Red Hat Linux 9

Red Hat Linux 参照ガイド

Red Hat Linux 9: Red Hat Linux 参照ガイド
製作著作‘
‘
2003 ：Red Hat, Inc.


Red Hat, Inc.

1801 Varsity Drive
Raleigh NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

rhl-rg(JA)-9-Print-RHI (2003-02-13T19:20)

security@redhat.comキーのGPG fingerprint は:
CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

目次

はじめに

Red Hat Linux 参照ガイドへようこそ。

Red Hat Linux 参照ガイドには、Red Hat Linuxシステムに関する役に立つ情報が含まれています。Red Hat Linuxファイルシステムの構造など基本的な概念から、システムセキュリティや認証制御の細かい点に至るまで、本書を貴重な資料としてご活用ください。

本書はRed Hat Linuxシステムの機能をもう少し詳しく知りたいときに役立ちます。おもな項目の中から、以下の項目について説明します：

- ファイルシステムの構造
- 起動プロセス
- X Window System
- セキュリティツール
- ネットワークサービス

1. このマニュアルへの変更

このマニュアルは、明確な説明とRed Hat Linux 9の特徴を最新のものに更新する為に再構成されています。主な変更には以下のようなものがあります：

*X Window System*の章の更新

‘ *X Window System*は完全に改訂されており、明確性を目的に再構成されています。新しいフォントの設定法も追加されています。

新しい`sysconfig`の章

‘ ブートプロセス、*Init*、シャットダウンの章の`sysconfig`のセクションは拡張され、それ自身が章になっています。

*TCP*ラッパーと`xinetd` の章の更新

‘ 新規に更新された*TCP*ラッパーと`xinetd`の章は、全面的に入れ換えとなり明確な表現の為に再構成されました。

ユーザーとグループの章の更新

‘ ユーザーとグループの章は更新され、より明確になり、再構成されています。

ネットワークインターフェイス の章の更新

‘ ネットワークインターフェイスの章は更新され再構成されています。

Apache HTTP サーバーの章の更新

‘ Apache HTTP サーバーのバージョン1.3からバージョン2.0への移行の為のガイドは更新されています。サーバーの更新オプションのリストは更に更新され、再構成されました。**Gary Benson**氏と**Joe Orton**氏にはApache HTTP サーバーの移行ガイド更新に対して多大な貢献をして頂きました。

本ガイドを読む前に、インストールに関する問題について*Red Hat Linux* インストールガイド、基本的なLinuxの概念に付いて*Red Hat Linux* 入門ガイド、そして全般的なカスタマイズに関して*Red Hat Linux* カスタマイズガイドの内容をそれぞれ知っておく必要があります。*Red Hat Linux* 参照ガイドには、高度なユーザーの為のトピックについての情報が含まれています。

HTML版とPDF版のすべてのRed Hat Linuxマニュアルは以下のオンラインサイトで入手できます：http://www.redhat.com/docs

注意

このマニュアルはできる限り最新の情報を反映していますが、この文書が最終段階になるまでに入手できなかった内容の情報に関してはRed Hat Linuxリリースノートをお読み下さい。リリースノートは、Red Hat LinuxのCD 1枚目の中と以下のURLで読むことができます：

```
http://www.redhat.com/docs/manuals/linux
```

2. 自分に最適のマニュアルを見つける

自分のLinuxの知識レベルに適したマニュアルを参照することが重要です。正しいマニュアルがなければ、とかく圧倒されがちになったり、逆に疑問に答えてくれる情報が見つからないこともあります。*Red Hat Linux* 参照ガイドでは、より技術的に掘り下げた内容と、Red Hat Linuxシステムのオプションについて説明します。本章では、必要な情報が本書で見つかるか、又はオンラインソースを含めて他のRed Hat Linuxマニュアルを探すべきかかどうかの、判断材料を提供します。

Red Hat Linuxを使用するユーザーには3つの異なるカテゴリーがあり、それぞれのカテゴリーはそれぞれ異なるドキュメントと情報源が必要です。どこから開始すべきかの判定を助けるには、自分の経験レベルを確認することです：

*Linux*を初めて使用する

‘ これまでにLinux（またはLinuxに類する）オペレーティングシステムをまったく使用したことがないか、Linuxの経験がごくわずかしかないユーザーです。他のオペレーティングシステム（Windowsなど）を使用した経験があるユーザーも、ないユーザーも含みます。これに該当しますか。該当する場合は、項2.1に進んでください。

*Linux*の経験が少しある

‘ これまでにLinuxをインストールし、正常に使用したことがある（ただしRed Hat Linuxは未経験な）ユーザーと、Linuxに類する他のオペレーティングシステムで同様な経験があるユーザーです。これに該当しますか。該当する場合は、項2.2に進んでください。

*Linux*の十分な経験がある

‘ これまでにRed Hat Linuxをインストールし、正常に使用したことがあるユーザーです。該当する場合は、項2.3に進んでください。

2.1. Linuxが初めてのユーザーのためのマニュアル

Linuxが初めてのユーザーのために、プリント、システムの起動、ハードディスクのパーティション設定など、あらゆるテーマに関して用意された情報の量は、たぶん混乱を招くでしょう。最初は1歩下がって、これらの上級レベルの情報をアタックする前に、Linuxの動作に中心を置いた知識を基本から習得していくことがこれからの助けになります。

