されます。なお、アクセス許可が蓄積されて 動作することはありません。

ある特定のプロセスが複数のグループに所属し、それらのグループに該当する ようなアクセス許可があった場合はより複雑な仕組みになります。この場合は 必要なアクセス許可に対して、該当する項目をランダムで抽出します。これは どの項目が最終的に「アクセスを許可する」のかに関係なく動作 します。同様に必要なアクセス許可を含むグループ項目が存在しない場合は、ランダムで選択された項目が最終的な「アクセス拒否」を発生 させます。

9.5 アプリケーションにおける ACL サポート

ACL は、新しいアプリケーションに対して、とても複雑なアクセス許可シナリオを 実装するために使用するものです。従来のファイルパーミッションの考え方と ACL は、賢い方法で組み合わせることが可能です。基本的なファイルコマンド (cp, mv, ls など) では ACL に対応しているほか、Samba や Konqueror などでも対応しています。

残念ながら、多くのエディタやファイルマネージャには ACL への対応が用意されて いません。emacs などではファイルをコピーすると、これらのファイルの ACL は 失われます。エディタなどでファイルを修正する場合は、ファイルの ACL が 保持される場合と保持されない場合がありますが、これはエディタが使用する バックアップ手法によって決まります。エディタが元々のファイルに対して変更 点を書き込む場合は、

ACL は元のまま保持されます。エディタが更新された ファイルを新しいファイルに保存し、あとから古いファイルと新しいファイルとの 間で名前変更を行なうような仕組みの場合は、エディタ側で ACL に対応していない限り、ACL の情報は失われます。また star アーカイバを除き、ACL を保持できるバックアップアプリケーションは現時点では存在していません。

9.6 さらになる情報

ACL についてさらに詳しい情報は、`getfacl(1)`, `acl(5)`, `setfacl(1)` の各マニュアル ページをお読みください。

パーティションとファイルの暗号化

10

多くのユーザは自分のマシン内に秘密の情報を保持していて、それらは第三者に読み取られるべきではないものです。異なる環境やネットワークで作業を行なうモバイルコンピューティングに依存すればするほど、それらのデータに対する扱いは厳重なものにしなければなりません。お使いのシステムにネットワーク経由や物理的にアクセス可能な環境の場合、ファイルやパーティション全体の暗号化が必要となります。ラップトップや外付けのハードディスク、USBメモリのようなりムーバブルメディアは、いずれも紛失や盗難の危険性があるものであるため、秘密の情報を保持するファイルシステムを暗号化しておくことをお勧めします。

データの保護を暗号化で行なう場合、下記のような方法があります：

ハードディスクのパーティションを暗号化する方法

インストール時、およびインストール後のシステムから YaST を利用することで、暗号化パーティションを作成することができます。詳しくは 10.1.1 項「インストール時の暗号化パーティション作成」(139 ページ) と 10.1.2 項「実行中のシステムに対する暗号化パーティションの作成」(140 ページ) をお読みください。この方法は外付けのハードディスクなど、リムーバブルメディアに対しても使用することができます。こちらに関する詳細は 10.1.4 項「リムーバブルメディアの内容の暗号化」(141 ページ) をお読みください。

コンテナとして暗号化ファイルを作成する方法

お使いのハードディスクやリムーバブルメディア内に、YaST を利用して暗号化ファイルを作成することができます。暗号化ファイルを作成したあとは、他のファイルやフォルダをその中に保存することができるようになります。詳しくは 10.1.3 項「コンテナとしての暗号ファイルの作成」(141 ページ) をお読みください。

ホームディレクトリを暗号化する方法

openSUSE では、暗号化されたホームディレクトリを作成することができます。ユーザがシステムにログインする、暗号化されたホームディレクトリ がマウントされ、内容をユーザから走査できるようになります。詳しくは 10.2項「暗号化されたホームディレクトリの使用」(142 ページ) をお読みください。

単一の ASCII テキストファイルを暗号化する方法

機密情報や秘密のデータを含む ASCII テキストファイルが少数だけ存在する場合、Kpgp や vi エディタを利用することで、それらを個別にパスワードで 保護することができます。詳しくは 10.3項「vi を使用した単一 ASCII テキストファイルの暗号化」(143 ページ) をお読みください。

警告: 暗号化メディアの保護範囲について

本章で言及している方法は、いずれも限定的な保護しか提供しないことに注意してください。たとえばお使いのシステムを攻撃から防ぐような機能はありません。また、暗号化メディアを正しくマウントすると、適切なアクセス権を持つユーザであれば、中身のデータに自由にアクセスすることができる点にも注意してください。ただし、暗号化されたメディアはコンピュータを紛失した場合や盗難にあった場合に、秘密のデータを不当に読み取られる行為から保護することができます。

10.1 YaST を利用した暗号化ファイルシステムの設定

YaST を利用することで、インストール時やインストール後のシステムから、お使いのシステム内にあるパーティションやファイルシステムの一部を暗号化 することができます。ただし、インストール済みのシステムでパーティションを 暗号化するのは、パーティションのサイズ変更と設定変更を伴うことになるため、さらに難しい手順を行なう必要があります。このような場合は、既存の パーティション内に暗号化ファイルを作成し、他のファイルやファイルシステムの 一部を 包含させる ほうが便利です。パーティション全体を 暗号化するには、パーティションレイアウト内に専用のパーティションを用意する 必要があります。なお、YaST 側で提示する標準のパーティション設定の提案では、既定では暗号化パーティションの設定を行いません。このような場合は、パーティション設定ダイアログから手作業で設定してください。

10.1.1 インストール時の暗号化パーティション作成

警告: パスワード入力について

暗号化パーティションに設定するパスワードは、忘れることの無いように注意してください。パスワードを忘れてしまうと、暗号化されたデータにアクセスできなくなるほか、復元も行なうことができません。

YaST での熟練者向けパーティション設定ダイアログでは、暗号化パーティション を設定するのに必要なオプションを提供しています。新しい暗号化パーティションを 作成するには、下記の手順で行ないます:

- 1 システム > パーティション設定 を選択し、熟練者向けパーティション設定ダイアログを表示させます。
- 2 ハードディスクを選択して *追加* を押し、プライマリパーティション または拡張パーティションを選択します。
- 3 次にパーティションのサイズを設定するか、パーティションに割り当てるディスク範囲を 設定します。
- 4 さらにファイルシステムと、対象のパーティションに設定するマウントポイントを設定 します。
- 5 デバイスの暗号化 のチェックボックスにチェックを入れます。

注記: 追加のソフトウェア要件について

デバイスの暗号化 にチェックを入れると、追加のソフトウェア をインストールする旨のポップアップウィンドウが表示される場合があります。暗号化パーティションを正しく動作させるには、提示されたパッケージを全て インストールしてください。

- 6 次へ を押し、パーティションの暗号化に使用する パスワードを入力します。パスワードは画面に表示されないことに注意してください。また、入力ミスを防ぐため、パスワードは 2 度入力してください。
- 7 完了 を押して処理を完了させます。暗号化パーティション が作成されます。

/etc/fstab 内で暗号化パーティションを自動マウントするように設定している場合は、起動処理時にオペレーティングシステムから パスワードを尋ねます。いったんマウントが成功すれば、そのパーティションは すべてのユーザから利用できるようになります。

起動時の暗号化パーティションのマウントを行なわないようにするには、パスワードプロンプトで何も入力せず、Enter だけを押して いただきます。この方法では、暗号化されたファイルシステムはマウント されない状態で、オペレーティングシステムの起動が続けられます。この場合は暗号化されたパーティション内のデータにはアクセスできません。

起動処理時にマウントしなかった暗号化パーティションを、後からマウント したい場合は、お使いのファイルマネージャを開いてから、対象のパーティション を選択して開いてください。開く際にはパスワードを尋ねられますので、正しいパスワードを入力するとマウントが行なわれます。

既にパーティションの存在するマシンにシステムをインストールしようとしている場合は、インストール時に既存のパーティションを暗号化することも できます。この場合は、10.1.2項「実行中のシステムに対する暗号化パーティションの作成」(140 ページ)にある手順に従って作業を行なってください。なお、この作業を行なうと、対象のパーティション内にある全てのデータが破壊されることに注意してください。

10.1.2 実行中のシステムに対する暗号化パーティションの作成

警告: 実行中のシステムに対する暗号化設定について

実行中のシステムでも暗号化パーティションを作成することができますが、既存のパーティションに対して暗号化を設定すると、そのパーティション内にある全てのデータは破壊されるほか、既存のパーティションに対するサイズ 変更や再構成が必要となります。

実行中のシステムから YaST コントロールセンターを起動し、システム > パーティション設定 を選択します。ポップアップメッセージが表示されますので、はい を押して先に進みます。熟練者向けパーティション設定 が表示されたら、暗号化するパーティションを選択して 編集 を押します。残りの手順は 10.1.1項「インストール時の暗号化パーティション作成」(139 ページ)と同じです。

10.1.3 コンテナとしての暗号ファイルの作成

パーティションを使用する代わりに、秘密のファイルやフォルダを保持することのできる暗号化ファイルを作成する方法もあります。このようなコンテナ 型のファイルも、YaST の熟練者向けパーティション設定ダイアログから 設定できます。設定を行なうには、*暗号ファイル > 暗号ファイルの追加* を選択し、暗号 ファイルのフルパスとサイズを指定します。新しくコンテナ型のファイルを作成する場合は、*ループファイルの作成* を選択します。あとはフォーマットの設定とファイルシステムの種類について、提案内容をそのまま受け入れるか変更するかしたあと、マウントポイントを指定します。なお、*デバイスの暗号化* のチェックボックスにチェックが入っていることを確認してください。

次へ を押し、暗号化に使用するパスワードを入力して *完了* を押します。

暗号化パーティションと比べて暗号化ファイルは、ハードディスクの パーティション設定の変更を行なう手間が無い、という利点があります。これは暗号化ファイルがループバックデバイスと呼ばれる仕組みを利用して マウントしているため、これによって通常のパーティションとほぼ同じ 動作をすることができるようになっています。

10.1.4 リムーバブルメディアの内容の暗号化

YaST ではリムーバブルメディア (外付けのハードディスクや USB メモリ など) をハードディスクと同様に取り扱います。もちろんコンテナ型の暗号化 ファイルや暗号化パーティションは、上述のとおりの手順で作成し利用することができます。ただし、リムーバブルメディアは通常、システムの動作中に 取り付けられるものであるため、システム起動時のマウントは設定しないで ください。

YaST でリムーバブルメディアに対して暗号化を行なうと、KDE や GNOME の デスクトップでは暗号化パーティションを自動的に認識し、デバイスが接続されたときにパスワードを尋ねるプロンプトを表示します。また、KDE や GNOME で FAT 形式のリムーバブルメディアを接続すると、パスワードを入力したユーザ (つまりデスクトップを利用しているユーザ) がデバイスの所有者となり、ファイルの読み書きを行なうことができるようになります。FAT 以外のファイル システムの場合は、root から所有者情報を変更して、デバイス上にある ファイルを読み書きできるように設定してください。

10.2 暗号化されたホームディレクトリの使用

盗難や不正なアクセスからホームディレクトリ内のデータを保護するため、YaST のユーザ管理モジュールを利用してホームディレクトリの暗号化を行なうことができます。暗号化ホームディレクトリは、新しく作成するユーザと既存のユーザの両方に対して設定することができます。既存のユーザに対してホームディレクトリの暗号化や暗号化の解除を設定するには、対象となるユーザのログインパスワードを知っておく必要があります。詳しくは 項「暗号化ホームディレクトリの管理」(第10章 *YaST* を利用したユーザ管理, ↑ スタートアップ) をお読みください。

暗号化ホームディレクトリは、10.1.3項「コンテナとしての暗号ファイルの作成」(141 ページ) で説明しているコンテナ型のファイルとして作成されます。それぞれ暗号化 ホームディレクトリ 1 つに対して、/home 以下に 下記の 2 つのファイルが作成されます：

ユーザ名.img

ディレクトリを保持するイメージファイル。

ユーザ名.key

上記のイメージファイルに対する鍵で、ユーザのログイン用パスワードで保護されるファイル。

ホームディレクトリを暗号化した場合は、ユーザがログインしたときに暗号化が解除されます。内部的には *pam_mount* と呼ばれる PAM のモジュールが、これを実施します。暗号化ホームディレクトリ以外にログインの処理が必要となる場合は、それらを /etc/pam.d/ 内の設定ファイルで指定する必要があります。詳しくは 第2章 *PAM* を利用した認証 (17 ページ) および *pam_mount* のマニュアルページをお読みください。

警告: セキュリティの制限

ユーザのホームディレクトリの暗号化は、同じシステムを利用する他のユーザからの機密保護にはならないことに注意してください。他のユーザに対しても機密を保護したい場合は、システムを物理的に共有すべきではありません。

また、セキュリティ機密保護をさらに強化したい場合は、swap (スワップ) パーティションや /tmp, /var/tmp についても暗号化を行なうことができます。これらのパーティションやディレクトリには、秘密のデータを処理する際に 利用する一時

ファイルが存在していて、その中には秘密のデータが含まれる 可能性があるためです。swap, /tmp, /var/tmp の暗号化方法 には YaST パーティション設定モジュールを利用して行ないます。詳しくは 10.1.1 項「インストール時の暗号化パーティション作成」(139 ページ) または 10.1.3 項「コンテナとしての暗号ファイルの作成」(141 ページ) をお読みください。

10.3 vi を使用した単一 ASCII テキストファイルの暗号化

暗号化パーティションを利用するにあたっての欠点は明確です。パーティションが マウントされてしまえば、少なくとも root はそれらのデータにアクセスできてしまうという点です。これを防ぐには、vi を暗号化モードで使⽤します。

vi を暗号化モードで使⽤するには、新しいファイルを編集する際 `vi -x ファイル名` のようにして 起動します。起動するとパスワードを設定するように促されますので、そこで パスワードを入力すると、保存時にそのパスワードで暗号化して保存するよう になります。なお、対象のファイルにアクセスする際は、正しいパスワードを入力 するよう促されます。

さらに厳しいセキュリティを必要とする場合は、暗号化パーティション内に 暗号化し たテキストファイルを配置してください。vi による暗号化はそれほど 強度の強い ものではないため、暗号化パーティションを⽤しておくこと を お勧めします。

AIDE を利用した侵入検知

システムの機密を保つことは、重要性の高いシステムの管理者には必須の作業項目です。システムに侵入されないことを保証することは不可能であるため、定期的に (たとえば cron などを利用して) 普通以上のチェックを行ない、システムの安全性を確認しておくことが重要になります。これが AIDE (*Advanced Intrusion Detection Environment* (高度な 侵入検知環境)) の存在する理由です。

11.1 AIDE を利用する理由

簡易的なチェックとして、RPM のチェック機能を利用して不用意な変更が なされていないかどうかを調べる方法があります。パッケージマネージャには 内蔵の検証機能が備わっているため、管理下にあるファイルすべてに対して変更を チェックすることができます。管理下にあるすべてのファイルを検証するには、rpm -Va を実行します。しかしながら、このコマンドは 設定ファイルに対する変更も検出してしまうため、重要な変更点を検出するには さらなるフィルタを作らなければなりません。

また、RPM を使用するにあたってのもう 1 つの問題は、高度な知識を持つ 攻撃者であれば rpm コマンドそれ自身を改ざんしてしまい、攻撃者に対して侵入や root 権限の奪取を許すような rootkit のようなものを、検出から逃れるようにしてしまう可能性があるという点です。これを解決するには、インストール済みのシステムとは完全に独立した第 2 のチェック機構を 用意する必要があります。

11.2 AIDE データベースの設定

重要: インストール後の AIDE データベースの初期化

お使いのシステムにインストールを行なう前に、メディアのチェックサムが正しいことを確認し、正当なものであることをご確認ください (詳しくは 項「メディアのチェック」(付録A ヘルプとトラブルシューティング, ↑ スタートアップ) をお読みください)。システムへのインストールが行なわれると、AIDE のデータベースが初期化されますが、インストールやインストール後の作業を正しく動作させるため、コンピュータに対して接続されているネットワークケーブル類をすべて外し、コンソールから直接作業を行なってください。また、コンピュータを無人の状態にしたりすることもせず、AIDE のデータベースが作成されるよりも前にネットワークに接続することも避けてください。

AIDE は openSUSE の既定のインストール作業でインストールされることはありません。インストールを行なうには、YaST を起動して **ソフトウェア > ソフトウェア管理** を選択するか、もしくは root の状態からコマンドラインで `zypper install aide` と入力してください。

AIDE に対してどのファイルに対してどの属性をチェックするのかを指定するには、設定ファイル `/etc/aide.conf` から設定を行ないます。このファイルは実際の設定として使用する前に必ず修正されていなければなりません。最初のセクションでは、AIDE のデータベースファイルの場所など、一般的なパラメータ設定を行ないます。ローカル側の設定で重要となる設定は Custom Rules と Directories and Files の各セクションです。一般的には下記のようなルールを指定します:

```
BinLib      = p+i+n+u+g+s+b+m+c+md5+sha1
```

BinLib の変数について設定を行なったあとは、ファイルの 章でチェックオプションを設定します。重要なオプションは下記のとおりです:

表 11.1 重要な AIDE チェックオプション

オプション	説明
p	選択したファイルやディレクトリに対する、ファイルパーミッションのチェック
i	inode 番号についてのチェック。それぞれのファイル名には、固有の

オプション	説明
	inode 番号があり、それは変わるべきではないものです。
n	対象のファイルを指すリンク数のチェック。
u	ファイルの所有者が変化しているかどうかのチェック。
g	ファイルの所有グループが変化しているかどうかのチェック。
s	ファイルサイズが変化しているかどうかのチェック。
b	ファイルが使用しているブロック数が変化しているかどうかのチェック。
m	ファイルの修正日時が変化しているかどうかのチェック。
c	ファイルのアクセス日時が変化しているかどうかのチェック。
md5	ファイルの MD5 チェックサムが変化しているかどうかのチェック。
sha1	ファイルの SHA1 (160 ビット) チェックサムが変化しているかどうかのチェック。

下記は /sbin ディレクトリ内にあるすべての ファイルに対してチェックを行なう設定で、BinLib として定義したオプションでチェックを行ない、/sbin/conf.d/ ディレクトリを除外する設定です:

```
/sbin BinLib
!/sbin/conf.d
```

AIDE データベースを作成するには、下記の手順で行ないます:

- 1 /etc/aide.conf を開きます。
- 2 チェックオプションを設定して、どのファイルをチェックすべきかを設定します。
利用可能なチェックオプションについて、完全な一覧は /usr/share/doc/packages/aide/manual.html をお読みください。なお、ファイル選択の設定を行なうにあたって、正規表現の知識が少し必要となります。設定を行なったら保存します。
- 3 設定ファイルに正しく記述できているかどうかを確認するには、下記を実行します:

```
aide --config-check
```

このコマンドからの出力は、設定ファイルが正しくない場合にその解決のためのヒントとなるものです。たとえば下記のような出力が発生したと仮定します:

```
aide --config-check
35:syntax error:!
35:Error while reading configuration:!
Configuration error
```

上記の出力は /etc/aide.conf ファイルの 36 行目に エラーがあることを示しているものです。エラーメッセージには、設定ファイルの中で正しく読み込まれた最後の行を表示していることに注意してください。

- 4 AIDE データベースを初期化するには、下記のコマンドを実行します:

```
aide -i
```

- 5 生成されたデータベースは CD-R や DVD-R、もしくはリモートのサーバや USB メモリなどにコピーして、後日使用します。

重要:

この手順は、データベースを改ざんされたりしないために必要な作業です。また、データベースが書き換えられたりすることを防ぐため、一度だけ書き込むことのできるメディアを使用しておくことをお勧めします。データベースを監視対象のマシンにだけ残したりすることは、絶対にしないでください。

11.3 ローカルの AIDE チェック

ファイルシステムのチェックを行なうには、下記の手順で行ないます:

1 データベースをリネームします:

```
mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db
```

2 設定を変更したりした場合は毎回、AIDE のデータベースを再度初期化して新しく生成したデータベースを移動します。データベースをバックアップしておくことも良い方法です。詳しくは 11.2 項「AIDE データベースの設定」(146 ページ)をお読みください。

3 下記のコマンドでチェックを行ないます:

```
aide --check
```

何も出力されない場合は、改ざんが行なわれていないことを示します。AIDE が何らかの改ざんを検知すると、下記のような変更点の概要が表示されます:

```
aide --check
AIDE found differences between database and filesystem!!
```

```
Summary:
Total number of files:      1992
Added files:                0
Removed files:             0
Changed files:              1
```

実際の変更点を確認するには、パラメータ `-V` を追加して 冗長出力レベルを上げて実行します。上記の例について 冗長性を上げると、下記のような出力になります:

```
aide --check -V
AIDE found differences between database and filesystem!!
Start timestamp: 2009-02-18 15:14:10
```

```
Summary:
Total number of files:      1992
Added files:                0
Removed files:             0
Changed files:              1
```

```
-----
Changed files:
-----
```

```
changed: /etc/passwd
```

```
-----
Detailed information about changes:
-----
```

```
File: /etc/passwd
```

```
Mtime      : 2009-02-18 15:11:02          , 2009-02-18 15:11:47
Ctime      : 2009-02-18 15:11:02          , 2009-02-18 15:11:47
```

この例では、`/etc/passwd` ファイルの変更日時とアクセス 日時が変化したことを示しています。

11.4 システムに依存しないチェック

リスクを嫌う管理者 (管理者以外でもかまいません) の場合は、信頼のおけるソースが提供する AIDE のバイナリを実行しておくことをお勧めします。このようにすることで、悪意のある攻撃者が AIDE のバイナリまでも改ざんしてしまい、本来の改ざんチェックを動作できなくなってしまうような問題を未然に回避することができます。

これを行なうには、AIDE をインストール済みシステムから独立した、レスキューシステムから実行しなければなりません。openSUSE では 比較的任意のプログラムを追加しやすいレスキューシステムが用意されていますので、ここから作業を行います。

レスキューシステムを利用して起動する前に、2 種類のパッケージを用意する必要があります。これはシステムに対してドライバ更新ディスクを追加するのと同じ書式で追加できるもので、`linuxrc` から行ないます。詳しくは <http://en.opensuse.org/SDB:Linuxrc> をお読みください。下記ではこの手順を説明しています。

手順 11.1 AIDE を利用したレスキューシステムの起動

1 2 台目のマシンを用意し、FTP サーバを起動します。

2 FTP サーバのディレクトリ内に `aide` と `mhash` の各パッケージをコピーします。openSUSE の FTP サーバ機能であれば `/srv/ftp/` になります。具体的にはアーキテクチャとバージョンの項目を実際の値に置き換え、下記のように実行します:

```
cp DVD1/suse/アーキテクチャ/aide/バージョン.アーキテクチャ.rpm /srv/ftp
cp DVD1/suse/アーキテクチャ/mhash/バージョン.アーキテクチャ.rpm /srv/ftp
```

3 レスキューシステムに対して必要な起動パラメータを提供するため、`/srv/ftp/info.txt` ファイルを下記の内容で 作成します:

```
dud:ftp://ftp.example.com/aide/バージョン.アーキテクチャ.rpm
dud:ftp://ftp.example.com/mhash/バージョン.アーキテクチャ.rpm
```

それぞれ `ftp.example.com` は実際の FTP サーバのドメイン名に、バージョンとアーキテクチャは実際の値に置き換えてください。

- 4 次に AIDE チェックを行なう必要のあるマシンを再起動し、お持ちの DVD からレスキューシステムを起動します。このとき、下記の起動パラメータを設定してください:

```
info=ftp://ftp.example.com/info.txt
```

このパラメータを設定すると、`linuxrc` に対して `info.txt` ファイル内の情報を読み込むように指示します。

レスキューシステムが起動したら、AIDE プログラムが利用できるようになります。

11.5 さらなる情報

AIDE に関する情報は、それぞれ下記の場所にあります:

- AIDE の Web ページ (ただし英語であることに注意してください): <http://aide.sourceforge.net>.
- `/etc/aide.conf` ファイル内のドキュメント。
- aide パッケージをインストールすることで作成 される、`/usr/share/doc/packages/aide` 以下のファイル。
- AIDE ユーザのメーリングリスト (ただし英語であることに注意してください): <https://mailman.cs.tut.fi/mailman/listinfo/aide>.

パート III. ネットワークセキュリティ

SSH: 機密を保護する通信

ネットワークに接続された環境の場合、ネットワーク上離れた場所から特定のホストにアクセスするような要件がしばしばあります。認証時に、利用者がログイン(ユーザ名)とパスワードの文字列を暗号化せずに送信する場合、それらの情報はネットワーク上で傍受することができてしまい、それらの情報をもとに許可していない利用者が特定のユーザアカウントを乗っ取ることができてしまいます。これにより、利用者のファイルを攻撃者が開くことができるほか、場合によっては管理者(root)の権限を奪取するための足がかりとなりますし、他のシステムに攻撃を加えるための踏み台としても利用できるようになってしまいます。以前はリモートからの接続はtelnetやrsh, rloginなどを利用してきましたが、暗号化やその他の仕組みによる保護が存在していませんでした。またtelnet以外にも、以前より使用されているFTPプロトコルやrcpのようなりモートファイルコピーのプログラムなど、通信データを保護していないものはいくつか存在しています。

SSHスイートでは、認証文字列(通常はログイン名とパスワード)について暗号化を行なうほか、ホスト間の通信そのものについても暗号化を行ない、必要な保護機能を提供しています。SSHでもデータの通信そのものは第三者が傍受することができてしまいますが、内容は暗号化されているため、暗号鍵を知らない限りはそれらを元のデータ(暗号化前のデータ)に戻すことはできません。そのため、SSHは機密の保護されていないネットワーク、たとえばインターネットなどにおいて、機密を保護できるようにすることができます。openSUSEにおけるSSHの実装として、OpenSSHが利用できます。

openSUSEでは既定でOpenSSHパッケージがインストールされるようになっていて、それぞれssh, scp, sftpの各コマンドを利用することができます。また、既定の設定ではopenSUSEシステムへは、sshdを起動させてファイアウォールでの

アクセス許可を設定した状態で、OpenSSH ユーティリティを利用した場合にのみアクセス可能です。

12.1 ssh—Secure Shell (機密を保持するシェル)

ssh プログラムを利用することで、リモートのシステムに ログインして対話的な作業を行なうことができます。たとえば sun というホストに対して tux というユーザでログインしたい場合は、下記のコマンドラインのうちのいずれかを入力します：

```
ssh tux@sun
ssh -l tux sun
```

接続元と接続先でユーザ名が同じである場合は、ユーザ名を省略することもできます (ssh sun)。いずれの場合も、リモート側の ユーザに対するパスワード入力を求められます。認証が成功すると、リモート側のコマンドラインや対話的なアプリケーション (たとえば YaST など) を利用した作業を行なうことができます。

さらに ssh では、ssh ホスト コマンド のように入力することで、対話機能を備えていないコマンドを実行することもできます。ただし、コマンド には適切に引用符を付ける必要があります。また、シェルでの作業のように、複数のコマンドを繋げて実行することも できます。

```
ssh root@sun "dmesg | tail -n 25"
ssh root@sun "cat /etc/issue && uptime"
```

12.1.1 X アプリケーションのリモートホストでの実行

SSH では、ネットワーク上離れた場所にある X アプリケーションの利用を 簡単に 行なうことができます。ssh コマンドを実行する 際に -X オプションを設定すると、リモート側の DISPLAY 環境変数が自動的に設定され、X の出力を SSH 接続経由で手元に持ってくるすることができます。また同時に、自分以外の ユーザが、この出力を悪用できないようにも設定します。

12.1.2 エージェント転送

-A オプションを指定すると、ssh-agent によって認証 情報が次のマシンにも引き継がれるようになります。この方法を利用することで、それぞれ異なるマシンに対してログインする際、パスワードの入力を省略することができます。ただし、この方法を利用するには、接続先のホストに対して あらかじめ公開鍵を配布しておき、適切な場所に保存しておく必要があります。

この仕組みは既定の設定では無効化されていますが、システム全体の設定ファイルである /etc/ssh/sshd_config ファイル内で AllowAgentForwarding yes と設定することで、恒久的に 有効化することができます。

12.2 scp一機密保護機能付きのファイルコピー

scp はローカル側のファイルをリモート側のマシンにコピー したり、逆にリモート側のマシンをローカルにコピーしたりすることができます。jupiter 上のユーザが sun 上のユーザと異なる場合でも、ユーザ名@ホスト の形式で指定することで解決できます。リモート側のホームディレクトリ以外にファイルをコピーしたい場合は、sun:ディレクトリ のように指定します。下記の例では、ローカルからリモートに対するファイルコピーと、その逆を行なっています。

```
# ローカル -> リモート
scp ~/MyLetter.tex tux@sun:/tmp
# リモート -> ローカル
scp tux@sun:/tmp/MyLetter.tex ~
```

ヒント: -l オプションについて

ssh コマンドでは -l オプションを利用することで、リモート側のユーザを指定することができます (ユーザ名@ホスト 形式での指定と同じ意味です)。ですが、scp での -l は、scp で使用するネットワーク 帯域を制限する意味で使います。

正しいパスワードを入力すると、scp はデータ転送を開始します。開始と同時に進捗 状況が表示され、コピー対象のファイル転送が終了するまでの時間を表示します。-q オプションを指定すると、これらの出力を省略することができます。

scp ではディレクトリ全体をコピーするための再帰コピー機能が用意されています:

```
scp -r src/ sun:backup/
```

上記を実行すると、src ディレクトリとそのサブディレクトリ 内にある全ての内容を、sun 内の ~/backup ディレクトリにコピーする意味になります。なお、サブディレクトリが存在しない 場合は、自動的に作成されます。

なお、-p オプションを指定すると、ファイルのタイムスタンプを 維持するようになります。また、-C オプションを指定すると、データ転送時に圧縮機能を利用するようになります。この圧縮機能により転送すべき データ量を減らすことができますが、プロセッサに対してそれなりの負荷がかかります。

12.3 sftp—機密保護機能付きのファイル転送

複数のプログラムを異なる場所にコピーするような場合は、scp の代わりに sftp が便利です。これは通常の ftp コマンドの ように、対話的にコマンドを入力して実行するシェルとして動作します。利用可能な コマンドの一覧を表示するには、sftp のシェルから help と入力してください。また、さらに詳しい説明は sftp (1) のマニュアルページにあります。

```
sftp sun
Enter passphrase for key '/home/tux/.ssh/id_rsa':
Connected to sun.
sftp> help
Available commands:
bye                               Quit sftp
cd path                           Change remote directory to 'path'
[...]
```

12.4 SSH デーモン (sshd)

ssh や cp のような SSH クライアントプログラムで作業を行なうためには、サーバ (SSH デーモン) が裏で動き、TCP/IP ポート 22 で接続を 待ち受けていなければなりません。デーモンは初回の起動時に 3 種類の鍵対を 生成します。それぞれの鍵対は機密鍵と公開鍵から構成されるもので、一般には 公開鍵ベースの手順と呼ばれます。SSH を介した通信の機密を保持するため、機密鍵へのアクセスはシステム管理者に限定しておく必要があります。既定の インストールでは、それらのファイルパーミッションは自動的に適切な値が設定 されます。機密鍵は SSH デーモンのみが読み取ればよいものであるため、SSH デーモン以外には読み取れないようになっている、公開鍵 (.pub という拡張子になっています) は接続が開始された

時にクライアントに送信されるものであるため、すべてのユーザから読み取ることができるようになっています。

接続は SSH クライアント側から行なわれます。接続が行なわれると、待ち受けていた SSH デーモンが SSH クライアントの間でプロトコルやソフトウェア バージョンなどの識別情報を交換し、正しくないポートに接続してしまったようなことを防ぐようになっています。なお、SSH デーモンから起動された子プロセスが リクエストに対する応答を行なうため、デーモン側では複数の SSH 接続を同時に 受け入れることができます。

SSH サーバとクライアントの間の通信では、OpenSSH はバージョン 1 と 2 の SSH プロトコルに対応しています。既定ではバージョン 2 を使用するようになっていますが、-1 オプションを指定することでバージョン 1 に強制することもできます。

SSH のバージョン 1 を使用する場合、サーバはまず公開鍵とサーバ鍵を送信します。サーバ鍵は毎時生成される鍵で、これら 2 種類の鍵を利用し SSH クライアント 鍵が自由に設定し、サーバ側に送信するセッション鍵を暗号化できるようになっています。SSH クライアントでは、使用する暗号化方法もサーバ側に送信します。SSH プロトコルのバージョン 2 ではサーバ鍵を必要とせず、Diffie-Hellman と呼ばれるアルゴリズムを利用して鍵の交換を行ないます。

ホスト側の機密鍵とサーバ鍵は、いずれもセッション鍵の暗号を解除するのに絶対に必要となるもので、公開鍵からは決して導き出すことのできないものです。これにより、接続先の SSH デーモンだけが機密鍵を利用して、セッション鍵の暗号を解除できる 仕組みになっています。接続当初の通信のやりとりを見たい場合は、SSH クライアント 側で -v オプションを指定し、デバッグモードを有効にしてください。

/etc/ssh/ 内に保存されている公開鍵と機密鍵については、機密を保持できる外部媒体に保管しておくことをお勧めします。このようにすることで、鍵が改変されてもそれを検出することができるほか、新しくシステムをインストールした場合でも、元の鍵を使い続けることができるようになります。

ヒント: 既存の SSH ホスト鍵について

既に何らかの Linux がインストールされているマシンに openSUSE をインストールした場合、インストール処理内でもっともアクセス日時の新しい SSH の鍵を自動的に取り込みます。

あるリモートのホストに対してはじめて SSH 接続を行なうと、クライアント側ではそのホストの公開鍵を ~/.ssh/known_hosts ファイル内に 保存します。これにより、man-in-the-middle 攻撃 (中間者攻撃) を防ぐことができます。中間者攻撃とは、

ホスト名や IP アドレスを偽って本来とは異なる偽の サーバに接続させる攻撃で、あらかじめ `~/.ssh/known_hosts` 内に正しい公開鍵を登録していれば、鍵が変化したことによってこれを検知することができます。サーバ側でも、正しい公開鍵に対応する機密鍵を持っていないことから、セッション情報を解読することができなくなり、クライアント側の機密を維持することができます。

ホスト側の公開鍵が変更された場合 (このようなサーバに接続する際は、あらかじめ何らかの方法でこれを確かめておく必要があります) は、`ssh-keygen -r ホスト名` で古いほうの鍵を削除することができます。

12.5 SSH 認証の仕組み

認証でもっとも単純な形態は、ユーザがログインする際にそのユーザの パスワードを入力させる形態です。しかしながら、リモートのマシン上に 登録されたユーザのパスワードだけでは保護が不十分です。また、これらのパスワードは変更される場合があります。たとえば `root` のアクセスを許可するような場合、管理者は `root` のパスワードを 変更したりすることなく、より明示的に不正なアクセスを拒否する必要があります。

ユーザのパスワードを入力せずにログインさせたいような場合は、ユーザ側で 生成した SSH 鍵対を利用して認証を実施します。ユーザ側で生成した鍵対は、公開鍵 (`id_rsa.pub` または `id_dsa.pub`) と機密鍵 (`id_rsa` または `id_dsa`) が対になっています。

パスワード無しでのログインを行なうには、生成した鍵対のうち公開鍵を、SSH サーバの「ユーザ」のホームディレクトリ内、`~/.ssh/authorized_keys` ファイルに保存します。このような方法を採用することで、ログインに関して完全な制御を実現することができます。これは公開鍵を追加したり削除したりするには、そのユーザで (パスワードなどによる) ログインを行なう必要があるためです。

最大限のセキュリティを実現するため、これらの鍵はパスフレーズで保護しておくべきです。このパスフレーズは `ssh`, `scp`, `sftp` を利用するたびに入力しなければならないもので、単純なパスワード認証に比べると、パスフレーズは 直接ユーザに紐づく情報ではなく独立した情報であるため、より安全性を 高めることに繋がります。

上述の鍵ベースの認証以外にも、SSH ではホストベースの認証に対応しています。ホストベースの認証を利用すると、一方のホストにいるユーザ (信頼される側) は 他方のホストにいるユーザ (信頼する側) としてログインすることができるようになります。これには両方のホストでその機能を有効にする必要があるほか、信頼する側とされる側で同じユーザ名である必要があります。openSUSE では 鍵ベース

の認証が設定されているため、ホストベースの認証については、ここでは 説明していません。

注記: ホストベースの認証を利用する場合のファイルパーミッションについて

ホストベースの認証を使用する場合、`/usr/lib/ssh/ssh-keysign` (32 ビットシステムの場合) または `/usr/lib64/ssh/ssh-keysign` (64 ビットシステムの場合) に対して、`setuid` ビットを設定する必要があります。これは openSUSE では既定で設定されない項目です。ホストベースの認証を使用する場合は、ファイルのパーミッションを手作業で設定してください。対応の際には、`/etc/permissions.local` ファイルを使用すると、`openssh` のセキュリティ更新が発生したような場合でも `setuid` ビットを保持し 続けることができます。

12.5.1 SSH 鍵の生成

- 1 既定のパラメータ (RSA, 2048 ビット) で鍵を生成するには、`ssh-keygen` コマンドを実行します。
- 2 Enter を押して既定の場所 (`~/.ssh/id_rsa`) に鍵を保存するか、もしくは独自の場所を指定します。
- 3 10 から 30 文字程度で構成されるパスフレーズを入力します。パスフレーズについては必ず設定しておくことを強くお勧めします。

なお、自分自身以外には機密鍵にアクセスできないようになっていることを、必ず確かめてください (ファイルのパーミッションを `0600` であれば問題ありません)。また、機密鍵を他のユーザに対して開示しては なりません。

既存の鍵対に対するパスフレーズを変更するには、`ssh-keygen -p` コマンドを使用します。

12.5.2 SSH 鍵のコピー

作成した公開鍵をリモート側の `~/.ssh/authorized_keys` にコピーするには、`ssh-copy-id` コマンドを使用します。ローカル側で `~/.ssh/id_rsa.pub` 以下に保存されている鍵であれば、単純なコマンドで実施できます。DSA 鍵や他のユーザの鍵を コピーしたい場合は、パスを指定する必要があります:

```
# ~/.ssh/id_rsa.pub  
ssh-copy-id -i tux@sun
```

```
# ~/.ssh/id_dsa.pub
ssh-copy-id -i ~/.ssh/id_dsa.pub tux@sun

# ~/.ssh/id_rsa.pub
ssh-copy-id -i ~/.ssh/id_rsa.pub tux@sun
```

鍵をコピーするには、リモートユーザ側のパスワードを入力してください。登録済みの鍵を削除するには、`~/.ssh/authorized_keys` ファイルを手作業で編集します。

12.5.3 ssh-agent コマンドの使用

様々な SSH 接続を利用して作業を行なう場合、それぞれの接続に対してパスフレーズを入力するのは、非常に面倒な作業です。この問題を解決するため、SSH パッケージには `ssh-agent` と呼ばれるツールが付属しています。これは X や端末セッションが維持されている間、機密鍵を保持 することができるツールです。その機密鍵を必要とする他のウィンドウや プログラムを `ssh-agent` の子プロセスとして 動作させることで、`ssh`, `scp`, `sftp` に必要な環境変数が設定され、自動的なログインを実現することができます。詳しくは `man 1 ssh-agent` をお読みください。

`ssh-agent` を起動したあとは、`ssh-add` コマンドで鍵を追加します。この時点でパスフレーズの入力を求められます。いったんパスフレーズを入力すると、それ以降はパスフレーズを再度入力したり することなく SSH の各コマンドを使用できるようになります。

12.5.3.1 ssh-agent の X セッション内での使用

openSUSE では、GNOME や KDE のディスプレイマネージャが自動的に `ssh-agent` を起動します。起動したエージェントに 対して、X セッションの起動時に `ssh-add` を実行して 鍵を追加するには、下記の手順で行ないます：

- 1 設定対象のユーザでログインし、`~/.xinitrc` ファイルが存在しているかどうかを確認します。

- 2 存在していない場合は、既存の雛形を使用するか、`/etc/skel` ディレクトリからコピーします：

```
if [ -f ~/.xinitrc.template ]; then mv ~/.xinitrc.template ~/.xinitrc; ¥
else cp /etc/skel/.xinitrc.template ~/.xinitrc; fi
```

- 3 雛形をコピーした場合は、下記のような行を検索し、その行のコメント マーク (#) を削除します。既に `~/.xinitrc` が 存在していた場合は、下記の行を追加します (コメントマークは外してください)。


```
# if test -S "$SSH_AUTH_SOCK" -a -x "$SSH_ASKPASS"; then
#     ssh-add < /dev/null
# fi
```

- 4 あとは新しく X セッションを起動すると、SSH のパズフレーズ入力を求められるようになります。

12.5.3.2 ssh-agent の端末セッション内での使用

端末セッションから利用する場合は、手作業で ssh-agent を起動して ssh-add を実行する必要があります。ssh-agent の起動方法には 2 種類の方法があり、1 つは既存のシェル上で新しい bash シェルを起動する方法、もう 1 つは既存のシェルの置き換わる形で新しいシェルを起動し、必要な環境変数を設定する方法です。

```
ssh-agent -s /bin/bash
eval $(ssh-agent)
```

エージェントを起動した後は、ssh-add を実行して必要な鍵を追加します。

12.6 ポート転送

ssh は TCP/IP 接続を転送する用途でも使用することができます。この機能は SSH トンネルとも呼ばれ、暗号化された通信経路を介して、あるポート宛の通信を別のマシンに転送します。

上記のコマンドを実行すると、jupiter のポート 25 (SMTP) に宛てた接続が、暗号化された状態で sun の SMTP ポートに転送されるようになります。これは特に SMTP-AUTH や POP-before-SMTP の機能を持たない SMTP サーバを利用している場合に便利な仕組みです。接続を転送することで、ネットワーク上離れた任意のホストからの接続を、「自宅」などのメールサーバに転送することができます。

```
ssh -L 25:sun:25 jupiter
```

同様に、jupiter からの全 POP3 リクエスト (ポート 110) を sun 上の POP3 ポートに転送するには、下記のように実行します:

```
ssh -L 110:sun:110 jupiter
```

なお、いずれのコマンドとも root で実行しなければなりません。これは特権ポートと呼ばれるポートを使用しているためです。また、電子メールは通常のユーザから SSH 接続を通して送受信することができます。メールソフトからは localhost に設

定してお使いください。上述のプログラムについて、追加情報はマニュアルページと `/usr/share/doc/packages/openssh` にある OpenSSH のドキュメンテーションに記述されています。

12.7 YaST を利用した SSH デーモンの設定

YaST の SSHD 設定モジュールは、既定のインストールには含まれていません。利用するには `yast2-sshd` パッケージをインストールしてください。

`sshd` サーバを YaST で設定するには、YaST を起動して **ネットワークサービス > SSHD の設定** を選択してから、下記の手順を行ないます：

- 1 一般 タブでは、`sshd` が待ち受けるポートを *SSHD TCP ポート* で選択します。既定値は 22 で、複数のポートを指定することもできます。新しいポートを追加するには、*追加* を押してからポート番号を入力し、*OK* を押します。ポートを削除するには、表内から削除するポートを選んで *削除* を押したあと、確認メッセージに応答します。
- 2 また `sshd` デーモンがサポートすべき機能を選択することもできます。TCP 転送機能を無効化するには、*TCP 転送を許可する* のチェックを外します。ただし、TCP 転送機能を無効化しても、ユーザがシェルにアクセスできる限り、セキュリティの改善にはならないことに注意してください。シェルにアクセスできてしまえば、独自の転送プログラムを起動できてしまうためです。TCP 転送について、詳しくは 12.6 項「ポート転送」(163 ページ) をお読みください。

X Window System の転送を無効化するには、*X11 転送を許可する* のチェックを外します。このオプションが無効化されていると、クライアント から X11 転送の要求が行なわれた際にエラーを返します。ただし上記同様に 独自の転送プログラムを起動できてしまうことに注意してください。X Window System の転送について、詳しくは 12.1 項「ssh—Secure Shell (機密を保持するシェル)」(156 ページ) をお読みください。

また *圧縮を許可する* では、サーバとクライアント間の 通信を圧縮するかどうかを選択します。

- 3 **ログイン設定** のタブでは、一般的なログインや認証に関する設定を行ないます。*ログイン時に '本日のメッセージ' を表示する* では、ユーザが対話的にログインした場合に `/etc/motd` 内のメッセージを表示するかどうかを設定 することができ

ます。また、root ユーザのログインを無効化したい場合は、*root ログインを許可する* のチェックを外します。

最大認証試行回数 の設定では、1 回の接続あたりに 許可される、最大の認証試行回数を設定します。*RSA Authentication* の設定では、純粋な RSA による認証 を許可するかどうかを設定します。このオプションは SSH プロトコルのバージョン 1 にのみ適用されます。また、*Public Key Authentication* では、公開鍵による認証を許可するかどうかを設定します。こちらの設定は プロトコルバージョン 2 にのみ適用されます。

- 4 *プロトコルと暗号化* のタブでは、対応すべき SSH プロトコル のバージョンを設定することができます。バージョン 1 のみ、バージョン 2 のみ、バージョン 1 と 2 の両方の中から選択することができます。

許可する暗号化方法 では、許可するすべての暗号化が一覧で 表示されます。一覧から特定の暗号化方法を削除するには、一覧から選んで *削除* を押します。逆に一覧に暗号買う方法を追加するには、ドロップダウンリストから選んで *追加* を押します。

- 5 最後に *OK* ボタンを押すと、設定を保存することができます。

12.8 さらなる情報

<http://www.openssh.com>

OpenSSH の Web ページ

<http://en.wikibooks.org/wiki/OpenSSH>

OpenSSH Wikibook

man sshd

OpenSSH デーモンのマニュアルページ

man ssh_config

OpenSSH のクライアント側設定ファイルのマニュアルページ

man scp , man sftp , man slogin , man ssh , man ssh-add , man ssh-agent , man ssh-copy-id , man ssh-keyconvert , man ssh-keygen , man ssh-keyscan

ファイルのコピー (scp, sftp) やログイン (slogin, ssh) 、鍵管理などの各マニュアルページ。

マスカレードとファイアウォール

Linux をネットワーク環境で利用している場合、カーネル機能を利用してネットワークパケットの操作を行ない、内部と外部のネットワーク領域に関する区分を管理することができます。Linux の netfilter フレームワークでは、効果的なファイアウォールを提供し、ネットワークを正しく区切ることができます。また、iptables と呼ばれる、ファイアウォールのルールセット定義のために用意された汎用 テーブル構造を使用することで、ネットワークインターフェイスに対して通過を許すパケットを厳密に設定することができます。このようなパケットフィルタ機能については、SuSEfirewall2 と関連する YaST モジュールを利用することで、簡単に設定できるようになっています。