最初の目標は、いくつかの役立つマニュアルを用意することです。この点はいくら強調しても、強調しすぎることはありません。マニュアルがなければ、思いどおりにRed Hat Linuxシステムを動かすことができずに、目的を達成するには難しくなります。

必要な種類のLinuxマニュアルは以下のとおりです。

- *Linux*の簡単な歴史—現在のLinuxは多くの側面で、歴史的先例があってこその姿になっています。Linux文化もまた過去のイベント、必要、要求に根ざしています。Linuxの歴史に関する基本的な知識があれば、顕在化する前に多くの潜在的な問題を解決する方法がわかります。
- *Linux*の動作に関する説明—Linuxカーネルの奥にある側面を詮索する必要はありませんが、Linuxがどのようにできているのかを知ることは重要です。これまで他のオペレーティングシステムを使用していた場合には、コンピュータが動作する仕組みについて前提と考えていたことのなかに、そのオペレーティングシステムからLinuxには当てはめて考えられない点もあるため、とくにこれが重要になります。
- コマンドの概要に関する手引き（例を含む）— これはおそらく、Linuxマニュアルで探すもののなかで最も重要なものです。ジョブ全体を実行する少数の巨大な（そして複雑な）コマンドを使用するよりも、多くの小さいコマンドをさまざまな組み合わせで結合して使用するほうがよい、というのがLinuxに底流する設計思想です。ただし、Linuxが作業を進めるこうしたアプローチの例示がなければ、Red Hat Linuxシステムで使用できる膨大な数のコマンドに圧倒されかねません。

  使用可能なLinuxコマンドすべてを記憶する必要はありません。タスクの遂行に必要な特定のコマンドを見つけるテクニックは、複数存在します。必要なのは、Linuxが機能する一般的な仕組み、遂行するタスク、コマンドの実行に必要な正しい指示が得られるツールの呼び出し方を知ることだけです。

Red Hat Linux インストールガイドは、Red Hat Linuxシステムを正しくインストールし、初期化するのを助ける優れたリファレンスです。*Red Hat Linux* 入門ガイドは、基本的なシステムコマンド、グラフィカルデスクトップ環境、それに数多くの根本的な概念までカバーします。この2冊の本からスタートして、Red Hat Linuxの知識の基盤を作ることができるように使用して下さい。基本を理解すればしばらくすると、より長くて複雑な概念も明確に判るようになります。

Red Hat Linuxマニュアルを読むほかにも、ほとんど費用がかからないか無料で、優れたマニュアルリソースを利用できます。

2.1.1. Linux Webサイト

- http://www.jp.redhat.com—Red Hat社のWebサイトには、Linux Documentation Project（LDP）、Red Hat Linuxマニュアルのオンライン版、FAQ（よくある質問）、最寄りのLinuxユーザーグループの検索に役立つデータベース、Red Hat Support Knowledge Baseに格納された技術情報、などへのリンクがあります。
- http://www.linuxheadquarters.com—Linux本社のWebサイトで、多様なLinuxタスクのステップバイステップによるガイドが特長です。

2.1.2. Linuxニュースグループ

問題の解決に挑戦する参加者の議論をウォッチするか、あるいは積極的に質問をポストしたり、他の参加者の質問に答えたりすることで、ニュースグループに参加できます。経験の深いLinuxユーザーは、さまざまなLinux問題で新規ユーザーを手助けできる際立った存在として知られています。— 特に妥当な場所での真剣な疑問に対しては、なおさらです。ニュースリーダーアプリケーションが利用できない場合は、以下のWebサイトでこの情報にアクセスすることが出来ます。http://groups.google.com/. Linux関連のニュースグループは数十も存在します。主に以下のようなものがあります：

- linux.help—経験豊富なLinuxユーザーから助言が得られる最高の場所です。
- linux.redhat—主として、Red Hat Linux固有の諸問題を議論するニュースグループです。
- linux.redhat.install—インストールについての質問をポストする、また他のユーザーが同様の問題をどう解決したか検索する、などに適したニュースグループです。

- linux.redhat.misc—上に掲げた在来の分類から外れる質問や、ヘルプをポストするニュースグループです。
- linux.redhat.rpm—特定の目的遂行のため**RPM**を使用していて問題が発生した場合に、ヘルプを求める最適な場所です。

2.1.3. Linuxの入門書

- *Red Hat Linux for Dummies, 2nd Edition*Jon "maddog" Hall著、IDG刊
- *Special Edition Using Red Hat Linux*Alan Simpson、John Ray、Neal Jamison共著、Que刊
- *Running Linux*Matt Welsh、Lar Kaufman共著、O'Reilly & Associates刊
- *Red Hat Linux 8 Unleashed* Bill Ball、Hoyle Duff共著、Pearson Education刊

上に挙げた書籍は、いずれもRed Hat Linuxシステムの基本知識に関する優れた最初の情報源です。本書で説明するさまざまな項目に関して、さらに詳しい情報が知りたい場合は、おもな各章のその他のリソースのセクションに掲げる書籍一覧を参考にしてください。

2.2. 経験のあるユーザーのためのマニュアル

他のLinuxディストリビューションを使った経験がある場合、最もよく使用されるコマンドについては基本的な知識があるものと考えられます。自分でLinuxシステムをインストールしたことがあるか、インターネットで見つけたソフトウェアをダウンロードしてコンパイルしたことがあるユーザーもいるはずです。しかし、Linuxのインストール後も、システム設定の問題で混乱することがしばしばあります。

Red Hat Linux カスタマイズガイドは、特定の目的を達成するように設定することが出来るRed Hat Linuxシステムのさまざまな活用法について説明できるように設計されています。このマニュアルを使用して特殊な設定オプションとそれの利用法について学ぶことができます。

Red Hat Linux カスタマイズガイドで扱われていないソフトウェアをインストールする場合は、他のユーザーが同様な環境にどう対応したかが、しばしば参考になります。この場合、次のサイトから利用できるLinux Documentation ProjectのHOWTO文書が役立ちます。http://www.redhat.com/mirrors/LDP/HOWTO/HOWTO-INDEX/howtos.htmlHOWTO文書は、低レベルカーネルの難解な変更から、アマチュア無線局の運営にLinuxを使用する方法まで、Linux独自のさまざまな側面を説明しています。

2.3. Linux精通者のためのマニュアル

古くからのRed Hat Linuxユーザーであれば、特定のプログラムを理解する最善の方法の1つが、ソースコードあるいは設定ファイルの精読だとわかっているはずです。しかもRed Hat Linuxの最大の利点は、読もうと思えばだれにでもソースコードが入手できることです。

明らかに、すべての人がプログラマーではありませんから、ソースコードは多分、ユーザーには役に立たないかも知れません。しかし、これを読むだけの知識と能力が有れば、ソースコードはすべての答えを持っています。

3. 表記方法

本マニュアルを読むと、特定の単語が、異なるフォント、書体、サイズ、太さで表記されていることにお気づきになるはずです。この強調表示は規則にしたがって行われています。異なる単語であって

も、同じスタイルで表記されている場合は、特定のカテゴリに含まれることを示しています。この様に表記されている単語のタイプには次のような物があります：

`command`

‘ Linux コマンド(場合によっては、その他のオペレーティングシステムコマンド)はこの様に表記します。この様に表記されている場合、その文字列をコマンドラインから入力し、[Enter]キーを押せば、そのコマンドを実行することができます。コマンドの中には、それとは異る表記の部分(例えば、ファイル名)が含まれていることもあります。この場合は、その部分もコマンドの一部であり、全体として1つのコマンドを構成します。例えば：

`cat testfile`コマンドは、現在の作業ディレクトリにある`testfile`という名前のファイルの内容を表示するのに使用します。

`filename`

‘ ファイル名、ディレクトリ名、パス、RPMパッケージ名は、この様に表記します。このスタイルはその名前の特定のファイルやディレクトリがRed Hat Linuxシステム上に存在することを示しています。例えば：

ホームディレクトリの`.bashrc`ファイルには、そのユーザー用のbashシェル定義とエイリアスが保存されています。

`/etc/fstab`ファイルには、各システムデバイスとファイルシステムの情報が保存されています。

Webサーバーのログファイル解析プログラムを使用するためには`webalizer` RPMをインストールしてください。

application

‘ この表記はプログラムがエンドユーザーアプリケーションである(システムソフトウェアではない)ことを示します。例えば：

Mozillaを使用してWebを閲覧します。

[key]

‘ キーボード上のキーは以下のように表記します。例えば：

[Tab]キーによる補完機能を使用するには、1文字入力してから[Tab]キーを押します。端末は、ディレクトリ内のその文字で始まるファイルのリストを表示します。

[key]-[combination]

‘ キーの組み合わせは、次のように表記されます。例えば：

[Ctrl]-[Alt]-[Backspace] キーの組合せはグラフィカル操作を終了させて、グラフィカルログイン画面、又は、コンソールに戻します。

GUI インターフェイス上にあるテキスト

‘ GUIの画面やウィンドウ上に使われる見出しや文字列は、次の様に表記します。この様に表記されている場合、それは特定のGUI画面か、そこにある特定の項目を指す為に使われています。(チェックボックスやフィールドに付けられた文字列など) 例えば：

スクリーンセーバーを停止するときにパスワードを要求するようにしたいときは**パスワードを要求**チェックボックスを選択します。

GUI画面、又はウィンドウ上のメニュー上部

‘ この表記がある時は、それがプルダウンメニューの最上位の項目だということを表します。GUI画面上にあるその文字列をクリックすると、そのメニューの残りが表示されます。例えば：

GNOMEターミナル上の**ファイル**の下に、同じウィンドウ内に複数のシェルプロンプトを開くことが出来る**新規タブ**オプションがあります。

GUIメニューを連続して操作する必要があるときは、次の例のように表記します：

(パネル上の)**メインメニューボタン**=> **プログラム** => **Emacs**と進んで**Emacs**テキストエディタを開始します。

GUI画面、又はウィンドウ上のボタン

‘ この表記は、GUI画面上にクリックできるボタン上にテキストがあることを示します。例えば：

戻る ボタンを押して、最後に表示したウェブページに戻ります。

`computer output`

‘ この表記のテキストがある場合、それはコマンドライン上でコンピュータが表示するテキストを示します。コマンドを入力した結果や、エラーメッセージ、及びスクリプトやプログラムへのユーザー入力の為の対話式プロンプトなど、この表記になります。例えば：

lsコマンドを使用してディレクトリの内容を表示します：

```
$ ls
Desktop       about.html   logs    paulwesterberg.png
Mail        backupfiles   mail    reports
```