13.1 iptables を利用したパケットフィルタ

netfilter と iptables の各コンポーネントは、それぞれネットワークパケットのフィルタや操作に関する機能を提供するほか、ネットワークアドレス変換 (NAT) の機能も提供しています。フィルタする条件とその動作 (受け入れるか捨てるかなど) は、チェーンと呼ばれる領域に保管します。チェーンは上から順に該当するかどうかを判断するようになっています。また、チェーンはテーブルと呼ばれる領域に含まれ、iptables コマンドでは、これらのテーブルやチェーンに対して変更を行なうことができます。

Linux カーネルには 3 種類のテーブルが存在しています。各テーブルは、パケットフィルタに関する各カテゴリに対応しています:

filter

このテーブルにはフィルタルールが含まれています。一般にパケット フィルタリングと呼ばれるルールを記述する場所で、あるパケットを通す (ACCEPT) か捨てる (DROP) かを判断します。

nat

このテーブルには発信元のアドレスや送信先のアドレスについて、それらを変更するためのルールを記述します。このテーブルを利用することで、マスカレードと呼ばれる、プライベートなネットワークをインターネットに接続するような NAT の特殊ケースに対応することができるようになります。

mangle

このテーブル内には、IP ヘッダの値 (たとえば ToS フィールド)などを編集するルールを保管します。

これらのテーブルには、役割ごとに決められたチェーンが存在しています:

PREROUTING

このチェーンは、到着するパケットに対して適用されます。

INPUT

このチェーンは、そのマシンが送信先となっているパケットに対して適用されます。

FORWARD

このチェーンは、そのマシンを単純に経由するだけのパケットに対して適用されます。

OUTPUT

このチェーンは、そのマシンを発信元とするパケットに対して適用されます。

POSTROUTING

このチェーンは、そのマシンから出て行くパケットに対して適用されます。

図13.1「iptables: ありうるパケット経路」(169 ページ)では、どのような経路でネットワークパケットが処理されるのかを示しています。単純化のため、図ではチェーンの一部としてテーブルを描いていますが、実際にはチェーンはテーブル自身の中に存在しています。

この仕組みを最も簡単に表わすとなると、まずシステムに到着するパケットは、eth0 のようなインターフェイス宛に到着します。すると到着したパケットは最初に mangle テーブル内の PREROUTING チェインを参照し、続いて nat テーブル内の

PREROUTING チェインを参照します。次にパケットのルーティング (経路制御) に関する処理を行ない、システム内のどのプロセスがパケットを受け取るのかを判断します。その後 `mangle` と `filter` の各テーブル内にある INPUT チェインを参照し、`filter` テーブル内のルールで指定した最終的な送付先に到着します。

図 13.1 *iptables: ありうるパケット経路*

13.2 マスカレードの基礎

マスカレードとは Linux 固有の NAT (ネットワークアドレス変換) 機能です。小規模な LAN (プライベートアドレスと呼ばれる IP アドレスを使用するホスト。詳しくは 項「ネットマスクとルーティング」(第13章 ネットワークの基礎, ↑リファレンス) をお読みください) からインターネット (公式の IP アドレスを使用する領域) に対して、接続を行なう場合に利用します。LAN 内のホストがインターネットに接続できるようにするには、プライベートなアドレスを公式のものに書き換える必要がありますが、この作業は一般に LAN とインターネットを繋ぐルータが実施するものです。ルータは非常にシンプルな仕組みで動作しています。ルータには複数のネットワークインターフェイスがあり、一方 (複数である場合もあります) を LAN に、他方をインターネットに接続しています。LAN に接続されたホスト (たとえば `eth0`などを介して) から、デフォルトゲートウェイ (またはルータ) として上述のルータを指定することで、インターネットに接続できるようになっています。

重要: 正しいネットマスクの使用について

ネットワークの設定にあたっては、LAN 内のすべてのホストでブロードキャストアドレスとネットマスクが正しく設定されていることを確認してください。これらが正しく設定されていないと、パケットのルーティング (経路制御) が正しく動作しなくなってしまうです。

上述のとおり、LAN 内のホストからインターネットアドレス宛のパケットを送信した場合、それらはデフォルト (既定の) ルータに送信されます。そのため、そのようなパケットを転送できるよう、ルータを設定しなければなりません。ですが、セキュリティ上の理由から、既定のインストールではルータとして動作するようになっていません。ルータ機能を有効にするには、`/etc/sysconfig/sysctl` 内の `IP_FORWARD` の項目について、`IP_FORWARD=yes` を設定してください。

接続先のホストは上述のルータと直接通信できますが、実際のパケット発信元である内部のネットワークについては知る術がありません。これがマスカレードの存在す

る理由です。マスカレード (アドレス変換) では、接続先のホストが応答を返す際、その宛先アドレスはルータのものに設定されます。ルータは到着するパケットを判断し、それが LAN 内からのパケットに対する応答であることがわかった場合は、パケットの宛先アドレスを変換して、ローカルネットワーク内の正しいホストに転送します。

到着するパケットのルーティング (経路制御) がマスカレードのテーブルに依存している仕組みにより、外側 (インターネット) から内部 (LAN) のホストに対して接続する手段は全く提供されないことになります。外側から内側のホストに接続しようとしても、マスカレードのテーブル内には、その情報が全く存在しないためです。加えて、マスカレードのテーブル内に設定済みの接続であっても、そこには状態に関する情報が保持されているため、他の接続からその接続を使用するようなこともできなくなっています。

また、これらの影響により、いくつかのアプリケーションプロトコルを利用する際に問題が生じる場合があります。たとえば ICQ, cucme, IRC (DCC, CTCP), FTP (PORT モード) などがそれに該当します。Web ブラウザや標準の FTP プログラムの場合は、PASV (passive) と呼ばれるモードを使用しますが、このモードであればパケットフィルタリングやマスカレードに対して問題を発生しにくくすることができるようになっています。

13.3 ファイアウォールの基礎

ファイアウォールとはおそらく、ネットワーク間を流れるデータの制御を行なう仕組みを説明する用語として、もっとも広く使用されているものであると思われます。厳密に言えば、この章で説明していることはパケットフィルタと表現します。パケットフィルタは特定の条件、たとえばプロトコルやポート、IP アドレスなどを元にして、データの流れを規制するためのものです。これにより、お使いのネットワークに届いて欲しくないパケットを、それらのアドレスをもとに防ぐことができるようになります。たとえばお使いの Web サーバに外部からのアクセスを許可したい場合は、Web サーバに使用しているポートを指定し、開くことで許可を実現します。しかしながら、パケットフィルタでは内容を調べることができないため、公開している Web サーバなどのように、正しいアドレスを持つものはそのまま通してしまいます。たとえば Web サーバ内にある CGI プログラムの脆弱性への攻撃などについては、パケットフィルタで防ぐことはできません。

より効果的で複雑な方法として、複数のシステムを組み合わせる方法があります。たとえばパケットフィルタと、アプリケーションゲートウェイやプロキシを併用するな

どの方法があります。この方法ではまず、パケットフィルタが無効なポートへのアクセスを遮断し、アプリケーションゲートウェイの通信のみを受け入れます。このゲートウェイ (プロキシ) がクライアントを代行する役割を果たし、本来のサーバへアクセスを行ないます。言い換えれば、このようなプロキシ ("代理" という意味です) は元のホストに対する代理役で、アプリケーションが使用するプロトコルを中継することになります。このようなプロキシサーバとして、たとえば Squid がありますが、これは HTTP や FTP に対応したプロキシサーバです。Squid を使用するには、ブラウザ側の設定を変更してプロキシサーバを使用するようにしなければなりません。任意の HTTP ページや FTP ファイルへのアクセスのうち、プロキシ内部が持つキャッシュ (一時保存データ) 内に存在しないものについてのみ、プロキシサーバがインターネットにアクセスします。

下記の章では、openSUSE におけるパケットフィルタについて説明しています。パケットフィルタとファイアウォールについて、より詳しい説明は `howto` パッケージ内に含まれる `Firewall HOWTO` をお読みください。パッケージがインストールされていれば、下記のコマンドで内容を表示することができます。

```
less /usr/share/doc/howto/en/txt/Firewall-HOWTO.gz
```

13.4 SuSEfirewall2

SuSEfirewall2 は `/etc/sysconfig/SuSEfirewall2` 内に設定された変数を読み込んで、`iptables` のルール群を生成するスクリプトです。SuSEfirewall2 では 3 種類のセキュリティゾーンを定義しますが、下記に続く例では 1 つめと 2 つめだけを使用しています:

外部ゾーン

外部ネットワークからの攻撃は発生しうるものであることから、ホストはそれらのネットワークから保護する必要があります。多くの場合、外部ネットワークとはインターネットのことを指しますが、場合によってはその他の安全でないネットワーク、たとえば無線 LANなどを指す場合もあります。

内部ゾーン

内部ゾーンとは一般にプライベートなネットワークのことで、多くの場合 LAN の意味になります。このネットワーク上のホストがプライベートアドレスの範囲 (項「ネットマスクとルーティング」(第13章 ネットワークの基礎, ↑リファレンス) をお読みください) にある場合、ネットワークアドレス変換 (NAT) を有効にし、内部ネットワーク内のホストから外部にアクセスできるようにします。また、内部ゾーンに対しては全てのポートを開きます。このような仕組みにより、ファイアウォール

ルを停止 させずに内部向けのサービスを提供できるほか、新しいインターフェイスを追加 したような場合でも、それらは自動的に外部ゾーンに割り当てられることになる ため、一時的であっても間違ってポートを「開いて」しまうようなトラブルを防ぐことができます。

非武装ゾーン (DMZ)

非武装ゾーンとは、外部ネットワークからも内部ネットワークからもアクセス 可能な領域のことで、非武装ゾーンからは内部にアクセスできないタイプの ものを指します。このような設定は、内部ネットワークよりも制限の緩い 第二の防衛線を作るために使用するもので、DMZ システムは内部ネットワーク とは分離した構成になっています。

フィルタを行なうルールで明示的に許可されていない限り、すべてのネットワークトラフィックは iptables 側で遮断されます。そのため、他のホストから通信を 受け付けるインターフェイスについては、すべてのインターフェイスを上記 3 種類のゾーンのうち、いずれかに設定しなければなりません。また、それぞれの ゾーンでは許可するサービスやプロトコルを設定します。なお、ルールセットは ネットワーク上離れたホストから生成されたものに対してのみ適用され、自分自身が 生成したパケットがファイアウォール機能で捕捉されることはありません。

設定作業は YaST から実施することができる (13.4.1 項「YaST を利用したファイアウォールの設定」(172 ページ) をお読みください) ほか、`/etc/sysconfig/SuSEfirewall2` ファイルをエディタなどで 編集する (詳しくは同ファイル内のコメントをお読みください) ことによっても 設定することができます。それ以外にも、シナリオごとの設定例が `/usr/share/doc/packages/SuSEfirewall2/EXAMPLES` 内に配置されています。

13.4.1 YaST を利用したファイアウォールの設定

重要: 自動ファイアウォール設定

インストールを行なうと、YaST は設定したすべてのインターフェイスに 対してファイアウォール機能を有効にします。YaST でサーバを設定し有効化している場合は、YaST の各サーバ設定モジュール内で *選択した インターフェイスのファイアウォールでポートを開く* または *ファイアウォールでポートを開く* を選択すると、自動で 生成されたファイアウォール設定を変更することもできます。サーバモジュールによっては、*ファイアウォールの詳細* ボタンから、追加のサービスやポートを

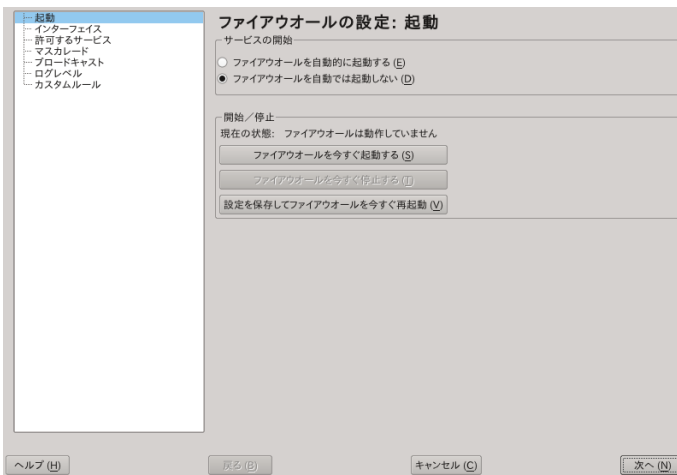
設定することもできます。YaST のファイアウォール モジュールでは、ファイアウォールの有効化や無効化、再設定などを行なうことができます。

グラフィカルな表示で設定を行なうには、YaST のコントロールセンターから モジュールを起動します。セキュリティとユーザ > ファイアウォール を選択してください。設定は全部で 7 つのセクションに分割されていて、それらは左側にあるツリー構造からアクセスすることができます。

起動

このダイアログでは、起動時の処理内容を設定します。既定のインストールでは、SuSEfirewall2 を自動的に起動します。ここから開始や停止を制御することができますほか、新しい設定を反映させたい場合は、設定を保存してファイアウォールを今すぐ再起動を使用することもできます。

図 13.2 YaST ファイアウォール設定



インターフェイス

ここでは、既知のネットワークインターフェイスがすべて表示されます。ゾーンからインターフェイスを取り除くには、インターフェイスを選択して **変更** を押し、どのゾーンにも割り当てないを選択します。ゾーンにインターフェイスを追加するには、インターフェイスを選択してから **変更** を押し、追加したいゾーンを選択します。また **カスタム** を押すと、独自の設定を作成することもできます。

許可するサービス

お使いのシステムから、保護対象のゾーンに対してサービスを提供したい場合、このオプションを利用する必要があります。既定ではシステムは外部ゾーン

からの 保護のみを行なうため、外部ゾーンにつながっているホストからアクセスできるようにするには、ここで明示的に指定する必要があります。まずは **設定対象のゾーン** を選んだあと、一覧からサービスを選択 してください。

マスカレード

マスカレード機能を利用すると、内部のネットワークを外部 (たとえばインターネット) から隠蔽することができるほか、内部ネットワークのホストからは透過的な形で外部にアクセスできるようになります。外部ネットワークから内部ネットワーク への要求はブロックされますが、内部ネットワークから外部ネットワーク に対しては、マスカレードサーバを介して自由に行なうことができるようになります。内部 ネットワークにあるマシンに対して、ある特定のサービスを外部ネットワークに 公開したい場合は、それらのサービスに対して特別な転送ルールを設定します。

ブロードキャスト

このダイアログでは、ブロードキャストと呼ばれる全体通知用の UDP ポートを設定します。それぞれ必要なゾーンに対して、ポート番号やサービスを指定してください。複数のポートやサービスを指定する場合は半角スペースで区切ってください。サービスについて、詳しくは `/etc/services` をお読みください。

受け入れられなかったブロードキャストに対して、ログ記録を行なうことも できます。ただし Windows ホストが存在する環境でこの機能を利用すると、それらのマシンはブロードキャストを利用してお互いを認識する仕組みである ことから、受け入れられなかったパケットが多数生成され、結果として多数の ログ記録が生成されることに注意してください。

IPsec サポート

このダイアログでは、外部ネットワークに対して IPsec を有効にするかどうか を設定します。また、**詳細** では、どのパケットを信頼 するかを設定することができます。

なお、**詳細** 内でも IPsec まわりの設定を行なうことができます。IPsec パケットは暗号化された形式であるため、暗号を解読して からファイアウォールがどのように処理すべきかを設定することができます。**内部ゾーン** を選択した場合は、解読された IPsec の パケットは、外部ゾーンから来たものであっても、内部ゾーンからのものとして信頼されます。このような状況を避けるには、**発信元のネットワークと同じゾーン** を選択します。

ログレベル

ログレベルでは 2 つのルールを設定することができます。それぞれ許可した パケットと許可しなかったパケットです。許可しなかったパケットには、パケットそ

のものを廃棄した場合や、拒否した場合が含まれます。それぞれ に対して、*すべてを記録, 重要なものだけ記録, 何も記録しない* の中から選択します。

カスタムルール

ここでは接続を許可するルールについて、カスタムなルールを作成することができます。条件として設定できる項目としては、発信元ネットワークや プロトコル、宛先ポートや発信元ポートがあります。それぞれ外部、内部 非武装の各ゾーンに対して設定できます。

ファイアウォールの設定を終えたら、*次へ* を押して ダイアログを抜けてください。ファイアウォール設定について、ゾーンごとの 概要が表示されます。ここでは設定内容を確認してください。許可するすべての サービスやポート、プロトコルのほか、すべてのカスタムルールが概要に表示 されます。設定を修正するには *戻る* ボタンを、設定を 完了するには *完了* を押してください。

13.4.2 手作業での設定

下記の章では、正しい設定を行なうための手順を段階ごとに説明しています。それぞれの設定項目には、ファイアウォール関連の設定なのか、それとも マスカレード関連の設定なのかを示すマークが付けられています。また、ポート の範囲 (たとえば 500:510) については、それぞれの環境に あわせて適切に読み替えてください。また、DMZ (非武装ゾーン) 関連の設定は ここでは説明していません。このゾーンは、たとえば大規模なネットワーク (たとえば企業用のネットワーク) など、より複雑なネットワークインフラを 設定したい場合にのみ使用するもので、より難しい設定を必要とするほか、ネットワークに関してより深い知識を必要とするものです。

手作業での設定を行なうにあたって、まずは YaST の システムサービス (ランレベル) モジュール を利用して、SuSEfirewall2 のサービスをお使いのランレベル (一般的には 3 か 5 のはずです) で有効に設定してください。これにより、`/etc/init.d/rc?.d/` ディレクトリ内に `SuSEfirewall2_*` スクリプトに対するシンボリックリンクが作成されます。

FW_DEV_EXT (ファイアウォール, マスカレードの両方で要設定)

インターネットに接続されているデバイスを指定します。モデムで接続している場合は `ppp0` を、ISDN (YaST の ISDN モジュールで設定している場合のみ。日本国内の場合、一般的には モデム接続と同じ設定を使用します) で接続している場合は `ipp0` を、DSL 接続を行なっている 場合は `dsl0` をそれぞれ指定します。なお `auto` を指定すると、デフォルトルートに設定されている デバイスを使用します。

FW_DEV_INT (ファイアウォール, マスカレードの両方で要設定)

内部のプライベートネットワークに接続されているデバイス (たとえば eth0 など) を指定します。内部ネットワークが存在せず、ホスト単体を保護すれば十分である場合は、何も指定しなくてもかまいません。

FW_ROUTE (ファイアウォール, マスカレードの両方で要設定)

マスカレード機能を必要とする場合は、この設定には yes に設定します。内部のホストはプライベートアドレス (たとえば 192.168.x.x など) であることになるので、インターネットルータで無視されるため、特別な設定を行なわない限り、外部からアクセスすることはできなくなります。

マスカレード機能を使用しない場合で、内部ネットワークに対してアクセスを許可したい場合にも、yes を設定します。この場合は、内部のホストについても、公式に登録済みの IP アドレス (つまりプライベートアドレスではない IP アドレス) である必要があります。通常は外部から内部のネットワークにアクセスさせるべきではありません。

FW_MASQUERADE (マスカレードで要設定)

マスカレード機能を必要とする場合は、ここに yes を設定します。これにより、内部のホストから (アドレス変換を行なうことで仮想的に) 外部のインターネットにアクセスできるようになります。なお、内部のネットワークとインターネットをつなぐ場合、プロキシサーバを使用するとより機密を保持する環境を構築することができます。プロキシサーバがインターネットアクセスの機能を提供する場合、マスカレード機能は不要です。

FW_MASQ_NETS (マスカレードで要設定)

マスカレード機能を適用するホストやネットワークを指定します。複数の項目を指定する場合は、それぞれ半角のスペースで区切ります。たとえば下記のようになります:

```
FW_MASQ_NETS="192.168.0.0/24 192.168.10.1"
```

FW_PROTECT_FROM_INT (ファイアウォールで要設定)

内部ネットワークを発信元とする攻撃について、このファイアウォールで保護を行なう必要がある場合、この値を yes に設定します。yes に設定した場合は、内部ネットワークに公開するサービスを明示的に指定する必要があります。それぞれ FW_SERVICES_INT_TCP や FW_SERVICES_INT_UDP をお読みください。

FW_SERVICES_EXT_TCP (ファイアウォールで要設定)

外部ネットワークに対して公開する TCP ポートを指定します。いかなるサービスも提供しない通常のワークステーションの場合、ここには何も指定しないでください。

FW_SERVICES_EXT_UDP (ファイアウォールで要設定)

外部ネットワークに対して UDP のサービスを実行し、公開したい場合にのみ設定します。それ以外の場合は何も指定しないでください。なお、UDP のサービスとしては DNS サーバや IPsec, TFTP, DHCP などのサービスがあります。これらを 公開したい場合は、使用する UDP ポートを指定します。

FW_SERVICES_ACCEPT_EXT (ファイアウォールで要設定)

インターネットに対して公開するサービスの一覧を指定します。これは FW_SERVICES_EXT_TCP や FW_SERVICES_EXT_UDP での設定をより汎用的にした設定 項目で、FW_TRUSTED_NETS での設定よりはより詳しい 設定になっています。半角スペースで区切るタイプの一覧を指定し、それぞれの項目は ネットワーク, プロトコル [, 宛先ポート] [, 発信元ポート] のルールで記述します。たとえば 0/0, tcp, 22 や 0/0, tcp, 22, , hitcount=3, blockseconds=60, recentname=ssh のように指定します。後者は、1 つのアドレスから 1 分あたり 3 回までの接続を 許可する、というルールになります。

FW_SERVICES_INT_TCP (ファイアウォールで要設定)

この設定では、内部ネットワークに対して公開するサービスを指定します。記述方法は FW_SERVICES_EXT_TCP と同じですが、設定は 内部の ネットワークに対して適用されます。なお、この設定は FW_PROTECT_FROM_INT を yes に設定した場合のみ必要な項目です。

FW_SERVICES_INT_UDP (ファイアウォールで要設定)

FW_SERVICES_INT_TCP をお読みください。

FW_SERVICES_ACCEPT_INT (ファイアウォールで要設定)

内部のホストに対して公開するサービスの一覧を設定します。詳しくは FW_SERVICES_ACCEPT_EXT をお読みください。

FW_SERVICES_ACCEPT_RELATED_* (ファイアウォールで要設定)

これは netfilter における RELATED (関連パケット) に関する SuSEfirewall2 の設定項目です。

たとえば Samba のブロードキャストパケットに対して、より細かい許可を 設定する場合、従来は 関連する パケットを無条件に 受け入れることはできませんでした。FW_SERVICES_ACCEPT_RELATED_ で始まる設定を 使用すると、特定のネットワークやプロトコルに対して、関連する パケットを限定的に許可できるようになります。

これは接続追跡用 (コネクショントラッキング; conntrack) のモジュール を FW_LOAD_MODULES 内に設定することで実現 できるもので、これらのモジュール

がパケットの関連性を判断し、自動的に 受け入れるかどうかを指定することができます。また、FW_SERVICES_ACCEPT_RELATED_ で始まる設定も 適切に設定しなければなりません。

FW_CUSTOMRULES (ファイアウォールで要設定)

カスタムルールを設定する場合は、この行のコメントを外してください。設定例は /etc/sysconfig/scripts/SuSEfirewall2-custom にあります。

ファイアウォールを設定したあとは設定を確認します。root で rcSuSEfirewall2 start と入力すると、ファイアウォールのルールが作成されます。telnet などのツールを利用し、本当に外部のホストから接続ができなくなっているか どうかを確認してください。また、接続を拒否した場合は、下記のようなログが /var/log/messages 内に記録されているはずです:

```
Mar 15 13:21:38 linux kernel: SFW2-INext-DROP-DEFLT IN=eth0
OUT= MAC=00:80:c8:94:c3:e7:00:a0:c9:4d:27:56:08:00 SRC=192.168.10.0
DST=192.168.10.1 LEN=60 TOS=0x10 PREC=0x00 TTL=64 ID=15330 DF PROTO=TCP
SPT=48091 DPT=23 WINDOW=5840 RES=0x00 SYN URG=0
OPT (020405B40402080A061AFEB0000000001030300)
```

ファイアウォールをテストする場合は、nmap (ポートスキャナ) や OpenVAS (Open Vulnerability Assessment System; オープン脆弱性調査システム) も利用できます。nmap のドキュメンテーションは、nmap のパッケージをインストールしたあと、/usr/share/doc/packages/nmap 内にあります。また OpenVAS のドキュメンテーションは、<http://www.openvas.org> をお読みください。

13.5 さらなる情報

SuSEfirewall2 パッケージに関する最新情報や その他のドキュメンテーションについては、/usr/share/doc/packages/SuSEfirewall2 をお読みください。それ以外にも、netfilter と iptables のプロジェクトの Web ページ <http://www.netfilter.org> には、多数の言語で数多くの ドキュメンテーションが提供されています。

VPN サーバの設定

現在ではインターネット接続は安価なものとなり、ほとんどどこからでも利用できるようになりました。そして接続に対してできる限りの機密を求めるため、仮想 プライベートネットワーク (VPN) という仕組みを、インターネットや無線 LAN のような機密の保持が難しいネットワークに対して利用するようにもなってきました。VPN は様々な方法で実装／提供されているものであることから、本章では VPN を利用して機密を保持するワイドエリアネットワークを構築し、企業のオフィスに 接続する場合を想定して説明しています。

14.1 考え方の概要

本章では VPN に関する各種の用語の定義と、いくつかのシナリオに沿った 大まかな概要について説明を行なっています。

14.1.1 用語

エンドポイント

トンネルの「両端」を指す言葉です。それぞれ発信元と 送信先の両方を意味します。

tap デバイス

tap デバイスはイーサネットデバイス (OSI 参照モデルにおけるレイヤ 2) を擬似します。tap デバイスは、ネットワークブリッジを構成する場合に 使用し、イーサネットのフレームを取り扱います。

tun デバイス

tun デバイスは一對一のネットワーク (OSI モデルにおけるレイヤ 3) を擬似します。tun デバイスはルーティングを扱う場合に利用するもので、IP パケットを取り扱います。

トンネル

一般に公衆ネットワークを介した、2 拠点 (端末) の接続を意味する用語です。より技術的な観点で言うと、トンネルはクライアント側のデバイスと サーバ側のデバイスの接続を意味します。通常トンネルは暗号化されていますが、必須ではありません。

14.1.2 VPN のシナリオ

VPN 接続を設定するには、機密を保持したトンネルを用意し、その中に IP パケットを流します。トンネルは *tun* または *tap* デバイスと呼ばれるものを利用します。これらはイーサネットフレームや IP パケットを流すことのできる機能を 実装した、仮想的なネットワークカーネルドライバです。

OpenVPN に対応した任意のユーザスペース (カーネル内蔵のものではない別個の) プログラムでは、tun デバイスや tap デバイ스에接続することで、OS からのパケットを受け取ることができます。これらのプログラムはデバイスにパケットを 書き込むこともできます。

VPN の接続設定と環境構築には、多数の解決方法があります。本章では OpenVPN パッケージを利用して解決する場合を想定します。他の VPN ソフトウェアと比較すると、OpenVPN は 2 種類のモードで動作させることができるようになっていきます:

ルーティング型 VPN

ルーティングはもっとも設定が簡単な方法です。ブリッジ型と比較すると、効率と性能の両面で効率的になります。さらにユーザに対して MTU (Maximum Transfer Unit; 最大転送単位) の調整を許すことになるので、より効率を上げる余地を与えることになります。しかしながら、ゲートウェイ上に Samba サーバが存在しない場合、ルーティング型では NetBIOS のブロードキャストパケットを通すことができません。また、IPv6 を必要とする環境では、VPN サーバとクライアントの両方の tun ドライバで IPv6 に対応する必要があります。詳しくは 図14.1「ルーティング型 VPN」(180 ページ) をご覧ください。

図 14.1 ルーティング型 VPN

ブリッジ型 VPN

ブリッジ型はより複雑な設定方法です。Samba や WINS サーバを利用しないネットワークで、VPN を介して Windows のファイル共有を参照する必要がある場合、ブリッジ型を選択する必要があります。また、ブリッジ型 VPN は IP 以外のプロトコル (たとえば IPX) を利用する場合や、ネットワーク ブロードキャストを必要とするアプリケーションを利用するような場合にも 選択する必要があります。しかしながら、ルーティング型 VPN に比べると 効率が悪くなってしまうほか、大規模環境にも不向きです。詳しくは 下記をご覧ください。

図 14.2 ブリッジ型 VPN - シナリオ 1

図 14.3 ブリッジ型 VPN - シナリオ 2

図 14.4 ブリッジ型 VPN - シナリオ 3

また、ブリッジ型とルーティング型の大きな違いとして、ルーティング型 VPN では IP ブロードキャストを扱うことができないのにたいし、ブリッジ型 VPN ではそれを行なうことができるという点もあります。

14.2 最も簡単な VPN の作成例

下記の例では、一対一 (Point-To-Point) 接続の VPN トンネルを作成しています。この実例を元にして、1 台のクライアントと 1 台のサーバから構成される VPN トンネルの作成方法を学習してください。なお、VPN サーバとクライアントはいずれもプライベート IP アドレスの環境で、それぞれサーバ側が 192.168.1.120、クライアント側が 192.168.2.110 となっている場合を 想定しています。お使いの環境で上記のアドレスをそのまま利用することも できますが、あらかじめ他の IP アドレスと矛盾がないことを確認しておいてください。

警告: テスト用途にのみお使いください

この設定例は VPN について学習するためのものとして、テスト用に用意されているものです。下記の設定をそのまま使用してしまうと、お使いのマシンの 安全を脅かすことになるばかりか、ネットワーク全体に危険をもたらす 可能性があり得るため、下記の設定をそのまま使用することは おやめください。

14.2.1 VPN サーバの設定

To configure a VPN server, proceed as follows:

手順 14.1 VPN サーバ設定

- 1 まずは VPN サーバとして利用したいマシンに対して、`openvpn` パッケージをインストールします。

- 2 シェルを開いて `root` になり、下記のように入力して VPN の 機密鍵を作成します:

```
openvpn --genkey --secret /etc/openvpn/secret.key
```

- 3 機密鍵をクライアント側にコピーします:

```
scp /etc/openvpn/secret.key root@192.168.2.110:/etc/openvpn/
```

- 4 `/etc/openvpn/server.conf` ファイルを作成し、下記のように入力します:

```
dev tun
ifconfig 192.168.1.120 192.168.2.110
secret secret.key
```

- 5 ファイアウォールをお使いの場合は、YaST を起動してから *セキュリティとユーザ* > *ファイアウォール* > *許可するサービス* を選び、UDP ポート 1194 を開きます。

- 6 `root` から OpenVPN サービスを起動します:

```
rcopenvpn start
```

14.2.2 VPN クライアントの設定

VPN クライアントを設定するには、下記のようにして行ないます:

手順 14.2 VPN クライアント設定

- 1 VPN クライアントマシンに対して、`openvpn` パッケージをインストールします。

- 2 下記のような内容で `/etc/openvpn/client.conf` ファイルを作成します:

```
remote サーバの IP アドレス
dev tun
ifconfig 192.168.2.110 192.168.1.120
secret secret.key
```

サーバの IP アドレス の欄には、VPN サーバのドメイン名 (FQDN) または IP アドレスを入力します。

- 3 ファイアウォールをお使いの場合は、手順14.1「VPN サーバ設定」(182 ページ) 内の ステップ 5 (182 ページ) の説明に従い、YaST を起動して UDP ポート 1194 を開きます。

- 4 root から OpenVPN サービスを起動します:

```
rcopenvpn start
```

14.2.3 VPN 設定のテスト

OpenVPN を正しく起動することができたら、tun デバイスが利用可能かどうか を下記のコマンドで確認します:

```
ifconfig tun0
```

さらに VPN の接続を確認するには、クライアントとサーバの両方で ping コマンドを利用し、お互いにパケットが届くかどうかを確認します。クライアントからサーバに対してテストを行なうには、下記の ように実行します:

```
ping -I tun0 192.168.1.120
```

逆にサーバからクライアントに対してテストを行なう場合は、下記のように 実行します:

```
ping -I tun0 192.168.2.110
```

14.3 証明機関を利用する VPN サーバの設定

14.2項 (181 ページ) で示した設定例は、単純にテストを行なう目的であれば簡単に実行できるものではありませんが、実際の環境で使用するにはセキュリティ上の理由から不適切です。この章では、同時に複数の接続を受け付ける VPN サーバの構築方法を説明しています。これは公開鍵インフラストラクチャ (PKI) と呼ばれる仕組みを利用して行なうもので、それぞれサーバとクライアントに対して、公開鍵と機密鍵からなる 鍵対を配置します。さらに、マスター証明機関 (CA) と呼ばれる仕組みを用意し、サーバやクライアントの証明書を検証させるようにするものです。

設定手順を大まかに分けると、下記ようになります:

- 1 14.3.1項「証明書の作成」(184 ページ)
- 2 14.3.2項「サーバの設定」(186 ページ)
- 3 14.3.3項「クライアントの設定」(188 ページ)

14.3.1 証明書の作成

VPN 接続が確立する前の段階で、クライアントはサーバに対して認証を行わなければなりません。また逆に、サーバもクライアントに対して認証を行わなければなりません。これは *相互認証* と呼びます。

それぞれ必要な証明書と鍵を作成するには、下記の 2 つの方法のうちのいずれかを実行します：

- YaST の証明機関モジュール (詳しくは 第15章 *X.509 証明書の管理* (195 ページ) をお読みください) を使用する
- `openvpn` パッケージに含まれる スクリプトを使用する

14.3.1.1 easy-rsa を利用した証明書の作成

`easy-rsa` ユーティリティは `/usr/share/openvpn/easy-rsa/バージョン` ディレクトリ内にある `openssl.cnf` ファイルを使用します。ほとんどの場合、このファイルをそのまま使用するだけで設定が完了します。

手順 14.3 マスター証明機関と鍵の作成

- 1 シェルを開いて `root` に移行します。
- 2 `/usr/share/openvpn/easy-rsa/バージョン/` ディレクトリに移動します。ここで、バージョンには `1.0` や `2.0` など、現在のバージョンが入ります。
- 3 `vars` ファイルを `/etc/openvpn` ディレクトリにコピーし、`/usr/share/openvpn/easy-rsa` 内で `export EASY_RSA` を設定します：
`export EASY_RSA="/usr/share/openvpn/easy-rsa/VER"`
- 4 `vars` ファイル内で、それぞれ `KEY_COUNTRY`, `KEY_PROVINCE`, `KEY_CITY`, `KEY_ORG`, and `KEY_EMAIL` の値を必要に応じて修正します。
- 5 PKI (公開鍵インフラストラクチャ) を初期化します：

```
source /etc/openssl/vars && ./clean-all && ./build-ca
```

- 6 build-ca スクリプトで要求されたデータをそれぞれ 入力します。通常は ステップ 4 (184 ページ) で設定した 既定値を利用することができます。追加で、左記では設定していなかった Organizational Unit Name (部署名) と Common Name (汎用名) を入力することもできます。

完了すると、マスター証明書と鍵が /usr/share/openssl/easy-rsa/バージョン/keys/ca.* に保存されます。

手順 14.4 プライベートサーバ鍵の作成

- 1 /usr/share/openssl/easy-rsa/バージョン/ ディレクトリに移動します。

- 2 下記のスクリプトを実行します:

```
./build-key-server server
```

パラメータ (ここでは サーバ) には、機密鍵の ファイル名を指定します。

- 3 パラメータは基本的に既定値をそのまま受け入れてかまいませんが、Common Name (汎用名) の項目だけは サーバのホスト名 (FQDN) または IP アドレスを指定してください。
- 4 続く 2 つの質問 ((「Sign the certificate? [y/n]」(証明書に署名しますか?) と 「1 out of 1 certificate requests certified, commit? [y/n]」(1 つのうち 1 つの証明書要求に署名 しました。確定してよろしいですか?)) に対しては、いずれも y (はい) を入力します。

作業を完了すると、サーバの機密鍵は /usr/share/openssl/easy-rsa/バージョン/keys/server.* に保存されます。

手順 14.5 クライアントに対する証明書と鍵の作成

- 1 /usr/share/openssl/easy-rsa/バージョン/ ディレクトリに移動します。ここで、バージョン には 1.0 や 2.0 など、現在のバージョンが 入ります。

- 2 手順 14.4「プライベートサーバ鍵の作成」(185 ページ) の ステップ 2 (185 ページ) にある手順に従い、鍵を作成します:

```
./build-key client
```

- 3 上記の手順を、VPN サーバへの接続を許可するクライアントの数だけ繰り返します。なお、それぞれ実行するごとに「client」と 問い合わせの途中で表示される Common Name の欄には、クライアントごとに異なる名前を指定してください。

作業を完了すると、クライアントの証明書と鍵は `/usr/share/opensvpn/easy-rsa/keys/client.*` に保存されます (`build-key` コマンドに指定した名前で 作成されます)。

手順 14.6 最後の設定作業

1 まずはカレントディレクトリが `/usr/share/opensvpn/easy-rsa/バージョン/` であることを確認します。

2 Diffie-Hellman パラメータを作成します:

```
./build-dh
```

3 `/etc/opensvpn/ssl` ディレクトリを作成します。

4 下記のファイルを `/etc/opensvpn/ssl` 内に コピーします:

```
cp keys/ca.{crt,key} keys/dh1024.pem keys/server.{crt,key} /etc/opensvpn/ssl
```

5 作成しておいたクライアントの鍵を、クライアントにコピーします。具体的には、各クライアントの `/etc/opensvpn/ssl` ディレクトリ内に、`client.crt` と `client.key` ファイルを配置します。

14.3.1.2 YaST CA モジュールを利用した証明書の設定について

この時点で `easy-rsa` ユーティリティを利用して証明書を作成してある場合は、本章を読み飛ばしてかまいません。

14.3.2 サーバの設定

設定ファイルは、`/usr/share/doc/packages/opensvpn/sample-config-files/server.conf` にあるサンプルを雛形にして作成することができます。なお、パス設定なども一部修正する 必要がある場合があります。

例 14.1 VPN サーバの設定ファイル

```
# /etc/opensvpn/server.conf
port 1194 ❶
proto udp ❷
```



```

dev tun0 ❸

# セキュリティ ❹
ca /etc/openvpn/ssl/ca.crt
cert /etc/openvpn/ssl/server.crt
key /etc/openvpn/ssl/server.key
dh /etc/openvpn/ssl/dh1024.pem

server 192.168.1.120 255.255.255.0 ❺
ifconfig-pool-persist /var/run/openvpn/ipp.txt ❻

# 権限 ❼
user nobody
group nobody

# その他の設定 ❽
keepalive 10 120
comp-lzo
persist-key
persist-tun
status /var/log/openvpn-status.log
log-append /var/log/openvpn.log
verb 4

```

- ❶ OpenVPN が待ち受ける TCP/UDP ポートを指定します。ファイアウォールをお使いの場合は、ファイアウォールの設定でポートを開く必要があります (第 13 章 マスカレードとファイアウォール (167 ページ) をお読みください)。なお、VPN で使用する標準のポートは 1194 で、特に要件がないかぎり、そのままの設定で かまいません。
- ❷ プロトコルを指定します。UDP または TDP を指定します。
- ❸ tun または tap デバイスを指定します。違いについては 14.1.1 項「用語」(179 ページ) をお読みください。
- ❹ この行以後には、それぞれルート CA (証明機関) の証明書 (ca) とルート CA の鍵 (cert)、プライベートサーバ鍵 (key) と Diffie-Hellman パラメータ (dh) を指定しています。これらは 14.3.1 項「証明書の作成」(184 ページ) の手順で作成してあるものです。
- ❺ VPN のサブネットを指定します。このとき、サーバは 192.168.1.120 のアドレスにあるものとします。
- ❻ クライアントと割り当てた IP アドレスの一覧を、指定したファイルに保存する設定です。サーバがいったん再起動するような状況で、再起動後もクライアントが 同じアドレスで通信できるようにしたい場合に便利です。
- ❼ セキュリティ保持の目的から、OpenVPN は制限された権限下で動作させておく ことをお勧めします。この例では nobody という ユーザ/グループを指定しています。
- ❽ その他のいくつかの設定です。詳しくは設定ファイルの雛形 /usr/share/doc/packages/openvpn/sample-config-files にあるコメントをお読みください。

設定を完了したら、OpenVPN サーバのログメッセージが `/var/log/openvpn.log` 内に記録されるようになります。初回に起動した場合は、下記のような出力が現われているはずです：

```
... Initialization Sequence Completed
```

もしも上記のようなメッセージが表示されていない場合は、まずログをよくお読みください。通常は OpenVPN 側で、設定ファイルのどこに問題があるのかヒントを表示しているはずです。

14.3.3 クライアントの設定

設定ファイルは、`/usr/share/doc/packages/openvpn/sample-config-files/client.conf` にあるサンプルを雛形にして作成することができます。なお、パス設定なども一部修正する 必要がある場合があります。

例 14.2 VPN クライアントの設定ファイル

```
# /etc/openvpn/client.conf
client ❶
dev tun ❷
proto udp ❸
remote IP アドレスまたはホスト名 1194 ❹
resolv-retry infinite
nobind

# 権限 ❺
user nobody
group nobody

# 再起動しても同じ状態を維持しようとするための設定
persist-key
persist-tun

# セキュリティ ❻
ca /etc/openvpn/ssl/ca.crt
cert /etc/openvpn/ssl/client.crt
key /etc/openvpn/ssl/client.key

comp-lzo ❼
```

- ❶ このマシンをクライアントとして動作させる指定です。
- ❷ ネットワークデバイスを指定します。クライアントとサーバの両方で、同じ デバイスを使用しなければなりません。
- ❸ プロトコルを指定します。サーバと同じ設定を使用します。
- ❹ まず *IP アドレスまたはホスト名* の項目は、お使いの VPN サーバのホスト名または IP アドレスを指定します。これに 続いて、ポート番号を指定します。

複数の `remote` 行を設定して、複数の VPN サーバを使用するように指定することもできます。この場合は、複数の VPN サーバ間で負荷分散を行なう場合に有効です。

- ⑤ セキュリティ保持の目的から、OpenVPN は制限された権限下で動作させておくことをお勧めします。この例では `nobody` というユーザ／グループを指定しています。
- ⑥ クライアント側の証明書類を指定しています。セキュリティ保持の目的から、各クライアントに対して共通の証明書を使用したりはせず、別々のファイル にしておくことをお勧めします。
- ⑦ 圧縮機能を有効にする設定です。サーバ側でも同様に圧縮機能を有効にしている場合にのみ有効です。

14.4 VPN 内でのネームサーバの変更

VPN セッションの前または途中でネームサーバを変更する必要がある場合は、`netconfig` を使用して行ないます。

ネームサーバを変更するには、下記の手順を実行します：

手順 14.7 ネームサーバの変更

1 シェルを開いて `root` に移行します。

2 `/etc/openvpn/client.up` ファイルを作成し、下記の 内容を入力します：

```
/sbin/netconfig modify -i "${1}" -s openvpn <<EOT
DNSSEARCH='${domain}'
DNSSERVERS='${dns[*]}'
EOT
```

3 `rcopenvpn start` を実行し、VPN 接続を開始します。

4 `/etc/openvpn/client.down` ファイルを作成し、下記の 内容を入力します：

```
/sbin/netconfig remove -i "${1}" -s openvpn
```

5 `netconfig` コマンドを実行します。なお、`DNSSERVERS` の行には設定したいネームサーバを指定します：

```
netconfig modify -i tun0 -s openvpn <<EOT
DNSSEARCH='mt-home.net'
DNSSERVERS='192.168.1.116'
EOT
```

正しく設定されたかどうかを確認するには、`/etc/resolv.conf` ファイル内に設定したはずの項目が挿入されているかどうかを確認します。下記のように実行してください:

```
grep -v ^# /etc/resolv.conf
search mt-home.net mat-home.net
nameserver ...
nameserver ...
nameserver 192.168.1.116
```

6 DNS の設定を削除するには、下記のように実行します:

```
netconfig remove -i tun0 -s openvpn
```

上記以外の設定例については、`/usr/share/doc/packages/openvpn/contrib/pull-resolv-conf/` をお読みください。

フォールバック (あるものが使用できない場合の代替) サービスの順位リストを設定する必要がある場合は、`/etc/sysconfig/network/config` 内の `NETCONFIG_DNS_RANKING` 変数を使用します。既定値は `auto` で、下記の設定になっています:

```
+strongswan +openswan +racoon +openvpn -avahi
```

優先するサービス名に対しては頭に `+` を、フォールバック サービスに対しては頭に `-` を付与します。

14.5 クライアント向けの KDE および GNOME アプレット

下記の章では、GNOME や KDE のデスクトップツールを利用して OpenVPN の接続を設定する方法を説明しています。

14.5.1 KDE

KDE4 で OpenVPN の設定を行ったり、有効と無効の切り替えを簡単に行ったりするには、下記の手順を実施します:

- 1 まずは `NetworkManager-openvpn-kde4` パッケージと、このパッケージが必要とするパッケージをすべてインストールしていることを確認します。

- 2 パネル内のウィジェットでマウスの右ボタンを押し、*パネルのオプション > ウィジェットの追加...* を選択します。
- 3 ネットワーク を選択します。
- 4 アイコンの上でマウスの右ボタンを押し、*接続の管理* を選択します。
- 5 *追加 > OpenVPN* を選択し、新しい VPN 接続を作成します。新しいウィンドウが開きます。
- 6 *接続タイプ* では *X.509 証明書* または *パスワード付き X.509 証明書* を選択します。これはお使いの VPN サーバの設定に従って選択してください。
- 7 あとはそれぞれ関連するテキスト項目に必要なファイルを挿入します。上述の例では、設定は下記ようになります:

CA ファイル	/etc/openvpn/ssl/ca.crt
証明書	/etc/openvpn/ssl/client1.crt
鍵	/etc/openvpn/ssl/client1.key
ユーザ名	ユーザ名
パスワード	ユーザに対するパスワード

- 8 KDE ウォレットシステムを初めてお使いになる場合は、ここで設定を行なうかどうかを尋ねられます。ウィザードの示す手順に従って作業を行なってください。設定手順が終わったら、*ネットワーク設定* ダイアログに戻ってきます。
- 9 *Ok* を押して完了します。
- 10 あとは Network Manager のアプレットを利用することで、接続を行なうことができるようになります。

14.5.2 GNOME

GNOME で OpenVPN の設定を行なったり、有効と無効の切り替えを簡単に行なったりするには、下記の手順を実施します:

- 1 まずは NetworkManager-openvpn-gnome パッケージと、このパッケージが必要とするパッケージをすべて インストールしていることを確認します。
- 2 Alt + F2 を押してから、表示されたテキスト枠に nm-connection-editor と入力し、ネットワーク接続エディタを起動します。新しいウィンドウが 表示されます。
- 3 VPN タブを選択して Add を 押します。
- 4 まずは VPN の接続タイプを選択します。この場合は OpenVPN を選択します。
- 5 Authentication の種類では、Certificates (TLS) または Password with Certificates (TLS) を選択します。これはお使いの VPN サーバの設定に従って選択してください。server.
- 6 あとはそれぞれ関連するテキスト項目に必要なファイルを挿入します。上述の例では、設定は下記ようになります：

Username	ユーザ名 (Password with Certificates (TLS) を 選択した場合のみ)
Password	ユーザに対するパスワード (Password with Certificates (TLS) を選択した場合のみ)
User Certificate	/etc/openvpn/ssl/client1.crt
CA Certificate	/etc/openvpn/ssl/ca.crt
Private Key	/etc/openvpn/ssl/client1.key

- 7 最後に Apply と Close を 押せば完了です。
- 8 あとは NetworkManager のアプレットを利用することで、接続を行なう ことができるようになります。

14.6 さらなる情報

VPN について、さらに詳しい情報は下記をお読みください：

- <http://www.openvpn.net>: VPN の Web ページ
- `/usr/share/doc/packages/openvpn/sample-config-files/`: 様々なシナリオに対応する設定例
- `/usr/src/linux/Documentation/networking/tuntap.txt`: kernel-source パッケージをインストールすると閲覧できるドキュメント (英語)

X.509 証明書の管理

大多数の認証の仕組みでは、暗号化の手順をベースにした作りになっています。たとえばデジタル証明書では、所有者に対して暗号鍵を割り当てる仕組みで、暗号化はデジタル証明書の仕組みの中で重要な役割になっています。これらの証明書は通信で利用されているほか、企業の ID カードなどでも利用されています。証明書の生成と管理は、多くの場合商用サービスを提供する公式の機関によって処理されていますが、必要であれば自身でその作業を行なうことも可能です。たとえば、企業が個人情報を第三者に渡したくない場合などが、それに当てはまります。

YaST には、デジタル X.509 証明書を管理する作業のうち、基礎的な機能を提供する 2 種類のモジュールがあります。下記の章では、デジタル証明書の基礎と、デジタル証明書の作成や管理に関する YaST の使用方法のそれぞれを説明しています。

15.1 デジタル証明書の仕組み

デジタル証明書では、暗号化処理を利用して暗号化を行ない、権限のないユーザに対するデータへのアクセスから保護します。また、ユーザデータは 2 番目のデータレコード (鍵と呼びます) を利用して暗号化を行ないます。鍵はユーザデータに対して数学的な処理を行なうもので、変換されたデータは (鍵無しでは) 解読できなくなります。今では、一般的にはこのような非対称の暗号化方法 (公開鍵手順と呼びます) を利用します。鍵は必ず 1 対のものとして存在しています：

機密鍵

機密鍵は、鍵の所有者が安全な場所に保存しておかなければならないものです。機密鍵を誤って公開してしまうと鍵対による暗号化の安全性が保てなくなり、機密鍵を持っていれば誰でもデータを解読できてしまいます。

公開鍵

公開鍵は第三者が利用できるよう、鍵の所有者が公開して配布するものです。

15.1.1 鍵の正当性

公開鍵の作業は広く用いられている仕組みであるため、多数の公開鍵が流通しているのが現状です。このシステムをうまく動かすには、それぞれのユーザが互いの間違いのない公開鍵を知っておく必要があることから、信頼できる機関が各ユーザの公開鍵を検証し、その検証結果を機関の公開鍵付きの証明書で示す必要があります。このような証明書には公開鍵のほか、証明書を発行した機関のデジタル 証明書が含まれます。

公開鍵を含む証明書に対して、公開や署名を行なう信頼機関は、証明書の発行や取り消し、更新などの証明書管理に対しても責任を持つ機関で、証明書の仕組みにおける重要な役割を担っています。このような仕組みを *公開鍵 インフラストラクチャ* や *PKI* などと呼びます。PKI として知られているものの 1 つに、*OpenPGP* というものがあります。これは中央の認証ポイントを用意することなく、証明書を自由に発行できる仕組みです。これらの証明書は、「既に信頼済みの」他者が署名することによって信頼を受けることができます。

X.509 公開鍵インフラストラクチャ (PKIX) は *IETF* (Internet Engineering Task Force) が定義するもう 1 つのモデルで、現在ではほとんどすべての公開鍵ベースの PKI のモデルとして 使用されています。このモデルでは、認証は *証明機関 (CA)* と呼ばれる機関が階層構造の形で実施します。もっとも根幹にある CA をルート CA と呼び、すべての子 CA (下位 CA) の正当性を証明します。もっとも階層の低い (木構造で言うところの末端) の CA が、ユーザに対する証明書を発行します。ユーザの証明書はルート CA から順に証明構造をたどることによって、正当性を 検証することができるようになっていきます。

このように、PKI の仕組みは CA の証明書の信頼性に依存した作りになっています。PKI の利用者に対して証明書の発行基準を明らかにするため、PKI の運用者は *認証局運用規定 (CPS)* を規定し、証明書の管理手順を 表明しておくのがよいでしょう。これにより、PKI が信頼性のある証明書を 発行できるかどうかを確認することができます。

15.1.2 X.509 証明書

X.509 証明書は複数のフィールドから構成されるデータ構造で、任意の拡張を含むこともできるようになっています。固定のフィールドには主に鍵の所有者に関する情報や公開鍵、発行元の CA に関する情報 (名前と署名) などがあります。セキュリティ上の理由から、証明書は有効期限が限定されているべきもので、有効期限を示すフィールドも証明書内に存在しています。CA は、この有効期限が切れるまでの間、証明書の正当性を保証することになります。また、CPS についても、有効期限が切れるまでに新しい証明書を作成して配布する必要があることから、PKI (発行元 CA) で必要になります。

拡張フィールドには任意の追加情報を含めることができます。その項目が *critical* (重要) であるものと記されている場合にのみ、項目を解釈するためのアプリケーションが必要となります。アプリケーションが重要項目を解釈できない場合、証明書を拒否しなければなりません。また、拡張によっては、署名や暗号化など、特定のアプリケーションに有用な項目です。

表 15.1 では、バージョン 3 の X.509 証明書で使用される基本的な項目について説明を記しています。

表 15.1 X.509v3 証明書

項目	内容
バージョン	証明書のバージョン。例: v3
シリアル番号	ユニークな証明書 ID (整数)
署名	証明書に署名する際に使用したアルゴリズムの ID
発行者	発行した機関 (CA) のユニーク名 (DN)
有効期限	証明書が有効である範囲
サブジェクト	所有者のユニーク名 (DN)
サブジェクト公開鍵情報	所有者の公開鍵と、アルゴリズムの ID

項目	内容
発行者ユニーク ID	発行元 CA のユニーク ID (オプション)
サブジェクトユニーク ID	所有者のユニーク ID (オプション)
拡張	任意の拡張情報。たとえば「鍵の使用法」や「基本制限」など。

15.1.3 X.509 証明書の無効化

証明書が有効期限切れよりも前に信頼のできない状態になってしまった場合、それらの証明書は即時に無効化されなければなりません。このような措置は、たとえば機密鍵が誤って漏洩してしまったような場合があげられます。証明書の無効化は、特に機密鍵がユーザの証明書ではなく CA に属するものであるような場合に重要です。この場合、その CA で発行されたすべてのユーザ証明書は即時に無効化されなければなりません。証明書が無効化されると PKI (対象の CA) は、その無効化情報を *証明書失効リスト* (CRL) を利用して配布しなければなりません。

これらの一覧は、CA が一定の間隔で提供するもので、公式の CRL 配布ポイント (CDP) と呼ばれる箇所に配置されます。CDP には任意で証明書内の拡張を命名することもできるため、確認を行ないたい利用者は検証目的で現在の CRL にアクセスすることができます。これを *オンライン証明書状態プロトコル* (OCSP) と呼びます。なお CRL の正当性は、CA 自身が発行する署名によって保証をおこなうことになります。表 15.2 には、X.509 CRL で使用する基本的な項目を示しています。

表 15.2 X.509 証明書失効リスト (CRL)

項目	内容
バージョン	CRL のバージョン。例: v2
署名	CRL に署名する際に使用したアルゴリズムの ID
発行者	CRL を発行した機関 (一般に発行元の CA) のユニーク名 (DN)

項目	内容
更新日時	CRL の発行日時
次回更新日時	次回の CRL 発行日時
失効済み証明書の一覧	各項目には証明書のシリアル番号と失効した日時、およびその他の拡張 (CRL 項目の拡張) が含まれています。
拡張	任意の CRL 拡張

15.1.4 証明書と CRL のリポジトリ

CA における証明書と CRL は、*リポジトリ* を介して 広くアクセスできなければなりません。証明書や CRL は、自身の偽造を防ぐ目的 で署名されていることから、リポジトリ自身を特別な方法で保護したりする必要はありません。その代わり、できるだけシンプルかつ高速なアクセス手段を提供 する必要があります。この理由から、証明書は LDAP や HTTP サーバ上で提供する のがよいでしょう。LDAP についての説明は 第4章 *ディレクトリサービス LDAP* (39 ページ) をお読みください。また、第20章 *Apache HTTP サーバ* (↑リファレンス) には HTTP サーバに関する情報が記載されています。

15.1.5 プロプライエタリな PKI

YaST には X.509 証明書を管理するための基本的な機能を提供するモジュールがあります。これは CA や副 CA の作成のほか、それらの証明書を作成する 機能も備えています。PKI のサービスは単純に証明書や CRL を作成したり 配布したりするだけではなく、証明書や CRL の更新を継続して行なうための厳密に 設計された管理構造を必要とします。このような構造は商用の PKI 製品として 提供されていて、部分的ではありますがこれらを自動化することができます。YaST では CA や証明書の作成や配布機能を提供していますが、現時点では 上述の管理構造までは提供していません。小規模な PKI を構築するにあたっては YaST のモジュールで十分ですが、「公式の」PKI や商用目的の PKI を構築する場合は、製品のご利用をお勧めします。

15.2 CA 管理用の YaST モジュール

YaST では 2 種類の基本的な CA 管理機能を提供しています。ここではこれらのモジュールを利用した主な作業について説明しています。

15.2.1 ルート CA の作成

PKI を設定するにあたっての最初の手順は、ルート CA を作成することです。下記の手順で行なってください:

- 1 YaST を起動し、**セキュリティとユーザ** > **CA 管理** を選択します。
- 2 **ルート証明機関の作成** を押します。
- 3 最初のダイアログ (図 15.1) では、CA に対する基本情報を入力します。それぞれのテキスト項目の意味は以下の通りです:

図 15.1 YaST CA モジュールルート CA に対する基礎データ

新規作成Root CA (ステップ 1/3)

証明機関名 (C):

共通名 (C):

メールアドレ

組織 (R):

部署 (G):

市区町村 (L):

都道府県 (S):

国 (O):

証明機関名

CA の技術名を入力します。ディレクトリ名などは、ここで入力した名前をもとにして既定値が作成されます。また、ヘルプに表示されているとおり、利用可能な文字に制限があることに注意してください。また、技術名はモジュールを起動したときに表示される名前でもあります。

共通名

CA を参照する際に使用する名前を指定します。

メールアドレス

CA のユーザから見ることのできるメールアドレスには、複数のアドレスを設定することができます。これらは問い合わせの際に便利な項目です。

国

CA の運用している国を選択します。

組織, 部署, 市区町村, 都道府県

いずれも任意指定の値です。

次へ を押して進みます。

- 4 2 番目のダイアログでは、パスワードを入力します。ここで入力した パスワード は、CA を使用する際に求められるもので、副 CA の作成や 証明書の作成の際に必要となります。残りの項目の意味は下記の通りです:

鍵長

鍵長 には自動的に値が設定されていて、アプリケーション 側でその長さの鍵を利用できない場合を除き、既定値を変更する必要は ありません。より長い鍵 (数値の大きな鍵) ほど、入力したパスワードの 機密性が高まります。

有効期間 (日)

CA の場合、有効期間 (日) の既定値は 3650 日 (約 10 年) になっています。このような長い期間になっているのは、無効になった CA の 証明書を置き換えるのは、管理上の手間が非常に大きくかかるためです。

なお、詳細オプション を押すと、X.509 拡張 (図15.4「YaST CA モジュール拡張設定」(207 ページ)) について、より詳しい値を設定する ためのダイアログが表示されます。これらの値にはそれぞれ既定値が存在していて、特段の理由が無い限り変更すべきではありません。次へ を押すと先に進みます。

- 5 最後に概要を確認します。YaST では確認のため、現時点での設定を表示します。作成 を押して作成処理を行なうと、概要画面にルート CA が表示されるようになります。

ヒント

一般的には、ルート CA に対してユーザの証明書を発行するのは不適切です。少なくとも 1 段階の副 CA を用意し、ユーザ証明書はその副 CA から発行してください。このようにすることで、ルート CA が独立した存在となるため、機密の保護がより容易になり、ルート CA への攻撃が非常に難しいものとなります。

15.2.2 パスワードの変更

お使いの CA に対するパスワードを変更するには、下記の手順で行ないます：

- 1 YaST を起動し、CA モジュールを開きます。
- 2 パスワードを変更したいルート CA を選び、*証明機関に入る* を押します。
- 3 まずは CA の現在のパスワードを入力します。すると YaST は、CA の情報を *説明* タブ内に表示します (図 15.2 をご覧ください)。
- 4 *詳細設定* ボタンを押し、*証明機関のパスワード変更* を選択します。ダイアログボックスが表示されます。
- 5 それぞれ既存のパスワードと、新しく設定するパスワードを入力します。
- 6 最後に *OK* を押せば完了です。

15.2.3 下位 CA の作成と失効化

下位 CA の作成方法は、ルート CA と全く同じです。

注記

ただし、下位 CA の有効期限は、「親の (下位 CA の発行元の)」CA の有効期限内に存在しなければならないことに注意してください。下位 CA は 常に「親の」CA より後に作成しなければならないため、既定値のまま作成しようとするとエラーメッセージが表示されてしまいます。このような エラーを避けるには、有効期限の値を許容範囲内で設定してください。

下記の手順で行ないます:

- 1 YaST を起動し、CA モジュールを開きます。
- 2 下位 CA を作成したいルート CA を選び、*証明機関に入る* を押します。
- 3 初めて CA に入る場合は、パスワードを入力します。すると YaST は、CA の情報を *説明* タブ内に表示します (図 15.2 をご覧ください)。

図 15.2 YaST CA モジュール—CA の使用



- 4 *詳細設定* を押してから *下位証明機関の作成* を選択します。ルート CA を作成したときと同じダイアログが表示されます。
- 5 15.2.1項「ルート CA の作成」(200 ページ) と同じ手順で作業を行ないます。

すべての CA に対して同じパスワードを設定することもできます。この場合は、*証明機関のパスワードを証明書のパスワードとして使用する* を選択してください。このようにすることで、ルート CA と同じパスワードを 下位 CA に設定することができるため、CA に対するパスワード管理の手間を減らすことができます。

注記: 正しい有効期間の設定について

下位 CA 用に設定する有効期間は、ルート CA の有効期間よりも短くしなければ ならないことに注意してください。

- 6 作成が完了したら、**証明書** のタブを選択します。ここでは、誤って発行してしまった場合など、不要な証明書をリセットすることができます。それぞれ証明書を選択して、**取り消し** ボタンを押してください。なお、取り消しの作業だけでは、下位 CA を無効化することができません。CRL 内に 取り消した下位 CA の情報を配布する必要があります。CRL の作成方法については 15.2.6 項「証明書失効リスト (CRL) の作成」(207 ページ) をお読みください。
- 7 最後に **OK** を押せば完了です。

15.2.4 ユーザ証明書の作成と失効化

クライアントやサーバの証明書の作成は、15.2.1 項「ルート CA の作成」(200 ページ) で説明している CA の作成ととてもよく似ていて、同じようなやり方で作成することができます。たとえば電子メール署名用の証明書であれば、電子メールのプログラムに取り込む 際、そのユーザのメールアドレス (機密鍵の所有者) が証明書に書かれていなければなりません。また、暗号化の際に証明書を利用するような場合では、受信者 (公開鍵 の所有者) 電子メールアドレスが証明書に書かれていなければなりません。サーバやクライアントに対する証明書の場合は、それぞれサーバやクライアントの ホスト名が **共通名** に書かれていなければなりません。いずれの場合も、有効期限の既定値は 365 日です。

サーバやクライアントの証明書を作成するには、下記の手順で行ないます：

- 1 YaST を起動し、CA モジュールを開きます。
- 2 証明書を作成したいルート CA を選び、**証明機関に入る** を押します。
- 3 初めて CA に入る場合は、パスワードを入力します。すると YaST は、CA の情報を **説明** タブ内に表示します。
- 4 **証明書** タブに移動します (図 15.3 をご覧ください)。

図 15.3 CA の証明書

5 追加 > サーバ証明書の追加 を選択し、サーバ証明書を作成します。

6 追加 > クライアント証明書の追加 を選択し、クライアント証明書を作成します。
このとき、電子メールアドレスの入力は忘れずに実施してください。

7 OK を押せば完了です。

誤って発行してしまったりした不要な証明書を取り消すには、下記の手順で行ない
ます:

- 1 YaST を起動し、CA モジュールを開きます。
- 2 証明書の発行元ルート CA を選び、証明機関に入る を押します。
- 3 初めて CA に入る場合は、パスワードを入力します。すると YaST は、CA の情報を説明 タブ内に表示します。
- 4 証明書 タブに移動します (15.2.3 項「下位 CA の作成と失効化」(202 ページ) をお読みください)。
- 5 取り消したい証明書を選択し、取り消し を押します。
- 6 証明書の取り消し理由を選択します。

7 OK を押せば完了です。

注記

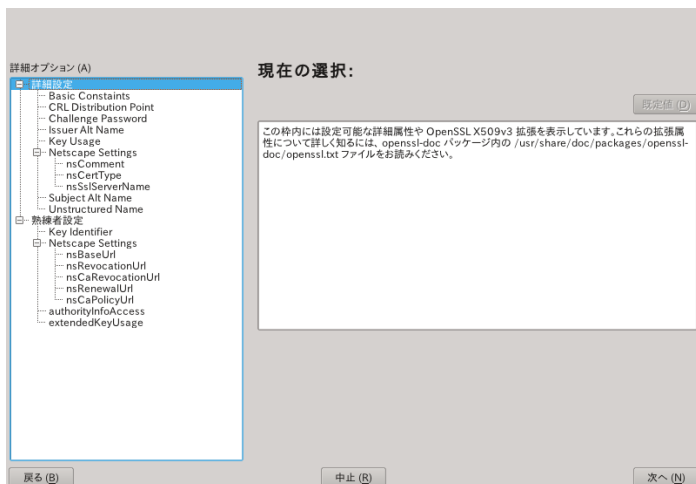
なお、取り消しの作業だけでは、証明書を無効化することができません。CRL 内に取り消した証明書の情報を配布する必要があります。CRL の作成 方法については 15.2.6 項「証明書失効リスト (CRL) の作成」(207 ページ) をお読みください。CRL に公開したあとであれば、**削除** ボタンで取り消した証明書を完全に消去できるようになります。

15.2.5 既定値の変更

今までこれまでの章では、下位 CA やクライアント、サーバの各証明書の作成 手順を説明してきました。X.509 の証明書の拡張部分を扱うには、特別な設定が必要となります。これらの値にはそれぞれ既定値が存在していて、特段の理由が 無い限り変更すべきではありません。しかしながら、特別な理由から変更をする 必要があります。このような場合は既定値を変更してください。それ以外の場合は、証明書の作成時に毎回設定してください。

- 1 YaST を起動し、CA モジュールを開きます。
- 2 設定を変更したいルート CA に入ります。15.2.3 項「下位 CA の作成と失効化」(202 ページ) の手順で行なってください。
- 3 **詳細設定 > 既定値の編集** を押します。
- 4 変更を行ないたい対象を選択します。既定値の変更ダイアログが 図15.4「YaST CA モジュール拡張設定」(207 ページ) のように表示されます。

図 15.4 YaST CA モジュール—拡張設定



- 5 左側で設定を変更したい項目を選び、右側でそれぞれの値を変更します。
- 6 次へ を押すと概要を表示することができます。
- 7 最後に *Save* を押すと設定を保存することができます。

注記

ここで設定した既定値は、これ以降に作成した証明書だけに適用されます。既に存在する CA や証明書については、変更は行ないません。

15.2.6 証明書失効リスト (CRL) の作成

証明書を誤って発行してしまった場合など、不要な証明書は使用できないようにするため、取り消し処理を行わなければなりません。それぞれ取り消し処理の手順については、15.2.3項「下位 CA の作成と失効化」(202 ページ) (下位 CA の場合) や 15.2.4項「ユーザ証明書の作成と失効化」(204 ページ) (ユーザ証明書の場合) で説明を行なっています。それぞれの作業後、CRL を作成して取り消した情報を公開しなければなりません。

なお、システムはそれぞれの CA に対して 1 つの CRL のみを管理します。CRL を作成したり更新したりするには、下記の手順で行ないます:

- 1 YaST を起動し、CA モジュールを開きます。
- 2 CRL を作成したいルート CA に入ります。15.2.3項「下位 CA の作成と失効化」(202 ページ) の手順で行ってください。
- 3 CRL を選択します。対象の CA に対して、最近作成した CRL の概要を表わすダイアログが表示されます。
- 4 新たに取り消した下位 CA や証明書がある場合は、CRL の生成 を押して新しい CRL を作成します。
- 5 まずは新しく生成する CRL の有効期間を指定します (既定値: 30 日) 。
- 6 OK を押して CRL を生成して表示します。この後、生成した CRL を公開しなければなりません。

注記

CRL を検証することのできるアプリケーションは、CRL が利用できないような場合や有効期限が切れているような場合、すべての証明書を受け付けなくなります。PKI の提供元としては、現在の CRL の有効期限が切れる前に新しい CRL を生成し、公開する義務を負うことに注意してください。YaST では、このような作業の自動化までは行なっていません。

15.2.7 LDAP に対する CA オブジェクトのエクスポート

LDAP にエクスポートするには、対象のコンピュータを YaST LDAP クライアントを利用して設定する必要があります。ダイアログの各項目に入力を行なうことで、使用する LDAP サーバの情報を実行時に取得するようになります。YaST LDAP クライアントモジュールを利用しない場合は (それでもエクスポート機能は動作しますが)、すべての LDAP データを手作業で入力しなければなりません。また、複数のパスワードを入力する必要もあります (詳しくは 表15.3「LDAP エクスポート時のパスワード」(209 ページ) をお読みください) 。

表 15.3 LDAP エクスポート時のパスワード

パスワード	意味
LDAP パスワード	LDAP ツリー内の項目を作成する際の認証
証明書パスワード	証明書をエクスポートする際の認証
新しい証明書パスワード	LDAP エクスポートの際には PKCS12 形式を利用します。この形式の場合、エクスポートした証明書に対して、新しいパスワードの付与が強制されます。

証明書, CA, CRL をそれぞれ LDAP にエクスポートすることができます。

CA の LDAP へのエクスポート

CA をエクスポートするには、まず 15.2.3 項「下位 CA の作成と失効化」(202 ページ) の手順で CA を設定します。その後、続くダイアログで *詳細設定* > *LDAP にエクスポート* を選択します。すると、LDAP の情報を入力するダイアログが表示されます。既に YaST LDAP クライアントで設定が完了している場合は、表示された項目が部分的に記入済みの状態で表示されます。それ以外の場合はすべてのデータを手作業で入力してください。必要なデータを入力すると、「caCertificate」という属性項目が個別のツリー内に作成されます。

証明書の LDAP へのエクスポート

エクスポートしたい証明書を含む CA (証明機関) に入り、*証明書* タブを選択します。表示されたダイアログの上半分で証明書を選択し、*エクスポート* > *LDAP にエクスポート* を選択します。ここで入力する LDAP データは、CA をエクスポートする場合と同様です。証明書は LDAP ツリー内で指定したユーザオブジェクト内の、「userCertificate」(PEM 形式) と「userPKCS12」(PKCS12 形式) の属性に保存されます。

CRL の LDAP へのエクスポート

エクスポートしたい CRL を含む CA (証明機関) に入り、*CRL* タブを選択します。必要であれば新しい CRL を作成し、*エクスポート* を押します。すると、各種のパラメータ設定を行なうためのダイアログが表示されます。エクスポート処理は一度だけ行なうことができるほか、一定の時間間隔で繰り返し実行すること

もできます。まずは *LDAP にエクスポート* にチェックを入れ、必要な LDAP 情報を入力してください。一定の時間間隔でエクスポートしたい場合は、*再発行を繰り返してエクスポート* のラジオボタンを選択し、間隔を指定してください。

15.2.8 CA オブジェクトのファイルへのエクスポート

お使いのコンピュータに CA を管理するためのリポジトリを設定している場合、このオプションを利用することで、CA オブジェクトを直接ファイルに作成することができます。この場合、PEM や DER、PKCS12 など、複数の出力形式に対応しています。PEM の場合は、証明書とともに鍵を含めるかどうかと、鍵を暗号化するかどうかを選択することができます。また、PKCS12 の場合は、証明書をあわせてエクスポートすることもできます。

ファイルへのエクスポートは、LDAP に対する CA のエクスポートや証明書のエクスポートと同じで、15.2.7 項「LDAP に対する CA オブジェクトのエクスポート」(208 ページ) の手順で作業を行ないます。このとき、*LDAP にエクスポート* ではなく *ファイルにエクスポート* を選択する必要があることに注意してください。これにより、出力形式とパスワード、およびファイル名を指定できるようになります。最後に *OK* を押すと、指定した場所に証明書が保存されます。

CRL の場合は *エクスポート* を押した後 *ファイルにエクスポート* を選択します。あとは出力形式 (PEM または DER) を選択し、出力先パスを指定します。最後に *OK* を押すと、指定した場所に出力されます。

ヒント

エクスポート処理は、ファイルシステム内であれば任意の場所を指定できます。それ以外にも、たとえば USB メモリなど、外部メディアに CA オブジェクトを保存することもできます。お使いのシステムに内蔵されているハードディスク以外の場所を指定する場合、一般的には */media* 以下に表示されます。

15.2.9 共通サーバ証明書のインポート

CA を管理するサーバ上で YaST を利用してサーバ証明書を作成した場合、メディアなどにエクスポートして対象のサーバに複製する必要があります。対象のサーバではこれを *共通サーバ証明書* と呼びます。このインポート作業はインストール中に実施することができるほか、YaST を利用して後から行なうこともできます。

注記

証明書を正しくインポートするには、PKCS12 形式のうちのいずれかを利用 する必要があります。

共通サーバ証明書は /etc/ssl/servercerts ディレクトリ内に 保管され、CA 機能に対応するすべてのサービスから利用できるようになります。また、この証明書の有効期限が切れた場合も、同じ手順で簡単に入れ替えることができるようになっています。証明書を入れ替えた後は、対象のサービスを再起動 することで新しい証明書を利用できるようになります。

ヒント

インポート を選択すると、ファイルシステム内の任意の 場所を選択することができます。インポートの処理はハードディスク のほか、USB メモリなどの外部メディアも選択することができます。

共通サーバ証明書をインポートするには、下記の手順で行ないます：

- 1 YaST を起動し、*セキュリティとユーザ* 内にある *共通サーバ証明書* を選択します。
- 2 YaST のモジュールが起動すると、説明欄に現在の証明書に関するデータが 表示されます。
- 3 *インポート／置換* を押し、証明書を選択します。
- 4 パスワードを入力して *次へ* を押します。証明書が取り 込まれ、説明欄に新しい証明書の情報が表示されるようになります。
- 5 最後に *完了* を押し、YaST を完了します。

15.3 さらなる情報

X.509 証明書について、詳しい情報は <http://www.ietf.org/html.charters/pkix-charter.html> をお読みください。

パート IV. AppArmor を利用した権利制限

AppArmor の紹介

信頼のおける プログラムであっても、バグに起因する 数多くのセキュリティ脆弱性が存在しています。信頼のおけるプログラムは、攻撃者が望むだけの権限を持って動作しているため、プログラム内にバグが存在すると、攻撃者に必要な権限を与えてしまうことになり、信頼を損なってしまいます。

AppArmor® は、疑念のあるプログラムに対して権利の制限を行なうことに特化して設計された、アプリケーションセキュリティソリューションです。AppArmor は管理者 に対してプログラムが実施できる動作領域を指定できるようにするためのもので、動作領域はセキュリティ判断 (AppArmor では *プロファイル* と呼びます) によって設定を行ないます。プロファイルとは、プログラムがアクセス する可能性のあるファイルや、プログラムが実施する可能性のある操作の一覧です。これにより、AppArmor は実際の攻撃を検出することなく、アプリケーションに対して 正しい動作を強制することになり、たとえ未知の脆弱性に対する攻撃であっても それを防ぐことができるようになります。

AppArmor は下記のコンポーネントから構成されています:

- 一般的な Linux* アプリケーション向けの AppArmor プロファイルライブラリ。プログラムがアクセスする必要のあるファイルを列挙しています。
- AppArmor プロファイル基礎クラス (プロファイル構築ブロック)。DNS の参照やユーザ認証など、一般的なアプリケーション動作に必要となります。
- AppArmor プロファイルの開発や拡張のためのツールスイート。これらを利用することで、既存のプロファイルを必要に応じて修正することができるほか、独自のアプリケーションに対して、新しいプロファイルを作成したりすることもできます。

- AppArmor が有効化されている特別版のアプリケーションで、ユニークなサブプロセス 制限の形でセキュリティの拡張可能を提供するもの (Apache や Tomcat などを含む)。
- AppArmor それ自身。読み込み可能なカーネルモジュールと、関連する制御用スクリプト から構成され、お使いの openSUSE® 製品に対して AppArmor のポリシーを適用する ために使用します。

16.1 AppArmor プロファイリングの関連情報

AppArmor の技術やセキュリティについて、詳しくは下記の書籍をお読みください:

SubDomain: Parsimonious Server Security by Crispin Cowan, Steve Beattie, Greg Kroah-Hartman, Calton Pu, Perry Wagle, and Virgil Gligor
AppArmor の初期デザインと実装について説明した書籍です。2000 年 12 月の New Orleans, LA における USENIX LISA カンファレンスで発行された報告書 です。内容は既に古いものになってしまっていて、文法や機能に関する説明は、現在の AppArmor 製品とは異なっています。この書籍は背景となる知識を得るために有用なものであり、技術文書としては使用すべきではありません。

Defcon Capture the Flag: Defending Vulnerable Code from Intense Attack by Crispin Cowan, Seth Arnold, Steve Beattie, Chris Wright, and John Viega

とても短い期間に発生した重要なセキュリティ問題を AppArmor で解決する際の、戦略／戦術的に良いガイドです。2003 年 4 月に Washington, DC で開催された DARPA Information Survivability Conference and Expo (DISCEX III) で発行 された報告書です。

AppArmor for Geeks by Seth Arnold

このドキュメントは、AppArmor の技術詳細についてよりよく理解するための文書です。こちらは http://en.opensuse.org/SDB:AppArmor_geeks からアクセスできます。

前置き

お使いのシステムに AppArmor をうまく導入するには、下記の要件について注意深く 考慮する必要があります:

- 1 プロファイルを行なうアプリケーションを決めること。詳しくは 17.3項「プロファイルを行なうアプリケーションの選択」(219 ページ) をお読みください。
- 2 17.4項「プロファイルの構築と修正」(220 ページ) の概要に従って、必要なプロファイルを構築すること。必要であれば、結果を確認してプロファイルを 調整する必要もあります。
- 3 お使い間環境が変化したような場合にはプロファイルを更新するか、もしくは AppArmor のレポートツールが記録するセキュリティイベントに対して何らかの 処理を行なうこと。詳しくは 17.5項「プロファイルの更新」(222 ページ) をお読みください。

17.1 AppArmor のインストール

AppArmor は openSUSE を既定値でインストールした場合は、インストールされません。AppArmor を完全にインストールするには、パターン `apparmor` を選択してインストールしてください。インストールには YaST ソフトウェア管理 モジュールを利用するか、下記の `zypper` コマンドを利用します:

```
zypper in -t pattern apparmor
```

- `apparmor-docs`
- `apparmor-parser`
- `apparmor-profiles`

- `apparmor-utils`
- `audit`
- `libapparmor1`
- `perl-libapparmor`
- `yast2-apparmor`

17.2 AppArmor の有効化と無効化

AppArmor は openSUSE をインストールしていれば、既定で起動するように 設定されています。AppArmor の起動設定を変更するには、下記の 2 種類の 方法があります:

YaST システムサービス (ランレベル) の使用

システムの起動時に実行されるスクリプト群に対して、AppArmor の起動処理を追加したり削除したりすることで、AppArmor を有効にしたり無効にしたりすることができます。設定の変更は、再起動後に反映されます。

AppArmor コントロールパネルの使用

YaST AppArmor コントロールパネルを利用することで、動作中のシステムに対して AppArmor の状態を有効または無効に切り替えることができます。ここでの 変更は即時に反映されます。コントロールパネルでは、イベントの開始や停止のほか、システムの起動処理内に AppArmor の起動を追加するかどうかを設定することもできます。

AppArmor を恒久的に無効化する (システム起動時に実行されないようにする) には、下記の手順で行ないます:

- 1 YaST を起動します。
- 2 システム > システムサービス (ランレベル) を選択します。
- 3 熟練者モード を選択します。
- 4 `boot.apparmor` と書かれている行を選択してから、**セット／リセット > サービスを無効にする** を押します。
- 5 あとは **OK** を押して YaST ランレベルエディタを終了します。

これで AppArmor は次回の再起動では初期化されなくなり、再度有効化しない限り 停止状態になります。YaST のランレベルツールを利用すれば、サービスを 再度有効化するのも上記と似た手順で行なうことができます。

AppArmor のコントロールパネルを利用すると、動作中のシステムで AppArmor の状態を 切り替えることができます。これらの変更は即時に反映され、システムの再起動 後にもその設定が維持されます。AppArmor の状態を切り替えるには、下記の手順で 実施します:

- 1 YaST を起動します。
- 2 *AppArmor > AppArmor コントロールパネル* を選択します。
- 3 *AppArmor を有効にする* を 選択します。無効にしたい場合は選択を外します。
- 4 あとは *完了* を押して AppArmor コントロールパネルを閉じます。

17.3 プロファイルを行なうアプリケーションの選択

プログラムに対する保護は、お使いの環境において攻撃を受ける可能性のあるプログラムに対してのみ、実施する必要があります。そのため、実際に 実行するアプリケーションのみをプロファイルしてください。攻撃を受ける 可能性のあるアプリケーションとしては、下記のようなものがあります:

ネットワークでの通信を行なうもの
Web アプリケーション
cron ジョブ

現時点でネットワークポートを開いている動作中のプロセスが何であるのかを調べて、制限を行なうためのプロファイル作成に役立てるには、`root` で `aa-unconfined` を実行します。

例 17.1 *aa-unconfined* の出力

```
19848 /usr/sbin/cupsd not confined
19887 /usr/sbin/sshd not confined
19947 /usr/lib/postfix/master not confined
29205 /usr/sbin/sshd confined by '/usr/sbin/sshd (enforce)'
```

上記の出力例では、それぞれのプロセスが `not confined` と示されていて、制限を行なうのに独自のプロファイルが必要である旨が示されています。`confined by` と書かれているものは、既に AppArmor で保護されていることを示しています。

ヒント: さらなる情報

プロファイルを行なうにあたって、正しいアプリケーションを選択するための 詳細については、18.2項「免疫を与えるプログラムの判断」(226 ページ) をお読みください。

17.4 プロファイルの構築と修正

openSUSE に同梱されている AppArmor は、最も重要なアプリケーション向けのプロファイルが事前に設定された形で公開されています。この状態から AppArmor を利用し、任意のアプリケーションに対する独自のプロファイルを作成することもできるようになっています。

プロファイルを管理する方法としては 2 種類の方法があります。1 つは YaST の AppArmor モジュールで、グラフィカルなフロントエンドを利用する方法です。もう 1 つは AppArmor スイート自身で提供されている、コマンドラインツールを利用する方法です。いずれの方法とも、基本的には同じ動作です。

プロファイルを作成するには、それぞれのアプリケーションに対して下記の 手順を実施します:

- 1 まずは root になった状態で AppArmor を利用し、アプリケーションの 大まかなプロファイルを作成します。これは `aa-genprof プログラム名` のように実行することで作成できます。

上の手順のほか、下記の手順でもかまいません。

YaST > AppArmor > AppArmor プロファイルウイザード を実行して 基本的なプロファイルを作成します。このとき、プロファイル対象の アプリケーションはフルパスで指定します。

基本的なプロファイルについて概要を作成すると、AppArmor は学習モードに移行します。学習モードでは実行したプログラムの動作が記録されますが、制限は行なわれません。

- 2 AppArmor に対して動作概要を正確に学習させるため、アプリケーションのすべての 機能を使用します。
- 3 `aa-genprof` では S を押し、AppArmor によって ステップ 2 (220 ページ) で生成したログについて、ファイルの分析作業を 実施します。

もしくは

AppArmor プロファイルウィザード内でシステムログをスキャンして *AppArmor* イベントを検出 を押し、ログファイルの分析を行ないます。ウィザードが示す手順に従い、プロファイルを完成させてください。

AppArmor は、アプリケーションが動作していたときに記録したログファイルを検索し、それぞれのイベントに対して正しいアクセス権を設定するように問い合わせが表示されます。それぞれのファイルに対して設定することができるほか、グループを利用して複数を一括指定することもできます。

- 4 お使いのアプリケーションの複雑さにもよりますが、ステップ 2 (220 ページ) や ステップ 3 (220 ページ) の手順を繰り返す必要がある場合もあります。この場合はアプリケーションの制限を設定し、制限された環境下でアプリケーションの動作をテストし、新しいログイベントを処理し直してください。アプリケーションのすべての機能に対して正しく制限を行なうには、この手順を繰り返し実施する必要となる場合があります。
- 5 すべてのアクセス許可を設定したら、お使いのプロファイルを強制モードに切り替えてください。作成したプロファイルが適用され、AppArmor はプロファイルに記されたとおりの制限を行ないます。

不平 (complain) モードで実行し、既存のプロファイルが存在するアプリケーションに対して aa-genprof を起動すると、プロファイルはその学習サイクルを終了すると学習モードにあり続けます。プロファイルのモード切替について、詳しくは 22.6.3.2 項「aa-complain—不平モード／学習モードへの突入」(288 ページ) と 22.6.3.3 項「aa-enforce—強制モードへの突入」(289 ページ) をお読みください。

制限された環境で動作させた際は、対象のアプリケーションに対して必要な作業を行ない、プロファイル設定をテストしてください。通常は制限された環境下でも問題なく動作し、AppArmor が動作しているかどうかは気になることはありません。しかしながら、お使いのアプリケーションの動作に明らかな問題が見つかった場合は、まずシステムログを確認して AppArmor がアプリケーションを制限しすぎているかどうかを確認してください。お買いのシステムで使用しているログ機構に依存しますが、AppArmor のログを確認する際は下記の場所をご覧ください:

```
/var/log/audit/audit.log  
/var/log/messages  
dmesg
```

また、プロファイル进行调整するには ステップ 3 (220 ページ) の手順に 従い、対象のアプリケーションに対するログメッセージを分析してください。問い合わせがあった場合は、アクセス権や制限についても判断を行なってください。

ヒント: さらなる情報

プロファイルの構築や修正について、詳しくは 第19章 *プロファイルの構成と文法* (233 ページ)、第21章 *YaST を利用したプロファイルの構築と管理* (259 ページ)、第22章 *コマンドラインからのプロファイル構築* (279 ページ) をお読みください。

17.5 プロファイルの更新

ソフトウェアやシステムの更新は、常日頃から発生しうるものです。そのため、AppArmor 向けに設定したプロファイルは、それらの更新にあわせて細かい調整を加える 必要がある場合があります。AppArmor ではシステムログを調査してポリシー違反などの AppArmor イベントを確認し、必要に応じてプロファイル进行调整することができるようになっています。プロファイル定義以外の方法でアプリケーションの動作を 調整するには、*プロファイルの更新ウィザード* を使用します。

プロファイルセットを更新するには、下記の手順で行ないます:

- 1 YaST を起動し、*AppArmor > プロファイルの更新ウィザード* を選択します。
- 2 問い合わせがあった場合は、記録のあった任意のリソースや実行形式に対するアクセス権限や実行権限を調整します。
- 3 すべての質問に回答したら、YaST を終了します。変更点は関連する プロファイルに適用されます。

ヒント: さらなる情報

システムログを利用してプロファイルを更新する方法について、より詳しい情報は 21.5項「ログ項目からのプロファイル更新」(275 ページ) をお読みください。

プログラムに対する免疫付与

コンピュータシステムを効率よく堅牢にするには、権限を媒介するプログラムの数を最小化し、可能な限りプログラムの機密性を高める必要があります。AppArmor では、お使いの環境で攻撃に晒される可能性のあるプログラムをプロファイルするだけで作業は終わるため、お使いのコンピュータを堅牢にする作業の量を劇的に減らすことができます。また AppArmor はプログラムが実行する可能性のある処理を確かめるためにプロファイルを行なうだけで、それ以外にはプロファイルを使用することはありません。

AppArmor® はアプリケーションに対して、生まれながらに持っているような脆弱性から保護するための免疫技術を提供します。AppArmor をインストールしてプロファイルの設定を完了し、コンピュータを再起動すれば、AppArmor のセキュリティポリシーが強制されることから、お使いのシステムに免疫が備わることになります。AppArmor を利用したプログラムの保護は、*免疫* と呼びます。

管理者は、攻撃に対して脆弱なアプリケーションに対して、プロファイルの生成にのみ注力するだけで済みます。つまり、システムを堅牢にする作業は、AppArmor のプロファイルセットを構築または管理し、ポリシー違反を監視するか、AppArmor のレポート機能で例外を記録するなどの作業に置き換わることになります。

ユーザは AppArmor を全く気にする必要はありません。AppArmor は「裏でこっそり」動作しているもので、ユーザ側に問い合わせを行なうようなことはありません。AppArmor の使用によって性能が顕著に落ちるようなこともありませんし、何らかのアプリケーション動作が AppArmor のプロファイル側で考慮範囲外のものであったり、禁止されているものであったりした場合も、管理者がプロファイルを調整して、そのような振る舞いに対応するようにするだけです。

AppArmor は標準の Linux サービスを保護するため、多数のアプリケーションに対して、既定のプロファイル集を設定します。他のアプリケーションを保護するには、AppArmor ツールを利用して、保護したいアプリケーションのプロファイルを作成してください。この章では、プログラムに免疫を与えるための考え方を紹介しています。既に AppArmor のプロファイルを構築していたり管理していたりする場合は、第19章 *プロファイルの構成と文法* (233 ページ)、第21章 *YaST を利用したプロファイルの構築と管理* (259 ページ)、第22章 *コマンドラインからのプロファイル構築* (279 ページ) を読み進めてください。

また、AppArmor はネットワークサービスに対して、それぞれのプログラムがどのファイルに 読み込み／書き込み／実行を許可するのか、そしてどの種類のネットワークに アクセスを許可するのかを指定することで、合理的なアクセス制御を行なうことができます。この仕組みにより、それぞれのプログラムが実行する可能性のある 処理を確認するだけで堅牢性を高めることができます。AppArmor はプログラムを 検疫し、危険な処理によってシステム全体にダメージを与えてしまうようなことから保護することができます。

AppArmor はホスト侵入阻止と強制アクセス制御機能の両方を備えたシステムです。従来、アクセス制御は巨大なタイムシェアリング (時間共有型) のシステム向けに 構築された、ユーザにとっては集中管理型の仕組みでした。その後、ネットワークサーバは直接的なログインを許可する代わりに、様々なネットワークサービス (Web, メール, ファイル共有, 印刷サービスなど) をユーザ向けに提供するようになりました。AppArmor はこのようなネットワークサービスに対するアクセスを制御し、その他のプログラムの弱点を突かれるようなことが無いように保護します。

ヒント: AppArmor の背景となる情報について

AppArmor に関するより深い概要や全体的な考え方について、詳しくは 16.1 項「AppArmor プロファイリングの関連情報」(216 ページ) をお読みください。

18.1 AppArmor フレームワークの紹介

この章では、AppArmor を動作させている際に「裏側で何が行なわれているのか」(および YaST インターフェイスの仕組み) について、非常に 基本的な部分を説明しています。

AppArmor のプロファイルは単純なテキストファイル形式で、パス項目とそれに対する アクセス権が書かれています。プロファイルについて、詳しくは 19.1 項

「AppArmor プロファイルの構成」(234 ページ) をお読みください。この テキスト ファイル内に含まれている設定は AppArmor ルーチン側で読み取られた あと、プロセスやプログラムを検疫するために利用されます。

AppArmor のプロファイルやポリシーを構築したり強制したりするにあたって、下記のようなツールを利用することができます:

aa-unconfined / unconfined

aa-unconfined はお使いのシステムでネットワーク接続を 待ち受けていて、かつ AppArmor のプロファイルで保護されていないアプリケーションを 検出します。このツールについて、詳しくは 22.6.3.8項「aa-unconfined—保護されていないアクセスの検出」(304 ページ) をお読みください。

aa-autodep / autodep

aa-autodep は、本番の環境で使用する前に肉付けを行なう 必要があるような、プロファイルの基本的なフレームワークを作成します。作成されたプロファイルは読み込まれたあと不平 (complain) モードに置かれ、AppArmor のルールではカバーしていないアプリケーションの動作を報告します。このツールについて、詳しくは 22.6.3.1項「aa-autodep—概要プロファイルの作成」(286 ページ) をお読みください。

aa-genprof / genprof

aa-genprof は基本的なプロファイルを生成し、対象の アプリケーションを起動してプロファイルを改良したあと、AppArmor のポリシー で注意する必要のある項目についてログを生成します。なお、アプリケーション の実行中に発生したログイベントについては、これらをどのように扱うのかについて、質問が表示されるようになっています。プロファイルが生成された 後は、これを読み込んで強制モードに移行します。このツールについて、詳しくは 22.6.3.4項「aa-genprof—プロファイルの生成」(290 ページ) をお読みください。

aa-logprof / logprof

aa-logprof は、不平 (complain) モードのプロファイルで 制限しているアプリケーションに対し、そこから生成されたログを対話的に スキャンして確認するためのツールです。プロファイルが生成したログ から、新しい設定項目の作成を支援するツールです。このツールについて、詳しくは 22.6.3.5項「aa-logprof—システムログのスキャン」(298 ページ) をお読みください。

aa-complain / complain

aa-complain は AppArmor プロファイルのモードを、強制モードから不平 (complain) モードに切り替えるためのツールです。プロファイル内に設定さ

れたルールの例外は記録されますが、強制され なくなります。このツールについて、詳しくは 22.6.3.2項「aa-complain—不平モード／学習モードへの突入」(288 ページ)をお読みください。

aa-enforce / enforce

aa-enforce は AppArmor プロファイルのモードを、不平 (complain) モードから強制モードに切り替えるためのツールです。プロファイル内に設定されたルールの例外は記録されますが、許可されなくなり、プロファイルが強制されるようになります。このツールについて、詳しくは 22.6.3.3項「aa-enforce—強制モードへの突入」(289 ページ)をお読みください。

いったんプロファイルを構築して読み込むと、そのプロファイルに対する処理方法を下記のいずれかから選択できるようになります：

aa-complain / complain

不平 (complain) モードでは、AppArmor のプロファイルルールに対する違反、たとえば対象のプログラムが、プロファイルで許可していないファイルにアクセスしようとした場合、違反内容は許可されますが記録されるようになります。プロファイルを改善するには、不平モードを有効にしてからプログラムの機能を一通り使ってみて、プログラムの動作特性に関するログを生成させてください。生成されたログは AppArmor のツール (YaST または aa-logprof) を使用して後処理することで、ログイベントを改善済みのプロファイルに変換することができます。

aa-enforce / enforce

強制モードでは AppArmor のプロファイルルールに対する違反、たとえば対象のプログラムが、プロファイルで許可していないファイルにアクセスしようとした場合、違反内容は記録されて禁止されるようになります。なお、既定では強制モードが有効になるようになっています。違反事例の記録だけを行ない、禁止させたくない場合は、不平 (complain) モードをお使いください。

18.2 免疫を与えるプログラムの判断

AppArmor に関して基本的な知識を得ることができたら、次はプロファイルを生成するアプリケーションを選択する番です。プロファイル処理が必要なプログラムは特権を取り扱うもので、下記のようにプログラムを使用するユーザはその資源へのアクセス許可を持っていないものの、そのツールの使用時にそれらへのアクセスが許可されるようなプログラムが該当します：

cron ジョブ

cron で定期的に実行されるプログラムです。このようなプログラムでは、様々な情報源から入力を受け取り、特別な権利 (時には root の権利) でプログラムを実行します。たとえば cron では /usr/sbin/logrotate を日々実行し、システムログをローテートさせたり圧縮したり、場合によってはメールで送信したりします。この種類のプログラムを発見する方法について、詳しくは 18.3 項「cron ジョブへの免疫付与」(227 ページ) をお読みください。

Web アプリケーション

Web ブラウザからのアクセスによって実行されるタイプのプログラムです。CGI Perl スクリプトや PHP のページ、もしくはさらに複雑な Web アプリケーションも該当します。この種類のプログラムを発見する方法について、詳しくは 18.4.1 項「Web アプリケーションへの免疫付与」(230 ページ) をお読みください。

ネットワークエージェント

ネットワークポートを開くプログラム (サーバとクライアント) です。メールクライアントや Web ブラウザなど、特権を取り扱うユーザ クライアントを含みます。これらのプログラムは、ユーザのホーム ディレクトリに書き込む際に特権を利用する場合があるほか、悪意を持ったリモートからの情報、たとえば Web サイトに対する攻撃や不正な コードを含む電子メールなどを受け付ける可能性があります。この種類のプログラムを発見する方法について、詳しくは 18.4.2 項「ネットワークエージェントへの免疫付与」(232 ページ) をお読みください。

その一方、特権を使用しないプログラムはプロファイルを行なう必要はありません。たとえばシェルスクリプトは cp プログラムを使用する場合があります。cp プログラムは独自の プロファイルを持たないため、呼び出し元のシェルスクリプトのプロファイルを引き継ぐこととなりますので、親のシェルスクリプトで読み込みや書き込みを許可しているファイルであれば、任意のファイルをコピーすることができます。

18.3 cron ジョブへの免疫付与

cron で実行するプログラムを調べるには、まずお使いの環境で cron の設定を調べる必要があります。残念ながら cron の設定は若干複雑な仕組みになっているため、調査対象のファイルは複数存在します。定期的な cron ジョブは、下記のファイルに書かれています:

```
/etc/crontab
/etc/cron.d/*
/etc/cron.daily/*
/etc/cron.hourly/*
/etc/cron.monthly/*
```

/etc/cron.weekly/*

また、root の cron ジョブについては `crontab -e` で編集を行なうことができるほか、`crontab -l` でジョブの一覧を表示することができます。それぞれ root になった状態で実行してください。

プログラムを見つけることができたなら、あとは *AppArmor* プロファイル ウィザードでプロファイルを作成してください。ここから先は 21.1 項「ウィザードを使用したプロファイルの追加」(261 ページ)をお読みください。

18.4 ネットワークアプリケーションへの免疫付与

プロファイル対象とすべきネットワークサーバとして、動作しているデーモンを発見する自動的な方法として、`aa-unconfined` ツールが用意されています。

`aa-unconfined` ツールは `netstat -nlp` コマンドを使用して、お使いのコンピュータの内側から接続可能なポートを調査したあと、そのポートを使用しているプログラムを判断し、読み込んだ *AppArmor* のプロファイル集を調べます。調査が終了すると、`aa-unconfined` プログラムは各プログラムとそれに関連する *AppArmor* のプロファイル (プロファイルが 無い場合は「なし」) を表示します。

注記

新しいプロファイルを作成する場合は、*AppArmor* で効率よく制限を受けられるようにするため、プロファイルされているプログラムを再起動しなければなりません。

下記は `aa-unconfined` の出力例です:

```
2325 /sbin/portmap not confined
3702❶ /usr/sbin/sshd❷ confined
    by '/usr/sbin/sshd❸ (enforce)'
4040 /usr/sbin/ntpd confined by '/usr/sbin/ntpd (enforce)'
4373 /usr/lib/postfix/master confined by '/usr/lib/postfix/master (enforce)'
4505 /usr/sbin/httpd2-prefork confined by '/usr/sbin/httpd2-prefork (enforce)'
5274 /sbin/dhcpd not confined
5592 /usr/bin/ssh not confined
7146 /usr/sbin/cupsd confined by '/usr/sbin/cupsd (complain)'
```

- ❶ 表示の冒頭は番号で始まります。この番号は待ち受けているプログラムのプロセス ID (PID) を示しています。
- ❷ 2 つめは、待ち受けているプログラムの絶対パスを表わす文字列です。
- ❸ 3 つめは、プログラムを制限しているプロファイルを示します (もしあれば)。

注記

aa-unconfined を実行するには root の権限が必要であるほか、AppArmor のプロファイルで制限されたシェルから実行してはいけません。

aa-unconfined はネットワークインターフェイスを識別することはしていないため、内部の LAN インターフェイスに対してのみ待ち受けているプロセスであっても、AppArmor で制限されていない場合は常に表示されることに注意してください。

ネットワーククライアントとして動作しているアプリケーションの検出は、お使いの環境に依存します。aa-unconfined ツールでもクライアントアプリケーションで開いているネットワークポートを検出し、報告する機能を備えていますが、aa-unconfined による分析を行なったタイミングで動作しているクライアントアプリケーションしか検出することができません。これは、ネットワークサービスは常時稼働しているのに対し、ネットワーククライアントは必要な時にしかネットワークに接続しないためです。

AppArmor のプロファイルをお使いのネットワーククライアントアプリケーションに適用する際も、お使いの環境に依存します。そのため、ネットワーククライアントに対するプロファイルについては、ユーザ側でさらに調整する余地を残した作りになっています。

デスクトップアプリケーションに対して積極的に制限を行ないたい場合、aa-unconfined コマンドに対して paranoid (偏執) オプションを設定してください。これにより動作中のすべてのプロセスとそれに対応する AppArmor のプロファイルが、各プロセスに関連しているかどうかに関わらず表示されるようになります。ユーザはその情報から、それらのプログラムに対して AppArmor のプロファイルを設定する必要があるかどうかを判断できます。

また、新しく作成したプロファイルや修正済みのプロファイルをお持ちの場合は、`apparmor-general@forge.novell.com` [<mailto:apparmor-general@forge.novell.com>] のメーリングリストに、アプリケーションの使用事例とともに送信することもできます。これにより AppArmor チームがプロファイルを分析し、openSUSE 内に取り込んでもらうことができる場合があります。お送りいただいたすべてのプロファイルを取り込む保証はできませんが、できる限り openSUSE に

同梱できるように努力させていただきます。なお、上記のメーリングリストは英語でのみサービスを提供させていただいております。あらかじめご注意ください。

上記以外にも、AppArmor のプロファイルリポジトリを使用して、作成した プロファイルを他のユーザに公開したり、他の AppArmor ユーザや開発者が作成した プロファイルをダウンロードしたりすることもできます。AppArmor のプロファイル リポジトリの使い方について、詳しくは 第20章 *AppArmor のプロファイルリポジトリ* (257 ページ) をお読みください。

18.4.1 Web アプリケーションへの免疫付与

Web アプリケーションを見つけるには、お使いの Web サーバの設定を調査するところから始まります。Apache Web サーバは高度な設定を行なうことができるシステムであるため、Web アプリケーションを設定に応じて様々な場所に 配置することができます。openSUSE では、Web アプリケーションの既定の 配置場所として `/srv/www/cgi-bin/` というディレクトリが 用意されています。なお、最大限の保護を行なう目的から、それぞれの Web アプリケーションに対して AppArmor のプロファイルを用意しておくことをお勧めします。

これらのプログラムを見つけることができれば、AppArmor の *AppArmor* プロファイル ウィザードを使用して、それらに対するプロファイルを作成します。詳しくは 21.1 項「ウィザードを使用したプロファイルの追加」(261 ページ) をお読みください。

CGI プログラムは Apache Web サーバから実行される仕組みであるため、Apache 自身に 対するプロファイル `usr.sbin.httpd2-prefork` (openSUSE 用の Apache2 の場合) を編集し、これら CGI プログラムに対する実行許可を追加しなければなりません。たとえば `/srv/www/cgi-bin/my_hit_counter.pl rpx` を追加すると、Apache に対して `my_hit_counter.pl` の実行を許可し、`my_hit_counter.pl` に対する独自のプロファイルを適用するように 指定することになります。`my_hit_counter.pl` に対する独自の プロファイルが存在しない場合、`/srv/www/cgi-bin/my_hit_counter.pl rix` は `usr.sbin.httpd2-prefork` のプロファイルを継承して使用する 意味になります。

上記のように、Apache が実行する可能性のある各 CGI スクリプトに対して実行許可を 指定するほうが便利な場合があるほか、CGI スクリプト全体に対して許可を設定する 方法もあります。たとえば `/srv/www/cgi-bin/*.{pl,py,pyc} rix` という行を追加すると、`/srv/www/cgi-bin/` ディレクトリ以下に 存在し、`.pl` (Perl スクリプト)、`.py`、`.pyc` (Python スクリプト) で終わる全てのファイルに対して、実行許可を与えることになります。上記の例ではルールに `ix` と書いていますが、これは

Python スクリプトに対して、独自のプロファイルが存在 しない場合、Apache 側のプロファイルを継承する意味になります。

注記

Web アプリケーションが Apache のモジュール (mod_perl や mod_php) で処理されている環境で、サブプロセス制限 モジュール (apache2-mod-apparmor) の機能を使用したい場合は、YaST やコマンドラインでプロファイルを追加する際、ChangeHat (ハット変更) 機能を利用してください。サブプロセスの制限に対して、さらに詳しく知るには 23.1 項「Apache のハット変更」(308 ページ) をお読みください。

mod_perl や mod_php を使用する Web アプリケーションのプロファイルは、通常とは少し異なる方法で行なう必要があります。この場合、「プログラム」はモジュールで直接解釈され Apache 内で実行されるスクリプトのことを指すため、プロセスの起動は行なわれ なくなります。その代わり、Apache の AppArmor バージョンが、要求された URI の 名前に応じてサブプロファイル (「ハット」) を利用するよう、change_hat() を呼び出します。

注記

実行するスクリプトに対する名前は、URI と異なる場合があります。これは Apache のモジュールスクリプトの参照先に依存して決まります。Apache で スクリプトの場所を異なる場所に指定した場合、AppArmor がアクセス違反を報告 する際に異なる名前が表示されます。詳しくは 第25章 *プロファイルを作成したアプリケーションの管理* (321 ページ) をお読みください。

mod_perl と mod_php を利用して 実行されるスクリプトの場合、これはリクエストされた Perl スクリプトや PHP ページの名前になります。たとえば下記のサブプロファイルでは、localtime.php のページに対して実行を許可し、ローカルの システム時刻へのアクセスを許可します:

```
/usr/bin/httpd2-prefork {
# ...
~/.cgi-bin/localtime.php {
    /etc/localtime                r,
    /srv/www/cgi-bin/localtime.php r,
    /usr/lib/locale/**            r,
}
}
```

サブプロファイルを何も指定しない場合は、Apache の AppArmor バージョンは DEFAULT_URI のハットを適用します。このサブ プロファイルは基本的に HTML の

Web ページを表示するには十分な設定に なっているはずのものです。AppArmor が既定で提供する DEFAULT_URI ハットは下記のとおりです:

```
^DEFAULT_URI {  
    /usr/sbin/suexec2                mixr,  
    /var/log/apache2/**              rwl,  
    @{HOME}/public_html              r,  
    @{HOME}/public_html/**          r,  
    /srv/www/htdocs                  r,  
    /srv/www/htdocs/**              r,  
    /srv/www/icons/*. {gif, jpg, png} r,  
    /srv/www/vhosts                  r,  
    /srv/www/vhosts/**              r,  
    /usr/share/apache2/**            r,  
    /var/lib/php/sess_*              rwl }
```

Apache で処理される全ての Web ページと CGI スクリプトに対して、単一の AppArmor プロファイルを使用する場合、DEFAULT_URI サブプロファイルを編集するのが最も良い方法です。

18.4.2 ネットワークエージェントへの免疫付与

プロファイルすべきネットワークサーバデーモンやネットワーククライアント (たとえば fetchmail, Firefox, Amarok, Banshee など) をを見つけるには、まずお使いのマシンにおけるポートの使用状況を調べる必要があります。その後 それらのポートを待ち受けているプログラムを調査し、それらに対して可能な 限りプロファイルを設定します。ネットワークポートを開いている全てのプログラム に対してプロファイルを設定すると、攻撃者は AppArmor のプロファイルポリシーを 介することでしかシステムにアクセスできないようになります。

お使いのサーバで開いているポートを調べるには、マシンの外側から実施するには スキャナ (nmap など) を使用できるほか、マシン内から実施する場合は netstat --inet -n -p コマンドを利用することで調べる ことができます。その後、検出された各ポートに対して、応答しているプログラムが 何であるのかを調べます。

ヒント

利用可能なオプションについての説明は、netstat の マニュアルページをお読みください。

プロファイルの構成と文法

アプリケーションの動作を制限するために AppArmor のプロファイルを構築する作業は、非常に直感的に行なうことができます。また、AppArmor にはプロファイル作成のためのツールが複数個同梱されているため、プログラミングやスクリプト作成の知識が無くても、プロファイルを作成できるようになっています。管理者がやるべき作業は、セキュリティを高める目的で各アプリケーションに対し、もっとも厳しいアクセス許可や実行許可を設定するよう、ポリシーの判断を行なうだけです。

アプリケーションのプロファイルを更新したり修正したりする作業は、ソフトウェアの設定や必要とする動作範囲の変更があった場合にのみ必要となります。AppArmor では、プロファイルの更新や修正を扱うための直感的なツールも提供しています。

プロファイル対象のプログラムを選択した後は、AppArmor のプロファイルを構築する作業を行ないます。これには、プロファイルの構成や文法を理解することが重要です。AppArmor のプロファイルには複数のブロックが含まれ、シンプルで再利用性の高いプロファイルが構築できるような仕組みになっています：

#include ファイル

#include ステートメントは、他の AppArmor プロファイルからの情報を引き出すためのもので、プロファイル構造を共通化して単純化するために使用します。

アブストラクト (概要)

アブストラクト (概要) は複数の #include ステートメントから構成されるもので、一般的なアプリケーションでの作業をまとめたものです。

プログラムチャンク

プログラムチャンクも複数の `#include` ステートメントから 構成されるものですが、特定のプログラムスイート固有のプロファイル集を 含んでいるものです。

機能項目

機能項目とは POSIX.1e Linux ケーパビリティに対するプロファイル項目で、制限されたプロセスから特権の必要なシステムコールを呼び出す際、どのような 呼び出しを許可するのかを詳しく制御することができるものです。

ネットワークアクセス許可項目

ネットワークアクセス許可項目は、アドレスの種類やファミリーをベースにして、ネットワークアクセスを制御するための項目です。

ローカル変数設定

ローカル変数は、パスへのショートカットを設定するためのものです。

ファイルアクセス制御項目

ファイルアクセス制御項目は、アプリケーションがアクセスできるファイルを 指定するためのものです。

rlimit 項目

rlimit 項目は、アプリケーションのリソースを制限するための制御を行なう 項目です。

プロファイルを行なうべきアプリケーションの判断については 18.2項「免疫を与えるプログラムの判断」(226 ページ) をお読みください。また、YaST を利用した場合の AppArmor プロファイルの作成 方法については、第21章 *YaST を利用したプロファイルの構築と管理* (259 ページ) をお読みください。AppArmor のコマンドラインインターフェイスを利用した場合の AppArmor プロファイルの作成 方法については、第22章 *コマンドラインからのプロファイル構築* (279 ページ) をお読みください。

19.1 AppArmor プロファイルの構成

プロファイルにどのような項目が含まれていて、どのように作成すべきなのかを説明するのに最も簡単な方法は、サンプルのプロファイルについて詳細を説明することです。今回は `/usr/bin/foo` という名前のアプリケーションがあった場合を想定して説明します:

```
#include <tunables/global>❶
```



```

# (対象のアプリケーションに関する説明)
/usr/bin/foo②
{③
    #include <abstractions/base>④

    capability setgid⑤,
    network inet tcp⑥,

    link /etc/sysconfig/foo -> /etc/foo.conf,⑦
    /bin/mount                ux,
    /dev/{,u}③random           r,
    /etc/ld.so.cache          r,
    /etc/foo/*                 r,
    /lib/ld-*.so*             mr,
    /lib/lib*.so*             mr,
    /proc/[0-9]**             r,
    /usr/lib/**               mr,
    /tmp/⑧                     r,
    /tmp/foo.pid              wr,
    /tmp/foo.*                lrw,
    /@{HOME}⑩/.foo_file        rw,
    /@{HOME}/.foo_lock        kw,
    owner⑪ /shared/foo/**    rw,
    /usr/bin/foobar           cx,⑫
    /bin/**                   px -> bin_generic,⑬

    # foo のローカル (子となる) /usr/bin/foobar 向けプロファイル。

    profile /usr/bin/foobar⑭ {
        /bin/bash             rmix,
        /bin/cat               rmix,
        /bin/more              rmix,
        /var/log/foobar*       rwl,
        /etc/foobar            r,
    }

    # foo のハット "bar"
    ^bar⑮ {
        /lib/ld-*.so*          mr,
        /usr/bin/bar           px,
        /var/spool/*           rwl,
    }
}

```

- ① 変数宣言を含むファイルを読み込んでいます。
- ② 制限されるべきプログラムのフルパスを指定しています。
- ③ 中括弧 ({}) は、include ステートメント／サブプロファイル／パス項目／機能項目／ネットワーク項目に対するコンテナとして動作する記号です。
- ④ このディレクティブでは、プロファイルを簡略化するために AppArmor の プロファイルコンポーネントを取り込んでいます。

- ⑤ POSIX.1e ドラフトで規定されているケーパビリティ (capability) を有効にする設定です。
- ⑥ アプリケーションに対してネットワークアクセスを許可する ディレクティブです。詳しくは 19.5項「ネットワークアクセス制御」(241 ページ) をお読みください。
- ⑦ ソースとリンクの宛先を指定する、リンク対ルールです。詳しくは 19.7.6項「リンク対」(247 ページ) をお読みください。
- ⑧ ここでの中括弧 ({}) は、カンマで区切られたそれぞれの文字列について、これら両方のパターンをルールに含める意味になります。
- ⑨ プログラムに対して、ファイルシステムのどの領域へのアクセスを許可するかを指定しています。最初の項目として指定するのはファイルに対する絶対パスで、正規表現による一括指定も行なうことができます。2 つめの項目は許可する アクセスモード (たとえば *r* は読み込みを、*w* は書き込みを、*x* は実行をそれぞれ 表わします) です。スペース類 (スペースまたはタブ) はファイル名の前にも 入力できるほか、ファイル名とアクセスモードの間にも入力できます。複数の アクセスモードを指定する場合のスペースやカンマは、入力してもしなくても かまいません。利用可能なアクセスモードについて、詳しい説明は 19.7 項「ファイルのアクセス許可とアクセスモード」(245 ページ) をお読みください。
- ⑩ ここで指定している変数は値に展開されるもので、プロファイル全体を変更することなく複数の設定を変更できるようになっています。
- ⑪ 所有者条件ルールと呼ばれるもので、自分が所有するファイルへの読み書き許可を 与える意味になります。詳しくは 19.7.7項「所有者条件ルール」(247 ページ) をお読みください。
- ⑫ この項目は、`/usr/bin/foobar` のローカルプロファイルへの 遷移を指定しています。利用可能な実行モードについて、詳しくは 19.8項「実行モード」(248 ページ) をお読みください。
- ⑬ 名前付きのプロファイル遷移を指定しています。グローバルスコープにある `bin_generic` というプロファイルへの遷移を指定しています。詳しくは 19.8.7 項「名前付きプロファイル遷移」(251 ページ) をお読みください。
- ⑭ ここではローカルプロファイル `/usr/bin/foobar` を宣言しています。
- ⑮ このセクションでは「ハット」と呼ばれる、アプリケーションの サブプロファイルを読み込む指定を行なっています。AppArmor のハット変更機能 について、詳しくは 第23章 *ハット変更を利用した Web アプリケーションのプロファイル作成* (307 ページ) をお読みください。

プログラムに対してプロファイルが作成されると、プログラムはプロファイルに 書かれたファイルやモード、POSIX ケーパビリティでのアクセスだけが許可される ようになります。これらの制限は Linux に元から備わっているアクセス制御機能に 追加される形で行なわれます。

例: たとえばカーパビリティ CAP_CHOWN を利用した処理を行なう場合、通常の Linux アクセス制御における CAP_CHOWN の権限 (通常は root) と、プロファイル内での chown 許可の両方が必要になります。同様に、プログラムが /foo/bar ファイルに書き込むには、ファイルのパーミッションで指定されたアクセス許可 (それぞれ chmod と chown のマニュアルページをお読みください) が必要になるほか、それに加えてプロファイル内に /foo/bar w が書かれている必要があります。

AppArmor のルールに対する違反情報は、audit パッケージがインストールされている場合は /var/log/audit/audit.log ファイルに、インストールされていない場合は /var/log/messages ファイルにそれぞれ書き込まれます。多くの場合、AppArmor のルールは攻撃に必要なファイルへのアクセスが禁止されることによって防ぐことができるほか、少なくとも AppArmor の制限は AppArmor が許可するファイルにしかアクセスさせない 仕組みであるため、ダメージを軽減することができます。

19.2 プロファイルの種類

AppArmor では 4 種類のプロファイルが用意されています。それぞれ標準プロファイル、独立プロファイル、ローカルプロファイル、ハットの 4 つです。標準プロファイルと独立プロファイルは単独で動作するプロファイルで、それぞれ /etc/apparmor.d/ ディレクトリ内に配置されます。ローカル プロファイルとハットは、親となるプロファイルに含まれる形で存在するもので、アプリケーション内の特定処理について、より厳しい制限や代替の制限を実施するために使用します。

19.2.1 標準プロファイル

既定の AppArmor プロファイルは指定した名前のプログラムに対して適用されるものであるため、プロファイル名は制限を行なうアプリケーションに対するパスを指定しなければなりません。

```
/usr/bin/foo {  
...  
}
```

このプロファイルは、制限を受けていないプロセスが /usr/bin/foo を実行すると、自動的に使用されます。

19.2.2 独立プロファイル

独立プロファイルは、プロファイル名がファイルシステムのパス名では書かれていないプロファイルのことで、これにより自動ではアプリケーションに割り当てられることはありません。独立プロファイルの名前は `profile` というキーワードで始まります。プロファイル名は自由に命名することができますが、下記の制限に従う必要があります: 名前は `:` や `.` の文字で始まってはならないこと。名前に空白を含む場合は、引用符を付けて記すこと。名前が `/` で始まる場合は 標準プロファイルとして扱われること。そのため、下記に示す 2 種類の プロファイルは、同じ意味になります:

```
profile /usr/bin/foo {  
...  
}  
/usr/bin/foo {  
...  
}
```

独立プロファイルは自動では使用されることがありませんし、`px` のルールを介して遷移することもあります。独立 プロファイルを使用するには、名前付きプロファイル遷移 (詳しくは 19.8.7項「名前付きプロファイル遷移」(251 ページ) をお読みください) または `change_profile` ルール (詳しくは 19.2.5項「ルール変更」(239 ページ) をお読みください) を使用する必要があります。

独立プロファイルは、通常システム全体のプロファイルでは制限できないようなシステムユーティリティ (たとえば `/bin/bash` など) に対して、特殊なプロファイルを設定する際に使用します。また、役割やユーザを 制限する際にも使用することができます。

19.2.3 ローカルプロファイル

ローカルプロファイルは、制限下にあるアプリケーションから起動するユーティリティプログラムに対して、特殊な制限を行なうための便利な方法です。標準プロファイルのような形でプロファイルを指定するものの、それらの制限は親となるプロファイルに 組み込まれて動作します。また `profile` というキーワードを 付けるのも特徴です:

```
/parent/profile {  
...  
    profile local/profile {  
        ...  
    }  
}
```

ローカルプロファイルへの遷移を行なうには、`cx` (詳しくは 19.8.2項「独立ローカルプロファイル実行モード (`cx`)」(249 ページ) をお読みください) ま

たは名前付きプロファイル遷移 (詳しくは 19.8.7項「名前付きプロファイル遷移」(251 ページ) をお読みください) を使用します。

19.2.4 ハット

AppArmor の "ハット" とは、ローカルプロファイルに対していくつかの制限を付けたもので、それらへの遷移時に `change_hat` と呼ばれる間接的な ルールを使用するものです。詳しくは 第23章 *ハット変更を利用した Web アプリケーションのプロファイル作成* (307 ページ) をお読みください。

19.2.5 ルール変更

AppArmor では `change_hat` (ハット変更) と `change_profile` (プロファイル変更) というルールが用意されていて、これらは領域の遷移を制御するのに使用します。 `change_hat` はプロファイル内でハットを定義することで 指定するもの、 `change_profile` ルールは他のプロファイルを 参照する際に使用し、 `change_profile` というキーワードで 書き始めます:

```
change_profile /usr/bin/foobar,
```

`change_hat` と `change_profile` のいずれ ともアプリケーション主導のプロファイル 遷移機能を提供するもので、個別の アプリケーションを立ち上げたりする必要はありません。 `change_profile` は、既に読み込まれている任意のプロファイルから一方 向の遷移機能を提供する のに対して、 `change_hat` は親子型で復帰可能な遷移機 能を 提供し、アプリケーションが親のプロファイルから子のハットに遷移した後、 正しい機密鍵が提供されていれば、後から親のプロファイルに戻ってくることが できるようになっています。

`change_profile` はアプリケーションの設定段階では信頼して おき、あとから権限 レベルを落としたいような場合にベストな選択です。起動 フェーズでマップされているか、もしくは開いている任意のリソースは、プロファイル 変更が発生しても変わらずアクセス可能ですが、新しいプロファイルではリソース を開く際の制限を規定することになるほか、切り替え前から開いていたいくつかの リソースを制限することも行ないません。特にメモリ資源については、機能設定や ファイルリソースが制限されていても (それらがメモリマップ型のものでない 限り)、以前と変わらずアクセスできるようになります。

`change_hat` は対象のアプリケーションを仮想マシンで動作 させるような場合や、アプリケーション資源に対する直接アクセスを提供しない ようなインタプリタ (たとえば `mod_php` など) を動作させる ような場合にベストな選択です。 `change_hat` はア

アプリケーションのメモリ内に戻り用の機密鍵を保存するので、制限された権限のもとではメモリへの直接アクセスを行なうことができなくなります。ファイルアクセスについても、ハットは閉じられていないファイルハンドルに対して制限を行なうことができるので、同様に適切に区切ることができます。なお、アプリケーションが何らかのバッファ処理を行なっていたり、バッファリング機能のあるファイルアクセス機能を提供していたりするような場合は、これらのバッファリングされたファイルへのアクセスはカーネル側では検知できず、新しいプロファイルでは制限できないことに注意してください。

警告: 領域遷移の安全性について

`change_hat` と `change_profile` の領域遷移は、`exec` を利用した領域遷移ほど安全性が高くないことに注意してください。これは、これらの領域遷移はメモリマッピングに対して作用することができないほか、既に開いているリソースを閉じることもしないためです。

19.3 #include ステートメント

`#include` ステートメントは他の AppArmor プロファイルにある内容を引き出すためのディレクティブで、プロファイルの単純化のために使用します。`#include` ファイルではプログラムに対するアクセス許可を取り出すもので、これを使用すると、他のプログラムでも同様に設定する必要のある共通的なアクセス許可をまとめることができます。これにより、プロファイル全体のサイズを小さくすることにもつながります。

既定では AppArmor は、`#include` に書かれたファイルに対して `/etc/apparmor.d` のパスを設定します。つまり、取り込む `#include` で取り込むファイルがあれば、それを `/etc/apparmor.d` 内にあると期待する意味になります。他のプロファイルステートメントとは異なり (C 言語のプログラムのように)、`#include` の行末はセミコロンで終わることはありません。

お使いのアプリケーションのプロファイル作業を支援する目的で、AppArmor では `#include` に対して 3 種類の分類を用意しています。それぞれ アブストラクトとプログラムチャンク、チューナブルと呼ばれます。

19.3.1 アブストラクト (概要)

アブストラクトは一般的なアプリケーション処理を `#include` でまとめたものを指します。これらの処理には、認証機構へのアクセスやネーム サービスへのアクセス、一

一般的なグラフィック環境の使用やシステムアカウンティングなどの処理が含まれます。これらアブストラクト内に一覧表示されているファイルは、いずれも特定の処理に固有のものです。アブストラクトのうちの1つを必要とするプログラムでは、通常は他のアブストラクトファイルに書かれた他のファイルへのアクセスも必要となります (ローカル側の設定のほか、プログラムの要件に依存します)。アブストラクトは `/etc/apparmor.d/abstractions` 内に配置されています。

19.3.2 プログラムチャンク

プログラムチャンクのディレクトリ (`/etc/apparmor.d/program-chunks`) には、プログラムスイート固有の複数のプロファイル部品 (チャンク) が含まれていて、スイートの外側では有益ではないものになっています。そのため、プロファイル ウィザード (aa-logprof や aa-genprof) などから、この中にあるプロファイルの使用は提案されることはありません。現時点では postfix プログラムスイート 向けのプログラムチャンクのみ利用できます。

19.3.3 チューナブル

チューナブル (`/etc/apparmor.d/tunables`) のディレクトリ には、いくつかのグローバル変数定義が書かれています。プロファイル内で使用する 場合、これらの変数はプロファイル全体を変更することなく値を変更できる仕組みになっています。すべてのプロファイルから利用できるチューナブル定義は、`/etc/apparmor.d/tunables/global` に配置します。

19.4 機能項目 (POSIX.1e)

機能項目とは、単純に POSIX.1e で規定されている ケーパビリティ を設定するためのものです。詳しくは `capabilities(7)` のマニュアルページをお読みください。

19.5 ネットワークアクセス制御

AppArmor では、アドレスの種類やファミリを利用して、ネットワークアクセスを 制御することができます。下記にはネットワークアクセスルールの文法を示します:

```
network [[<domain>❶][<type>❷][<protocol>❸]]
```

- ❶ サポートされるネットワーク領域: inet, ax25, ipx, appletalk, netrom, bridge, x25, inet6, rose, netbeui, security, key, packet, ash, econet, atmshvc, sna, irda, pppox, wanpipe, bluetooth
- ❷ サポートされるネットワークタイプ: stream, dgram, seqpacket, rdm, raw, packet
- ❸ サポートされるプロトコル: tcp, udp, icmp

なお、AppArmor ツールはファミリとタイプの仕様のみに対応しています。また、AppArmor のモジュールは「access denied」(アクセスが拒否されました) のメッセージで network ドメイン タイプ のメッセージのみを出力します。これらはプロファイルの生成ツールである YaST とコマンドラインの両方で 出力させることができます。

下記では、AppArmor のプロファイルで使用されるネットワーク関連のルールについて、例を示しています。ただし、最後の 2 つのルールの文法は、現時点では AppArmor ツールで対応していないことに注意してください。

```
network❶,  
network inet❷,  
network inet6❸,  
network inet stream❹,  
network inet tcp❺,  
network tcp❻,
```

- ❶ すべてのネットワーク処理を許可します。ドメインやタイプ、プロトコルに対して、制限はありません。
- ❷ 一般的な IPv4 ネットワーク処理を許可します。
- ❸ 一般的な IPv6 ネットワーク処理を許可します。
- ❹ IPv4 の TCP ネットワーク処理を許可します。
- ❺ IPv4 の TCP ネットワーク処理を許可します。上と同じ意味です。
- ❻ IPv4, IPv6 両方の TCP ネットワーク処理を許可します。

19.6 パスとグロブ

AppArmor ではディレクトリのパス名とファイルのパス名を明示的に区別します。ディレクトリパスを指定する場合は、末尾に / を付けて 明示的に区別してください:

```
/some/random/example/* r  
/some/random/example ディレクトリ内に あるファイルに対して、読み込みアクセスを許可します。
```

```
/some/random/example/ r  
ディレクトリに対してのみ読み込みアクセスを許可します。
```


`/some/**/ r`
/some ディレクトリ以下の任意のサブ ディレクトリに対して、読み込みアクセスを許可します。

`/some/random/example/** r`
/some/random/example 以下にある 任意のファイルやディレクトリに対して、読み込みアクセスを 許可します。

`/some/random/example/**[^/] r`
/some/random/example ディレクトリ以下に ある任意のファイルに対して、読み込みアクセスを許可します。ディレクトリについては明示的に除外 (`[^/]`) しています。

グロブ (または正規表現のマッチング) は、ワイルドカードを利用してディレクトリ パスを修正し、複数のファイルやサブディレクトリをまとめて指定したい場合に 使用します。ファイルリソースの指定では、`csh`, `bash`, `zsh` などの有名なシェルで 利用できるグロブ文法を利用できます。

<code>*</code>	<p>/ を除く、任意の長さの文字列に置き換えることができます。</p> <p>例: ファイルパス要素の指定などに使用します。</p>
<code>**</code>	<p>/ を含む、任意の長さの文字列に置き換えることができます。</p> <p>例: ディレクトリ全体を含む、パス要素の任意箇所をまとめて表記できます。</p>
<code>?</code>	<p>/ を除く、任意の 1 文字に置き換えることができます。</p>
<code>[abc]</code>	<p>a, b, c のうちの、いずれか 1 文字に置き換えることができます。</p> <p>例: <code>/home[01]/*/.plan</code> というルールがあった場合、<code>/home0</code> と <code>/home1</code> の両方の ディレクトリ以下にある</p>

	.plan ファイルが該当する ことになります。
[a-c]	a, b, c のうちの、いずれか 1 文字に置き換えることができます。
{ab, cd}	ab または cd のどちらかに 置き換えることができます。 例: /{usr, www}/pages/** というルールがあった場合、/usr/pages と /www/pages の両方のディレクトリ以下にある Web ページにアクセスを許可する意味に なります。
[^a]	a を除く任意の 1 文字に置き換えることができます。

19.6.1 プロファイル内での変数の使用

AppArmor では、プロファイル内でパスを保持するのに変数を使用することができます。グローバル変数はお使いのプロファイルに対して可搬性を与えるのに、ローカル 変数はパスに対するショートカットとして使用できます。

グローバル変数が便利になる一例として、ユーザのホームディレクトリが 異なる場所にマウントされるネットワークシナリオを考えてみます。影響する すべてのプロファイルに対してホームディレクトリのホームディレクトリを 書き換える代わりに、変数の値だけを書き換えれば作業が終わります。グローバル変数は /etc/apparmor.d/tunables ディレクトリ以下で定義し、#include ステートメント 経由で利用します。この使用例 (@{HOME} と @{HOMEDIRS}) での変数定義は、/etc/apparmor.d/tunables/home ファイル内に 書かれています。

対して、ローカル変数はプロファイルの冒頭で定義します。これはたとえば、chroot 環境下のパスに対して、ベースを設定する際に便利です:

```
@{CHROOT_BASE}=/tmp/foo
/sbin/syslog-ng {
...
# chrooted applications
@{CHROOT_BASE}/var/lib/*/dev/log w,
@{CHROOT_BASE}/var/log/** w,
```

```
...  
}
```

注記

現在の AppArmor ツールでは、変数は手作業でのプロファイル編集および管理でのみ 使用することができます。

19.6.2 別名ルール

別名ルールは、プロファイルパスをサイト固有のレイアウトにマッピングする ための代替手段です。変数を使用してパスの書き換えを行なう方法もありますが、こちらは変数を値に置き換えた後に作用します:

```
alias /home/ -> /mnt/users/
```

注記

現在の AppArmor ツールでは、変数は手作業でのプロファイル編集および管理でのみ 使用することができます。また、無効化するとルールが適用されなくなります。別名ルールの定義は、`/etc/apparmor.d/tunables/alias` ファイルを編集して行ないます。

19.7 ファイルのアクセス許可とアクセスモード

ファイルのアクセス許可は、下記のモードの組み合わせで表現します:

r	読み込みモード
w	書き込みモード (a モードとは相互に排他関係)
a	追記モード (w モードとは相互に排他関係)
k	ファイルロック (施錠) モード

l	リンクモード
link ファイル -> 宛先	リンク対ルール (他のアクセスモードとは同時に指定できません)

19.7.1 読み込みモード (r)

プログラムに対して、特定のリソースへの読み込みアクセスを許可します。読み込みアクセスは、シェルスクリプトやその他のインタプリタ型言語を利用する場合に必要となるほか、実行中のプロセスがコアダンプを出力できるかどうかを判断する際にも必要となります。

19.7.2 書き込みモード (w)

プログラムに対して、特定のリソースへの書き込みアクセスを許可します。ファイルをアンリンク (削除) する場合、この許可が必要となります。

19.7.3 追記モード (a)

プログラムに対して、特定のファイルへの追記を許可します。w モードとは異なり、追記モードではデータの上書きやファイル名の変更、ファイルの削除を行なうことができません。追記のアクセス許可は一般に、ログファイルへの書き込み許可が必要なアプリケーションに対して付与されるもので、既存のログ情報を操作させたくない場合に設定します。追記のアクセス許可は単純に書き込みモードのサブセットであるため、w と a の許可フラグを同時に指定することはできません (相互に排他の関係です)。

19.7.4 ファイルロック (施錠) モード (k)

アプリケーションがロック (施錠) を行なうことができますようにします。AppArmor の従来バージョンでは、アプリケーションから特定のリソースへアクセスする許可があれば、ロックは特に何もすることなく許可されてきました。個別のファイルロックモードを使用することで、AppArmor はロックを行なう必要のあるファイルに対してのみロックを許可し、サービス拒否攻撃 (DoS) のような状況に陥らないようにしています。

19.7.5 リンクモード (l)

リンクモードはハードリンクへのアクセスを制御するものです。リンクを作成すると、リンク先のファイルと作成したリンクは同じアクセス許可を持つことになります (ただし、リンク先でリンクアクセスの許可が必要になることはありません)。

19.7.6 リンク対

リンクモードは、任意のファイルに対してリンクを作成する許可を与えるもので、リンクはリンク先のアクセス許可に対するサブセットの形になります (アクセス 許可テストのサブセット)。リンク対では、リンク元とリンク先を指定することで、どのような形でハードリンクの作成を許可するのかを制御することができます。リンク対ルールは既定では、標準のリンク許可モードが必要とするようなアクセス 許可テストのサブセットを実施しません。このテストを強制するには、subset キーワードを使用します。それぞれ下記に示す ルールは等価な意味を持ちます:

```
/Link l,  
Link subset /link -> /**,
```

注記

現時点では、リンク対ルールは YaST やコマンドラインツールで利用することはできません。これらを使用する際は、プロファイルを手作業で編集してお使いください。リンク対ルールのあるプロファイルでも、これらのツールを利用した 更新であれば問題なく動作します。これは、リンク対ルールについてはツール側で 一切処理されず、そのままにされるためです。

19.7.7 所有者条件ルール

ファイルルールを拡張して、対象のファイルに対して所有者であった場合 (fsuid がファイルの uid と一致した場合) のルールを設定することができます。これを 設定するには、ルールの前に owner というキーワードを設定 します。所有者条件ルールは通常のファイルルールとして動作します。

```
owner /home/*/** rw
```

リンクルールと所有者条件ルールを同時に利用すると、所有者の確認はリンク先となるファイルに対して実施され、リンクを行なうには、リンク先のファイルに対して 所有者でなければなりません。

注記: 通常のファイルルールとの関係性について

所有者条件ルールは、通常のファイルルールに対するサブセットとして扱われます。通常のファイルルールが所有者条件ルールと重複すると、結果として適用されるルールは通常のファイルルールと同じものになります。

19.7.8 拒否ルール

拒否ルールは通知を行なう拒否を設定することができるほか、通知を行なわない拒否を設定することもできます。プロファイル生成ツールでは、拒否ルールの取り扱いとして既知の拒否かどうかを尋ねることはありません。このような拒否を設定したい場合は、拒否時の監査ルールについても表示されることはなく、ログファイルの肥大化を防いでいます。このような動作がお望みのものでない場合は、拒否項目に対して `audit` というキーワードを追加してください。

拒否ルールは、許可ルールと混在させて使用することもできます。これにより、幅の広い許可ルールを設定しておいて、その中から部分的に許可されないファイルだけを除外するような使い方ができます。拒否ルールは所有者ルールとも混在させ、自分自身が所有するファイルに対して、アクセスを拒否させることができます。たとえば下記の例では、ユーザのディレクトリにあるすべてのファイルに対して読み書きアクセスを許可するが、`.ssh/` 以下にあるファイルに対しては例外的にアクセスを拒否するルールです：

```
deny /home/*/.ssh/** w,  
/home/*/** rw,
```

拒否ルールを多用すると、プロファイルの動作を理解するのが難しくなってしまうことから、お勧めできません。ただし、拒否ルールを賢く使用することで、プロファイルを単純化することができます。そのため、ツール類では特定のファイルに対する拒否ルールのみを生成し、グロブを利用した拒否ルールは作成しないようになっています。グロブを利用して拒否ルールを記述したい場合は、手作業でプロファイルを編集し、設定してください。これは、拒否ルールについてはツール側で一切処理されず、そのままにされるためです。

19.8 実行モード

実行モードは、名前付きプロファイル遷移とも呼ばれ、下記のモードが含まれます：

px	独立プロファイル実行モード
----	---------------

cx	独立ローカルプロファイル実行モード
ux	無制限実行モード
ix	継承実行モード
m	mmap(2) での PROT_EXEC の許可

19.8.1 独立プロファイル実行モード (px)

このモードは AppArmor の領域遷移の際、実行中のリソースに対して独立した セキュリティプロファイルを必要とします。プロファイルが定義されていない 場合は、アクセスが拒否されます。

警告: 独立プロファイル実行モードの使用について

px は LD_PRELOAD のような環境変数を 取り除く処理は行ないません。そのため、呼び出し側の領域は呼び出された 領域からの影響を大きく受けることになります。

なお、Ux, ux, Px, ix とはそれぞれ互換性がありません。

19.8.2 独立ローカルプロファイル実行モード (cx)

px のようにグローバルのプロファイルセットを検索する 代わりに、cx では現在のプロファイルが属するローカル プロファイルのみを検索対象とします。このプロファイル 遷移は、ヘルパー アプリケーション向けに代替プロファイルを設定するのに使用 することができます。

注記: 独立ローカルプロファイル実行モード (cx) の制限について

現時点では cx の遷移はトップレベルのプロファイルに限定されていて、ハットや 子プロファイル内で使用することはできません。この制限は将来取り除かれる予 定 です。

Ux, ux, Px, px, Cx, ix とは それぞれ互換性がありません。

19.8.3 無制限実行モード (ux)

プログラムに対して、AppArmor プロファイルを全く適用せずにリソースを 実行できるようにします。このモードは、制限を受けているプログラムが コンピュータの再起動など、特権操作を必要となる場合に便利な機能です。他の実行ファイル内で特権付きのセクションを用意して、そこに対して無制限の 実行権利を与えると、すべての制限付きプロセスに対して設定されていた、必須の制限を迂回することができるようになります。制限内容について、詳しくは `apparmor(7)` のマニュアルページをお読みください。

警告: 無制限実行モード (ux) の使用について

ux は特別な理由がある場合にのみ利用してください。これにより、子プロセスが AppArmor の保護無しで動作することになってしまいます。また、ux は LD_PRELOAD のような環境 変数を取り除く処理も行ないません。そのため、呼び出し側の領域は呼び出された 領域からの影響を大きく受けることになります。さらに、このモードは無制限 での実行をどうしても必要とする状況で、LD_PRELOAD の環境 変数を使用しなければならない場合に限って使用してください。このモードを使用したプロファイルは、ほとんどセキュリティが担保されないため、ご自身の 責任範囲のもとでご利用ください。

このモードは、Ux, px, Px, ix とそれぞれ互換性がありません。

19.8.4 クリーン実行モード

クリーン実行モードは指定した名前のプログラムを px, cx, ux の各モードで実行できるように するためのものですが、AppArmor が Linux カーネルの機能である `unsafe_exec` ルーチンを呼び出す仕組みになっていて、`setuid` プログラムを実行するときのように環境変数を取り除く処理を行ないます。クリーン実行モードはそれぞれ大文字 (Px, Cx, Ux) で表わします。`setuid`, `setgid` での環境変数の取り除きについて、詳しくは `ld.so(8)` の マニュアルページをお読みください。

19.8.5 継承実行モード (ix)

ix モードは、プロファイルが用意されているプログラムが 特定の名前のプログラムを実行する際、`execve(2)` による 通常の AppArmor 領域遷移を阻害するための

ものです。通常の領域遷移の代わりに、実行されたりソースは現在のプロファイル
を継承します。

このモードは、制限付きのプログラムが他の制限付きプログラムを呼び出す際、呼
び出し先のプロファイルの許可を得たり、現在のプロファイルの許可を失ったりする
ことなく実行する必要がある場合に、有用なモードです。ix での実行は特権を変更
することはないので、環境変数を取り除く機能は用意されていません。