コマンドの実行結果として表示される出力(この場合は、ディレクトリの内容)は、上記の様に表示されます。

`prompt`

‘ コンピュータが入力待ちであることを示すプロンプトは、この表記で示されます。例えば：

```
$
#
[stephen@maturin stephen]$
leopard login:
```

`user input`

‘ コマンドラインかGUI画面上のテキストボックスにユーザーが入力しなければならない文字列は、このように表記します。次の例では、**`text`**がこの表記で示されています：

システムでテキストベースのインストールプログラムに起動するには、`boot:` プロンプトで、**`text`**と入力する必要があります。

さらには、特定の情報について、ユーザーの注意を引くために幾つかの特策があります。システムに対する重要度に応じて、これらの項目は、ヒント、注意、重要、用心、警告と区分されています。例えば：

注意

Linuxは、大文字/小文字を区別します。つまりROSEとrOsEは異なります。

ヒント

/usr/share/docディレクトリには、システムにインストールされているパッケージの為の追加のドキュメントが含まれています。

重要

DHCP設定ファイルを変更する場合は、その変更はDHCPデーモンを再起動するまで、有効になりません。

用心

日常の操作はrootで実行しないで下さい。—システム管理の作業に、rootアカウントで操作をする必要があるとき以外は、通常のユーザーアカウントを使用して下さい。

警告

手動でパーティション設定を行わない場合、サーバーシステムインストールを実行すると、インストール先のハードディスクドライブ上にある既存のパーティションはすべて削除されます。保存する必要のあるデータがないことが確実である場合以外は、このインストールクラスは選択しないでください。

4. マウスの使い方

Red Hat Linuxは、3ボタンマウスの使用を前提として設計されています。2ボタンマウスを使っている場合、インストールの工程で3ボタンエミュレーションを選択しておく必要があります。3ボタンエミュレーションを選択をすると、両方のマウスボタンを同時に押すことによって、3番目のボタン（中央ボタン）を押すことと同じ操作になります。

本ガイドで単に「マウスでクリックする」とある場合は、左マウスボタンをクリックすることを意味します。中央マウスボタンか右マウスボタンを使うときは、そのように明記します（マウスを左利き用に設定している場合は、逆になります）。

「ドラッグアンドドロップ」という表現にはなじんでいるユーザーも多いと思われます。GUIデスクトップ上で何かをドラッグアンドドロップするようにと指示がある場合は、まず、そのアイテムをクリックします。その際、マウスボタンは押したままにし、そのままマウスを移動して、アイテムをドラッグ(移動)させます。目的の位置まで移動したら、マウスボタンから指をはなし、つかんでいたアイコンをドロップします。

5. Xでのテキストのコピーと貼り付け

X Window Systemでマウスを使うとテキストのコピーと張り付けは簡単にできます。テキストをコピーするには、まずテキストをマウスでクリックして、コピーする範囲の上をドラッグします。ドラッグした範囲は強調表示されます。テキストを貼り付けるには、貼り付ける場所の決めた位置で、中央マウスボタンをクリックします。

6. 今後の発行予定

Red Hat Linux 参照ガイドは、Red Hat Linuxユーザーに有益でタイムリーなサポートを提供するというRed Hatのお約束の一環です。今後発行するマニュアルでは、システム構造、組織の変更、新しいパワフルなセキュリティツール、そしてユーザーがRed Hat Linuxシステムでパワーとそれを活用する能力を拡張することに役立つその他のリソース、などに関する詳しい情報を提供する予定です。

このマニュアルはユーザーの助言を反映させる場所でもあります。

6.1. フィードバックを募集します

Red Hat Linux 参照ガイドの中でエラーを発見したり、又はこのマニュアルを改善する方法を思い付いた場合などに、貴方の御連絡をお待ちしています。報告はBugzilla(http://bugzilla.redhat.com/bugzilla)に提出して下さい。この場合*rhl-rg*のコンポーネントを指定して提出をお願いします。

次に示す本マニュアルのIDを忘れずに明記してください。

```
rhl-rg(JA)-9-Print-RHI(2003-02-13T19:20)
```

このIDによって、お手持ちの本ガイドの正確なバージョンがわかります。

改善策をお寄せいただく場合は、改善点をできるだけ具体的にお書きください。エラーのご指摘の場合は、誤記部分が容易にわかるように、章や節とともに周辺テキストの一部をお書き添えください。

7. サポートを受ける為のユーザー登録

Red Hat Linux 9のエディションのいずれかをお持ちの場合は、忘れずに登録をして、Red Hatの登録ユーザーとしての特典をご利用ください。

購入されたRed Hat Linux製品の種類にしたがって、以下の特典のいくつか、またはすべてをご利用いただけます：

- Red Hat サポート— インストール時の疑問について、Red Hat, Inc.のサポートチームからのサポートが受けられます。
- Red Hat ネットワーク— 簡単にパッケージをアップデートしたり、お使いのシステム用にカスタマイズされたセキュリティ通知を受けることができます。詳細については、http://rhn.redhat.com を参照してください。
- *Under the Brim: The Red Hat E-Newsletter* — 毎月、最新のニュースと製品情報が直接Red Hatから送信されます。