それぞれ cx, ux, px との互換性がありません。また、m の機能を包含します。

19.8.6 実行ファイルのマッピング許可 (m)

このモードは、mmap(2) の PROT_EXEC フラグを介してファイルをメモリ内にマッピン
グできるようにします。このフラグは 対象のページに対して実行許可を与えるもので
す。この機能は、いくつかの アーキテクチャで実行不可能なデータページを提供す
る際に使用されているもので、不正なコードを実行しにくくするために使用します。
AppArmor では、このモードを ld(1) での無効な -L フラグのほか、ld.so(8) で
の無効な LD_PRELOAD, LD_LIBRARY_PATH 設定の効果を 制限するために使用し、
正しい振る舞いをするプログラム (または実行不可能な メモリアクセス制御を強制
するアーキテクチャでの全プログラム) を制限する のに使用します。

19.8.7 名前付きプロファイル遷移

既定では、px と cx (および左記の クリーン実行モード) のプロファイル遷移を行な
う際、実行ファイルの名前で プロファイルを探します。名前付きプロファイル遷移で
は、遷移先のプロファイル 名を指定できるようになります。これは、複数の
バイナリで単一の プロファイルを共有する必要がある場合に便利な機能であるほ
か、名前以外の 方法で異なるプロファイルに遷移したほうが都合がいい場合に便
利です。名前 付きプロファイル遷移は、それぞれ cx, Cx, px, Px と同時に指定する
ことができます。なお、現時点では 1 プロファイルあたり 12 個までの名前付きプロ
ファイル遷移までしか使用できません。

名前付きプロファイル遷移では -> を使用し、移行先の プロファイル名を指定しま
す:

```
/usr/bin/foo
{
  /bin/** px -> shared_profile,
  ...
  /usr/*bash cx -> local_profile,
  ...
}
```

```
profile local_profile
{
    ...
}
}
```

注記: 通常の遷移と名前付き遷移の違いについて

グロブとともに利用した場合、通常の遷移は「一対多」の関係を構築することになります。つまり `/bin/** px` は、プログラムが実行されている環境に依存して `/bin/ping` や `/bin/cat` などの遷移を指定することになります。

名前付き遷移の場合は、「多対一」の関係を構築することになります。つまり、その名前とは関係なく、条件に該当したすべてのプログラムが指定したプロファイルに遷移することになります。

名前付きプロファイル遷移は、ログでは `Nx` というモードで表示されます。また、遷移先のプロファイル名は `name2` という項目で表示されます。

19.8.8 プロファイル遷移における継承フォールバック

`px` や `cx` の遷移には「強力な 依存関係」が存在していて、指定したプロファイルが存在しない場合は実行が失敗します。これに対して、継承フォールバックを利用すると、現在のプロファイルが適用された状態のまま実行を続けることができるようになります（つまりプロファイルを継承することになります）。継承フォールバックを指定するには、それぞれ `cx`, `Cx`, `px`, `Px` に対して `ix` を組み合わせて指定します。これにより、それぞれ `cix`, `Cix`, `pix`, `Pix` の各モードになります。フォールバックモードは名前付きプロファイル遷移でも使用することができます。

19.8.9 実行モードにおける変数設定

`Px`, `Cx`, `Ux` のいずれかの実行モードを選択した場合、子プロセスが環境を引き継ぐにあたり、下記の環境変数が取り除かれます。その結果、プロファイルに対して `Px`, `Cx`, `Ux` のいずれかの実行モードを設定すると、下記の環境変数に依存して動作するアプリケーションやプロセスは動作しなくなります：

- `GCONV_PATH`
- `GETCONF_DIR`

- HOSTALIASES
- LD_AUDIT
- LD_DEBUG
- LD_DEBUG_OUTPUT
- LD_DYNAMIC_WEAK
- LD_LIBRARY_PATH
- LD_ORIGIN_PATH
- LD_PRELOAD
- LD_PROFILE
- LD_SHOW_AUXV
- LD_USE_LOAD_BIAS
- LOCALDOMAIN
- LOCPATH
- MALLOC_TRACE
- NLSPATH
- RESOLV_HOST_CONF
- RES_OPTIONS
- TMPDIR
- TZDIR

19.9 リソース制限制御

AppArmor では、アプリケーションが使用するリソースについて制限を設定する機能が あります (rlimits や ulimits としても知られているものです)。既定では

AppArmor はアプリケーションの rlimits を制限することはなく、プロファイルに書かれた 制限のみを適用します。リソースの制限について、詳しくは `setrlimit(2)`, `ulimit(1)`, `ulimit(3)` の各マニュアルページをお読みください。

AppArmor ではシステムの rlimits を制御しますが、通常発生するような追加の監査 機能は提供しません。また、システムが設定した rlimit を上昇させることもできず、AppArmor ではその時点でのアプリケーションのリソース制限を下降させる ことだけが実施可能です。

これらの値はプロセスの子供に対して継承する仕組みになっているため、子の プロセスで新しいプロファイルに遷移した場合や、子のアプリケーションが無制限の 設定になっていた場合でも、リソース制限が残ることになります。アプリケーションが 新しいプロファイルに遷移した場合は、遷移先のプロファイルでさらなるリソース 制限を設定することも可能です。

AppArmor の rlimits ルールでは、アプリケーションのハードリミットの調停を行ない、上昇させることもできます。ただし、アプリケーションは、プロファイル内で指定されているものよりも、ハードリミットを上げることはできません。ハードリミットを 上げる調停は値の設定として継承されることはないのので、いったんアプリケーションが 新しいプロファイルに遷移すると、プロファイル内で指定された制限まで自由に 上げることができるようになります。

AppArmor のリソース制限制御は、アプリケーションのハードリミットと等しいか、もしくはより小さいソフトリミットに対して、影響を与えることはありません。

AppArmor のハードリミットのルールは、一般的には下記のような書式で記述します:

```
set rlimit 資源 <= 値,
```

それぞれ 資源 と 値 には下記の値を指定します:

cpu

現時点ではサポートされていません

fsize, data, stack, core, rss, as, memlock, msgqueue

バイト単位で指定するか、もしくは K (キロバイト), M (メガバイト), G (ギガバイト) の接尾辞を付けた数値。たとえば下記のようになります:

```
rlimit data <= 100M,
```

fsize, nofile, locks, sigpending, nproc*, rtprio

0 以上の整数

nice

-20 から 19 までの値

* nproc のリソース制限はその他のリソース制限とは異なる方法で処理されます。標準のプロセスリソース制限の代わりに、一時点で 特定のプロファイル下で実行している、最大のプロセス数として制限されます。いったん制限を超過すると、実行中のプロセス数が減るまでの間、プロファイル 適用下での新規プロセス生成は失敗するようになります。

注記

現時点では、プロファイルに対してリソース制限ルールを追加するにあたって、ツールを利用することはできません。リソース制限ルールを追加するには、テキストエディタなどを利用して、手作業でプロファイルを編集する必要があります。リソース制限ルールの書かれたプロファイルをツールで読み込ませて取り扱わせることもできますが、リソース制限ルールは編集されることも削除されることもあります。そのため、リソース制限ルールを含むプロファイルでも、ツールで 更新することは可能です。

19.10 監査ルール

AppArmor では特定のルールをもとに監査を行なう機能に対応していて、特定の条件に 該当した場合、監査ログ内にメッセージを出力することができます。特定のルール で監査メッセージを有効にするには、ルールの前に audit キーワードを指定します:

```
audit /etc/foo/*          rw,
```

特定の許可に対して監査だけを行ないたい場合、ルールを 2 つに分割することができます。下記の例では、ファイルが書き込み用に開かれた場合に監査メッセージが生成され、読み込み用にひらされた場合には監査メッセージが生成されないルールを示しています:

```
audit /etc/foo/* w,  
/etc/foo/*          r,
```

注記

監査メッセージは、ファイルに対して読み込みや書き込みが発生するごとに 生成されることはありません。それぞれ読み込みや書き込み用に開いた場合にのみ生成されます。

監査の設定は、所有者条件ルールと組み合わせることもできます。この場合は、自分自身が所有するファイルにアクセスした場合に監査が生成されることになります (現時点では、所有していないファイルに対して監査を生成することはできません):

```
audit owner /home/*/.ssh/**          rw,
```

AppArmor のプロファイルリポジトリ

20

AppArmor には、既定で有効化されているプロファイル集が用意されています。これらは AppArmor の開発チームが作成したもので、`/etc/apparmor.d` 内に配置されています。これ以外にも openSUSE では、アプリケーションに同梱される形で、それらアプリケーション向けの プロファイルが付属しているものもあります。これらのプロファイルは既定では 有効に設定されておらず、標準の AppArmor プロファイルとは異なるディレクトリ `/etc/apparmor/profiles/extras` に配置されます。

20.1 ローカルリポジトリの使用

AppArmor ツール (YaST や `aa-genprof`, `aa-logprof` など) では、ローカルリポジトリを使用するようになっていました。新しいプロファイルを何もない状態から作成した場合、既にローカルリポジトリ内に無効なプロファイルが存在すると、`/etc/apparmor/profiles/extras` 内にある無効なプロファイルの中から、いずれかのプロファイルを使用し、それをベースにするかどうかを 尋ねます。いずれかのプロファイルを使用するように選択すると、既定で有効化 されるプロファイルのディレクトリ (`/etc/apparmor.d`) に 内容がコピーされ、AppArmor の起動時に読み込まれるようになります。残りの 細かい調整は、`/etc/apparmor.d` ディレクトリ内の プロファイルに対して行ないます。

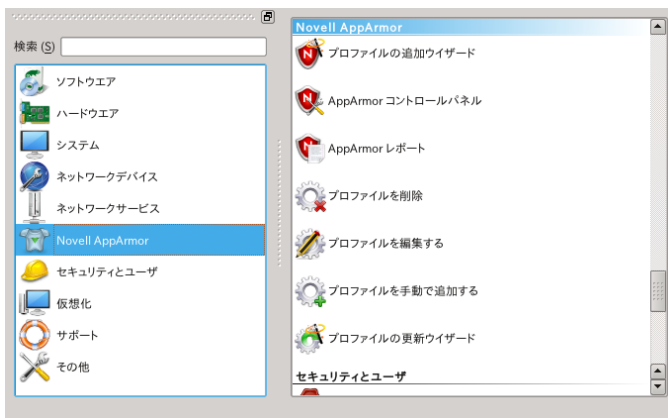
YaST を利用したプロファイルの構築と管理

21

YaST では、AppArmor® のプロファイルを構築したり管理したりするのに便利な仕組みを提供しています。また、YaST では 2 種類のインターフェイスが用意されていて、それぞれグラフィカルなインターフェイスとテキストベースのインターフェイスです。テキストベースのインターフェイスは、コンピュータの 資源を大きく消費することなく、ネットワーク帯域も小さくて済むため、リモートからの管理や何らかの事情で GUI が使用できない場合に便利な方法 です。それぞれのインターフェイスは異なる外観ですが、機能も操作手順も全く 同一になっています。それ以外の方法として存在するのが、AppArmor のコマンドを 利用する方法です。端末ウィンドウやリモート接続から AppArmor を制御する方法で、こちらの使い方は 第22章 コマンドラインからのプロファイル構築 (279 ページ) で説明しています。

メインメニューから YaST を起動し、ダイアログが表示された場合は root のパスワードを入力します。これ以外にも、端末ウィンドウを開いて root になったあと、yast2 または yast と入力 することで YaST を起動することもできます。前者の場合はグラフィカルな インターフェイスが、後者の場合はテキストベースのインターフェイスが表示されます。

図 21.1 AppArmor に対する YaST の制御



右側の枠には AppArmor のオプションが表示されます:

AppArmor プロファイルウィザード

詳しい手順については、21.1 項「ウィザードを使用したプロファイルの追加」(261 ページ) をお読みください。

手作業でプロファイルを追加

お使いのシステムにあるアプリケーションに対して、ウィザードの助けを 借りずに AppArmor のプロファイルを追加します。詳しい手順については、21.2 項「手作業でのプロファイル追加」(268 ページ) をお読みください。

プロファイルの編集

お使いのシステムにある既存の AppArmor プロファイルを編集します。詳しい手順については 21.3 項「プロファイルの編集」(269 ページ) をお読みください。

プロファイルの削除

お使いのシステムにある AppArmor プロファイルを削除します。詳しい手順については、21.4 項「プロファイルの削除」(274 ページ) をお読みください。

プロファイルの更新ウィザード

詳しい手順については、21.5 項「ログ項目からのプロファイル更新」(275 ページ) をお読みください。

AppArmor Control Panel

詳しい手順については、21.6 項「AppArmor の管理」(275 ページ) をお読みください。

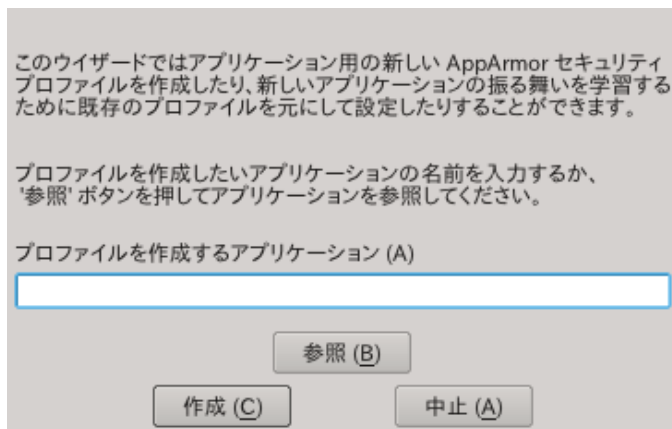
21.1 ウィザードを使用したプロファイルの追加

AppArmor プロファイルウィザードは、*AppArmor* のプロファイルツールである *aa-genprof* (プロファイルの生成) や *aa-logprof* (学習モードのログファイルからのプロファイル更新) を利用したプロファイル設定ツールです。これらのツールについて、詳しい使い方は 22.6.3 項「プロファイル作成ツールの概要」(286 ページ)をお読みください。

- 1 プロファイル内に起動時の情報を含めるため、プロファイル作業前に アプリケーションを停止します。対象のアプリケーションやデーモンが、動作していない状態であることを確認してください。

アプリケーションを停止するには、端末ウィンドウを開いて *root* になり、*rc* プログラム名 *stop* (または */etc/init.d/プログラム名 stop*) を実行します。*プログラム名* にはプロファイル対象のプログラムの名前を指定してください。

- 2 *YaST* を起動し、*AppArmor* > *AppArmor* プロファイルウィザード を選択します。



このウィザードではアプリケーション用の新しい AppArmor セキュリティプロファイルを作成したり、新しいアプリケーションの振る舞いを学習するために既存のプロファイルを元にして設定したりすることができます。

プロファイルを作成したいアプリケーションの名前を入力するか、'参照' ボタンを押してアプリケーションを参照してください。

プロファイルを作成するアプリケーション (A)

参照 (B)

作成 (C) 中止 (A)

- 3 アプリケーションの名前を入力するか、もしくはプログラムの場所を指定します。
- 4 *作成* を押します。aa-autodep と呼ばれる AppArmor ツールが 起動し、プロファイル対象のプログラムに対して静的な分析が行なわれます。その

後、必要なプロファイルが AppArmor モジュールとして読み込まれます。aa-autodep について、詳しくは 22.6.3.1 項「aa-autodep—概要プロファイルの作成」(286 ページ)をお読みください。

作成するプロファイルが、既にローカルのプロファイルリポジトリ内に存在する場合 (詳しくは 20.1 項「ローカルリポジトリの使用」(257 ページ)をお読みください) や、外部のプロファイルリポジトリ内に存在する場合 (詳しくは 第20 章 *AppArmor のプロファイルリポジトリ* (257 ページ)をお読みください)、もしくはどこにも存在していない場合によって、それぞれ異なる手順を実施します:

- ローカルのプロファイルリポジトリ内に既存のプロファイルが存在している場合は、これを利用して調整することができます。手順については ステップ 5 (262 ページ)をお読みください。
 - 外部のプロファイルリポジトリ内に既存のプロファイルが存在している場合は、これを利用して調整することができます。手順については ステップ 6 (262 ページ)をお読みください。
 - 何もない状態からプロファイルを作成するには、ステップ 7 (263 ページ)の手順を実施してください。
- 5 /etc/apparmor/profiles/extra ディレクトリにある ローカルプロファイルリポジトリ内に、既にプロファイルが存在する場合、YaST は有効化されていないプロファイルが存在する旨を表示し、これを ベースとして使用するか、もしくはこれをそのまま使用するかを尋ねます。

もちろんこれらのプロファイルを使用せず、何もない状態からプロファイルを作成することもできます。いずれの場合とも、ステップ 7 (263 ページ)に読み進めてください。

- 6 外部プロファイルリポジトリ内に既にプロファイルが存在していて、既に リポジトリ内に存在するプロファイルをはじめて作成しようとしている場合は、サーバへのアクセス設定を行なうことができます。設定は下記のようにして 行ないます:

- 6a まずは外部リポジトリへのアクセスを有効にするか、もしくはこの決定を先延ばしにするかを選択します。*リポジトリの有効化*を選択すると、次のステップではアクセスモード (ダウンロード／アップロード) の選択を行ないます。先延ばしにしたい場合は *後で確認* を押し、ステップ 7 (263 ページ)に進んでください。

6b プロファイルが配置されているリポジトリサーバについて、このアカウント 情報 (ユーザ名とパスワード) を入力し、サーバの登録を行いません。

6c あとは ステップ 7 (263 ページ) の手順に従って プロファイルを使用してください。

7 プロファイルを行ないたいアプリケーションを起動します。

8 可能な限りアプリケーションの機能を多く使用して、学習モードからその アプリケーションが必要とするファイルやディレクトリへのアクセスを判断 できるようにします。なお、アプリケーションの再起動や停止についても 学習させてください。AppArmor では他のプログラム同様、これらのイベントに ついても処理を行なう必要があるためです。

9 システムログをスキャンして *AppArmor イベントを検出* を押し、学習モードでのログファイルを処理させます。この処理では、セキュリティ プロファイルを作成するにあたって、ウィザードから多数の質問が表示されることがあります。

ハットの追加を求められた場合は、第23章 *ハット変更を利用した Web アプリケーションのプロファイル作成* (307 ページ) をお読みください。

なお、質問は大きく分けて 2 つの種類に分けられます：

- プロファイル内には書かれていない資源へのアクセスが、プロファイル対象のプログラムから行なわれた場合 (詳しくは 図21.2「学習モードの例外: 特定のリソースに対するアクセス制御」(264 ページ) をお読みください)。それぞれのリソースに対して、アクセスを許可するかどうかを選択します。
- プロファイル対象のプログラムからあるプログラムが実行されたものの、セキュリティドメインの遷移が設定されていない場合 (詳しくは 図21.3「学習モードの例外: 項目に対する実行許可の設定」(264 ページ) をお読みください)。対象の項目に対して実行許可を設定します。

それぞれの質問に対して答えることで、プロファイル内にリソースへのアクセスか プログラムの追加を行なうことになります。それぞれの選択について、詳しくは 図21.2「学習モードの例外: 特定のリソースに対するアクセス制御」(264 ページ) と 図21.3「学習モードの例外: 項目に対する実行許可の設定」(264 ページ) をお読みください。続くステップでは、これらの質問に対する回答の説明を記述します。

注記: 処理オプションの違いについて

処理している項目の種類によって、利用可能なオプションが異なります。

図 21.2 学習モードの例外: 特定のリソースに対するアクセス制御



図 21.3 学習モードの例外: 項目に対する実行許可の設定



- 10 *AppArmor* プロファイルウィザード はプロファイル対象の アプリケーションがアクセスしたディレクトリパスを提案する (図21.2「学習モードの例外: 特定の

リソースに対するアクセス制御」(264 ページ)) か、もしくはそれぞれの項目に対して実行許可の設定を求めます (図21.3「学習モードの例外: 項目に対する実行許可の設定」(264 ページ))。

- 図 21.2: 学習モードの例外: 特定のリソースに対するアクセス制御 に対しては、それぞれ必要なアクセス許可パターンを選択します。include 句で他のファイルを参照させるか、ワイルドカード表記を利用するか、もしくは実際のパス名を指定するかを選択することができます。状況に応じて、それぞれ以下のオプションを利用できます:

#include

include ファイルと呼ばれる、他のファイルへの参照を設定します。include ファイルはプログラムに対してアクセス許可を設定するためのもので、これを利用すると、他のプログラムでも利用することのできる共通のファイルを作成することができます。また、include を利用するとプロファイルのサイズを小さくすることにもなります。提案があった場合は、これを選択しておくのが良いでしょう。

ワイルドカード表記

グロブを押すことでアクセスできるものです。グロブの書式について、詳しくは 19.6項「パスとグロブ」(242 ページ)をお読みください。

実際のパス名

プログラムがアクセスする必要のあるパスについて、これをありのまま記述します。

ディレクトリパスを選択したあとは、許可 または 拒否 のいずれかを押して AppArmor のプロファイルに設定します。表示されたパスが期待通りのものでない場合は、グロブや編集を押してそれぞれ設定を調整することができます。

学習モードで項目を処理し、プロファイルを構築する場合、下記のオプションを利用することができます:

許可

プログラムに対して、指定したディレクトリパスに対するアクセスを許可します。AppArmor プロファイルウイザードでは、ファイルへのアクセス許可を提案します。詳しくは 19.7項「ファイルのアクセス許可とアクセスモード」(245 ページ)をお読みください。

拒否

拒否 を押すと、プログラムに対して指定したパスへの アクセスを禁止することができます。

グロブ

これを押すと、ディレクトリパスに対してワイルドカードを適用して、指定したディレクトリ内の全ファイルを含めるようにすることができます。ダブルクリックを行なうと、ディレクトリパス以下にある全てのファイルとディレクトリが含まれるようになります。グロブの文法について、詳しくは 19.6 項「パスとグロブ」(242 ページ) をお読みください。

拡張子付きグロブ

元のディレクトリパス表記を修正して、ファイル名の拡張子だけを保持するようにします。1 回だけ押した場合は `/etc/apache2/file.ext` のようなパスが `/etc/apache2/*.ext` となり、ファイル名部分にワイルドカードが設定されるようになります。この場合、左記のディレクトリ内にある `.ext` という拡張子の 全ファイルが含まれるようになります。ダブルクリックを行なうと、特定の拡張子を持つ全てのファイルとサブディレクトリが含まれるようになります。

編集

ハイライト表示された項目を編集します。編集後の行は、一覧内の最後に 現われます。

中止

`aa-logprof` を中止し、これまでに設定した全てのルール変更を取り消します。これにより、全てのプロファイルは修正前の状態に戻ります。

完了

`aa-logprof` を閉じ、これまでに設定した全てのルール変更を保存します。

それぞれの学習モードの項目に対して、**許可** または **拒否** を押します。これにより AppArmor のプロファイルの構築 を助けることになります。

注記

学習モードの項目に表示される項目数は、アプリケーションの複雑さに依存して変化します。

- 図 21.3: 学習モードの例外: 項目に対する実行許可の設定 に対しては、それぞれ必要なアクセス許可を選択します。利用可能なオプションについて、詳しくは 19.7項「ファイルのアクセス許可とアクセスモード」(245 ページ)をお読みください。

継承

親と同じセキュリティプロファイルを適用します。

プロファイル

実行されたプログラムに対して、別途のプロファイルを求めるようにします。このオプションを選択する場合は、プロファイルを切り替える際に AppArmor が 特定の環境変数を取り除き (消毒し)、子プロセスの動作に影響が及ばないようにするかどうかを選択することができます。子プロセスの動作について、これらの環境変数がどうしても必要な場合を除き、常により安全なオプションを選択しておくことをお勧めします。

無制限

セキュリティプロファイル無しでプログラムを実行します。問い合わせがあった場合、親のプロセスから特定の環境変数を引き継ぐことでセキュリティリスクを高めるようなことがないよう、環境変数の取り除き処理を行なうことも できます。

警告: 無制限モードでの実行のリスク

どうしても必要な場合を除いて、無制限モードでは動作させない ください。 *無制限* オプションを選択すると、新しいプログラムは AppArmor の保護無しで動作することになります。

拒否

拒否 を押すと、指定したパスに対するプログラムの アクセスを禁止することができます。

中止

aa-logprof を中止し、これまでに設定した全てのルール変更を取り消します。これにより、全てのプロファイルは修正前の状態に戻ります。

完了

aa-logprof を閉じ、これまでに設定した全てのルール変更を保存します。

- 11 アプリケーションにあるさらなる機能を試す必要がある場合は、上記までの手順を繰り返し実施します。

全てが完了したら **完了** を押します。続いてローカルのプロファイル セットに対して、変更点を保存するかどうかを選択します。以前の手順で、お使いのプロファイルを外部のプロファイルリポジトリにアップロードするように選択していた場合は、作業内容に関する短い変更履歴情報を入力し、アップロード作業を行ないます。以前の手順でプロファイルをアップロードするかどうかを選択しなかった場合は、YaST はここで再度プロファイルのアップロード用にアカウントを作成するかどうか、および今すぐアップロードするかどうかをそれぞれ選択することができます。

プロファイル作成ウィザードを終了すると、プロファイルは すぐにローカルとリポジトリサーバ (アップロードの設定を行なっていれば) の 両方に保存されます。また、プロファイルは AppArmor のモジュール内に読み込まれます。

21.2 手作業でのプロファイル追加

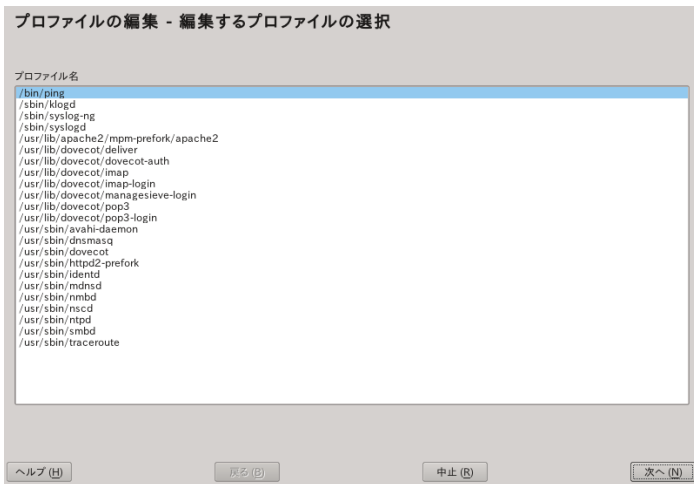
AppArmor では、プロファイル内に手作業で項目を追加する方法でのプロファイル構築 にも対応しています。プロファイルを作成したいアプリケーションを選択してから、項目を追加してください。

- 1 YaST を起動し、*AppArmor > 手作業でプロファイルを追加* を選択します。
- 2 プロファイルを作成するアプリケーションを、お使いのファイルシステム内から 選択します。
- 3 アプリケーションを選択したあとは **開く** を押します。*AppArmor プロファイルダイアログ* 内には、基本的な構造だけを持つ何もないプロファイルが表示されます。
- 4 *AppArmor プロファイルダイアログ* 内では、AppArmor のプロファイル を追加／編集／削除することができます。それぞれ対応するボタンを押して作業を行なってください。また、21.3.1項「項目の追加」(270 ページ)、21.3.2項「項目の編集」(274 ページ)、21.3.3項「項目の削除」(274 ページ) についてもそれぞれお読みください。
- 5 作業が完了したら **完了** を押せば完了です。

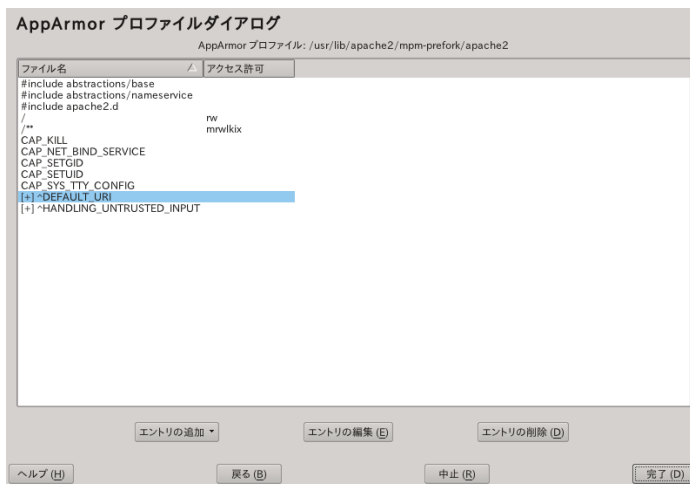
21.3 プロファイルの編集

AppArmor ではプロファイルの項目を手作業で追加／編集／削除することができます。プロファイルを編集するには、下記の手順で実施します：

- 1 YaST を起動し、*AppArmor* > *プロファイルの編集* を選択します。



- 2 プロファイル済みのアプリケーション一覧が表示されますので、編集したいプロファイルを 選択します。
- 3 次へ を押します。*AppArmor* プロファイルダイアログ ウィンドウ内にプロファイルが表示されます。



4 AppArmor プロファイルダイアログ 内では、AppArmor のプロファイル を追加／編集／削除することができます。それぞれ対応するボタンを押して作業を行なってください。また、21.3.1項「項目の追加」(270 ページ), 21.3.2項「項目の編集」(274 ページ), 21.3.3項「項目の削除」(274 ページ) についてもそれぞれお読みください。

5 作業が完了したら、完了 を押します。

6 ポップアップが表示されたら、はい を押します。するとプロファイルへの変更が保存され、AppArmor のプロファイルセットが 再読み込みされます。

ヒント: AppArmor 内での文法チェック

AppArmor には文法チェック機能があり、YaST の AppArmor ツールを利用して作業を行なっていればプロファイル内の文法エラーが通知されるようになっています。エラーが発生した場合は root でプロファイルを修正したあと、`rcapparmor reload` コマンドでプロファイル セットを再読み込みしてください。

21.3.1 項目の追加

項目の追加 オプションは、21.2項「手作業でのプロファイル追加」(268 ページ) や 21.3項「プロファイルの編集」(269 ページ) で 利用でき

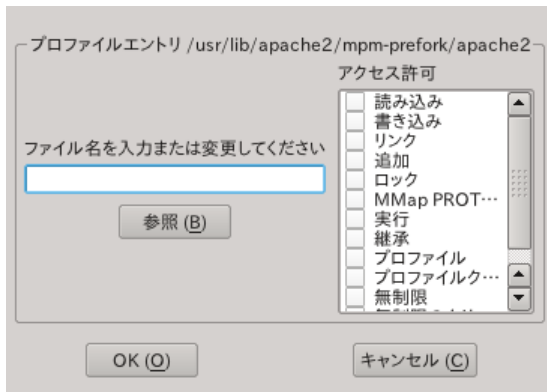
まず、**項目の追加** を選択すると、AppArmor プロファイルに 追加する項目種類の一覧が表示されます。

一覧からいずれかを選択してください:

ファイル

ポップアップウィンドウ内では、アクセスを許可するファイルについて、その絶対パスを 指定します。完了したら **OK** を押します。

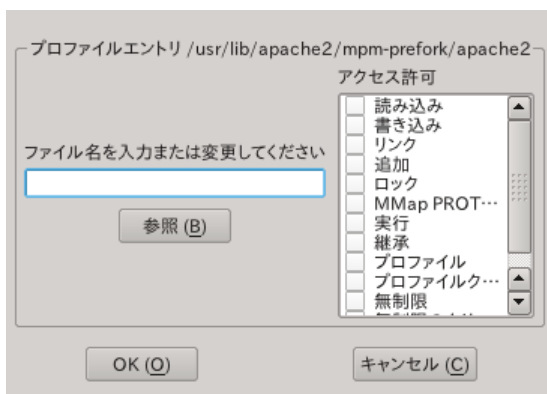
必要であればワイルドカードを利用したグロブ表記を行なうこともできます。グロブ について、詳しくは 19.6項「パスとグロブ」(242 ページ) をお読みください。また、ファイルのアクセス許可情報について、詳しくは 19.7項「ファイルのアクセス許可とアクセスモード」(245 ページ) をお読みください。



ディレクトリ

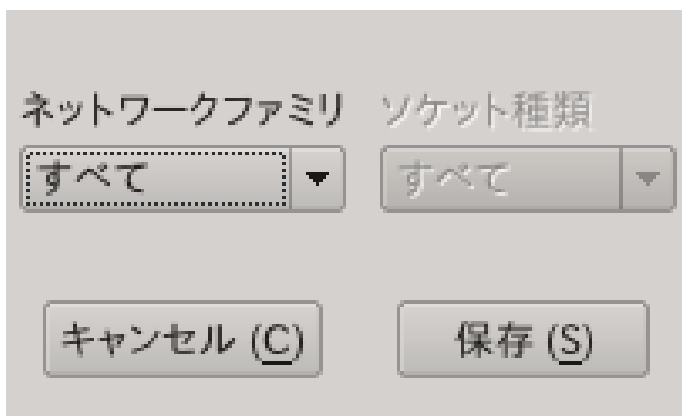
ポップアップウィンドウ内では、アクセスを許可するディレクトリについて、その絶対パスを指定します。必要であればワイルドカードを利用したグロブ表記を行なうこともできます。完了したら **OK** を押してください。

グロブについての、詳しくは 19.6項「パスとグロブ」(242 ページ) をお読みください。また、ファイルのアクセス許可情報について、詳しくは 19.7項「ファイルのアクセス許可とアクセスモード」(245 ページ) をお読みください。



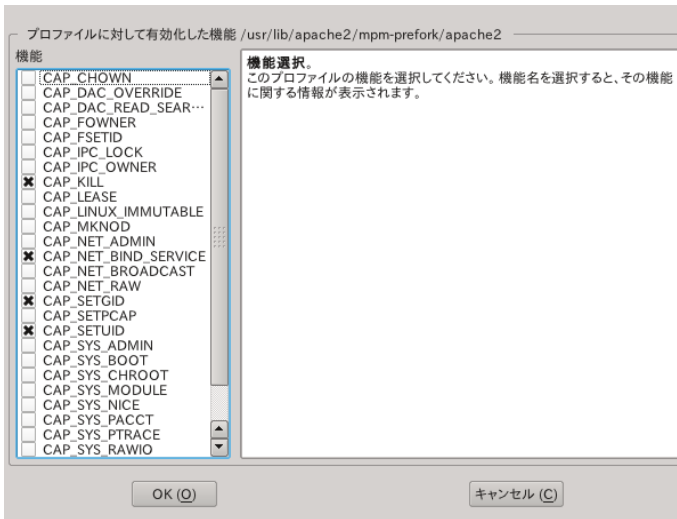
ネットワークルール

ポップアップウィンドウ内では、ネットワークファミリーとソケットの種類を選択します。詳しくは 19.5項「ネットワークアクセス制御」(241 ページ) をお読みください。



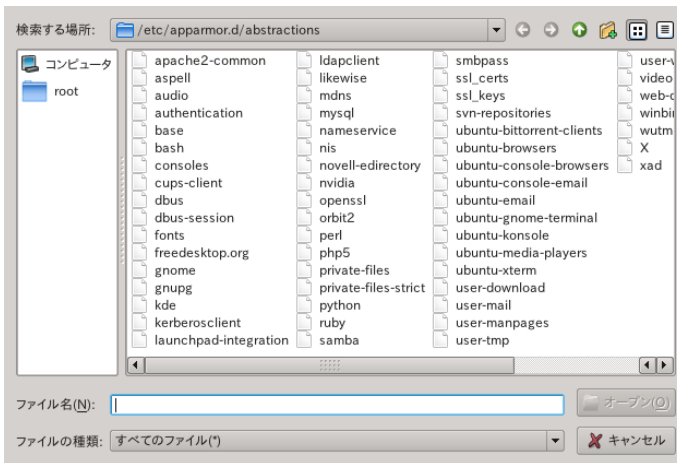
機能 (ケーパビリティ)

ポップアップウィンドウ内では、必要な機能 (ケーパビリティ) を選択します。これらはそれぞれ 32 POSIX.1e ケーパビリティを有効にするためのものです。ケーパビリティについて、詳しくは 19.4項「機能項目 (POSIX.1e)」(241 ページ) をお読みください。選択が完了したら *OK* を押します。



Include

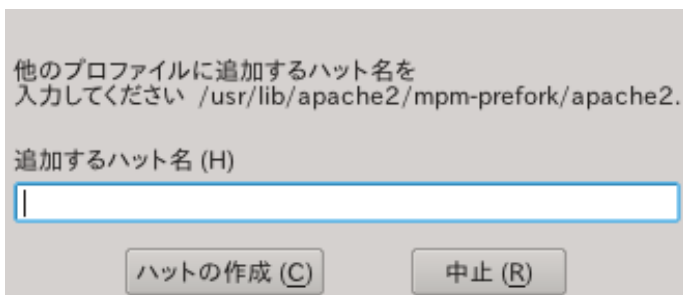
ポップアップウィンドウ内では、取り込みたいファイルを選択します。include とは他の AppArmor プロファイル内に書かれている情報を引用して使用するための機能です。詳しくは 19.3 項「#include ステートメント」(240 ページ) をお読みください。



ハット

ポップアップウィンドウ内では、現在のプロファイルに追加するサブプロファイル (ハット) の名前を指定します。名前を入力したら **ハットの作成** を押してください

い。ハットについて、詳しくは 第23章 *ハット変更を利用した Web アプリケーションのプロファイル作成* (307 ページ) をお読みください。



21.3.2 項目の編集

項目の編集 を選択すると、ファイルブラウザのポップアップ ウィンドウが表示されます。ここから選択した項目を編集することができます。

ポップアップウィンドウ内では、アクセスを許可するファイルについて、その絶対パスを指定します。必要であれば、ワイルドカードを利用したグロブ表記を行なうことができます。作業が完了したら *OK* を押します。

グロブ表記について、詳しくは 19.6項「パスとグロブ」(242 ページ) をお読みください。また、ファイルアクセス許可について、詳しくは 19.7項「ファイルのアクセス許可とアクセスモード」(245 ページ) をお読みください。

21.3.3 項目の削除

指定したプロファイルで項目を削除するには、*項目の削除* を選択します。AppArmor は選択したプロファイル項目を削除します。

21.4 プロファイルの削除

AppArmor ではプロファイルを手作業で削除することもできます。単純にプロファイルを削除したいアプリケーションを選択するため、下記の手順で削除してください:

- 1 YaST を起動し、*AppArmor* > *プロファイルの削除* を選択します。
- 2 削除するプロファイルを選択します。
- 3 次へ を押します。
- 4 ポップアップウィンドウが開いたら はい を押します。するとプロファイルが削除され、AppArmor のプロファイルセットが再読み込みされます。

21.5 ログ項目からのプロファイル更新

AppArmor プロファイルウィザードでは aa-logprof と呼ばれるツールを使用します。このツールはログファイルを読み込み、プロファイルを更新することができます。また、このツールでは AppArmor モジュールからのメッセージを追跡する機能があり、お使いのシステムで発生した全てのプロファイルに対する違反事例を追跡することができます。これらの違反事例は、プログラムに対する何らかのプロファイル 設定が足りていないことを示しているため、本ツールを利用してプロファイルを更新すると、それらの違反に対して設定を追加することができるようになっています。

- 1 YaST を起動し、*AppArmor* > *プロファイルの更新ウィザード* を選択します。

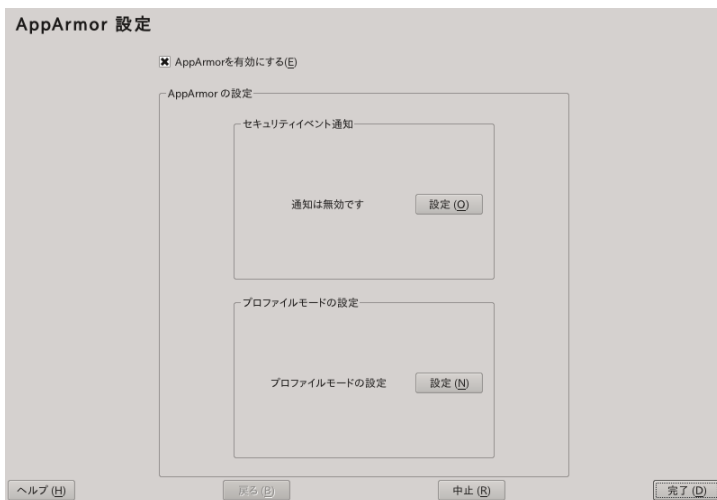
プロファイルの更新ウィザード (aa-logprof) では、学習モードの ログファイルを読み込んで処理します。この処理時間中は、セキュリティプロファイルに 正しく反映させるための問い合わせメッセージが、数多く表示されます。このときの 選択は、新しいプロファイルを作成する際と同じように実施してください。詳しくは 21.1 項「ウィザードを使用したプロファイルの追加」(261 ページ) 内の ステップ 9 (263 ページ) をお読みください。

- 2 作業が完了したら 完了 を押します。続いて表示される ポップアップで はい を選択すると、*プロファイルの更新ウィザード* を終了することができます。これでプロファイルが保存され、AppArmor モジュール内に読み込まれるようになります。

21.6 AppArmor の管理

AppArmor はその状態を切り替えて、有効化や無効化を行なうことができます。AppArmor を有効に するとお使いのシステムを保護して、潜在的なプログラム保護を受けることができるようになります。AppArmor を無効にすると、プロファ

イルが設定されていた場合でも、お使いのシステムに対する保護は行なわれなくなります。AppArmor の状態を変更するには、YaST を起動して *AppArmor* > *AppArmor* コントロールパネル を選択します。



AppArmor の状態を変更するには、21.6.1項「AppArmor の状態変更」(276 ページ) の手順に従って操作します。また、個別のプロファイルについて状態を変更するには、21.6.2項「個別のプロファイルに対するモード変更」(277 ページ) の手順に従って操作します。

21.6.1 AppArmor の状態変更

AppArmor の状態は、有効と無効とを切り替えることができます。AppArmor が有効になっていれば、AppArmor のセキュリティポリシーがインストールされて動作し、強制されるようになります。

- 1 YaST を起動して、*AppArmor* > *AppArmor* コントロールパネル を選択します。
- 2 *AppArmor* を有効にする にチェックを入れると有効に設定することができ、チェックを外すと無効に設定することができます。
- 3 設定が終わったら *AppArmor* の設定 内で *完了* を押します。
- 4 YaST コントロールセンターで *ファイル* > *終了* を選択します。

21.6.2 個別のプロファイルに対するモード変更

AppArmor はプロファイルを 2 種類のモードで適用することができます。不平モードと学習モードでは、プロファイルでは許可されていないファイルへのアクセスなど、AppArmor のプロファイル ルールの違反は、検出され記録されますが、許可されます。このモードはプロファイル の構築時に便利なモードで、AppArmor ツールがプロファイルを生成する場合にも 利用しているモードです。一方の強制モードでは、プロファイルで規定されたポリシーが強制され、ポリシーへの違反は `syslogd` を利用して記録されます。

プロファイルモード ダイアログでは、現在読み込まれている AppArmor のプロファイルに対して、それぞれのモードを閲覧したり編集したりすることができます。期の機能は、プロファイル開発時にお使いのシステムの状態を判断するのに 便利な仕組みです。もちろんシステム全体のプロファイル作業 (詳しくは 22.6.2 項「統合型プロファイル作成」(284 ページ) をお読みください) にも便利な仕組みで、複数のプロファイルを調整したり監視したりしたい場合にも 利用できます。

アプリケーションのプロファイルモードを編集するには、下記の手順を行ないます：

- 1 YaST を起動し、*AppArmor* > *AppArmor* コントロールパネル を選択します。
- 2 **プロファイルモード設定** のセクション内で、**設定** を選択します。
- 3 モードを切り替えたいプロファイルを選択します。
- 4 **モード切替** を選択し、それぞれ **不平モード** や **強制モード** を選択します。
- 5 最後に **完了** を押すと、設定を適用して YaST を終了 することができます。

全てのプロファイルを切り替えたい場合は、それぞれ **全てを強制モードに設定** や **全てを不平モードに設定** を利用してください。

ヒント: 表示されるプロファイルについて

既定では有効になっているプロファイル (お使いのシステムに対して、対象のアプリケーションがインストールされているプロファイル) だけが一覧表示されます。アプリケーションのインストールを行なう前にプロファイルの設定を行ないたい場合は、**全てのプロファイルを表示** を押し、表示された 一覧から設定したいプロファイルを選択してください。

コマンドラインからのプロファイル構築

AppArmor® では、システムのセキュリティを管理したり設定したりするのに、グラフィカルなインターフェイスだけでなく、コマンドラインインターフェイスを利用することもできます。コマンドラインツールを利用することで、AppArmor の状態を追跡したり、AppArmor のプロファイルを作成／削除／修正したりすることができます。

ヒント: 背景となる情報

AppArmor のコマンドラインツールを利用してプロファイルの管理を行なう前に、まずは 第18章 プログラムに対する免疫付与 (223 ページ) と 第19章 プロファイルの構成と文法 (233 ページ) で説明している AppArmor の一般的な 情報をお読みください。

22.1 AppArmor のモジュール状態の確認

AppArmor のモジュールは、下記の 3 種類のうちのいずれかの状態になります:

読み込み前

AppArmor のモジュールがカーネルに読み込まれていない状態です。

実行中

AppArmor のモジュールがカーネル内に読み込まれ、AppArmor のプログラム ポリシーが強制されている状態です。

停止中

AppArmor のモジュールはカーネル内に読み込まれていますが、プログラム ポリシーは強制されていない状態です。

AppArmor のモジュールの状態は、`/sys/kernel/security/apparmor/profiles` ファイルを調べることで判別できます。`cat /sys/kernel/security/apparmor/profiles` を実行することでプロファイルの一覧が表示されるようであれば、AppArmor は実行中であると言えます。何も表示されない場合は、AppArmor は停止中で、左記のファイルが存在していない場合は AppArmor が読み込み前の状態であると言えます。

スクリプト `rcapparmor` を利用した管理では、下記の操作を行なうことができます：

`rcapparmor start`

AppArmor のモジュール状態によって動作が異なります。読み込み前の状態であった場合、`start` はモジュールを読み込んで起動し、実行中の状態に移行します。停止中の状態であった場合、`start` は通常 `/etc/apparmor.d` ディレクトリ内にある AppArmor プロファイルを再読み込みし、実行中の状態に移行します。既に実行中であった場合、`start` は警告メッセージを表示するだけで何も行ないません。

`rcapparmor stop`

実行中の場合は、すべてのプロファイルをカーネルメモリから取り除き、すべてのアクセス制御を無効化してモジュールを停止状態にすることで、AppArmor モジュールの停止を行ないます。AppArmor のモジュールが読み込み前の状態であった場合や、既に停止していた場合、`stop` は再度プロファイルの読み込みを解除しようとはしますが、何も起こりません。

`rcapparmor restart`

AppArmor のモジュールに対して、実行中のプロセスが制限を受けることの無いようにしながら、`/etc/apparmor.d` 内にあるプロファイルの再スキャンを行なわせます。新しく作成されたプロファイルは強制モードに置かれ、削除されたプロファイルは `/etc/apparmor.d` ディレクトリから削除されます。

`rcapparmor kill`

カーネルから無条件に AppArmor のモジュールを取り除きます。ただし、このコマンドでの Linux カーネルからのモジュール読み込み解除は安全ではありません。このコマンドはデバッグ目的のほか、緊急にモジュールの読み込みを解除するために利用するものです。

警告

AppArmor はパワフルなアクセス制御システムであり、レスキューメディア (たとえば openSUSE の 1 枚目のメディア) から起動して制御を取り戻さない限り、あなた自身がマシンから締め出される可能性もありうるシステムです。

このような締め出しの問題を回避するには、AppArmor のモジュールを再起動する際に 無制限に動作するようにし、root で対象のマシンにログインしたあと 設定をやり直してください。ログインさえもできない状態にまで (たとえば SSH デーモンに関連づけられたプロファイルを壊してしまうなど) ダメージを与えてしまった場合は、左記の手順で獲得した root プロンプトを利用してダメージを修復し、AppArmor モジュールを再起動してください。

22.2 AppArmor プロファイルの構築

AppArmor モジュールのプロファイル設定は、`/etc/apparmor.d` ディレクトリ内にテキストファイル形式で保存されます。これらのファイルの 書式について、詳しくは第19章 *プロファイルの構成と文法* (233 ページ) を

`/etc/apparmor.d` ディレクトリ内にあるすべてのファイルは、プロファイルとして解釈され、読み込まれます。ディレクトリ内でのファイル 名の変更は、プロファイルが読み込まれないようにする対策としては正しく ありません。特定のプロファイルについて、これを読み込まれないようにしたい 場合は、ディレクトリから取り除いて対応してください。

プロファイルへのアクセスや変更を行なうには、vim などのテキストエディタを利用することができます。プロファイルを構築するにあたっては、下記のような 手段をとることができます：

AppArmor プロファイルの追加と作成

22.3項「AppArmor プロファイルの追加と作成」(282 ページ) をお読みください。

AppArmor プロファイルの編集

22.4項「AppArmor プロファイルの編集」(282 ページ) をお読みください。

AppArmor プロファイルの削除

22.5項「AppArmor プロファイルの削除」(282 ページ) をお読みください。

22.3 AppArmor プロファイルの追加と作成

アプリケーションに対して AppArmor プロファイルを追加したり作成したりする際、必要に応じて統合型のプロファイル作成を行なうか、もしくは単独でプロファイル作成を行なうかを選択することができます。それぞれのやり方について、詳しい説明は 22.6 項「プロファイルを作成する際の 2 つの方法」(283 ページ)をお読みください。

22.4 AppArmor プロファイルの編集

AppArmor のプロファイルを編集するには、下記の手順で行ないます：

- 1 root でログインしていない場合は、端末ウィンドウで `su` と入力します。
- 2 root のパスワードを尋ねられた場合は、それを入力します。
- 3 `cd /etc/apparmor.d/` を実行して、プロファイルのあるディレクトリに移動します。
- 4 現在インストールされているすべてのプロファイルを一覧表示するには、`ls` と入力します。
- 5 `vim` などのテキストエディタを利用して、プロファイルを開きます。
- 6 必要に応じてプロファイルを修正し、保存します。
- 7 端末ウィンドウで `rcapparmor restart` と入力し、AppArmor を再起動します。

22.5 AppArmor プロファイルの削除

AppArmor のプロファイルを削除するには、下記の手順で行ないます：

- 1 root でログインしていない場合は、端末ウィンドウで `su` と入力します。
- 2 root のパスワードを尋ねられた場合は、それを入力します。

- 3 `cd /etc/apparmor.d/` を実行して、プロファイルのあるディレクトリに移動します。
- 4 現在インストールされているすべてのプロファイルを一覧表示するには、`ls` と入力します。
- 5 プロファイルを削除するには、`rm` プロファイル名 と入力します。
- 6 端末ウィンドウで `rcapparmor restart` と入力し、AppArmor を再起動します。

22.6 プロファイルを作成する際の 2 つの方法

第19章 *プロファイルの構成と文法* (233 ページ) に書かれている AppArmor プロファイルの文法を利用することで、ツールを利用することなくプロファイルを作成することができます。ただし、ツールを利用せずに作成するのは、それなりの困難を伴います。このような手間暇を省くには、AppArmor のツールを利用してプロファイルの作成や修正を自動化することをお勧めします。

AppArmor のプロファイルを作成するにあたっては、2 種類の方法が存在します。それぞれの方法向けにツールが用意されています。

単独プロファイル作成

限られた時間しか動作しない、メールクライアントなどのユーザクライアントアプリケーションなどの小規模なアプリケーションに適した方法です。詳しくは 22.6.1 項「単独プロファイル作成」(284 ページ) をお読みください。

統合型プロファイル作成

一括で多数のプログラムに対してプロファイルを作成する方法で、Web サーバやメールサーバなど、何日／何週間もの間、システムの再起動が発生するまで動作し続けるタイプのアプリケーションに適した方法です。詳しくは 22.6.2 項「統合型プロファイル作成」(284 ページ) をお読みください。

プロファイルの自動開発は、AppArmor のツールを利用することでより管理がやりやすくなります：

- 1 必要に応じて、どちらのプロファイル作成方法を利用するのかを決めます。
- 2 まずは静的な分析を行いません。選択した作成方法に応じて、`aa-genprof` または `aa-autodep` を動作させます。

- 3 動的な学習を有効にします。すべてのプロファイル作成済みプログラムに対して学習モードを有効にします。

22.6.1 単独プロファイル作成

単独でのプロファイル生成と改善の作業は、aa-genprof と呼ばれるプログラムで管理します。この方法は、aa-genprof がすべての事象を取り扱うため、簡単に実行することができる方法です。ただし、aa-genprof ではお使いのプログラムをテストする際、すべての機能を動作させる必要があるという制限があります（プロファイルを作成している間は、マシンを再起動することができません）。

単独プロファイル作成で aa-genprof を利用するには、22.6.3.4項「aa-genprof ープロファイルの生成」（290 ページ）をお読みください。

22.6.2 統合型プロファイル作成

この方法は、aa-genprof や単独プロファイル作成のように 1 つまたは少数のプロファイルに限定することなく、システム内に存在するすべてのプロファイルを一括更新することから、**統合システムプロファイル作成** とも呼ばれます。統合システムプロファイル作成では、プロファイルの構築と改善はほとんど自動化することができませんが、その分だけ柔軟性があるものになります。なお、この方法はシステムを再起動するまでの間動作し続けるようなプログラムや、一括で多数のプログラムが動作するようなプログラムの場合に適切な手段です。

複数のアプリケーションに対して AppArmor のプロファイルを作成するには、下記の手順で行ないます：

- 1 まずはお使いのアプリケーションを構成する各プログラムに対して、それぞれプロファイルを作成します。

本手順は統合型とは言うものの、AppArmor はプロファイルの存在するプログラムとその子プロセスしか監視しません。AppArmor に対してプログラムへの考慮を行なわせるには、少なくとも aa-autodep を利用して概要プロファイルを作成しておかなければなりません。大まかなプロファイルを作成するための方法については、22.6.3.1項「aa-autodep ー概要プロファイルの作成」（286 ページ）をお読みください。

- 2 それぞれ関連するプロファイルを学習モードか不平モードに切り替えます。

root でログインしてから端末ウインドウを開き、`aa-complain /etc/apparmor.d/*` と入力することで、すべての プロファイルを学習モードや不平モードに切り替えることができます。この機能は YaST のプロファイルモジュールでも利用できるもので、こちらについては 21.6.2 項「個別のプロファイルに対するモード変更」(277 ページ) をお読みください。

学習モードでは、たとえプロファイル側で明示的に拒否されていた場合であっても、アクセス要求がブロックされることはありません。これにより、いくつかのテスト (ステップ 3 (285 ページ) に 書かれているような) を実施することができるほか、プログラムが正しく動作するためのアクセス要件を学習することができます。この情報を利用することで、プロファイルをどれだけ堅牢にするかを決定することができます。

学習モードに不平モードの詳しい手順については、22.6.3.2 項「aa-complain — 不平モード／学習モードへの突入」(288 ページ) をお読みください。

3 アプリケーションを使用します。

アプリケーションを起動し、その機能を一通り動作させます。少なくとも、プログラムが必要とするファイルにはすべてアクセスするようにしてください。動作は `aa-genprof` が管理するものではないため、この手順はシステムの再起動を行なうまでの間、数日間や数週間の単位で動作させることができます。

4 次にログを分析します。

統合型のプロファイル作成では、単独プロファイルで動作させる `aa-genprof` の代わりとして、`aa-logprof` を直接起動します。`aa-logprof` は下記のようにして実行します：

```
aa-logprof [ -d プロファイルのパス ] [ -f ログファイルのパス ]
```

`aa-logprof` について、詳しくは 22.6.3.5 項「`aa-logprof`—システムログのスクラン

5 あとは ステップ 3 (285 ページ) と ステップ 4 (285 ページ) を繰り返します。

繰り返し作業を行なうことで、最適なプロファイルを作成することができます。繰り返しの作業では、学習してポリシーエンジンに読み込むデータを小さくまとめることができるほか、警告などのメッセージも少なくできるので、より高速に動作することになります。

6 プロファイルを編集します。

生成されたプロファイルを確認したい場合は、vim などを利用し、`/etc/apparmor.d/` ディレクトリ内にあるプロファイルを 開いて編集します。

7 強制モードに戻します。

単純にログだけを採取していた状態から、プロファイル内に書かれているルールを 適用するようにシステムを戻します。これはプロファイル内に書かれている `flags=(complain)` という行を削除するか、もしくは `aa-enforce` コマンドを利用することで実施することができます。これは `aa-complain` コマンドと同じ使い方 で、強制モードに切り替える作業を行なうものです。この機能は、YaST の プロファイルモードモジュールから行なうこともできます。こちらについて 詳しくは 21.6.2項「個別のプロファイルに対するモード変更」(277 ページ)をお読みく ださい。

すべてのプロファイルの不平モードを終了し、強制モードに戻すには、`aa-enforce /etc/apparmor.d/*` と入力します。

8 すべてのプロファイルを再読み込みします。

AppArmor に対してすべてのプロファイルの再読み込みを行ない、カーネルでの 強制モードを適用するには、`rcapparmor restart` と入力します。

22.6.3 プロファイル作成ツールの概要

AppArmor でのプロファイル作成ユーティリティは、`apparmor-utils` RPM パッ ケージとして提供されていて、それらは `/usr/sbin` 内に配置されます。これらは そ れぞれ異なる用途で使います。

22.6.3.1 aa-autodep—概要プロファイルの作成

このユーティリティは、選択したプログラムやアプリケーションに対して、概要プロ ファイルを作成します。バイナリの実行形式のほか、インタプリタ 型のスクリプトプ ログラムに対しても、プロファイルを作成することができます。作成されたプロファ イルが「概要プロファイル」と呼ばれる のは、AppArmor の制限下で動作するにあ たって必要となる、すべての項目が含まれて いるわけではないことによります。aa-autodep が生成する最低限のプロファイル には、ほとんどのプログラムで必要とな るプロファイル項目を含む、基本的な `include` ディレクティブが書かれます。プログ ラムの種類によっては、aa-autodep がさらに詳しいプロファイルを生成する場合も

あります。これは コマンドラインで指定した実行ファイルに対し、再帰的に ldd(1) を呼び出すことで生成されたプロファイルです。

概要プロファイルを作成するには aa-autodep プログラムを使用して行ないます。aa-autodep に指定するパラメータは、パス名を含まないプログラム名 (aa-autodep 内部で PATH 環境変数を参照して検索します) かフルパスのプログラム名で指定します。概要プロファイルを作成するプログラムには、任意の種類のもので指定できます (ELF 形式のバイナリ、シェルスクリプト、Perl スクリプトなど)。aa-autodep は後に続く動的なプロファイル改善処理を行なう目的で、概要 プロファイルを作成します。

作成された概要プロファイルは、AppArmor のプロファイル命名規約に従って /etc/apparmor.d ディレクトリ以下に作成されます。これはプログラムの絶対パスのうち、スラッシュ (/) をピリオド (.) に置き換えた名前になります。aa-autodep は通常、root でログインした状態で、端末ウィンドウから 下記のように入力して実行します:

```
aa-autodep [ -d /path/to/profiles ] [program1 program2...]
```

プログラム名を指定しない場合は、aa-autodep の起動後に問い合わせが表示されます。また、*/path/to/profiles* を指定すると、既定で /etc/apparmor.d に設定されているプロファイルのパスを上書きすることができます。これにより、既定の場所以外にプロファイルを作成することができます。

プロファイル作成を始めるには、まずお使いのアプリケーションを構成する それぞれの実行ファイルやサービスに対して、プロファイルを作成しなければなりません (既にプロファイルを作成してあるプログラムや、そこから起動 される子プロセスなどは除きます)。このようなプログラムを全て調べるのは 面倒な作業ですが、下記のようなやり方で調べることができます:

ディレクトリ

プロファイル対象のプログラムが全て 1 つのディレクトリにまとまっていて、そのディレクトリにはそれ以外のプログラムが存在していない場合、aa-autodep */path/to/your/programs/** というシンプルなコマンドを実行することで、そのディレクトリ内にある 全てのプログラムに対する基本的なプロファイルを作成することができます。

ps コマンド

アプリケーションを起動したあと、標準の Linux コマンドである ps を利用すると、実行中の全てのプロセスを確認することができます。あとは手作業でそれぞれのプログラムの場所を探し出し、それぞれに対して aa-autodep を実行すれば完了です。プログラムが PATH 環境変数で検索可能なディレクトリにある場合は、aa-autodep 側でプログラムを検索します。検索可能なディレクトリに

無い場合は、同じく標準の Linux コマンドである `find` を利用することで、プログラムを見つけることができます。`find / -name 'アプリケーション名' -print` のように実行し、アプリケーションのパスを探してください。なお、必要であればワイルドカードを利用することもできます。

22.6.3.2 aa-complain—不平モード／学習モードへの突入

不平モード／学習モードツール (`aa-complain`) は、プロファイル済みのプログラムが許可されていないファイルにアクセスした場合など、プロファイルルールの違反事例を検出するようにします。この場合、違反事例は許可されるものの、ログが記録されます。プロファイルを改善するには、まず不平モードに設定したあとプログラムを起動し、プログラムの振る舞いを記録させるために一通りの機能を動作させます。あとは AppArmor ツールを利用してログファイルを事後処理し、ログイベントをプロファイルに適用します。

不平モードを手作業 (コマンドライン経由) で有効にした場合、プロファイルの冒頭にフラグを追加します。これにより `/bin/foo` と書かれている行が `/bin/foo flags=(complain)` のようになります。不平モードを使用するには、端末ウィンドウを開いて `root` になり、下記のいずれかを入力します:

- プログラムが `PATH` 環境変数で書かれたパス内に存在する場合は、下記のように実行します:

```
aa-complain [プログラム 1 プログラム 2 ...]
```

- プログラムが `PATH` 環境変数で書かれたパス内に存在しない場合は、下記のように実行します:

```
aa-complain /sbin/プログラム 1
```

- プロファイルが `/etc/apparmor.d` 以外のディレクトリに存在する場合は、下記のようにして場所を指定します:

```
aa-complain /path/to/profiles/ プログラム 1
```

- プログラム 1 に対するプロファイルを指定するには、下記のようにします:

```
aa-complain /etc/apparmor.d/sbin.プログラム 1
```

上記のコマンドは、いずれも指定したプロファイルやプログラムに対して、不平モードを有効にします。プログラム名がフルパス表記でない場合、`aa-complain` はプログラムを `$PATH` 内で検索します。たとえば `aa-complain /usr/sbin/*` のように実

行すると、/usr/sbin 以下にある全てのプログラムに関連づけられているプロファイルを検索し、不平モードに切り替えます。また、aa-complain /etc/apparmor.d/* のように実行すると、/etc/apparmor.d 内にある全てのプロファイルを 不平モードに切り替えます。

ヒント: YaST を利用したプロファイルのモード切り替え

YaST では、不平モードや強制モードへのプロファイル切り替えについて、グラフィカルなフロントエンドを提供しています。詳しくは 21.6.2 項「個別のプロファイルに対するモード変更」(277 ページ)をお読みください。

22.6.3.3 aa-enforce—強制モードへの突入

強制モードでは、AppArmor のプロファイル済みのプログラムが許可されていないファイルにアクセスした場合など、プロファイルルールの違反事例を検出し、禁止します。強制モードは通常時に使用しておくべきものです。違反事例を検出だけして、禁止したくない場合は、不平モードをお使いください。

強制モードを手作業 (コマンドライン経由) で有効にした場合、プロファイルの冒頭にフラグを追加します。これにより /bin/foo と書かれている行が /bin/foo flags=(enforce) のようになります。強制モードを使用するには、端末ウィンドウを開いて root になり、下記のいずれかを入力します:

- プログラムが PATH 環境変数で書かれたパス内に存在する場合は、下記のように 実行します:
`aa-enforce [プログラム 1 プログラム 2 ...]`
- プログラムが PATH 環境変数で書かれたパス内に存在しない場合は、下記のように 実行します:
`aa-enforce /sbin/プログラム 1`
- プロファイルが /etc/apparmor.d 以外のディレクトリに 存在する場合は、下記のようにして場所を指定します:
`aa-enforce /path/to/profiles/プログラム 1`
- プログラム 1 に対するプロファイルを指定するには、下記のようにします:
`aa-enforce /etc/apparmor.d/sbin.プログラム 1`

上記のコマンドは、いずれも指定したプロファイルやプログラムに対して、強制モードを有効にします。

プログラム名やプロファイル名を指定しない場合は、起動後に問い合わせが表示されます。また、`/path/to/profiles` を指定すると、既定で `/etc/apparmor.d` に設定されているプロファイルのパスを上書きすることができます。

パラメータにはプログラム名のほか、プロファイルの一覧を指定することもできます。プログラム名がフルパス表記でない場合、`aa-enforce` はプログラムを `$PATH` 内で検索します。

ヒント: YaST を利用したプロファイルのモード切り替え

YaST では、不平モードや強制モードへのプロファイル切り替えについて、グラフィカルなフロントエンドを提供しています。詳しくは 21.6.2 項「個別のプロファイルに対するモード変更」(277 ページ) をお読みください。

22.6.3.4 aa-genprof—プロファイルの生成

`aa-genprof` は AppArmor のプロファイル生成ユーティリティです。指定したプログラムに対して `aa-autodep` を起動して概要プロファイルを作成 (プロファイルが存在していなかった場合) し、プロファイルを不平モードに切り替えます。その後 AppArmor の再読み込み処理を行なってログに印を付け、ユーザに対してプログラムを起動して一通りの機能を利用するように促します。`aa-genprof` の書式は下記のとおりで:

```
aa-genprof [ -d /path/to/profiles ] program
```

たとえば Apache Web サーバのプログラムである `httpd2-prefork` に対してプロファイルを作成するには、下記の手順を `root` で実行します:

- 1 まずは端末ウィンドウから `rcapache2 stop` と入力します。
- 2 次に `aa-genprof httpd2-prefork` と入力します。

`aa-genprof` は下記のように動作します:

1. まずはシェルの `PATH` 環境変数を利用して、`httpd2-prefork` のフルパスを解決しようとします。コマンドラインからフルパスを指定することで、解決を行わずにすませることも可能です。なお、openSUSE では、`httpd2-prefork` は `/usr/sbin/httpd2-prefork` にあります。
2. 次に、`httpd2-prefork` に対する既存のプロファイルが存在していないかどうかを確認します。既に存在する場合は更新し、存在しない場合は `aa-`

autodep を利用して作成 (詳しくは 22.6.3項「プロファイル作成ツールの概要」(286 ページ) をお読みください) します。

3. このプログラムに対するプロファイルを、学習モードまたは不平モードに 切り替えます。これにより、プロファイルに対する違反事例は記録されるものの、特に禁止されることもなく動作するようになります。記録される イベントは下記のようになります (/var/log/audit/audit.log にあります):

```
type=APPARMOR_ALLOWED msg=audit(1189682639.184:20816): operation="file_mmap"
requested_mask="::r" denied_mask="::r" fsuid=30 name="/srv/www/htdocs/index.html"
pid=27471 profile="null-complain-profile"
```

監査デーモンを起動していない場合、AppArmor のイベントは /var/log/messages 内に記録されます:

```
Sep 13 13:20:30 K23 kernel: audit(1189682430.672:20810): operation="file_mmap"
requested_mask="::r" denied_mask="::r" fsuid=30 name="/srv/www/htdocs/phpsysinfo/
templates/bulix/form.tpl" pid=30405 profile="/usr/sbin/httpd2-prefork//phpsysinfo/"
```

イベントは dmesg コマンドを利用して閲覧することも できます:

```
audit(1189682430.672:20810): operation="file_mmap" requested_mask="::r"
denied_mask="::r" fsuid=30 name="/srv/www/htdocs/phpsysinfo/templates/bulix/
form.tpl" pid=30405 profile="/usr/sbin/httpd2-prefork//phpsysinfo/"
```

4. ログイベントの冒頭には、判別しやすいようにマーカーが付けられています。下記のような形式です:

```
Sep 13 17:48:52 figwit root: GenProf: e2ff78636296f16d0b5301209a04430d
```

- 3 ツールから問い合わせがあったタイミングで、プロファイル対象の アプリケーションを別の端末ウインドウ内で起動し、アプリケーションに用意されている機能をできるだけ多く使用します。これにより学習モードは、正しく動作するのに必要な、各種ファイルやディレクトリへのアクセスをログに記録 します。たとえばこの場合、新しい端末ウインドウを起動して rcapache2 start と入力します。
- 4 プログラムの機能を一通り試した後は、aa-logprof を起動した端末 ウインドウで、下記のキーのうちのいずれかを入力します:
 - S を押すと、aa-logprof をシステムログに対して 起動し、aa-genprof がマージングしたところからログを処理したあと、プロファイルを読み込みなおします。ログ内にシステムイベントが存在 した場合は、AppArmor は学習モードのログファイルを処理します。この 処理では、aa-genprof が正しいセキュリティプロファイルを作成する ために必要な、各種の問い合わせメッセージが表示されます。

- F を押すと、ツールを終了してメインメニューに戻る ことができます。

注記

ハットの追加を示すメッセージが表示された場合は、第23章 *ハット変更を利用した Web アプリケーションのプロファイル作成* (307 ページ) をお読みください。

5 それぞれ 2 種類の質問に対して回答してください:

- プロファイル対象のプログラムが、プロファイルには書かれていない リソースを要求した場合の質問 (詳しくは 例22.1「学習モードの例外: 特定リソースに対するアクセス制御」(292 ページ) をお読みください)。
- プロファイル対象のプログラムから新しいプログラムが起動され、領域 遷移が設定されていない場合の質問 (詳しくは 例22.2「学習モードの例外: 項目に対する実行許可の設定」(294 ページ) をお読みください)。

それぞれ下記に示す分類の事象が発生した場合、リソースやプロファイル 対象のプログラムに追加するための質問が表示されます。それぞれ、例22.1「学習モードの例外: 特定リソースに対するアクセス制御」(292 ページ) と 例22.2「学習モードの例外: 項目に対する実行許可の設定」(294 ページ) に例が示されています。続く手順では、これらの質問に回答する際の選択肢 について説明しています。

- 扱っている実行アクセスが複雑な場合。この場合は、これに適用する実行 許可タイプを選択しなければなりません:

例 22.1 学習モードの例外: 特定リソースに対するアクセス制御

```
Reading log entries from /var/log/audit/audit.log.  
Updating AppArmor profiles in /etc/apparmor.d.
```

```
Profile: /usr/sbin/xinetd  
Program: xinetd  
Execute: /usr/lib/cups/daemon/cups-lpd  
Severity: unknown
```

```
[(I)nherit] / (P)rofile / (U)nconfined / (D)eny / Abo(r)t / (F)inish
```

継承 (ix)

子プロセスに対して、親プロセスのプロファイルを引き継ぐように指定 します。これにより、親と子で同じアクセス制御を持つようになります。この選

択肢は、制限を受けるプログラムが他のプログラムを呼び出す環境で、元のプロファイルから新しく権利や許可を受けたり失ったりしない状況で便利な選択です。このモードは、子プログラムがヘルパーアプリケーションである場合にしばしば使用される選択肢で、たとえば /usr/bin/mail のようなメールクライアントから、less のようなページャや Mozilla* Web ブラウザを起動したり、PDF ファイルを表示するために Adobe Acrobat* を起動したりする場合が該当します。

プロファイル (px)

子プロセスに対して、子プロセス独自のプロファイルを使用させるように指定します。プロファイルが存在しない場合は、子プロセスの起動は「permission denied」(許可がありません)として失敗します。この選択肢は、親のプログラムがグローバルなサービス、たとえば DNS を参照したり、お使いのシステムにある MTA を利用してメールを送信したりする場合などに特に便利な仕組みです。

なお、*profile with clean exec* (Px) を選択すると、子プロセスに渡される環境変数のうち、実行動作を変更することのできるものを取り除くことができます。

無制限 (ux)

子プロセスに対して、いかなる AppArmor のプロファイルをも適用しない、無制限の実行を行なうように指定します。

なお、*unconfined with clean exec* (Ux) を選択すると、子プロセスに渡される環境変数のうち、実行動作を変更することのできるものを取り除くことができます。この選択は、AppArmor の盲点を作ることになることから、セキュリティ上の脆弱性を作ってしまいます。この選択は、他の選択がうまくいかない場合の最後の選択肢としてください。

mmap (m)

この許可は、プロファイル下で動作するプログラムに対して、PROT_EXEC フラグ付きの mmap システムコールを利用し、リソースにアクセスできる許可を与えます。これにより、ここにマッピングされたデータをプログラムとして実行できるようになります。プロファイルの作成中にこれが要求された場合、許可するかどうかを尋ねられます。

拒否

指定されたディレクトリパス項目に対して、プログラムからのアクセスを拒否します。AppArmor は次のイベントに移動します。

中止

aa-logprof を中断し、今までに行なわれた全てのルール変更を取り消し、全てのプロファイルを変更前の状態に戻します。

完了

aa-logprof を閉じ、今までに行なわれた全てのルール変更を保存します。

- 例22.2「学習モードの例外: 項目に対する実行許可の設定」(294 ページ) には、プロファイル対象のアプリケーションがアクセスするディレクトリパス について、AppArmor が提案するディレクトリを示しています。場合によっては、実行許可を設定する必要がある場合もあります。

例 22.2 学習モードの例外: 項目に対する実行許可の設定

Adding /bin/ps ix to profile.

Profile: /usr/sbin/xinetd
Path: /etc/hosts.allow
New Mode: r

[1 - /etc/hosts.allow]

[(A)llow] / (D)eny / (N)ew / (G)lob / Glob w/(E)xt / Abo(r)t / (F)inish

AppArmor では 1 つまたは複数のパス、もしくは include を提示します。選択肢の 番号を入力することで選択を行ない、次のステップに進むことができます。

注記

なお、AppArmor のメニューには全ての選択肢が表示されない場合があります。

#include

これは AppArmor のプロファイルを構成する include ファイルを参照するセクションで、プログラムに対するアクセス許可を設定するためのものです。include を使用することで、ファイルやディレクトリに対するアクセス許可を他のプログラムでも共有できるようになります。また、include を使用することでプロファイルのサイズを小さくすることもできます。提案があった場合は include を選択しておくのが お勧めです。

グロブ表記

これは次のステップで **グロブ** を選択した場合に 利用できるようになります。グロブの文法について、詳しくは 19.6項「パスとグロブ」(242 ページ) をお読みください。

実際のパス

プログラムが必要としているパスそのものを指定し、アクセスできるようにします。

パスや **include** を選択したあとは、AppArmor プロファイル内での扱いを、**許可** または **拒否** で指定します。表示されたディレクトリパスが期待通りのものでない場合は、**グロブ** を選択してワイルドカード指定を行なうこともできます。

学習モードでの項目処理の選択肢には、下記のようなものがあります：

Enter キー入力

選択したディレクトリパスに対して、アクセスを許可します。

許可

選択したディレクトリパスに対して、アクセスを許可します。AppArmor はファイルのアクセス許可を提案します。詳しくは 19.7項「ファイルのアクセス許可とアクセスモード」(245 ページ) をお読みください。

拒否

指定したディレクトリパス項目に対して、プログラムからのアクセスを 拒否します。AppArmor は次のイベントに進みます。

新規

このイベントに対して、正規表現を利用する独自のルールを設定します。入力した正規表現が、最初に提示したイベントに対して十分なものではない場合、AppArmor は確認メッセージを表示して再入力を促します。

グロブ

より広いパスが含まれるようにするため、指定したパスを選択するか ワイルドカードでルールを作成します。提示されたパスの中から選択 したい場合はパスの前に表示された番号を入力したあと、選択した項目 についてどのように扱うのかを指定します。

グロブの書式について、詳しくは 19.6項「パスとグロブ」(242 ページ) をお読みください。

拡張子付きグロブ

元々のディレクトリパスに対して、ファイル名の拡張子部分を保持する形でグロブを設定します。たとえば `/etc/apache2/file.ext` というディレクトリパスであった場合、ファイル名の本体部分にワイルドカード (アスタリスク) が挿入されて `/etc/apache2/*.ext` のようになります。これにより、プログラムは指定したディレクトリ内に存在し、かつ `.ext` という拡張子を持つ全てのファイルにアクセスできるようになります。

中止

`aa-logprof` を中断し、今までに行なわれた全てのルール変更を取り消し、全てのプロファイルを変更前の状態に戻します。

完了

`aa-logprof` を閉じ、今までに行なわれた全てのルール変更を保存します。

- 6 `vim` を利用してプロファイルを閲覧したり編集したりしたい場合は、端末 ウィンドウから `vim /etc/apparmor.d/プロファイル名` のように入力します。
- 7 `rcapparmor restart` コマンドを入力して AppArmor を再起動し、新しく作成したものを含む全てのプロファイルセットを再読み込みします。

AppArmor のプロファイルを構築する際のグラフィカルなフロントエンドと同様に、YaST の AppArmor プロファイルウィザードと `aa-genprof` では、`/etc/apparmor/profiles/extras` ディレクトリ内にある ローカルプロファイル と、リモートの AppArmor プロファイルリポジトリ を利用することができます。

ローカルリポジトリ内のプロファイルを利用するには、下記の手順を実施します:

- 1 上記の手順で `aa-genprof` を起動します。

`aa-genprof` が有効に設定されていないローカルプロファイルを検出すると、端末ウィンドウ内に下記のようなメッセージを表示します:

```
Profile: /usr/bin/opera
```

```
[1 - Inactive local profile for /usr/bin/opera]
```

```
[(V)iew Profile] / (U)se Profile / (C)reate New Profile / Abo(r)t / (F)inish
```

- 2 表示されたプロファイルを単純に使用したい場合は、U (*Use Profile*) を選択してから、上述のプロファイル生成 手順を実施します。

プロファイルを有効化する前に確認したい場合は、V (*View Profile*) を選択します。

既存のプロファイルを使用せずに無視したい場合は、C (*Create New Profile*) を選択してから、上述のとおり 何もない状態からのプロファイル生成手順を実施します。

- 3 aa-genprof を閉じ、今までに行なわれた全てのルール変更を保存するには、F (*Finish*) を選択します。

aa-genprof でリモートの AppArmor プロファイルリポジトリを使用するには、下記の手順を実施します:

- 1 上記の手順で aa-genprof を起動します。

aa-genprof がリポジトリサーバ内に適切なプロファイルが存在すると判断 した場合は、端末ウィンドウ内に下記のようなメッセージを表示します:

```
Repository: http://apparmor.opensuse.org/backend/api
```

```
Would you like to enable access to the profile repository?
```

```
(E)nable Repository / (D)isable Repository / Ask Me (L)ater
```

- 2 リポジトリを有効にするには、E (*Enable Repository*) を選択します。

- 3 aa-genprof に対して、リポジトリサーバにプロファイルをアップロード するかどうかを指示するには、下記のメッセージに応答してください:

```
Would you like to upload newly created and changed profiles to  
the profile repository?
```

```
(Y)es / (N)o / Ask Me (L)ater
```

プロファイルのアップロードを有効にするには Y (*Yes*) を、単純にリポジトリからプロファイルの 取得だけを行なわせ、アップロードさせたくない場合は N (*No*) を選択します。

- 4 プロファイルをアップロードできるようにするには、プロファイルの リポジトリサーバ内に新規ユーザを作成する必要があります。それぞれ ユーザ名とパスワードを用意してください。

- 5 サーバからダウンロードしたプロファイルを使用するかどうかと、その内容を 確認するかどうか尋ねられます:

Profile: /usr/bin/opera

[1 - novell]

[(V)iew Profile] / (U)se Profile / (C)reate New Profile / Abo(r)t / (F)inish

表示されたプロファイルを単純に使用したい場合は、U (*Use Profile*) を選択してから、上述のプロファイル生成 手順を実施します。

プロファイルを有効化する前に確認したい場合は、V (*View Profile*) を選択します。

既存のプロファイルを使用せずに無視したい場合は、C (*Create New Profile*) を選択してから、上述のとおり 何もない状態からのプロファイル生成手順を実施します。

- 6 aa-genprof を閉じ、今までに行なわれた全てのルール変更を保存するには、F (*Finish*) を選択します。

プロファイルをアップロードするように選択した場合は、短い変更履歴を 入力してリポジトリにアップロードします。

22.6.3.5 aa-logprof—システムログのスキャン

aa-logprof は、学習モードや不平モードで /var/log/audit/audit.log や /var/log/messages (auditd が動作していない場合) に出力されたログ項目を確認するための対話的なツールで、ここから AppArmor のセキュリティプロファイルに対する新しい項目を作成することができます。

aa-logprof を実行すると、学習モードや不平モードで生成されたログファイル を読み込み、既存のプロファイルセットではカバーしていない、新しい セキュリティイベントが存在しないかどうかを確認したあと、プロファイルの 修正について提案を表示します。学習モードや不平モードはプログラムの動作を 追跡してそれらをログに記録する仕組みであるため、aa-logprof はこの情報を 利用してプログラムの振る舞いを監視します。

制限下にあるプログラムが fork() したり他のプログラムを実行したりした 場合、aa-logprof はこれを検知して、子プロセスの起動時に使用する実行モード をユーザに問い合わせます。実行モードはそれぞれ *ix*, *px*, *Px*, *ux*, *Ux* の中から選択します。子プロセスに対して個別の プロファイルが存在する場合、既定の選択肢は

px になります。個別のプロファイルが存在しない場合は、既定値は *ix* になります。個別のプロファイルが存在する子プロセス に対しても aa-autodep を動作させ、AppArmor 内に読み込む必要があります。

aa-logprof が終了すると、プロファイルは更新されます。AppArmor モジュールが動作している場合は更新されたプロファイルが再読み込みされます。また、セキュリティイベントを生成しているプロセスが、その時点でも NULL-不平 モードで動作していた場合は、それらのプロセスは適切なプロファイルで動作 するように設定されます。

aa-logprof を実行するには、root でログインしている状態で端末ウィンドウ から aa-logprof と入力します。aa-logprof に対しては下記の ようなオプションを指定 できます：

```
aa-logprof -d /path/to/profile/directory/
```

プロファイルが標準のディレクトリである /etc/apparmor.d/ に存在しない場合は、プロファイルのフルパスを指定することができます。

```
aa-logprof -f /path/to/logfile/
```

ログファイルが標準のディレクトリである /var/log/audit/audit.log や /var/log/messages (auditd が動作していない場合) に存在しない場合は、ログファイルのフルパスを指定することができます。

```
aa-logprof -m "string marker in logfile"
```

aa-logprof に対してシステムログ内に開始点のマーキングを付けるように 指示します。aa-logprof は指定したマークよりも前にあるシステムログ については、全てのイベントを無視するようになります。なお、マークに スペースが含まれる場合、正しく動作させるには引用符を付けて指定しな ければなりません。たとえば下記ようになります：

```
aa-logprof -m "17:04:21"
```

or

```
logprof -m e2ff78636296f16d0b5301209a04430d
```

aa-logprof がログを読み出すと、それぞれ記録されたイベントに対して、どのように 処理すべきかを尋ねてきます。それぞれの質問には AppArmor ルールに 対する番号付きリストが表示され、その番号を選択することでルールに追加する ことができます ようになっています。

既定では aa-logprof は、/etc/apparmor.d/ 内で プロファイルを検索し、/var/log/messages 内で ログを検索します。多くの場合、root で aa-logprof を起動すれば、プロファイルを作成するには十分な環境となります。

ただし、ログのローテーション (今まで書き込んでいたログの書き込みを停止し、新しいログへの書き込みを始めること) 期間をまたいでいる場合など、アーカイブされたログファイルを検索する必要がある場合は注意が必要です。この場合は `zcat -f `ls -ltr /var/log/messages*` | aa-logprof -f -` のようにして実行してください。

22.6.3.6 aa-logprof 実行例 1

下記に示す例は、aa-logprof が httpd2-prefork のログを調査している場合の例で、`/etc/group` ファイルへのアクセスを行なっています。なお、`[]` は既定値を示しています。

下記の例では `/etc/group` へのアクセスが httpd2-prefork のネームサービスアクセスの一部として扱っています。ここでの適切な応答は `1` で、AppArmor ルール集として事前に定義されているものを取り込むように指定します。`1` である `#include` を選択すると、DNS 参照に分類される全ての質問のうち、今後発生するものをネームサービスのパッケージで全て解決するようになります。また、このようにすることで、DNS の設定を変更してもプロファイルが役に立たなくなるような事態を防ぐことができるほか、関連するネームサービス関連の設定をひとまとめにすることができます。

```
Profile: /usr/sbin/httpd2-prefork
Path:    /etc/group
New Mode: r
```

```
[1 - #include <abstractions/nameservice>]
2 - /etc/group
[(A)llow] / (D)eny / (N)ew / (G)lob / Glob w/(E)xt / Abo(r)t / (F)inish
```

それぞれ下記のように入力します:

Enter の入力

既定の選択肢を選択したものとみなします。この場合、指定したディレクトリパス項目に対して、アクセスを許可します。

許可

指定したディレクトリパス項目に対して、アクセスを許可します。AppArmor はファイルアクセス許可を提案しています。アクセス許可について、詳しくは 19.7 項「ファイルのアクセス許可とアクセスモード」(245 ページ)をお読みください。

拒否

指定したディレクトリパス項目に対して、アクセスを拒否します。AppArmor は次のイベントに移動します。

新規

このイベントに対して独自のルールを設定します。ここでは正規表現による記述を行なうことができます。入力した正規表現が表示されたイベントに当てはまらないようなものであった場合、AppArmor は確認メッセージを表示して再入力を促します。

Glob

より広いパスが含まれるようにするため、指定したパスを選択するか ワイルドカードでルールを作成します。提示されたパスの中から選択したい場合はパスの前に表示された番号を入力したあと、選択した項目 についてどのように扱うのかを指定します。

グロブの書式について、詳しくは 19.6項「パスとグロブ」(242 ページ)をお読みください。

拡張子付きグロブ

元々のディレクトリパスに対して、ファイル名の拡張子部分を保持する 形でのグロブを設定します。たとえば `/etc/apache2/file.ext` というディレクトリパスであった場合、ファイル名の本体部分にワイルドカード (アスタリスク) が挿入されて `/etc/apache2/*.ext` のようになります。これにより、プログラムは指定したディレクトリ内に存在し、かつ `.ext` という拡張子を持つ全てのファイルにアクセスできるようになります。

中止

aa-logprof を中断し、今までに行なわれた全てのルール変更を取り消し、全てのプロファイルを変更前の状態に戻します。

完了

aa-logprof を閉じ、今までに行なわれた全てのルール変更を保存します。

22.6.3.7 aa-logprof 実行例 2

たとえば vsftpd に対してプロファイル作成を行なうと、下記のような 質問が表示されます:

```
Profile: /usr/sbin/vsftpd
Path:    /y2k.jpg
New Mode: r
```

[1 - /y2k.jpg]

(A)llow / [(D)eny] / (N)ew / (G)lob / Glob w/(E)xt / Abo(r)t / (F)inish

この質問にはいくつか注目すべき項目があります。まず 1 つめには、openSUSE 上で動作する vsftpd は、既定では /srv/ftp 内にあるファイルを公開するはずなのに、トップレベルのディレクトリにある項目のアクセス可否を尋ねていることがあります。これは vsftpd が chroot を使用しているためで、コードの一部が chroot 環境内で動作するためです。AppArmor はグローバルな絶対パスではなく、chroot 環境下のパスとしてファイル アクセスを確認するためです。

2 つめに注目すべき項目としては、このディレクトリ内にある全ての JPEG ファイルに対して FTP からの読み込みアクセスを許可したい場合、*拡張子付きグロブ* を利用し、`/*.jpg` というパスを設定できる点です。このようにして個別の .jpg ファイルに対するアクセス許可ルールを集約し、今後発生する .jpg ファイルの質問に対して、前もって手を打つことができます。

最後に、FTP で提供されているファイルに対してより広範囲のアクセス許可を設定することができます。最後の行に書かれている *Glob* を選択すると、`aa-logprof` は表示されているパス `/y2k.jpg` を `/*` に置き換えることができます。またこれ以外にも、ディレクトリツリー全体に対してアクセス許可を設定したい場合は、*New* を選択してから `/**/*.jpg` と入力する (サブディレクトリを含む全ての .jpg ファイルにアクセスを許可する意味) か、もしくは `/**` と入力して (サブディレクトリを含む全てのファイルにアクセスを許可する意味) ください。

ここまでの説明では読み込みアクセスについて記述してきましたが、書き込み アクセスに対しても同様です。ただし書き込みアクセスに対しては、正規表現を使用する場合でもより注意して作っておいたほうが良いでしょう。また、実行許可の設定はより複雑なやり方になります。詳しくは 例22.1「学習モードの例外: 特定リソースに対するアクセス制御」(292 ページ) をお読みください。

下記の例では、`/usr/bin/mail` というメールクライアントに対してプロファイルを作成していて、`/usr/bin/mail` が `/usr/bin/less` をヘルパーアプリケーションとして呼び出し、長いメールメッセージを表示している例です:

```
/usr/bin/nail -> /usr/bin/less
(I)nherit / (P)rofile / (U)nconfined / (D)eny
```

ヒント

`/usr/bin/mail` の実際の実行ファイルは `/usr/bin/nail` です。印字ミスなどはありません。

プログラム `/usr/bin/less` は 1 画面以上にわたる テキストファイルに対して、これをスクロールさせながら読むのに適した シンプルなソフトウェアです。`/usr/bin/mail` が使用していることからわかるとおり、シンプルではありますがパワフルで広範囲の用途を持った、`tar` や `rpm` のようなヘルパーアプリケーションを使用することもできる ソフトウェアです。

ヒント

`tar` 形式のファイルや RPM ファイルに対して `less` を 実行すると、これらのファイルに含まれる内容を表示することができます。

たとえばメールメッセージを表示したとき、`rpm` コマンドを 自動では動作させたくない場合 (RPM ファイルはシステムプログラムを インストールしたり修正したりする機能があるため、Microsoft* Outlook スタイルの直接的なウイルス攻撃になります) は、*継承* を選択しておくのがベストです。これにより、この環境下で `less` プログラムを実行すると、`/usr/bin/mail` 向けのプロファイルで 動作するようになるため、下記のような影響があります：

- `/usr/bin/less` に対するファイルアクセス許可のうち、基本的なものを全てを `/usr/bin/mail` の プロファイルに追加する必要があります。
- `tar` や `rpm` のようなヘルパーアプリケーションを追加する必要がなくなります。これは `/usr/bin/mail` と同じ環境で `/usr/bin/less` を動作させるため、`less` が AppArmor の 保護無しで動作させるよりはずっと安全になるためです。

その他の状況下では、*プロファイル オプション*を使用する こともできます。これは `aa-logprof` に対して 2 つの影響があります：

- プロファイルに対して書き込まれるルールが `px` を使用するようになります。これは子プロセス独自のプロファイルに強制的に遷移させるための設定です。
- `aa-logprof` は子プロセスに対してもプロファイルを生成し、構築します。それは親のプロセスを構築したのと同じ方法で行なわれるもので、子プロセス に対して発生したイベントを、`aa-logprof` からの問い合わせをもとに、子プロセスのプロファイルに反映させる手順です。

制限下にあるプログラムが `fork()` したり他のプログラムを実行したりした 場合、`aa-logprof` はこれを検知して、子プロセスの起動時に使用する実行モード をユーザに問い合わせます。実行モードはそれぞれ継承、プロファイル、無制限、もしくは実行の拒否を選択できます。

また、子プロセスに対して個別のプロファイルが存在する場合、既定の選択肢はプロファイルになります。個別のプロファイルが存在しない場合は、既定値は継承になります。継承 (ix) について、詳しくは 19.7 項「ファイルのアクセス許可とアクセスモード」(245 ページ) をお読みください。

プロファイルの選択肢は、子プロセスに対して独自のプロファイル下で実行させることを指しています。2 つめの質問では、親から引き継ぐ子プロセス 向けの環境変数について、実行動作を変更することのできるものを取り除く かどうか が問い合わせられます。取り除く処理を有効にすると、お使いの AppArmor プロファイル内にある実行オプションに Px が設定されます。取り除く処理を選択しなかった場合は実行オプションに px が設定され、環境変数はそのまま引き継がれます。プロファイルを選択した場合の実行モードの既定値は px です。

無制限モードでの実行は非推奨であり、その他の選択ではどうしても動作するプロファイルを生成できない場合にのみ選択すべきものです。無制限オプションを選択すると、警告ダイアログが表示されて確認が求められます。メッセージを確認して **はい** を押すと、実行動作を変更することのできる 環境変数について、これらを取り除く かどうかを選択します。 **はい** を選択すると、お使いのプロファイル内の実行オプションに Ux が設定されます。 **いいえ** を選択すると実行モードは ux が設定されます。無制限モードでの既定値は Ux です。

重要: 無制限モードでの実行

ux の選択は非常に危険なものであり、プログラムの実行 動作に対して一切の強制力 (セキュリティ観点での) が働かなくなります。

22.6.3.8 aa-unconfined—保護されていないアクセスの検出

aa-unconfined コマンドは、お使いのシステムから ネットワークに対して開いているポートを検出し、システムに読み込まれているプロファイル集との比較を行います。その後、AppArmor プロファイルが 適用されていないネットワークサービスについて報告を行います。この コマンドの実行には root の権限が必要で、AppArmor のプロファイルでの 制限を受けてはいけません。

aa-unconfined は /proc ファイルシステムからプロセス の実行ファイルへのリンクを探すため、root で動作させなければ なりません。なお、このプログラムは下記の ような状況では正しく動作しない 場合もあります:

- 削除された実行ファイルが正しく処理されない場合がある

- netstat(8) を実行してから詳しいチェックを行なうまでの間にプロセスが終了した場合、正しく処理されない場合がある

注記

このプログラムは TCP と UDP を使用しているプロセスのみを列挙します。つまり、このプログラムは危険箇所を把握するためのソフトウェアではなく、検証環境などからネットワークにアクセスできる全てのプログラムを検出し、プロファイルを作成するために用意されているものです。

22.7 重要なファイル名とディレクトリ

下記には AppArmor のフレームワークで使用される、最も重要なファイルとディレクトリを列挙しています。お使いのプロファイルについて手作業で管理したりトラブルの原因を探ったりしたい場合は、あらかじめこれらのファイルやディレクトリについて知識を得ておいてください:

`/sys/kernel/security/apparmor/profiles`

現在読み込まれているプロファイル集を表示することのできる、仮想ファイルです。

`/etc/apparmor/`

AppArmor の設定ファイルが含まれるディレクトリです。

`/etc/apparmor/profiles/extras/`

AppArmor に同梱されているローカルのプロファイルリポジトリです。ただし有効化されていないことに注意してください。

`/etc/apparmor.d/`

プロファイルが配置されるディレクトリです。ファイル名の規則として、パス内の `/` を `.` に置き換えた (ただしルートディレクトリの `/` は除く) 名前を利用するようになっていて、管理しやすくなっています。たとえば `/usr/sbin/ntpd` に対するプロファイルは `usr.sbin.ntpd` という名前になります。

`/etc/apparmor.d/abstractions/`

アブストラクト (抽象) が配置されるディレクトリです。

`/etc/apparmor.d/program-chunks/`

プログラムチャンクが配置されるディレクトリです。

/proc/*/attr/current

このファイルを表示させることで、プロセスに対する制限状態を確認することが
できるほか、プロセスを制限するのにどのプロファイルが使用されているのかを
確認することができます。ps auxZ のように実行することで、この情報を自動
的に採取することもできます。

ハット変更を利用した Web アプリケーションのプロファイル作成

AppArmor® のプロファイルは個別のプログラムインスタンスやプロセスに対するセキュリティポリシーを表現するためのものです。これは実行可能なプログラムに対して適用されるものですが、プログラムの一部分で他の部分とは異なるアクセス許可を必要とする場合、プログラムに対して「ハット変更」を適用して、メインプログラムのアクセス許可とは異なるセキュリティコンテキストを使用するように設定することができます。これを **ハット** や **サブプロファイル** と呼びます。

ハット変更を利用すると、AppArmor のプロファイル内からプログラムに対するハットを変更することができます。これはプロセス単位のものよりも、詳しくセキュリティレベルを設定することができます。この機能を利用するには、各アプリケーションが「ハット変更に対応した」作りになっている必要があります。これは AppArmor モジュールに対して、セキュリティ領域を切り替えるように要求する機能のことを指します。ハット変更に対応したアプリケーションには、たとえば Apache Web サーバや Tomcat などがあります。

プロファイルには任意の数のサブプロファイルを設定することができますが、2 段階しか存在しないことに注意する必要があります。つまり、サブプロファイルに対するサブプロファイルは設定することができません。また、サブプロファイルは個別のプロファイルとして記述するもので、プロファイル名に続いて ^ を付け、続けてサブプロファイル名を記述します。なお、サブプロファイルは親のプロファイルと同じファイルに保存しなければなりません。

また、ハットを利用する場合のセキュリティは、完全な（ハットではない）プロファイルに比べるとかなり弱いものであることにも注意する必要があります。例を挙げると、もしも攻撃者がプログラムに対する完全なバグを見つけた場合、プロファイル内に含まれるハットの適用から逃げることでできてしまいます。これは、ハットのセキュリ

ティはそのプロセスが処理する機密鍵によって決定 される仕組みであり、ハット内で動作するコードからは鍵へのアクセス方法が 用意されないためです。そのためハット変更は、言語のインタプリタ (Perl, PHP, Java など) を含むアプリケーションサーバで使用するのをもっとも 便利です。このようなアプリケーションサーバでは、親プロセスのメモリに 直接アクセスできないような形でコードが分離されるため、ハットを利用するには都合が良いためです。

以降では、`mod_perl` と `mod_php` の各 Web サーバコンポーネントを含む Apache で、ハット変更機能を利用した場合について説明を行なっています。`mod_apparmor` に 似たアプリケーションモジュールを利用することで、他のアプリケーションサーバでも同様のアプローチでハット変更を実施できます。詳しくは 23.2.2 項「Location ディレクティブと Directory ディレクティブ」(315 ページ)をお読みください。

注記: さらなる情報

詳しくは `change_hat` のマニュアルページをお読みください。

23.1 Apache のハット変更

AppArmor では、Apache 向けのモジュールとして `mod_apparmor` (`apache2-mod_apparmor` パッケージ) が提供されています。このモジュールは Apache Web サーバをハット変更に対応させるためのものです。Apache とともにインストールしておいてください。

Apache がハット変更に対応していれば、受信した各 URI 要求に対して、下記の順序でカスタマイズ済みの AppArmor セキュリティプロファイルを確認します。

- URI 固有のハット。たとえば `^phpsysinfo/templates/classic/images/bar_left.gif` など。
- `DEFAULT_URI`
- `HANDLING_UNTRUSTED_INPUT`

注記: Apache の設定

`apache2-mod_apparmor` をインストール済みであれば、下記のコマンドを実行することで Apache 内にモジュールを読み込むことができます:

```
a2enmod apparmor
```

23.1.1 ハット変更に対応したアプリケーションの管理

AppArmor の大部分のツールと同様に、ハット変更に対しても YaST とコマンド ラインツールの 2 種類の方法で管理を行なうことができます。ハット変更に対応したアプリケーションをコマンドラインから管理すると、非常に柔軟に対応することができますが、作業は少し複雑になります。いずれの方法でもアプリケーションのハットを管理できるほか、プロファイル項目への反映も 行なうことができます。

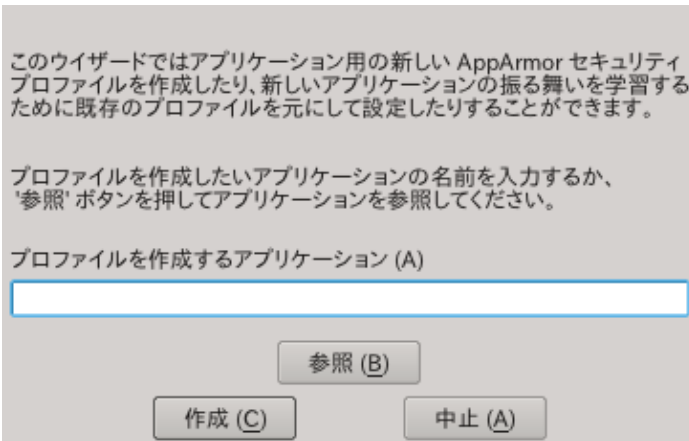
下記に示す手順は、YaST を利用して Apache のプロファイルにハットを追加するまでのデモンストレーションです。AppArmor プロファイルウィザードでは、AppArmor のプロファイル作成ユーティリティが個別の URI 要求に対して、新しいハットを作成するかどうかを尋ねます。新しいハットを作成するように 選択すると、それぞれの URI に対して個別のプロファイルを作成することができます。これにより、要求ごとにより厳しいルールを作成することができます。

処理された URI が特に意味のないものであったり、特にセキュリティリスクの 考えられるものではなかったりした場合は、既定のハット (つまり既定の セキュリティプロファイル) で対象の URI を処理するようにするため、*既定のハットの使用* を選択しておくことができます。

今回の例では、phpsysinfo という URI とそれに続くアクセスに対して、新しいハットを作成しています。プロファイル作成ユーティリティを使用することで、新しいハットに追加すべきものを代理で設定させることも できます。結果として生成されるハットは、Apache Web サーバで phpsysinfo という URI が処理された際、対象のサーバで行なわれる全ての処理に対して適用される、セキュリティの厳しいコンテナになります。

URI は phpsysinfo (詳しくは <http://phpsysinfo.sourceforge.net> をお読みください) というアプリケーションを実行します。phpsysinfo パッケージは新規インストールの openSUSE と AppArmor に対してインストールされたものとし、/srv/www/htdocs/phpsysinfo ディレクトリに配置されているものと仮定します。

- 1 phpsysinfo をインストール すれば Apache プロファイル内にハットを追加する準備は完了です。AppArmor の GUI から AppArmor プロファイルウィザードを選択してください。
- 2 プロファイルを作成するアプリケーション では、httpd2-prefork と入力します。
- 3 プロファイルの作成 を押します。



- 4 端末ウィンドウで `rcapache2 restart` と入力し、Apache を再起動します。
また、この時点でプロファイルを作成している全てのプログラムを再起動します。
- 5 Web ブラウザのウィンドウで `http://localhost/phpsysinfo/` と入力します。ブラウザのウィンドウにはネットワークの使用率とシステム情報が表示されます。

注記: データのキャッシュ

リクエストがサーバ側で間違いなく処理され、キャッシュされたデータを閲覧してしまうようなことをなくすため、お使いのブラウザではページの更新を行ってください。これを行なうには、ブラウザの **更新** ボタンを押します。

- 6 システムログをスキャンして *AppArmor イベントを検出* を押します。AppArmor は `aa-logprof` ツールを起動し、上記の手順で学習した情報を読み出します。これにより、プロファイルへの設定について質問が表示されます。
- 7 `aa-logprof` では、まず `phpsysinfo` の URI にアクセスされたことを検知し、**要求されたハットの追加** か **既定のハットの使用** のどちらかを選択するように促します。ここでは **要求されたハットの追加** を選択してください。
- 8 続いて **許可** を押します。

上記で **要求されたハットの追加** を選択すると、プロファイル内に新しいハットを作成し、続く質問の回答は、このアプリケーションに対する既定のハットではなく、新しく作成したハットに対して反映するようになります。

次の画面では、AppArmor はスクリプトが実行した外部プログラムを表示します。ここでは phpsysinfo のハット (*継承* を選択した場合) で 制限するか、もしくは個別のプロファイルで制限する (*プロファイル* を選択した場合) かを選択することができ、セキユリティプロファイルを 一切適用せずに無制限の動作を許す (*無制限*) 選択を行なうことができます。*プロファイル* の選択肢を選択した場合は、そのプログラムに対するプロファイルが存在しない場合、新規に作成することになります。

注記: セキユリティ面の考慮事項

無制限 を選択すると、それは明示的なセキユリティ ホールを作成することになってしまいます。注意してお使いください。

8a /bin/bash のパスに対しては *継承* を選択します。これにより /bin/bash (Apache からアクセスされるもの) は phpsysinfo のハットプロファイルが適用され、必要な許可が行なわれるようになります。

8b 続いて *許可* を押します。

9 あとの質問は新しいハットを生成する旨の問い合わせと、プロファイルやハットに追加する項目の問い合わせです。プロファイルに対する項目の追加については、21.1 項「ウィザードを使用したプロファイルの追加」(261 ページ) で詳細を説明しています。

全てのプロファイル生成時の質問に回答したら、*完了* を押すと設定を保存し、ウィザードを終了することができます。

下記は phpsysinfo のハット作成例です。

例 23.1 phpsysinfo のハット例

```
/usr/sbin/httpd2-prefork {
...
^phpsysinfo {
    #include <abstractions/bash>
    #include <abstractions/namespace>

    /bin/basename          ixr,
    /bin/bash               ixr,
    /bin/df                 ixr,
    /bin/grep               ixr,
    /bin/mount              Ux,
    /bin/sed                ixr,
```

```

/dev/bus/usb/                r,
/dev/bus/usb/**              r,
/dev/null                    w,
/dev/tty                      rw,
/dev/urandom                  r,
/etc/SuSE-release             r,
/etc/ld.so.cache              r,
/etc/lsb-release              r,
/etc/lsb-release.d/          r,
/lib/ld-2.6.1.so              ixr,
/proc/**                      r,
/sbin/lspci                   ixr,
/srv/www/htdocs/phpsysinfo/** r,
/sys/bus/pci/**               r,
/sys/bus/scsi/devices/        r,
/sys/devices/**               r,
/usr/bin/cut                   ixr,
/usr/bin/getopt                ixr,
/usr/bin/head                  ixr,
/usr/bin/lsb_release           ixr,
/usr/bin/lsscsi                ixr,
/usr/bin/tr                    ixr,
/usr/bin/who                   ixr,
/usr/lib/lib*so*               mr,
/usr/lib/locale/**             r,
/usr/sbin/lusb                  ixr,
/usr/share/locale/**           r,
/usr/share/pci.ids             r,
/usr/share/usb.ids             r,
/var/log/apache2/access_log    w,
/var/run/utmp                  kr,
}
}

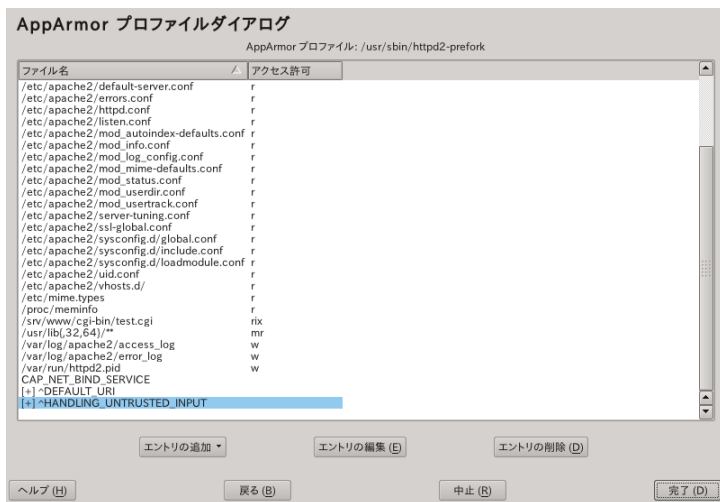
```

注記: ハットと親プロファイルの関係

プロファイル `^phpsysinfo` は、親プロファイルである `httpd2-prefork` のプロセスが実行中である場合にのみ有効となります。

23.1.2 ハットの追加とハットへの項目追加

プロファイルの編集 を使用した場合 (手順については 21.3 項「プロファイルの編集」(269 ページ) を参照してください) や *手作業でプロファイルを追加* (手順については 21.2 項「手作業でのプロファイル追加」(268 ページ) を参照してください) を使用した場合は、作業中の AppArmor プロファイルに対してハット (サブプロファイル) を追加することができます。下記のような *AppArmor プロファイルダイアログ* から、ハット 変更のサブプロファイルを追加してください。



- 1 AppArmor プロファイルダイアログ のウィンドウから、項目の追加 を押したあと ハット を選択します。すると、ハット名の入力 ダイアログが表示されます:



- 2 AppArmor のプロファイルに追加するハット名を入力します。名前はハット内で許可セットを受け取る際の URI を指定します。
- 3 ハットの作成 を押します。すると AppArmor プロファイルダイアログ の画面に戻ります。
- 4 新しいハットを追加したら、完了 を押せば終了です。

注記: さらなる情報

AppArmor プロファイルの例について、詳しくは 例23.1「phpsysinfo のハット例」(311 ページ)をお読みください。

23.2 mod_apparmor 向けの Apache 設定

Apache はテキスト形式の設定ファイルに対してディレクティブを配置することで設定を行ないます。主となる設定ファイルは通常、`httpd.conf` です。この設定ファイル名は Apache をコンパイルする際、場所を指定することができます。また Apache の動作を変更するためのディレクティブは、これらの設定ファイル内のいずれかに配置します。主となる設定ファイルに対して変更を行なった場合は、Apache を起動または再起動することで設定が認識されるようになります。

23.2.1 仮想ホストのディレクティブ

仮想ホストのディレクティブでは、パス名情報とそれに続く実際のファイル名について、これを受け入れるか拒否するかを制御します。仮想ホストのディレクティブに関する Apache のドキュメンテーションは、<http://httpd.apache.org/docs/2.2/mod/core.html#virtualhost> をお読みください。

ハット変更関係の設定キーワードは `AADefaultHatName` です。これは `AAHatName` に似た仕組みで、たとえば `AADefaultHatName My_Funky_Default_Hat` のような形式で既定のハット名を指定します。

上記の設定オプションはサーバディレクティブ内に書かれるべきもので、他のオプションの外側でも使用することができます。これにより、既定のサーバに対するハット設定を行なうことができます。仮想ホストは Apache の内部で別々の「サーバ」として扱われるため、既定のサーバに対する既定のハット名のほか、必要であれば各仮想ホストに対する既定のハット名を設定することもできます。

リクエストが到達すると、`mod_apparmor` は下記の手順で適用すべきハットを判断します。

1. `Location` や `Directory` ディレクティブ内で `AAHatName` キーワードが指定されていれば、そのハット
2. URI パス全体に対して命名されているハット
3. `AADefaultHatName` キーワードで指定されている既定のサーバのハット。
4. `DEFAULT_URI` (何も指定されていない場合は「親」となる Apache のハット)

23.2.2 Location ディレクティブと Directory ディレクティブ

設定ファイル内の Location と Directory の各ディレクティブでは、ハット名を指定することができ、そこから対応するセキュリティ用のハットを選択することができます。Apache の場合、Location や Directory のディレクティブに関する説明は <http://httpd.apache.org/docs/2.2/sections.html> に書かれています。

下記のような Location ディレクティブを指定すると、mod_apparmor に対して指定のハットを使用するように指示することができます:

```
<Location /foo/> AAHatName MY_HAT_NAME </Location>
```

上記の例では、/foo/ で始まる全ての URI (/foo/, /foo/bar, /foo/cgi/path/blah_blah/blah など) に対して、MY_HAT_NAME というハットを適用するように指示することになります。

Directory ディレクティブでも Location ディレクティブと同様に動作します。ただし下記の例のとおり、Directory ディレクティブではファイルシステム内のパスを参照する点が異なります:

```
<Directory "/srv/www/www.immunix.com/docs">
# 最後のスラッシュが無いことに注意
AAHatName immunix.com
</Directory>
```

例: 下記の例では、phpsysinfo というプログラムを例にして Location ディレクティブを使用しています。phpsysinfo のソースコードは、<http://phpsysinfo.sourceforge.net> からダウンロードできます。

- 1 ソースコードをダウンロードしたら、/srv/www/htdocs/phpsysinfo ディレクトリ内にインストールしてください。
- 2 /etc/apache2/conf.d/phpsysinfo.conf ファイルを作成し、下記のテキストを入力します:

```
<Location "/phpsysinfo">
AAHatName phpsysinfo
</Location>
```

あとは下記のようなハットを設定すれば、phpsysinfo に対して動作するようになります:

```
/usr/sbin/httpd2-prefork {
```

```

...
~phpsysinfo {
    #include <abstractions/bash>
    #include <abstractions/nameservice>

    /bin/basename                ixr,
    /bin/bash                    ixr,
    /bin/df                      ixr,
    /bin/grep                    ixr,
    /bin/mount                   Ux,
    /bin/sed                     ixr,
    /dev/bus/usb/                r,
    /dev/bus/usb/**              r,
    /dev/null                    w,
    /dev/tty                     rw,
    /dev/urandom                 r,
    /etc/SuSE-release            r,
    /etc/ld.so.cache             r,
    /etc/lsb-release            r,
    /etc/lsb-release.d/         r,
    /lib/ld-2.6.1.so            ixr,
    /proc/**                     r,
    /sbin/lspci                 ixr,
    /srv/www/htdocs/phpsysinfo/** r,
    /sys/bus/pci/**              r,
    /sys/bus/scsi/devices/       r,
    /sys/devices/**             r,
    /usr/bin/cut                 ixr,
    /usr/bin/getopt              ixr,
    /usr/bin/head                ixr,
    /usr/bin/lsb_release         ixr,
    /usr/bin/lsscsi             ixr,
    /usr/bin/tr                  ixr,
    /usr/bin/who                 ixr,
    /usr/lib/lib*so*             mr,
    /usr/lib/locale/**          r,
    /usr/sbin/lsusb             ixr,
    /usr/share/locale/**        r,
    /usr/share/pki.ids          r,
    /usr/share/usb.ids           r,
    /var/log/apache2/access_log  w,
    /var/run/utmp                kr,
}
}

```

- 3 root で端末ウィンドウを開き、`rcapparmor restart` と入力して AppArmor プロファイルの再読み込みを行います。
- 4 同じく root で端末ウィンドウから、`rcapache2 restart` と入力し、Apache を再起動します。

- 5 Web ブラウザを開いて `http://ホスト名/phpsysinfo/` と入力し、phpsysinfo が配信するシステム情報を表示させます。
- 6 設定エラーについては `/var/log/audit/audit.log` ファイルに書かれるほか、`dmesg` で拒否された情報を確認することができます。

pam_apparmor によるユーザに対する制限

AppArmor のプロファイルは実行されるプログラムに対して適用されるものです。もしも 特定のプログラムのうち一部分で、他の部分とは異なるアクセス許可を必要とする 場合は、プログラムに対してハット変更を適用することでハット (サブプロファイル と呼ばれる場合もあります) という役割を変更することができます。pam_apparmor という PAM モジュールでは、グループ名やユーザ名をベースにしたサブプロファイルを 適用するかどうかを設定することができます。なお、pam_apparmor を利用するには、あらかじめ PAM のセッションモジュールとして登録しておく必要があります。

pam_apparmor パッケージは既定ではインストールされません。YaST または zypper を利用してインストールしてください。pam_apparmor の設定方法については、パッケージに同梱されている /usr/share/doc/packages/pam_apparmor/README をお読みください。また、PAM について詳しくは 第2章 *PAM を利用した認証* (17 ページ) をお読みください。

pam_apparmor では役割ベースのアクセス制御 (RBAC) を設定することができます。RBAC の設定方法について、詳しくは <http://wiki.apparmor.net/index.php/AppArmorRBAC> をお読みください。

プロファイルを作成したアプリケーションの管理

25

プロファイルを作成してアプリケーションに免疫を付与したら、AppArmor® のプロファイル管理 (これにはログファイルの分析のほか、プロファイルの更新やプロファイル群のバックアップが含まれます) を行なうことで、お使いの openSUSE® がより効果的かつ適切に保護されるようになります。また、電子メールによるイベント通知を設定し、定期的なレポート作成を行なったあと、YaST 経由で aa-logprof ツールを起動してシステムログ項目からプロファイルの更新を行ない、メンテナンス上の問題に対策を打つことで、それらが大きな問題となる前に対策を打つこともできます。

25.1 セキュリティ拒否イベントへの対応

セキュリティ拒否イベントを受け取った場合は、アクセス違反の詳細を調査し、攻撃など予期せぬものであるのか、それともアプリケーションが通常どおりに動作した結果であるのかを判別する必要があります。この判断を行なうには、アプリケーション固有の知識が必要となります。もしも拒否動作が通常のアプリケーションの動作に由来するものであった場合は、コマンドラインから aa-logprof を実行するか、もしくは AppArmor の *プロファイルの更新ウィザード* を利用して、プロファイルを更新してください。

もしも拒否動作がアプリケーションの通常動作に由来するものでないと判断される場合は、そのアクセスはシステムへの不正侵入をしようとしたもの (ただし侵入は阻止されています) であると言えます。そのため、この通知をご利用の環境におけるセキュリティ責任者に転送する必要があります。

25.2 セキュリティプロファイルの管理

本番の環境では、配置されているすべてのアプリケーションに対するプロファイルについて、これらを管理することを検討すべきです。また、プロファイルなどのセキュリティポリシーは、お使いの環境で必要不可欠なものであるため、バックアップの採取やその復元、ソフトウェアの変更やそれに伴うセキュリティポリシーの修正を計画する必要があります。

25.2.1 セキュリティプロファイルのバックアップ

プロファイルのバックアップを行なうことで、ディスクがクラッシュした場合でもプロファイルをやり直す必要がなくなります。また、プロファイルを変更した場合も、バックアップされたプロファイルから元のプロファイルに戻すことができます。

プロファイルのバックアップは、単純にプロファイルのファイルを指定したディレクトリにコピーすることで行ないます。

- 1 まずはプロファイルを単一の書庫ファイルにまとめます。これを行なうには、`root` で端末ウィンドウを開き、下記のコマンドを入力します：

```
tar zcplf profiles.tgz /etc/apparmor.d
```

お使いのセキュリティポリシーのファイルを定期的にバックアップするのに、もっとも簡単な方法は、`/etc/apparmor.d` ディレクトリを指定してすべてのファイルを書庫にまとめることです。

- 2 `scp` や `Konqueror` または `Nautilus` のようなファイルマネージャを利用し、ストレージメディアやネットワーク、もしくは他のコンピュータなどにファイルを保存すれば作業は完了です。

25.2.2 セキュリティプロファイルの変更

セキュリティプロファイルの修正や変更は、お使いのシステムでアプリケーションに対するセキュリティを変更したい場合に行ないます。AppArmor でプロファイルを修正する方法については、21.3項「プロファイルの編集」(269 ページ)をお読みください。

25.2.3 お使いの環境に対するソフトウェアの導入

アプリケーションの新バージョンをインストールした場合や、お使いのシステムに修正をインストールした場合は、必要に応じてプロファイルを更新する必要があります。この更新作業にはいくつかの選択肢が存在し、それらはお使いの環境におけるソフトウェア配置ポリシーによって決まります。また、修正やアップグレードはテスト環境に配置してから本番の環境に配置することもできます。下記では、それぞれのやり方について説明しています。

テスト環境に修正やアップグレードを配置したい場合、プロファイルの更新は下記のいずれかの方法で実施するのが最適です:

- YaST から *AppArmor* プロファイルウィザードを選択し、プロファイル作成を行なう方法。この方法では、追加されたアプリケーションや修正を適用したアプリケーションに対して、新しいプロファイルを作成することができます。詳しい手順については 21.1 項「ウィザードを使用したプロファイルの追加」(261 ページ)をお読みください。
- root で端末ウィンドウを開き、aa-genprof と入力して aa-genprof を実行する方法。詳しい手順については、22.6.3.4 項「aa-genprof—プロファイルの生成」(290 ページ)をお読みください。

修正やアップグレードを本番環境に直接インストールしたい場合、プロファイルの更新は下記のいずれかの方法で実施するのが最適です:

- プロファイルに追加すべき新しい拒否事例が存在しないかどうか、システムを頻繁に確認し、必要であれば aa-logprof を実行して更新する方法。詳しい手順については、22.6.3.5 項「aa-logprof—システムログのスキャン」(298 ページ)をお読みください。
- YaST の *プロファイルの更新ウィザード* を実行し、新しい動作を学習させる方法 (すべてのアクセスが許可され記録される仕組みであるため、リスクの高い方法です)。詳しい手順については、21.5 項「ログ項目からのプロファイル更新」(275 ページ)をお読みください。

サポート

この章では、メンテナンス関連の作業について説明しています。たとえば AppArmor® の更新方法のほか、AppArmor が提供する各種コマンドラインツールの使用方法についてマニュアルページを読む方法などが書かれています。また、トラブルシューティングの章では、AppArmor を使用するにあたってよく発生する問題と、それらの解決策について説明しています。それ以外にも、AppArmor に対する不具合や機能拡張リクエストなどの説明もあります。

26.1 AppArmor のオンライン更新

AppArmor パッケージに対する更新は、その他の openSUSE 向け更新パッケージと同じ方法で提供されます。そのため、他のパッケージと同様に 取得し適用してください。

26.2 マニュアルページの使用

ヘルプの一部としてマニュアルページが用意されています。端末ウィンドウから `man apparmor` と入力すると、apparmor のマニュアルページを表示することができます。マニュアルページには 1 から 8 までのセクションが存在していて、それぞれ下記のような分類になっています:

表 26.1 マニュアルページ: セクションと分類

セクション	分類
1	ユーザ向けのコマンド
2	システムコール
3	ライブラリ関数
4	デバイスドライバの情報
5	設定ファイルの書式
6	ゲーム類
7	高レベルなコンセプト説明
8	管理者向けコマンド

このセクション番号は、それぞれのマニュアルページを区別するために使用します。たとえば `exit(2)` というマニュアルページは、`exit` という名前のシステムコールを説明していますが、`exit(3)` というマニュアルページでは、`exit` という名前の C 言語のライブラリ関数を説明しています。

AppArmor におけるマニュアルページは下記のとおりです:

- `unconfined(8)`
- `autodep(1)`
- `complain(1)`
- `enforce(1)`
- `genprof(1)`
- `logprof(1)`
- `change_hat(2)`
- `logprof.conf(5)`

- `apparmor.conf`(5)
- `apparmor.d`(5)
- `apparmor.vim`(5)
- `apparmor`(7)
- `apparmor_parser`(8)

26.3 さらなる情報

AppArmor 製品について、より詳しい情報は <http://wiki.apparmor.net> をお読みください。また、この文書を含む AppArmor 製品の文書をお読みにになりたい場合は、インストール済みシステムで `/usr/share/doc/manual` ディレクトリを開いてください。

AppArmor について、開発者とユーザの間で対話できるよう開設されている メーリングリストがあります。詳しくは <https://lists.ubuntu.com/mailman/listinfo/apparmor> をお読みください。なお、いずれのメーリングリストとも英語によるものであることにご注意ください。

26.4 トラブルシューティング

この章では、AppArmor でよく発生する問題やエラーメッセージについて説明しています。

26.4.1 奇妙なアプリケーション動作への対応

アプリケーションの動作が奇妙なものになってしまった場合や、アプリケーションを利用して何らかの問題が発生した場合、まずは AppArmor がアプリケーションを制限しすぎていないかどうかを調べるため、ログファイル内の拒否メッセージを確認してください。アプリケーションに対する AppArmor での制限が厳しすぎることによって拒否メッセージが生成されていた場合は、プロファイルを更新して正しく制限されるようにする必要があります。これを行なうには、YaST から **プロファイルの更新ウイザード** を利用して行ないます。詳しくは 21.5 項「ログ項目からのプロファイル更新」(275 ページ) をお読みください。

AppArmor の保護無しでアプリケーションやサービスを動作させたい場合は、`/etc/apparmor.d` ディレクトリ内にあるアプリケーションの プロファイルを削除するか、別の場所に移動させてください。

26.4.2 プロファイルが動作しなくなってしまった場合

以前のバージョンの AppArmor をお使いの場合で、お使いのシステムを更新した 場合 (ただしプロファイルは従来のまま) は、以前にきちんと動作していた アプリケーションが正しく動作しなくなったり、場合によっては 全く動作しなくなったりする場合があります。

このバージョンの AppArmor では、プロファイルの文法に対して新しい機能が導入されているため、古いバージョンの AppArmor プロファイルでは、AppArmor のツールを利用するにあたって、問題が発生する場合があります。対象となる機能は下記のとおりです:

- ファイルロック (施錠)
- ネットワークアクセス制御
- SYS_PTRACE 機能 (ケーパビリティ)
- ディレクトリパスへのアクセス

現在のバージョンの AppArmor では、ファイルロック (施錠) を取り扱うことができるようになっていて、(k) という許可モードが用意されています。ファイルロックを必要とするアプリケーションが、古いバージョン向けのプロファイル で制限されている場合、ファイルロックの許可が設定されていないために誤った 動作をしてしまったり、失敗してしまったりする場合があります。このような場合は、`/var/log/audit/audit.log` ファイル内に下記のような項目が 出力されているはず:

```
type=APPARMOR_DENIED msg=audit(1188913493.299:9304): operation="file_lock"
requested_mask="::k" denied_mask="::k" fsuid=1000 name="/home/tux/.qt/.qtrc.lock"
pid=25736 profile="/usr/bin/opera"
```

上記のような項目を確認したら、あとは YaST のプロファイル更新ウィザードや `aa-logprof` コマンドを利用し、プロファイルを更新してください。

また、新しいネットワークアクセス制御文法は、ネットワークのファミリと種類を ベースにしているため (詳細は 19.5 項「ネットワークアクセス制御」(241 ページ) をお

読みください)、アプリケーションが誤動作してしまったり、全く動作しなかったりする場合があります。ネットワーク関連のアプリケーションが正しく動作しない場合は、`/var/log/audit/audit.log` ファイル内に下記のような項目が出力されていることを確認してください:

```
type=APPARMOR_DENIED msg=audit(1188894313.206:9123): operation="socket_create"
family="inet" sock_type="raw" protocol=1 pid=23810 profile="/bin/ping"
```

上記のログ項目は `/bin/ping` というコマンドでの例ですが、ネットワークの接続を開くにあたって AppArmor の許可が得られなかったことを示しています。許可はアプリケーションがネットワークにアクセスするにあたって、明示的に設定されていなければなりません。YaST のプロファイル更新 ウィザードや `aa-logprof` コマンドを利用し、プロファイルを更新してください。

また、あるプロセスが `/proc/pid/fd/*` 内にあるファイルにアクセスしようとすると、現在のカーネルは `SYS_PTRACE` の機能を要求します。新しいプロファイルではファイルと機能の両方に対して項目を設定する必要がありますが、古いプロファイルでは、下記のようなファイル項目だけが作成されているはずです:

```
/proc/*/fd/** rw,
```

上記のような項目は、新しいプロファイルに変換するにあたって下記のように書き換えられるべきです:

```
capability SYS_PTRACE,
/proc/*/fd/** rw,
```

プロファイルを新しい文法に更新するには、YaST のプロファイル更新ウィザードを利用するか、もしくは `aa-logprof` コマンドをお使いください。

また、このバージョンの AppArmor では、プロファイルルールの文法が変更されていて、ディレクトリへのアクセスとファイルへのアクセスを明示的に区別するようになっています。そのため、以前のバージョンで利用していたルールのうち、ファイルとディレクトリの両方に該当するようなルールがあった場合、現在のバージョンではファイルパスにしか該当しないようになります。これにより AppArmor は必要なディレクトリに対してアクセスを拒否するようになるため、アプリケーションが誤動作したり、様々なログメッセージを生成する結果になってしまったりする場合があります。下記の例では、パス文法について最も重要な変更点を説明します。

古い文法では、下記のようなルールを指定することで、`/proc/net` 以下にあるファイルとディレクトリの両方に、アクセスすることができるようになっていました。このルールではディレクトリ内にあるファイルを読み込むことは許可されますが、そのディレクトリ内にあるファイルやディレクトリへのアクセスは許可されません。たとえば `/proc/net/dir/foo` のようなファイルやディレクトリはアスタリスク (*) の指定がある

ため、該当するようになるはずですが、foo は dir 以下にあるファイルまたはディレクトリであるため、アクセスは許可されません。

```
/proc/net/* r,
```

新しい文法を利用して同じような動作をさせたい場合は、2 つのルールに分割して指定する必要があります。1 つめに /proc/net 内にあるファイルに対してアクセス許可を設定し、2 つめに /proc/net 内にあるディレクトリに対してアクセス許可を設定します。ディレクトリへのアクセスは内容の一覧にのみ使用することができるもので、その中にある実際のファイルやディレクトリに対する許可には使用できません。

```
/proc/net/* r,  
/proc/net/*/ r,
```

また下記のルールは、古い文法と新しい文法の両方で似たような動作をするものです。これは /proc/net 以下にあるファイルとディレクトリの両方にアクセスを許可します:

```
/proc/net/** r,
```

新しい文法で、上記の表現を利用してファイルとディレクトリのアクセスを区別したい場合は、下記のような 2 つのルールを設定します。1 つめのルールで /proc/net 以下にあるディレクトリに対して再帰的なアクセス許可を設定し、2 つめのルールでファイルにのみ再帰的なアクセスを許可しています。

```
/proc/net/**/ r,  
/proc/net/**[^/] r,
```

また、下記のルールは古い文法と新しい文法の両方で似たような動作をするもので、/proc/net 以下にあって foo で始まるファイルやディレクトリに対し、アクセスを許可します:

```
/proc/net/foo** r,
```

新しい文法でファイルアクセスとディレクトリアccessを区別したい場合は、** というグロブパターンを使用して 2 つのルールを設定してください。1 つめのルールは、古い文法でファイルとディレクトリの両方に該当するルールですが、新しい文法では最後のスラッシュが書かれていないため、ファイルにのみ該当するルールになります。2 つめのルールは古い文法ではファイルとディレクトリのどちらにも該当しないルールですが、新しい文法ではディレクトリにのみ該当するルールになります:

```
/proc/net/**foo r,  
/proc/net/**foo/ r,
```

下記のルールは、? のグロブの使い方がどのように変更されたのかを示すものです。古い文法では 1 つめのルールはファイルとディレクトリの両方に該当するル

ルになります (4 文字で、最後の文字がスラッシュである 場合もあります)。新しい文法では、1 つめのルールはファイルにのみ該当する ルールになります (最後のスラッシュが無いため)。また、2 つめのルールは 古い文法ではいずれにも該当しないルールですが、新しい文法ではディレクトリ にのみ該当するルールになります。最後のルールは `/proc/net/foo?` 内にある `bar` ファイルにのみ該当するルールですが、古い文法ではファイルとディレクトリの両方に該当するルールになります:

```
/proc/net/foo? r,  
/proc/net/foo?/ r,  
/proc/net/foo?/bar r,
```

文法の仕様変更に関連する問題を発見して解決するには、更新作業後にしばらく時間をかけてプロファイルを確認し、各アプリケーションに対して下記の手順を 行ないます:

- 1 AppArmor が動作していて、アプリケーションのプロファイルが読み込まれていることを確認します。
- 2 YaST の AppArmor コントロールパネルを起動し、アプリケーションのプロファイル を不平モードに切り替えます。現在のプロファイルに対する違反事例に対してログが効くされるものの、プロファイルは強制されることはなく、アプリケーションの動作が阻害されることはありません。
- 3 そのアプリケーションに対して期待する、すべての処理を実行します。
- 4 YaST のプロファイル更新ウィザードを起動し、アプリケーションの起動中に 生成されたログ項目をプロファイルに反映させます。
- 5 プロファイルを更新したら、再度 YaST の AppArmor コントロールパネルから強制モードに切り替えます。

AppArmor のコマンドラインツールを使用する場合、下記の手順で行ないます:

- 1 まずはアプリケーションのプロファイルを不平モードに切り替えます:

```
aa-complain /path/to/application
```

- 2 アプリケーションを実行します。

- 3 アプリケーションの起動中に生成されたログ項目をプロファイルに反映させます:

```
aa-logprof /path/to/application
```

4 プロファイルを強制モードに戻します:

```
aa-enforce /path/to/application
```

26.4.3 AppArmor で KDE アプリケーションを制限する方法

現時点では、KDE は独自のプロセス管理方法をとっているため、他のアプリケーションと同じ方法では KDE アプリケーションを制限することができません。

KDE アプリケーションを制限したい場合は、下記に示すいずれかのアプローチで制限を行なってください。ただし、下記に示すものはいずれも、標準のセットアップには 適しません:

KDE デスクトップ全体に対して単一のプロファイルを作成する方法

すべての KDE プロセスは特定の親に対する子プロセスであるため、AppArmor では 個別のアプリケーションプロセスを識別することができません。そのため、巨大な 1 つのプロファイルを作成して、一括でデスクトップを制限する必要があります。このアプローチは用途が限定されている (キオスク型の) ような場合 にのみ有用なものであり、標準的な KDE デスクトップ (アプリケーション類を含む) に対してプロファイルを管理するのは不可能です。

KDE のプロセス処理を修正する方法

KDE_EXEC_SLAVES=1 と KDE_IS_PRELINKED=1 の変数を指定することで、KDE に対して個別のアプリケーションプロセスを 識別できるようにする方法です。このアプローチを利用すると、お使いの デスクトップ環境が目に見えて遅くなってしまうため、速度面の要件が厳しい 場合には不適です。なお上記の環境変数は、KDM/XDM/GDM や startx を起動 する前に設定する必要があります。これを設定するための方法として、/etc/security/pam_env.conf 内にそれらを記述する 方法があります。

26.4.4 Apache での問題を解決する方法

Apache に対して新しいモジュールをインストールしたり、設定を変更したりした 場合には、Apache が正しく起動しなかったり Web ページを正しく提供しなかったり

する場合があります。追加のモジュール (たとえば `apache2-mod_apparmor` など) をインストールした場合や、Apache の設定を変更した場合は、プロファイルに追加すべき項目を判断させるため、プロファイルを作成し直してください。

26.4.5 使用中のプロファイル群から特定のプロファイルを除外する方法

特定のプログラムに対するプロファイルを無効化するには、`aa-disable PROGRAMNAME` を実行します。ここで、`PROGRAMNAME` にはプログラムの名前を指定します。このコマンドは `/etc/apparmor.d/disable/` 内にシンボリックリンクを作成することで無効化しますので、無効化を解除したい場合は単純にシンボリックリンクを削除してください。

26.4.6 インストールされていないアプリケーションのプロファイルを管理する方法

AppArmor でプロファイルを管理するには、アプリケーションが動作しているシステムのログファイルに対して、アクセスを行なう必要があります。ログファイルにさえアクセスできれば、プロファイルを利用してアプリケーションを起動させる必要はありません。一方のシステムでアプリケーションを動作させ、ログ (`audit` がインストールされていれば `/var/log/audit.log`、インストールされていなければ `/var/log/messages`) をプロファイル構築側のホストに転送し、`aa-logprof -f path_to_logfile` を実行してください。

26.4.7 AppArmor の文法エラーを見つけて対処する方法

AppArmor を手作業で修正した場合、文法エラーが発生する場合があります。お使いのプロファイル内に文法エラーがある状態で AppArmor を起動したり再起動したりすると、エラーが表示されます。たとえば下記のように文法エラーが表示されま

```
localhost:~ # rcapparmor start
Loading AppArmor profiles AppArmor parser error in /etc/apparmor.d/usr.sbin.squid at line
410: syntax error, unexpected TOK_ID, expecting TOK_MODE
Profile /etc/apparmor.d/usr.sbin.squid failed to load
```

AppArmor の YaST ツールを利用すると、プロファイル内のどこにエラーがあるのかを グラフィカルに表示することができるため、そこから修正を行なうことができます。



文法エラーを修正するには、`root` で端末ウィンドウを開いて、そこから プロファイルを開いて修正してください。プロファイル集の再読み込みは、`rcapparmor reload` で行なうことができます。

ヒント: vi での AppArmor 文法ハイライト表示

openSUSE での `vi` には、AppArmor プロファイルの 文法ハイライト表示機能が備わっています。文法エラーの行は、赤い背景で表示 されます。

26.5 AppArmor のバグ報告について

AppArmor の開発者は AppArmor がもっとも高品質な製品であり続けることを望んでいます。フィードバックやバグレポートは品質を高めるための支援となるものであるため、AppArmor 内にバグと思われる事象を発見した場合は、下記の手順でバグをご報告ください。なお、バグ報告はすべて英語での対応となります。あらかじめご了承ください。

- 1 Web ブラウザを起動し、<https://bugzilla.novell.com/index.cgi> を開きます。
- 2 Novell アカウントに関する情報を入力し、*Login* を押します。

Novell アカウントをお持ちでない場合:

下記の手順で Novell アカウントを作成します:

2a *Login to Continue* ページにある *Create New Account* を押します。

2b ユーザ名とパスワードをそれぞれ入力し、追加のアドレスデータを入力したあと、*Create Login* を押すと、ログインを即時に作成 することができます。

もしくは、下記の手順を実施します

すべてのアカウントを一括で管理したい場合は、管理者のアカウント情報を 入力します。

- 3** *Search Reports* を押して、既に類似の問題が報告されていないかどうかを確認します。製品名やキーワードを入力して簡易検索するか、もしくは *Advanced Search* を使用します。
- 4** 既に報告済みの問題であった場合は、そのバグ報告を確認し、もしも追加できる情報があればそれを記入します。
- 5** まだ報告されていない問題であった場合は、上部のナビゲーションバーから *New* を押し、*Enter Bug* ページに 入ります。
- 6** バグを報告する対象製品を選択します。この場合は、お使いの製品リリースを 選択します。選択したら *Submit* を押します。
- 7** 次に製品のバージョンとコンポーネント (この場合は AppArmor) 、ハードウェアプラットフォームとバグの重大度を選択します。
- 8** 問題点の概要を入力し、ログファイルなどの詳細説明を記述します。スクリーンショットやログファイル、テスト手順などの添付ファイルをバグに 追加するとよりわかりやすくなります。
- 9** すべての詳細を記入したら *Submit* を押します。すると、記入した報告が開発者に送信されます。

AppArmor 用語集

Apache

Apache は無償公開されている UNIX ベースの Web サーバです。現時点では、インターネットにおいてもっともよく使用されている Web サーバです。Apache について、詳しくは Apache Web サイト <http://www.apache.org> をお読みください。

アプリケーションファイアウォール

AppArmor では、アプリケーション側に許可する動作を定義する機能が備わっています。これは権限を制限するためのもので、攻撃者が悪意のあるプログラムを動作させることを防ぐ一方、信頼できるアプリケーションを無人で動作させ続けるために使用します。

攻撃シグネーチャ

ウイルスや悪意のある攻撃が発生した可能性を示す、システム内やネットワーク内の動作パターンを指す用語です。侵入検知システムでは、このような攻撃シグネーチャを利用して、妥当なアクセスと潜在的に悪意のある攻撃とを区別しています。

AppArmor ではこのような "受動型" の攻撃シグネーチャは利用しておらず、"能動型" の防御を実現しています。攻撃シグネーチャは実際に攻撃が始まるよりも前に定義されていなければ、防御を行なうことができない、という構造上の問題が存在することから、このような仕組みになっています。

GUI

グラフィカルユーザインターフェイス (Graphical User Interface) の略称です。ユーザとアプリケーションの間で、対話的かつ使いやすいインターフェイスを提

供する フロントエンドのことを指します。たとえばウィンドウやアイコン、ボタンやカーソル、スクロールバーなどから構成されます。

グロブ

ファイル名の代用表記法です。

HIP

ホスト侵入防御 (Host Intrusion Prevention) の略称です。オペレーティングシステムのカーネルと協調的に動作し、期待とは異なる異常なアプリケーション動作をブロックします。またネットワークレベルでも、アプリケーションに損害を与えるような悪意のあるパケットがホストに到着した場合、これを未然に防ぐことができるものです。

強制アクセス制御

ユーザやファイルなどの要素に対して、固定のセキュリティ属性をベースにしたアクセス制限を課すための仕組みです。"強制" とは、ユーザやプログラム が制御を変更したりすることができない、という意味から名付けられています。

プロファイル基礎クラス

たとえば DNS 参照やユーザ認証など、一般的なアプリケーション動作に必要な プロファイル構築ブロックのことを指す用語です。

RPM

RPM パッケージマネージャのことを指す用語です。誰にでも利用できるように公開されている、オープンなパッケージングシステムです。Red Hat Linux や openSUSE 、その他の Linux や UNIX システムで利用されています。コンピュータソフトウェアパッケージのインストールやアンインストール、検証 (ペリファイ) や問い合わせ、更新などを行なうことができます。詳しくは <http://www.rpm.org/> をお読みください。

SSH

Secure SHell の略称です。リモートのコンピュータからお使いのサーバに対し、機密の保護された通信形態でコマンドを発行することができるサービスです。

合理的なアクセス制御

AppArmor では、プログラムに対してどのファイルを読み込み、書き込み、実行する のかを指定することで、ネットワークサービスに対する合理的なアクセス制御を実現しています。この仕組みにより、それぞれのプログラムが実行する可能性のある処理を確認するだけで堅牢性を高めることができます。

URI

統一資源識別子 (Universal Resource Identifier) の略称です。World Wide Web の世界において、特定の対象に付与される名前とアドレスの一般形式です。URL は URI の一種です。

URL

統一資源位置指定子 (Uniform Resource Locator) の略称です。World Wide Web の世界において、文書などの資源に付与されるアドレスです。

アドレス表記の冒頭部分には使用するプロトコルが記されていて、2 番目の部分には対象となる資源が存在する IP アドレスやドメイン名が書かれています。

たとえば `http://www.novell.com` では、http プロトコル を使用します。

脆弱性 (ぜいじゃくせい)

攻撃に対して無防備なシステムやネットワークの一部を指す用語です。個人に対して正当な操作を許可する一方、悪意のあるユーザに対してもシステムの制御権を譲り渡してしまうような性質を指します。このような性質は設計や管理、実装に由来するものであるほか、ハードウェアやファームウェア、ソフトウェア内にも存在しうるものです。もしもこのような脆弱性を利用されてしまうと、情報に対する不当なアクセスや重要な処理を停止させられてしまう など、とうてい容認のできない影響をもたらします。



GNU ライセンス

本付録には、GNU General Public License バージョン 2 と GNU Free Documentation License バージョン 1.2 を掲載しています。

なお、八田真行氏 (mhatta@gnu.org) [<mailto:mhatta@gnu.org>] による各ライセンスの日本語訳を併記しています。

ただし、各日本語訳は *非公式*なものであり、フリーソフトウェア財団 (the Free Software Foundation) によって発表されたものではないことにご注意ください。法的に有効なものは常に原文 (つまり英語版) 側であり、日本語訳は各ライセンスをよりよく理解する支援を行なう目的で 作成されたもの、という扱いです。

また、日本語訳は DocBook (novdoc) に合わせて段落を分割しているほか、引用符のタグ化 ("blah" -> <quote>blah</quote>) とリンクの生成 (ulink) を行なっています。

GNU General Public License

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc. 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The «Program», below, refers to any such program or work, and a «work based on the Program» means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term «modification».) Each licensee is addressed as «you».