ユーザー登録をするには、http://www.redhat.com/apps/activate/にアクセスして下さい。登録時に使用する製品番号(Product ID)は、Red Hat Linuxボックスの黒と赤と白のカードに記載されています。

Red Hat Linuxの技術サポートについては、*Red Hat Linux* インストールガイドの付録のテクニカルサポートのご利用方法を参照してください。

最後になりましたが、Red Hat Linuxをお選びいただきありがとうございました。

*Red Hat*ドキュメンテーションチーム一同

I. システムへの参照

システムを効率良く管理するには、システムのコンポーネントを知り、全体がどのようにうまく統合すべきかを考慮することが重要です。このセクションでは、システムの多くの重要な側面を簡単に説明しています。起動プロセス、基本的なファイルシステムのレイアウト、重大なシステムファイルとファイルシステムの場所、そしてユーザーとグループの使用の背後にある基本的概念を説明しています。更には、X Window Systemが詳細に説明してあります。

目次

第1章

ブートプロセス、Init、シャットダウン

Red Hat Linuxの重要でパワフルな側面の1つは、このオペレーティングシステムの開始に使用するオープンでユーザー設定可能な方法です。ユーザーは、ブート時に立ち上げるプログラムの指定を含む、ブートプロセスの多くの側面を自由に設定できます。同じように、停止のプロセスは殆んどカスタマイズを必要としませんが、システムのシャットダウンも組織化され設定可能な方法で優しくプロセスを停止します。

プロセスのブートとシャットダウンの仕組みを理解することは、Red Hat Linuxのカスタマイズを可能にするだけでなく、システムの開始とシャットダウンに関連する問題のトラブルシューティングをも簡単にします。

1.1. ブートプロセス

以下にx86システム用のブートプロセスの基本的ステージを示します：

1. システムBIOSが、システムをチェックしそれからプライマリハードディスクのMBRにある第1ステージのブートローダーを立ち上げます。
2. 第1ステージのブートローダーはそれ自身をメモリにロードして、`/boot/` パーティションから第2ステージのブートローダーを立ち上げます。
3. 第2ステージブートローダーはカーネルをメモリにロードし、今度はカーネルが必要なモジュールをロードして読み込み専用ルートパーティションをマウントします。
4. カーネルはブートプロセスの制御を`/sbin/init`プログラムに渡します。
5. `/sbin/init`プログラムは全てのサービスとユーザースペースツールをロードし、そして`/etc/fstab`にリストしてある全てのパーティションをマウントします。
6. ユーザーは、新しくブートしたLinuxシステムのログインプロンプトの表示を見ることが出来ます。

ブートプロセスの設定が、シャットダウンプロセスのカスタマイズよりももっと一般的であるため、この章の残りの部分では、ブートプロセスの仕組みと、それを特定のニーズに適合させるカスタマイズの方法を詳細に説明していきます。

1.2. ブートプロセスの詳細

ブートプロセスの始まりは使用しているハードウェアプラットフォームによって異なります。しかし、カーネルが見付かり、ブートローダーでロードされるというデフォルトのブートプロセスは全てのアーキテクチャーを通じて共通です。この章ではx86アーキテクチャーに焦点を置いています。

1.2.1. BIOS

x86コンピュータがブートされると、プロセッサはシステムメモリの最後部を見て*Basic Input/Output System*、いわゆる*BIOS* プログラムを探しそれを実行します。BIOSは、ブートプロセスの最初のステップだけでなく周辺機器への低レベルインターフェイスも提供します。この為、BIOSは読み込み専用の固定メモリとして書かれており、何時でも使用可能です。

他のプラットフォームは、x86システムのBIOSの働きとほぼ同じ様な低レベルタスクを演じるための別種のプログラムを使用します。例えば、Itaniumベースのコンピュータは*EFI* (*Extensible Firmware Interface*)シェル を使用し、Alphaシステムは*SRM* コンソールを使用します。

ロードされると、BIOSはシステムを検査し、周辺機器の検索とチェックを行い、システムをブートする為の有効なデバイスを調べます。BIOSは通常、起動可能なメディアを求めて存在するフロッピィディスクドライブとCD-ROMドライブを検査します。そしてそれがない場合、システムのハードドライブへと移動します。殆んどの場合、ブート過程でドライブを見て回る順序はBIOSで設定されておりそれがプライマリIDEバス上のマスターIDEデバイスを見に行きます。BIOSはそこで、このデバイスの最初のセクターにあるマスターブートレコード、いわゆる*MBR*のプログラムをメモリにロードします。MBRは、サイズとしては512バイトしかなく、マシンをブートするためのブートローダーと呼ばれるマシンコード指示文とパーティションテーブルを一緒に含んでいます。BIOSがブートローダープログラムを見付けて、メモリにロードすると、ブートプロセスの制御はブートローダーに任せます。

1.2.2. ブートローダー

このセクションでは、x86プラットフォーム用のブートローダーについて考察しています。システムのアーキテクチャーによって、ブートプロセスには少々相違がありますので、x86以外のブートローダーについての概要は項1.2.2.1で御覧下さい。

Red Hat Linuxでは、2種類のブートローダーが利用できます：*GRUB* か又は*LILO*です。GRUBがデフォルトのブートローダーですが、LILOもこれを必要な人又は、好む人の為に利用可能です。GRUB又はLILOの使用とその設定については第2章を参照して下さい。

x86プラットフォーム用のこれら両方のブートローダーは、少なくとも2つのステージに分割されています。第1ステージは、MBR上の小規模のマシンコードバイナリです。この唯一の仕事は第2ステージのブートローダーを見付けてその最初の部分をメモリにロードすることです。

GRUBは、より新しいブートローダーでext2 とext3[1]パーティションを読み込める優位性を持ち、その設定ファイル— `/boot/grub/grub.conf`—をブート時にロードします。このファイルの編集法のついては項2.7を御覧下さい。

LILOでは、第2ステージブートローダーは、MBRの情報を使用してユーザーが利用できるブートオプションを決定します。これは、設定の変更がされた場合やカーネルが手動で更新された場合などはいつも、MBRに適切な情報を書き込む為に`/sbin/lilo -v -v`コマンドが実行される必要があることを意味します。その実践法については項2.8を参照して下さい。

ヒント

Red Hat 更新エージェントを使用してカーネルをアップグレードする場合は、ブートローダー設定ファイルは自動的に更新されます。Red Hat ネットワークの詳細はオンラインのサイト、URL:https://rhn.redhat.comで御覧下さい。

第2ステージブートローダーがメモリにロードされると、ブートするように設定されている別のオペレーティングシステムやカーネルを表示したRed Hat Linuxの初期グラフィカル画面が提示されます。この画面で、ユーザーは矢印キーを使用してブートしたいオペレーティングシステムやカーネルを選択して[Enter]キーを押します。何もキーを押さなければ、ブートローダーは、設定してある待ち時間が経過した後にデフォルトの選択をロードします。

注意

SMP(Symmetric Multi-Processor)カーネルのサポートがインストールされている場合、システムが始めてブートする時に複数のオプションが表示されます。この状況では、LILOでは、SMPカーネル用である`linux`と、シングルプロセッサ用である`linux-up`が表示されます。GRUBは、SMPカー

1. GRUBは、そのジャーナルファイルを無視して、ext3ファイルシステムをext2として読み込みます。ext3ファイルシステムに関する詳細は*Red Hat Linux カスタマイズガイド* の中の*ext3* ファイルシステムの章を参照して下さい。

ネル用にRed Hat Linux(<*kernel- version*>-smp)と、シングルプロセッサ用に Red Hat Linux(<*kernel-version*>)を表示します。

SMPカーネルを使用中に何か問題が発生すれば、再起動してSMPカーネル以外を選択するようにします。

第2ステージブートローダーは、ブートするカーネルを決定すると、次に/boot/ディレクトリ内の対応するカーネルバイナリを見付けます。このカーネルバイナリは以下の形式を使用して名前を付けています。—/boot/vmlinuz- <*kernel-version*> ファイル(<*kernel-version*> はブートローダー設定に指定してあるカーネルバージョンに相当します)。

ブートローダーを使用してカーネルにコマンド行の引数を与える方法については第2章を参照して下さい。GRUB 又はLILOでランレベルを変更する方法については項2.10を御覧下さい。

ブートローダーは、それからinitrdと呼ばれる適切な初期*RAM*ディスクイメージをメモリに配置します。initrdはカーネルによってシステムのブートに必要なドライバをロードするのに使用されます。これは、SCSIハードドライブがある場合、又は、システムがext3ファイルシステムを使用する場合に特に重要になります。[2].

警告

いかなる理由があっても、ファイルシステムから/initrd/ ディレクトリを削除しないで下さい。このディレクトリを削除するとブート時にカーネルパニックエラーでシステムが停止する原因になります。

カーネルと*initrd*イメージがメモリにロードされるとブートローダーはブートプロセスの制御をカーネルに託します。

GRUB とLILOブートローダーのより詳しい概要については第2章を御覧下さい。

1.2.2.1. 他のアーキテクチャー用ブートローダー

Red Hat Linuxカーネルがロードしてブートプロセスをinitコマンドに渡すと全てのアーキテクチャーを通じて同じ工程の流れが起きます。各アーキテクチャーのブートプロセス間での主要な相違は、カーネルを探してロードするアプリケーションにあります。

例えば、Alphaアーキテクチャーはabootブートローダーを使用し、ItaniumアーキテクチャーはELILOブートローダーを使用します。

これらのプラットフォーム特有のブートローダー設定に関する情報は*Red Hat Linux* インストールガイドを参照して下さい。

1.2.3. カーネル

カーネルがロードされると、すぐにコンピュータのメモリを初期化して設定し、全てのプロセッサ、I/Oサブシステム、記憶装置を含むシステムに接続されている各種ハードウェアを設定します。カーネルはそれからメモリ内の事前設定してある場所の圧縮されたinitrdイメージを見付けて展開し、マウントし、全ての必要なドライバをロードします。次にLVM 又はソフトウェアRAIDなどのファイルシステム関連の仮想デバイスを初期化して、その後initrdディスクイメージをアンマウントし、ディスクイメージで使用されていたメモリ領域を空けます。