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a)** You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b)** You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c)** If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a)** Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and [any later version], you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the 「copyright」 line and a pointer to where the full notice is found.

```
one line to give the program's name and an idea of what it does.  
Copyright (C) yyyy name of author
```

```
This program is free software; you can redistribute it and/or  
modify it under the terms of the GNU General Public License  
as published by the Free Software Foundation; either version 2  
of the License, or (at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,  
but WITHOUT ANY WARRANTY; without even the implied warranty of  
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the  
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License  
along with this program; if not, write to the Free Software  
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author  
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details  
type `show w' . This is free software, and you are welcome  
to redistribute it under certain conditions; type `show c'  
for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a 「copyright disclaimer」 for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright
```

```
interest in the program `Gnomovision'
(which makes passes at compilers) written
by James Hacker.
```

```
signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License [<http://www.fsf.org/licenses/lgpl.html>] instead of this License.

GNU 一般公衆利用許諾契約書 (日本語訳)

バージョン 2, 1991年6月

Copyright (C) 1989, 1991 Free Software Foundation, Inc. 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

この利用許諾契約書を、一字一句そのままに複製し頒布することは許可する。しかし変更は認めない。

はじめに

ソフトウェア向けライセンスの大半は、あなたがそのソフトウェアを共有したり 変更したりする自由を奪うように設計されています。対照的に、GNU 一般公衆利用許諾契約書は、あなたがフリーソフトウェアを共有したり変更したりする自由を保証する--すなわち、ソフトウェアがそのユーザすべてにとってフリー であることを保証することを目的としています。この一般公衆利用許諾契約書は、フリーソフトウェア財団のソフトウェアのほとんどに適用されており、また GNU GPLを適用すると決めたフリーソフトウェア財団以外の作者によるプログラムにも適用されています(いくつかのフリーソフトウェア財団のソフトウェアには、GNU GPLではなくGNU ライブラリー一般公衆利用許諾契約書が適用されています)。あなたもまた、ご自分のプログラムにGNU GPLを適用することが可能です。

私たちがフリーソフトウェアと言うとき、それは利用の自由について言及している のであって、価格は問題にしていません。私たちの一般公衆利用許諾契約書は、あなたがフリーソフトウェアの複製物を頒布する自由を保証するよう設計されています (希望に応じてその種のサービスに手数料を課す自由も保証されます)。また、あなたがソースコードを受け取るか、あるいは望めばそれを入手することが可能であるということ、あなたがソフトウェアを変更し、その一部を新たなフリーの プログラムで利用できるとのこと、そして、以上で述べたようなことができる ということがあなたに知られるということも保証されます。

あなたの権利を守るため、私たちは誰かがあなたの有するこれらの権利を否定する ことや、これらの権利を放棄するよう要求することを禁止するという制限を加える 必要があります。よって、あなたがソフトウェアの複製物を頒布したりそれを変更 したりする場合には、そういった制限のためにあなたにある種の責任が発生することになります。

例えば、あなたがフリーなプログラムの複製物を頒布する場合、有料が無料に 関わらず、あなたは自分が有する権利を全て受領者に与えなければなりません。また、あなたは彼らもソースコードを受け取るか手に入れることができるよう 保証しなければなりません。そして、あなたは彼らに対して以下で述べる条件を示し、彼らに自らの持つ権利について知らしめるようにしなければなりません。

私たちはあなたの権利を二段階の手順を踏んで保護します。(1) まずソフトウェアに対して著作権を主張し、そして (2) あなたに対して、ソフトウェアの複製や頒布または改変についての法的な許可を 与えるこの契約書を提示します。

また、各作者や私たちを保護するため、私たちはこのフリーソフトウェアには 何の保証も無いということを誰もが確実に理解するようにし、またソフトウェアが 誰か他人によって改変され、それが次々と頒布されていったとしても、その受領者は 彼らが手に入れたソフトウェアがオリジナルのバージョンでは無いこと、そして 原作者の名声は他人によって持ち込まれた可能性のある問題によって影響される ことがないということを周知させたいと思います。

最後に、ソフトウェア特許がいかなるフリーのプログラムの存在にも不断の脅威を 投げかけていますが、私たちは、フリーなプログラムの再頒布者が個々に特許 ライセンスを取得することによって、事実上プログラムを独占的にしてしまうという 危険を避けたいと思います。こういった事態を予防するため、私たちはいかなる特許も 誰もが自由に利用できるようライセンスされるか、全くライセンスされないかの どちらかでなければならないことを明確にしました。

(訳注: 本契約書で「独占的(proprietary)」とは、ソフトウェアの利用や再頒布、改変が禁止されているか、許可を得ることが必要とされているか、あるいは厳しい 制限が課せられていて自由にそうすることが事実上できなくなっている状態のことを指す。詳しくは <http://www.gnu.org/philosophy/categories.ja.html#ProprietarySoftware> [<http://www.gnu.org/philosophy/categories.ja.html#ProprietarySoftware>] を参照せよ。)

複製や頒布、改変についての正確な条件と制約を以下で述べていきます。

複製、頒布、改変に関する条件と制約

0. この利用許諾契約書は、そのプログラム(またはその他の著作物)を この一般公衆利用許諾契約書の定める条件の下で頒布できる、という告知が 著作権者によって記載されたプログラムまたはその他の著作物全般に適用される。以下では、「プログラム」とはそのようにしてこの契約書が適用され プログラムや著作物全般を意味し、また「プログラムを基にした著作物」とは「プログラム」やその他の著作権法の下で派生物と見なされるもの 全般を指す。すなわち、「プログラム」かその一部を、全く同一の ままか、改変を加えたか、あるいは他の言語に翻訳された形で含む

著作物のことである(「改変」という語の本来の意味からはずれるが、以下では 翻訳も改変の一種と見なす)。それぞれの契約者は「あなた」と表現される。

複製や頒布、改変以外の活動はこの契約書ではカバーされない。それらはこの契約書の 対象外である。「プログラム」を実行する行為自体に制限はない。また、そのような「プログラム」の出力結果は、その内容が「プログラム」を基にした著作物を構成する場合のみ この契約書によって保護される(「プログラム」を実行したことによって 作成されたということは無関係である)。このような線引きの妥当性は、「プログラム」が何を
するのかに依存する。

1. それぞれの複製物において適切な著作権表示と保証の否認声明(disclaimer of warranty) を目立つよう適切に掲載し、またこの契約書および一切の保証の不在に触れた 告知すべてをそのまま残し、そしてこの契約書の複製物を「プログラム」のいかなる受領者にも「プログラム」と共に頒布する限り、あなたは「プログラム」のソースコードの複製物を、あなたが受け取った通りの形で複製または頒布することができる。媒体は問わない。

あなたは、物理的に複製物を譲渡するという行為に関して手数料を課しても良いし、希望によっては手数料を取って交換における保護の保証を提供しても良い。

2. あなたは自分の「プログラム」の複製物がその一部を改変して「プログラム」を基にした著作物を形成し、そのような改変点や 著作物を上記第1節の定める条件の下で複製または頒布することができる。ただし、そのためには以下の条件すべてを満たしていなければならない:

- a) あなたがそれらのファイルを変更したということと変更した日時が良く 分かるよう、改変されたファイルに告示しなければならない。
- b) 「プログラム」またはその一部を含む著作物、あるいは「プログラム」かその一部から派生した著作物を頒布あるいは 発表する場合には、その全体をこの契約書の条件に従って第三者へ無償で 利用許諾しなければならない。
- c) 改変されたプログラムが、通常実行する際に対話的にコマンドを読むようになっているならば、そのプログラムを最も一般的な方法で対話的に実行する際、適切な著作権表示、無保証であること(あるいはあなたが保証を提供するということ)、ユーザがプログラムをこの契約書で述べた条件の下で頒布することができる ということ、そしてこの契約書の複製物を閲覧するにはどうしたらよいかというユーザへの説明を含む告知が印刷されるか、あるいは画面に表示される ようにしなければならない(例外として、「プログラム」そのものは対話的であっても通常そのような告知を印刷しない場合には、「プログラム」を基にしたあなたの著作物に そのような告知を 印刷させる必要はない)。

以上の必要条件是全体としての改変された著作物に適用される。著作物の一部が「プログラム」から派生したのではないと確認でき、それら自身 別の独立した著作物であると合理的に考えられるならば、あなたがそれらを別の 著作物として分けて頒布する場合、そういった部分にはこの契約書とその条件は 適用されない。しかし、あなたが同じ部分を「プログラム」を基にした 著作物全体の一部として頒布するならば、全体としての頒布物は、この契約書が 課す条件に従わなければならない。というのは、この契約書が他の契約者に与える 許可は「プログラム」丸ごと全体に及び、誰が書いたかは関係なく各部分のすべてを保護するからである。

よって、すべてあなたによって書かれた著作物に対し、権利を主張したりあなたの 権利に異議を申し立てることはこの節の意図するところではない。むしろ、その趣旨は「プログラム」を基にした派生物ないし集合著作物の 頒布を管理する権利を行使することにある。

また、「プログラム」を基にしていないその他の著作物を「プログラム」(あるいは「プログラム」を基にした著作物)と一緒に集めただけのものを一巻の保管装置ないし頒布媒体に収めても、その他の 著作物までこの契約書が保護する対象になるということにはならない。

3. あなたは上記第1節および2節の条件に従い、「プログラム」(あるいは 第2節における派生物)をオブジェクトコードないし実行形式で複製または頒布 することができる。ただし、その場合あなたは以下のうちどれか一つを実施 しなければならない:

- a) 著作物に、『プログラム』に対応した完全かつ機械で読み取り可能な ソースコードを添付する。ただし、ソースコードは上記第1節および2節の条件に従いソフトウェアの交換で習慣的に使われる媒体で頒布しなければならない。あるいは、
- b) 著作物に、いかなる第三者に対しても、『プログラム』に対応した完全かつ 機械で読み取り可能なソースコードを、頒布に要する物理的コストを上回らない 程度の手数料と引き換えに提供する旨述べた少なくとも3年間は有効な書面 になった申し出を添える。ただし、ソースコードは上記第1節および2節の条件に 従いソフトウェアの交換で習慣的に使われる媒体で頒布しなければならない。あるいは、
- c) 対応するソースコード頒布の申し出に際して、あなたが得た情報を一緒に引き渡す (この選択肢は、営利を目的としない頒布であって、かつあなたが上記小節bで 指定されているような申し出と共にオブジェクトコードあるいは実行形式の プログラムしか入手していない場合に限り許可される)。

著作物のソースコードとは、それに対して改変を加える上で好ましいとされる 著作物の形式を意味する。ある実行形式の著作物にとって完全なソースコードとは、それが含むモジュールすべてのソースコード全部に加え、関連するインターフェース 定義ファイルのすべてとライブラリのコンパイルやインストールを制御するために 使われるスクリプトをも加えたものを意味する。しかし特別な例外として、そのコンポーネント自体が実行形式に付随するのでは無い限り、頒布されるものに 中、実行形式が実行されるオペレーティングシステムの主要なコンポーネント (コンパイラやカーネル等)と通常一緒に(ソースかバイナリ形式のどちらかで) 頒布されるものを含んでいる必要はないとする。

実行形式またはオブジェクトコードの頒布が、指定された場所からコピーするための アクセス手段を提供することで為されるとして、その上でソースコードも同等の アクセス手段によって同じ場所からコピーできるようになっているならば、第三者が オブジェクトコードと一緒にソースも強制的にコピーせられるようになっているとしてもソースコード頒布の条件を満たしているものとする。

4. あなたは「プログラム」を、この契約書において明確に提示された 行為を除き複製や改変、サブライセンス、あるいは頒布してはならない。他に「プログラム」を複製や改変、サブライセンス、あるいは頒布する 企てはすべて無効であり、この契約書の下でのあなたの権利を自動的に終結させる こととなる。しかし、複製物や権利をこの契約書に従ってあなたから得た人々に 関しては、そのような人々がこの契約書に完全に従っている限り彼らのライセンスまで 終結することはない。

5. あなたはこの契約書を受諾する必要は無い。というのは、あなたはこれに署名して いないからである。しかし、この契約書以外にあなたに対して「プログラム」やその派生物を改変または頒布する許可を与えるものは 存在しない。これらの行為は、あなたがこの契約書を受け入れない限り

法によって 禁じられている。そこで、「プログラム」(あるいは「プログラム」を基にした著作物全般)を改変しない頒布することにより、あなたは自分がそのような行為を行うためにこの契約書を受諾したということ、そして「プログラム」とそれに基づく著作物の複製や頒布、改変について この契約書が課す制約と条件をすべて受け入れたということを示したものと見なす。

6. あなたが「プログラム」(または「プログラム」を基にした著作物全般)を再頒布するたびに、その受領者は元々のライセンス許可者から、この契約書で指定された条件と制約の下で「プログラム」を複製や頒布、あるいは改変する許可を自動的に得るものとする。あなたは、受領者がここで認められた権利を行使することに関してこれ以上他のいかなる制限も課してはならない。あなたには、第三者がこの契約書に従うことを強制する責任はない。

7. 特許侵害あるいはその他の理由(特許関係に限らない)から、裁判所の判決あるいは申し立ての結果としてあなたに(裁判所命令や契約などにより)このライセンスの条件と矛盾する制約が課された場合でも、あなたがこの契約書の条件を免除されるわけではない。もしこの契約書の下であなたに課せられた責任と他の関連する責任を同時に満たすような形で頒布できないならば、結果としてあなたは「プログラム」を頒布することが全くてできないことである。例えば特許ライセンスが、あなたから直接間接を問わずコピーを受け取った人が誰でも「プログラム」を使用料無料で再頒布することを認めていない場合、あなたがその制約とこの契約書を両方とも満たすには「プログラム」の頒布を完全に中止するしかないだろう。

この節の一部分が特定の状況の下で無効ないし実施不可能な場合でも、節の残りの部分は適用されるよう意図されている。その他の状況では節が全体として適用されるよう意図されている。

特許やその他の財産権を侵害したり、そのような権利の主張の効力に異議を唱えたりするようあなたを誘惑することがこの節の目的ではない。この節には、人々によってライセンス慣行として実現されてきた、フリーソフトウェア頒布のシステムの完全性を護るという目的しかない。多くの人々が、フリーソフトウェアの頒布システムが首尾一貫して適用されているという信頼に基づき、このシステムを通じて頒布される多様なソフトウェアに寛大な貢献をしてきたのは事実であるが、人々がどのようなシステムを通じてソフトウェアを頒布したいと思うかはあくまでも作者/寄与者次第であり、あなたが選択を押しつけることはできない。

この節は、この契約書のこの節以外の部分の一掃になると考えられるケースを徹底的に明らかにすることを目的としている。

8. 「プログラム」の頒布や利用が、ある国においては特許または著作権が主張されたインターフェースのいずれかによって制限されている場合、「プログラム」にこの契約書を適用した元の著作権者は、そういった国々を排除した明確な地理的頒布制限を加え、そこで排除されていない国の中やそれらの国々の間でのみ頒布が許可されるようにしても構わない。その場合、そのような制限はこの契約書本文で書かれているのと同様に見なされる。

9. フリーソフトウェア財団は、時によって改訂または新版の一般公衆利用許諾書を発表することができる。そのような新版は現在のバージョンとその精神においては似たものになるだろうが、新たな問題や懸念を解決するため細部では異なる可能性がある。

それぞれのバージョンには、見分けが付くようにバージョン番号が振られている。「プログラム」においてそれに適用されるこの契約書のバージョン番号が指定されていて、更に「それ以降のいかなるバージョン(any later version)」も適用して良いとなった場合、あなたは従う条件と制約として、指定のバージョンが、フリーソフトウェア財団によって発行された指定のバージョン以降の版のどれか一つのどちらかを選ぶことが出来る。「プログラム」でライセンスのバージョン番号が指定されていないならば、あなたは今までにフリーソフトウェア財団から発行されたバージョンの中から好きに選んで構わない。

10. もしあなたが「プログラム」の一部を、その頒布条件がこの契約書と異なる他のフリーなプログラムと統合したいならば、作者に連絡して許可を求めよ。フリーソフトウェア財団が著作権を保有するソフトウェアについては、フリーソフトウェア財団に連絡せよ。私たちは、このような場合のために特別な例外を設けることもある。私たちが決定を下すにあたっては、私たちのフリーソフトウェアの派生物すべてがフリーな状態に保たれるということと、一般的にソフトウェアの共有と再利用を促進するという二つの目標を規準に検討されるであろう。

無保証について

11. 「プログラム」は代価無しに利用が許可されるので、適切な法が認める限りにおいて、「プログラム」に関するいかなる保証も存在しない。書面で別に述べる場合を除いて、著作権者、またはその他の団体は、「プログラム」を、表明されたか言外にかかわらず、商業的適性を保証するほのめかしやある特定の目的への適合性(に限られない)を含む一切の保証無しに「あるがまま」で提供する。「プログラム」の質と性能に関するリスクのすべてはあなたに帰属する。「プログラム」に欠陥があると判明した場合、あなたは必要な保守点検や補修、修正に要するコストのすべてを引き受けることになる。

12. 適切な法が書面で同意によって命ぜられない限り、著作権者、または上記で許可されている通りに「プログラム」を改変または再頒布したその他の団体は、あなたに対して「プログラム」の利用ないし利用不能で生じた通常損害や特別損害、偶発損害、間接損害(データの消失や不正確な処理、あなたが第三者が被った損失、あるいは「プログラム」が他のソフトウェアと一緒に動作しないという不具合などを含むがそれらに限らない)に一切の責任を負わない。そのような損害が生ずる可能性について彼らが忠告されていたとしても同様である。

条件と制約終わり

以上の条項をあなたの新しいプログラムに適用する方法

あなたが新しいプログラムを開発したとして、公衆によってそれが利用される可能性を最大にしたいなら、そのプログラムをこの契約書の条項に従って誰でも再頒布あるいは変更できるようフリーソフトウェアにするのが最善です。

そのためには、プログラムに以下のような表示を添付してください。その場合、保証が排除されているということを最も効果的に伝えるために、それぞれのソースファイルの冒頭に表示を添付すれば最も安全です。少なくとも、「著作権表示」という行と全文がある場所へのポインタだけは各ファイルに含めて置いてください。

one line to give the program's name and an idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

(訳)

プログラムの名前と、それが何をするかについての簡単な説明。Copyright (C) 西暦年 作者の名前

このプログラムはフリーソフトウェアです。あなたはこれを、
フリーソフトウェア財団によって発行された GNU 一般公衆利用許諾契約書
(バージョン2か、希望によってはそれ以降のバージョンのうちどれか)の
定める条件の下で再頒布または改変することができます。

このプログラムは有用であることを願って頒布されますが、*全くの無保証*
です。商業可能性の保証や特定の目的への適合性は、言外に示されたものも
含め全く存在しません。詳しくはGNU 一般公衆利用許諾契約書をご覧ください。

あなたはこのプログラムと共に、GNU 一般公衆利用許諾契約書の複製物を一部
受け取ったはずですが、もし受け取っていない場合は、フリーソフトウェア財団
まで請求してください(宛先は the Free Software Foundation, Inc., 59
Temple Place, Suite 330, Boston, MA 02111-1307 USA)。

電子ないし紙のメールであなたに問い合わせる方法についての情報も書き加えましょう。

プログラムが対話的なものならば、対話モードで起動した際に出力として 以下のような短い告知が表示されるようにしてください:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w'. This is free software, and you are welcome
to redistribute it under certain conditions; type `show c'
for details.

(訳)

Gnomovision バージョン 69, Copyright (C) 西暦年 作者の名前
Gnomovision は*全くの無保証*で提供されます。詳しくは
`show w' とタイプして下さい。
これはフリーソフトウェアであり、ある条件の下で再頒布することが
奨励されています。詳しくは `show c' とタイプして下さい。

ここで、仮想的なコマンド `show w' と `show c' は 一般公衆利用許諾契約書の適切な部分を表示するようになっていなければなりません。
もちろん、あなたが使うコマンドを `show w' や `show c' と呼ぶ必然性はありませんので、あなたのプログラムに 合わせてマウスのクリックやメ
ニューのアイテムにしても結構です。

また、あなたは、必要ならば(プログラマーとして働いていたら)あなたの 雇用主、あるいは場合によっては学校から、そのプログラムに関する「著作権放棄声明(copyright disclaimer)」に署名してもらうべきです。以下は例ですので、名前を変えてください:

```
Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision'
(which makes passes at compilers) written
by James Hacker.
```

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice

(訳)

Yoyodyne社はここに、James Hackerによって書かれた
プログラム `Gnomovision' (コンパイラへ通すプログラム)
に関する一切の著作権の利益を放棄します。

Ty Coon氏の署名、1989年4月1日
Ty Coon、副社長

この一般公衆利用許諾契約書では、あなたのプログラムを独占的なプログラムに 統合することを認めていません。あなたのプログラムがサブルーチンライブラリ ならば、独占的なアプリケーションとあなたのライブラリをリンクすることを 許可したほうがより便利であると考えられるかもしれません。もしこれがあなたの 望むことならば、この契約書の代わりに GNU ライブラリー一般公衆利用許諾契約書 [<http://www.fsf.org/licenses/lgpl.html>] を適用してください。

GNU Free Documentation License

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA.
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member

of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties; any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and

legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O.Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the

aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled "GNU
Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

GNU フリー文書利用許諾契約書

Copyright (C) 2000,2001,2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA この利用許諾契約書を、一字一句そのままに複製し頒布することは許可する。しかし変更は認めない。

This is an unofficial translation of the GNU Free Documentation License into Japanese. It was not published by the Free Software Foundation, and does not legally state the distribution terms for documents that uses the GNU FDL--only the original English text of the GNU FDL does that. However, we hope that this translation will help Japanese speakers understand the GNU FDL better.

(訳: 以下はGNU Free Documentation Licenseの非公式な日本語訳です。これはフリーソフトウェア財団 (the Free Software Foundation)によって発表されたものではなく、GNU FDLを適用した文書の頒布条件を法的に有効な形で述べたものではありません。頒布条件としてはGNU FDLの英語版テキストで指定されているもののみが有効です。しかしながら、私たちはこの翻訳が、日本語を使用する人々にとってGNU FDLをより良く理解する助けとなることを望んでいます。)

0. はじめに

この利用許諾契約書の目的は、この契約書が適用されるマニュアルや教科書、その他機能本位で実用的な文書を(無料ではなく)自由という意味で「フリー」とすること、すなわち、改変の有無あるいは目的の営利非営利を問わず、文書を複製し再頒布する自由をすべての人々に効果的に保証することです。加えてこの契約書により、著者や出版者が自分たちの著作物に対して相応の敬意と賞賛を得る手段も保護されます。また、他人が行った改変に対して責任を負わずに済むようになります。

この利用許諾契約書は「コピーレフト」的なライセンスの一つであり、この契約書が適用された文書から派生した著作物は、それ自身もまた原本と同じ意味でフリーでなければなりません。この契約書は、フリーソフトウェアのために設計されたコピーレフトなライセンスであるGNU一般公衆利用許諾契約書を補足するものです。

(訳注: コピーレフト(copyleft)の概念については <http://www.gnu.org/copyleft/copyleft.ja.html> を参照せよ)

この利用許諾契約書は、フリーソフトウェア用のマニュアルに適用することを目的として書かれました。フリーソフトウェアはフリーな文書を必要としており、フリーなプログラムはそのソフトウェアが保証するのと同じ自由を提供するマニュアルと共に頒布されるべきだからです。しかし、この契約書の適用範囲はソフトウェアのマニュアルに留まりません。対象となる著作物において扱われる主題が何であれ、あるいはそれが印刷された書籍として出版されるか否かに関わらず、この契約書は文字で書かれたいかなる著作物にも適用することが可能です。私たちとしては、主にこの契約書を解説や参照を目的とする著作物に適用することをお勧めします。

1. この利用許諾契約書の適用範囲と用語の定義

著作物がこの利用許諾契約書の定める条件の下で頒布される旨の告知を、著作権者がその中に書いたすべてのマニュアルあるいはその他の著作物は、いかなる媒体上にあってもこの契約書の適用対象となる。そのような告知を置くことで、全世界において、著作権使用料を必要とせず、許可の存続期間を限定されることなく、この契約書の中で述べられている条件の下で当該著作物を利用できるという許可を与えることとする。以下において、「『文書』(Document)」とはそのような告知が記載されたマニュアルないし著作物すべてを指す。公衆の一員ならば誰でも契約の当事者となることができ、この契約書中では「あなた」と表現される。あなたは、著作権法の下で許可を必要とするような方法で著作物を複製や改変、あるいは頒布することにより、この契約書を受諾することになる。

『文書』の「改変版 (Modified Version)」とは、一字一句忠実に複製したが、あるいは改変や他言語への翻訳を行ったかどうかに関わらず、その『文書』の全体あるいは一部分を含む著作物すべてを意味する。

「補遺部分 (Secondary Section)」とは、『文書』中でその旨指定された補遺ないし本文に先だって前付けとして置かれる一部分であり、『文書』の出版者あるいは著者と、『文書』全体の主題 (あるいはそれに関連する事柄)との関係のみを論じ、全体としての主題の範疇に直接属する内容を

全く含まないものである (たとえば、『文書』の一部が数学の教科書だった場合、補遺部分では数学について何も解説してはならない)。補遺部分で扱われる関係は、その主題あるいは関連する事柄との歴史的つながりのことかも知れないし、それらに関する法的、商業的、哲学的、倫理的、あるいは政治的立場についてかも知れない。

「変更不可部分 (Invariant Sections)」とは補遺部分の一種で、それらが変更不可部分であることが、『文書』をこの利用許諾契約書の下で発表する旨述べた告知中においてその部分の題名と共に明示されているものである。ある部分が上記のような「補遺」性の定義にそぐわない場合は、その部分を「変更不可」として指定することは認められない。『文書』は、変更不可部分を全く含まなくても良い。『文書』において変更不可部分が全く指定されていないければ、その『文書』に変更不可部分は存在しないということである。

「カバーテキスト (Cover Texts)」とは、『文書』がこの利用許諾契約書の指定する条件の下で発表される旨述べた告知において、「表カバーテキスト」あるいは「裏カバーテキスト」として列挙された短い文章のことを指す。表カバーテキストは最大で5語、裏カバーテキストは最大で25語までとする。

『文書』の「透過的」複製物とは、機械による読み取りが可能な『文書』の複製物のことを指す。透過的な複製物の文書形式は、その仕様が一般の人々に入手可能で、『文書』の内容を一般的なテキストエディタ、または(画素で構成される画像ならば)一般的なペイントプログラム、あるいは(図面ならば)いくつかの広く入手可能な製図エディタで簡単に改訂するのに適しており、なおかつテキストフォーマットへの入力に適する(あるいはテキストフォーマットへの入力に適する諸形式への自動的な変換に適する)ものでなければならない。透過的なファイル形式への複製であっても、マークアップ、あるいはマークアップの不在が読者によるそれ以降の改変をわざと邪魔し阻害するように仕組まれたものは透過的であるとは見做されない。ある画像形式が、相当量のテキスト文章を表現するために使われた場合、それは透過的ではない。透過的ではない複製は「非透過的」複製と呼ばれる。

透過的複製に適した形式の例としては、マークアップを含まないプレーンな ASCII 形式、Texinfo 入力形式、LaTeX 入力形式、一般に入手可能な DTD を用いた SGML あるいは XML、または人間による改変を想定して設計された、標準に準拠したシンプルなもの HTML や PostScript、PDF などが挙げられる。透過的な画像形式の例には、PNG や XCF、JPG が含まれる。非透過な形式としては、独占的なワードプロセッサでのみ閲覧編集できる独占的なファイル形式、普通には入手できない DTD または処理系を使った SGML や XML、ある種のワードプロセッサが生成する、出力のみを目的とした機械生成の HTML や PostScript、PDF などが含まれる。

「題扉 (Title Page)」とは、印刷された書籍に於いては、実際の表紙自身のみならず、この利用許諾契約書が表紙に掲載することを義務づける文章や図などを、読みやすい形で載せるのに必要なだけの、表紙に引き続き数ページをも意味する。表紙に類するものが無い形式で発表される著作物においては、「題扉」とは本文の始まりに先だって、その著作物の題名が最も目立つ形で現れる場所の近くに置かれる文章のことを指す。

「XYZ」と題された (Entitled XYZ) 部分とは、『文書』において「XYZ」と名付けられた一部分であり、その題名は正確に「XYZ」であるか、「XYZ」を他の言語に翻訳した上でその後ろに「XYZ」をそのまま括弧で括ったものを含む記述のどちらかである(ここでの「XYZ」とは、この利用許諾契約書において以下で言及される特定の部分名を意味している。例えば「謝辞 (Acknowledgements)」、「献辞 (Dedications)」、「推薦の辞 (Endorsements)」、「履歴 (History)」)。あなたが『文書』を改変する場合、そのような部分の「題名を保存する (Preserve the Title)」とは、「XYZ」と題された」部分として、ここでの定義に従い題名を残すということである。

『文書』は、「保証否認警告 (Warranty Disclaimers)」を、この利用許諾契約書が『文書』に適用されると述べた告知の次に含んでも良い。この種の保証否認警告は、この契約書からの言及という形で利用条件に含まれるものと解されるが、保証の否認に関することについてののみ有効とする。こういった保証否認警告で示うその他のいかなる含意も無効であり、この契約書の効能には何ら影響を持たない。

2. 逐語的に忠実な複製

この利用許諾契約書、著作権表示、この契約書が『文書』に適用される旨述べた告知の三つがすべての複製物に複製され、かつあなたがこの契約書で指定されている以外のいかなる条件も追加しない限り、あなたはこの『文書』を、商用であるか否かを問わずいかなる形で複製頒布することができる。あなたは、あなたが作成あるいは頒布する複製物に対して、閲覧や再複製を技術的な手法によって妨害、規制してはならない。しかしながら、複製と引き換えに代価を得てもかまわない。あなたが相当量の複製物を頒布する際には、本契約書第3項で指定される条件にも従わなければならない。

またあなたは、上記と同じ条件の下で、複製物を貸与したり複製物を公に開示することができる。

3. 大量の複製

もしあなたが、『文書』の印刷された (あるいは通常は印刷された表紙を持つ媒体における)複製物を100部を超えて出版し、また『文書』の利用許諾告知がカバーテキストの掲載を要求している場合には、指定されたすべてのカバーテキストを、表カバーテキストは表表紙に、裏カバーテキストは裏表紙に、はっきりと読みやすい形で載せた表紙の中に複製物本体を綴じ込まなければならない。また、両方の表紙において、それらの複製物の出版者としてのあなたをはっきりとかつ読みやすい形で確認できなければならない。表表紙では『文書』の完全な題名を、題名を構成するすべての語が等しく目立つようにして、視認可能な形で示さなければならない。それらの情報に加えて、表紙に他の文章や図などを加えることは許可される。表紙のみを変更した複製物は、それが『文書』の題名を保存し上記の条件を満たす限り、ほかの点では逐語的に忠実な複製物として扱われる。

もしどちらかの表紙に要求されるカバーテキストの量が多すぎて読みやすく収めることが不可能ならば、あなたはテキスト先頭の一文(あるいは適切に収まるだけ)を実際の表紙に載せ、続きは隣接したページに載せるべきである。

あなたが『文書』の「非透過的」複製物を100部を超えて出版あるいは頒布する場合、それぞれの非透過な複製物と一緒に機械で読み取り可能な透過的複製物を添付するか、それぞれの非透過な複製物(あるいはそれに付属する文書)中で、公にアクセス可能なコンピュータネットワーク上の所在地を記述しなければならない。その場所には、非透過な複製物と内容的に寸分違わず、余計なものが追加されていない完全な『文書』の透過的複製物が置かれ、またそこから、ネットワークを利用する一般公衆が、一般に標準的と考えられるネットワークプロトコルを使ってダウンロードすることができなければならない。もしあなたが後者の選択肢を選ぶならば、その版の非透過な複製物を公衆に(直接、あるいはあなたの代理人ないし小売業者が)最後に頒布してから最低1年間は、その透過的複製物が指定の場所でアクセス可能であり続けることを保証するよう、非透過な複製物の大量頒布を始める際に十分に慎重な手順を踏まなければならない。

これは要望であり必要条件ではないが、『文書』の著者に、『文書』の更新された版をあなたに提供する機会を与えるため、透過非透過を問わず大量の複製物を再頒布し始める前には彼らにきちんと連絡しておいてほしい。

4. 改変

『文書』の改変版を、この利用許諾契約書と細部まで同一の契約の下で発表する限り、すなわち原本の役割を改変版で置き換えた形での頒布と改変を、その複製物を所有するすべての人々に許可する限り、あなたは改変版を上記第2項および第3項が指定する条件の下で複製および頒布することができる。さらに、あなたは改変版において以下のことを行わなければならない。

- A. 題扉に(もしあればその他の表紙にも)、『文書』および『文書』のそれ以前の版と見分けがつく題名を載せること(もし以前の 版があれば、『文書』の「履歴 (History)」の部分に列記されているはずである)。もし元の版の出版者から許可を得たならば、以前の版と同じ題名を使ってもいい。
- B. 題扉に、改変版における改変を行った1人以上の人物 が団体名を列記すること。あわせて元の『文書』の著者として、最低5人(もし5人以下ならばすべて)の主要著者を列記すること。ただし元の著者たちが この条件を免除した場合は除く。
- C. 題扉に、改変版の出版者名を出版者として記載すること。
- D. 『文書』にあるすべての著作権表示を残すこと。
- E. 他の著作権表示の近くに、あなたの改変に対する適 当な著作権表示を追加すること。
- F. 著作権表示のすぐ後に、改変版をこの契約書の条件 の下で利用することを公衆に対して許可する告知を含めること。その形式は この契約書の末尾にある付記で示されている。
- G. 元の『文書』の利用許諾告知に書かれた、変更不可 部分の完全な一覧と、要求されるカパーテキストとを、改変版の利用許諾告知でもそのまま残すこと。
- H. この契約書の、変更されていない複製物を含めること。
- I. 「履歴 (History)」と題された部分とその題名を保存し、そこに改変版の、少なくとも題名、出版年、新しく変更した部分の著 者名、出版者名を、題扉に掲載するのと同じように記載した一項を加えること。もし『文書』中に「履歴」と題された部分が存在しない場合には、『文 書』の題名、出版年、著者、出版者を題扉に掲載するのと同じように記載した部分を用意し、上記で述べたような、改変版を説明する一項を加えること。
- J. 『文書』中に、『文書』の透過的複製物への公共的 アクセスのために指定されたネットワークの所在地が記載されていたならば、それを保存すること。同様に、その『文書』の元になった以前の版で指定 されていたネットワークの所在地も載っていたならば、それも保存すること。これらの情報は「履歴(History)」の部分に置いてもいい。ただし、それが『文書』自身より少なくとも4年前に出版された著作物の情報であったり、あるいは改変版が参考としている版の元々の出版者から許可を得たならば、その情報を削除してもかまわない。
- K. 「謝辞 (Acknowledgement)」あるいは「献辞 (Dedication)」等と題されたいかなる部分も、その部分の題名を保存し、そ の部分の内容(各貢献者への謝意あるいは献呈の意)と語調を保存すること。
- L. 『文書』の変更不可部分を、その本文および題名を 変更せずに保存すること。章番号やそれに相当するものは部分の題名の一 部とは見做さない。
- M. 「推薦の辞 (Endorsement)」というような章名が題 された部分はすべて削除すること。そのような部分を改変版に含めてはならない。
- N. すでに存在する部分を「推薦の辞 (Endorsement)」と題されるように改名したり、題名の点で変更不可部分のどれかと衝突する ように改名してはならない。
- O. 保証否認警告を保存すること。

もし改変版に、補遺部分としての条件を満たし、かつ『文書』から複製物された文章や図などをいっさい含んでいない、前書き的な章あるいは付録が新しく含まれるならば、あなたは希望によりそれらの部分の一部あるいはすべてを変更不可と宣言することができる。変更不可を宣言するためには、それらの部分の題名を改変版の利用許諾告知中の変更不可部分一覧に追加すれば良い。これらの題名は他の章名とは全く別のものでなければならない。

含まれる内容が、さまざまな集団によるあなたの改変版に対する推薦の辞のみである限り、あなたは、「推薦の辞 (Endorsement)」と題された章を追加することができる。推薦の辞の例としては、ピアレビューの陳述、あるいは文書がある標準の権威ある定義としてその団体に承認されたという声明などがある。

あなたは、5語までの一文を表カバーテキストとして、25語までの文を表裏紙テキストとして、改変版のカバーテキスト一覧の末尾に加えることができる。一個人ないし一団体が直接(あるいは団体内で結ばれた協定によって)加えることができるのは、表カバーテキストおよび裏カバーテキストとしてそれぞれ一文ずつのみである。もし以前すでにその文書において、表裏いずれかの表紙にあなたの(またはあなたが代表する同じ団体内で為された協定に基づく)カバーテキストが含まれていたならば、あなたが新たに追加することはできない。しかしあなたは、その古い文を加えた以前の出版者から明示的な許可を得たならば、古い文を置き換えることができる。

『文書』の著者あるいは出版者は、この利用許諾契約書によって、彼らの名前を利用することを許可しているわけではない。彼らの名前を改変版の宣伝に使ったり、改変版への明示的あるいは黙示的な保証のために使うことを許可するものではない。

5. 文書の結合

あなたは、上記第4項において改変版に関して定義された条件の下で、この利用許諾契約書の下で発表された複数の文書の一つにまとめることができる。その際、原本となる文書にある変更不可部分を全て、改変せずに結合後の著作物中に含め、それらをあなたが統合した著作物の変更不可部分としてその利用許諾告知において列記し、かつ原本にある全ての保証否認警告を保存しなければならない。

結合後の著作物についてはこの契約書の複製物一つ含んでいれよく、同一内容の変更不可部分が複数ある場合には一つで代用してよい。もし同じ題名だが内容の異なる変更不可部分が複数あるならば、そのような部分のそれぞれの題名の最後に、(もし分かっているならば)その部分の原著者あるいは出版者の名前、あるいは他と重ならないような番号を括弧で括って記載することで、それぞれ見分けが付くようにしなければならない。結合後の著作物の利用許諾告知における変更不可部分の一覧においても、章の題名に同様の調整をすること。

結合後の著作物においては、あなたはそれぞれの原本の「履歴 (History)」と題されたあらゆる部分をまとめて、「履歴 (History)」と題された一章にしなければならない。同様に、「謝辞 (Acknowledgements)」あるいは「献辞 (Dedications)」と題されたあらゆる部分もまとめなければならない。あなたは「推薦の辞 (Endorsements)」と題されたあらゆる部分も削除しなければならない。

6. 文書の収集

あなたは、この利用許諾契約書の下で発表された複数の文書で構成される収集著作物を作ることができる。その場合、それぞれの文書が逐語的に忠実に複製されることを保障するために他のすべての点でこの契約書の定める条件に従う限り、さまざまな文書中のこの契約書の個々の複製物を、収集著作物中に複製物一つ含めることで代用することができる。

あなたは、このような収集著作物から文書一つ取り出し、それをこの契約書の下で頒布することができる。ただしその際には、この契約書の複製物を抽出された文書に挿入し、またその他のすべての点でこの文書の逐語的に忠実な複製に関してこの契約書が定める条件に従わなければならない。

7. 独立した著作物の集積

『文書』あるいはその派生物を、他の別の独立した文書あるいは著作物と一緒にし、一巻の記憶装置あるいは頒布媒体に収めた編集著作物は、編集に起因する著作権が編集著作物に含まれる個々の著作物がその利用者に許可した法的権利を制限するよう行使されない限り、「集積」著作物と呼ばれる。『文書』が集積著作物に含まれる場合、この契約書は、『文書』と共にまとめられた他の独立した著作物には、それら自身が『文書』の派生物で無い限り適用されることにはならない。

このような『文書』の複製物において、この利用許諾契約書の第3項によりカバーテキストの掲載が要求されている場合、『文書』の量が集積著作物全体の2分の1以下であれば、『文書』のカバーテキストは集積著作物中で『文書』そのものの周りを囲む中表紙、あるいは『文書』が電子的形式である場合には表紙の電子的等価物にのみ配置するだけでよい。その場合以外は、カバーテキストは集積著作物全体を取り巻く印刷された表紙に掲載されなければならない。

8. 翻訳

翻訳は改変の一種と見做すので、あなたは『文書』の翻訳をこの利用許諾契約書の第4項の定める条件の下で頒布することができる。変更不可部分を翻訳によって置き換えるには著作権者の特別許可を必要とするが、元の変更不可部分に追加する形で変更不可部分の全てないし一部の翻訳を含めることはかまわない。この契約書や『文書』中の利用許諾告知、保証否認警告すべての英語原本も含める限り、あなたはこの契約書、告知、警告の翻訳を含めることができる。契約書や告知、警告に関して翻訳と英語原本との間に食い違いが生じた場合、英語原本が優先される。

典型的な例として、『文書』のある部分が原文で「Acknowledgements」、「Dedications」、あるいは「History」と題されていた場合、実際の題名を変更するには、題名を保存する(この契約書の第1項)ための条件(同第4項)を満たすことが必要となる。

9. 契約の終了

この利用許諾契約書の下で明確に提示されている場合を除き、あなたは『文書』を複製、改変、サブライセンス、あるいは頒布してはならない。このライセンスで指定されている以外の、『文書』の複製、改変、サブライセンス、頒布に関するすべての企ては無効であり、この契約書によって保証されるあなたの権利を自動的に終結させることとなる。しかし、この契約書の下であなたから複製物ないし諸権利を得た個人や団体に関しては、そういった人々がこの契約書に完全に従ったままである限り、彼らに与えられた許諾は終結しない。

10. 将来における本利用許諾契約書の改訂

フリーソフトウェア財団は、時によってGNU フリー文書利用許諾契約書の新しい改訂版を出版することができる。そのような新版は現在の版と理念においては似たものになるであろうが、新たに生じた問題や懸念を解決するため細部においては違ったものになるだろう。詳しくは <http://www.gnu.org/copyleft/> を参照せよ。

GNU フリー文書利用許諾契約書のそれぞれの版には、新旧の区別が付くようなバージョン番号が振られている。もし『文書』において、この契約書のある特定の版が「それ以降のどの版でも」適用して良いと指定されている場合、あなたはフリーソフトウェア財団から発行された(草稿として発表されたものを除く)指定の版かそれ以降の版のうちどれか一つを選び、その条項や条件に従うことができる。もし『文書』がこの契約書のバージョン番号を指定していない場合には、あなたはフリーソフトウェア財団から今までに出版された(草稿として発表されたものを除く)版のうちからどれか一つを選ぶことができる。

付録: この利用許諾契約書をあなたの文書に適用するには

Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

(訳:

Copyright (C) 西暦年 あなたの名前。
この文書を、フリーソフトウェア財団発行の GNU フリー文書利用許諾契約書(バージョン1.2かそれ以降から一つを選択)が定める条件の下で複製、頒布、あるいは改変することを許可する。変更不可部分、表カバーテキスト、裏カバーテキストは存在しない。この利用許諾契約書の複製物は「GNU フリー文書利用許諾契約書」という章に含まれている。
)

もし変更不可部分や表カバーテキスト、裏カバーテキストがあれば、「変更不可部分…は存在しない。」というところを以下で置き換えてください:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

(訳:

(章の題名を列記)は変更不可部分であり、(表カバーテキストを列記)は表カバーテキスト、(裏カバーテキストを列記)は裏カバーテキストである。

)

変更不可部分はあるがカバーテキストは存在しないなど、その他の三者の組み合わせに関しては、状況に合わせて上記二つの選択肢を混ぜてください。

あなたの文書に、他に類を見ない独自のプログラムコードのサンプルが含まれる場合、フリーソフトウェアにおいてそのコードを利用することを許可するために、そういったサンプルに関してはこの利用許諾契約書と同時にGNU 一般公衆許諾契約書のようなフリーソフトウェア向けライセンスのうちどれか一つを選択して適用してもよい、というような条件の下で発表することを推奨します。