カーネルはその後、ルートデバイスを作成し、読み込み専用のルートパーティションをマウントしてから、未使用のメモリを開放します。

2. *initrd*作成の詳細に関しては、*Red Hat Linux* カスタマイズガイドの中の*ext3*ファイルシステムの章を御覧下さい。

この時点で、カーネルはメモリにロードされ、機能できます。しかし、ここではシステムに価値のある入力をするようなユーザーアプリケーションが何もありませんので、することはありません。

ユーザー環境をセットアップするために、カーネルは/sbin/init プログラムを実行します。

1.2.4. /sbin/initプログラム

The /sbin/init プログラム(initとも呼ばれます)がブートプロセスの残りを統制して、ユーザーの為の環境を設定します。

initコマンドがスタートする時、それは、Red Hat Linuxシステム上で自動的に起動するすべてのプロセスの親か、親の親になります。それはまず、/etc/rc.d/rc.sysinitスクリプトを実行します。これは環境パスの設定、スワッピングの開始、ファイルシステムのチェックなどを実行します。そしてシステム初期化時に行っておく必要のあるすべてのことを処理します。例えば、殆んどのシステムはクロックを使用しますので、rc.sysinitは/etc/sysconfig/clockの設定ファイルを読み込んで、ハードウェアクロックを初期化します。もう1つの例としては、初期化すべき特殊なシリアルポートプロセスがある場合、rc.sysinitは/etc/rc.serial ファイルを実行します。

initコマンドは、それぞれの*SysV init* のランレベルがどの様にセットアップするかが記述してある/etc/inittabスクリプトを実行します。[3] 他の作業も含めて、/etc/inittabはデフォルトのランレベルを設定し、/sbin/updateが、あるランレベルが開始される時には必ず実行するように指示します。[4]

次に、initコマンドはシステム用に、ソース機能ライブラリ、/etc/rc.d/init.d/functionsをセットします。これにはプログラムをどのようにスタートする又はキルするか、そしてどの様にプログラムのPIDを決定するかが記述されています。

init プログラムは/etc/inittab内にデフォルトとして指定してあるランレベル用の適切なrc ディレクトリを調べて、すべてのバックグラウンドプロセスを開始します。rcディレクトリはそれが代表するランレベルに相当する番号を付けています。例えば、/etc/rc.d/rc5.d/はランレベル5のディレクトリとなります。

ランレベル5へブートする時、initプログラムは/etc/rc.d/rc5.d/ディレクトリ内を調べ、どのプロセスが開始と停止するのかを決定します。

以下に/etc/rc.d/rc5.d/ディレクトリのサンプル一覧を表示します：

```
K05innd -> ../init.d/innd
K05saslauthd -> ../init.d/saslauthd
K10psacct -> ../init.d/psacct
K12cWnn -> ../init.d/cWnn
K12FreeWnn -> ../init.d/FreeWnn
K12kWnn -> ../init.d/kWnn
K12mysqld -> ../init.d/mysqld
K12tWnn -> ../init.d/tWnn
K15httpd -> ../init.d/httpd
K15postgresql -> ../init.d/postgresql
K16rarpd -> ../init.d/rarpd
K20bootparamd -> ../init.d/bootparamd
K20iscsi -> ../init.d/iscsi
K20netdump-server -> ../init.d/netdump-server
K20nfs -> ../init.d/nfs
K20rstatd -> ../init.d/rstatd
K20rusersd -> ../init.d/rusersd
K20rwalld -> ../init.d/rwalld
K20rwhod -> ../init.d/rwhod
K24irda -> ../init.d/irda
```

3. SysV initランレベルの詳細情報は項1.4で御覧下さい。
4. updateコマンドは、変更されたバッファをディスクへフラッシュバックするために使用されます。

```
K25squid -> ../init.d/squid
K28amd -> ../init.d/amd
K34dhcrelay -> ../init.d/dhcrelay
K34yppasswdd -> ../init.d/yppasswdd
K35atalk -> ../init.d/atalk
K35dhcpd -> ../init.d/dhcpd
K35smb -> ../init.d/smb
K35vncserver -> ../init.d/vncserver
K35winbind -> ../init.d/winbind
K40mars-nwe -> ../init.d/mars-nwe
K45arpwatch -> ../init.d/arpwatch
K45named -> ../init.d/named
K45smartd -> ../init.d/smartd
K46radvd -> ../init.d/radvd
K50netdump -> ../init.d/netdump
K50snmpd -> ../init.d/snmpd
K50snmptrapd -> ../init.d/snmptrapd
K50tux -> ../init.d/tux
K54pxe -> ../init.d/pxe
K55routed -> ../init.d/routed
K61ldap -> ../init.d/ldap
K65identd -> ../init.d/identd
K65kadmin -> ../init.d/kadmin
K65kprop -> ../init.d/kprop
K65krb524 -> ../init.d/krb524
K65krb5kdc -> ../init.d/krb5kdc
K70aep1000 -> ../init.d/aep1000
K70bcm5820 -> ../init.d/bcm5820
K74ntpd -> ../init.d/ntpd
K74ups -> ../init.d/ups
K74ypserv -> ../init.d/ypserv
K74ypxfrd -> ../init.d/ypxfrd
K84bgpd -> ../init.d/bgpd
K84ospf6d -> ../init.d/ospf6d
K84ospfd -> ../init.d/ospfd
K84ripd -> ../init.d/ripd
K84ripngd -> ../init.d/ripngd
K85zebra -> ../init.d/zebra
K90isicom -> ../init.d/isicom
K92ipvsadm -> ../init.d/ipvsadm
K95firstboot -> ../init.d/firstboot
S00microcode_ctl -> ../init.d/microcode_ctl
S05kudzu -> ../init.d/kudzu
S08ip6tables -> ../init.d/ip6tables
S08ipchains -> ../init.d/ipchains
S08iptables -> ../init.d/iptables
S09isdn -> ../init.d/isdn
S10network -> ../init.d/network
S12syslog -> ../init.d/syslog
S13portmap -> ../init.d/portmap
S14nfslock -> ../init.d/nfslock
S17keytable -> ../init.d/keytable
S20random -> ../init.d/random
S24pcmcia -> ../init.d/pcmcia
S25netfs -> ../init.d/netfs
S26apmd -> ../init.d/apmd
S28autofs -> ../init.d/autofs
S44acpid -> ../init.d/acpid
```

```
S55sshd -> ../init.d/sshd
S56rawdevices -> ../init.d/rawdevices
S56xinetd -> ../init.d/xinetd
S80sendmail -> ../init.d/sendmail
S80spamassassin -> ../init.d/spamassassin
S84privoxy -> ../init.d/privoxy
S85gpm -> ../init.d/gpm
S90canna -> ../init.d/canna
S90crond -> ../init.d/crond
S90cups -> ../init.d/cups
S90xfs -> ../init.d/xfs
S95anacron -> ../init.d/anacron
S95atd -> ../init.d/atd
S97rhnsd -> ../init.d/rhnsd
S99local -> ../rc.local
S99mdmonitor -> ../init.d/mdmonitor
```

この一覧に表示されているように、実際にサービスを開始や停止をするスクリプトはどれも/etc/rc.d/rc5.d/ディレクトリにはありません。むしろ、/etc/rc.d/rc5.d/内のファイル全ては/etc/rc.d/init.d/ディレクトリ内に位置してあるスクリプトを指すシンボリックリンクなのです。シンボリックリンクは各rcディレクトリ内に使用され、それらが参照するスクリプトに影響を与えることなく、それらを作成、修正、削除したりしてランレベルを再構成できるようにします。

各シンボリックリンクの名前は、Kか、又はSで始まります。K リンクはそのランレベル上でキルされるプロセスで、Sで始まる物はスタートされます。

initコマンドは最初に、/etc/rc.d/init.d/ <*command*>stopコマンド(<*command*>はキルされるプロセス)を発行することにより、ディレクトリのK シンボリックリンクを停止します。そして/etc/rc.d/init.d/<*command*> startコマンドを発行してSシンボリックリンクを開始します。

ヒント

システムがブートを完了した後は、rootとしてログインして、同じスクリプトを使用してサービスの開始と停止を実行することが出来ます。例として、コマンド/etc/rc.d/init.d/httpd stopはApache Webサーバを停止します。

それぞれのシンボリックリンクは開始順を決定するために番号が付いています。サービスが開始、又は停止される順序は、この番号を変更することで変えることが出来ます。番号が低いとより早くスタートします。同じ番号を持つシンボリックリンク同士はアルファベット順に開始されます。

注意

initプログラムが実行する最後の役目の内の1つは/etc/rc.d/rc.localファイルです。このファイルはシステムのカスタマイズに役に立つものです。rc.local ファイルの使用についての詳細は項1.3で御覧下さい。

initコマンドがそのランレベルに適切なrcディレクトリを通過すると/etc/inittab スクリプトは、ランレベルに割り当てられた各仮想コンソール(ログインプロンプト)用に/sbin/mingettyプロセスをフォーク(分岐)します。ランレベル2～5は6つの仮想コンソールを取得し、ランレベル1(シングルユーザーモード)は1つしか取得しないで、また、ランレベルの0と6は仮想コンソールを取得しませ

ん。/sbin/mingetty プロセスは通信の経路を*tty*デバイス[5]に対して開き、そのモードを設定、ログプロンプトを表示、ユーザー名の取得をしてユーザーの為にログインプロセスを開始します。

ランレベル5では、/etc/inittabが/etc/X11/prefdmと呼ばれるスクリプトを実行します。prefdmスクリプトは、好みのX ディスプレイマネージャを実行します。—/etc/sysconfig/desktopファイルの内容に応じてgdm, kdm,xdmのいずれかとなります。

この時点で、システムはランレベル5で動作しており、ログイン画面を表示しています。

1.3. ブート時に追加プログラムを実行

/etc/rc.d/rc.localスクリプトは、ブート時、又はランレベルを変更する時にinitコマンドで実行されます。このスクリプトにコマンドを追加すると、簡単に特別なサービスを開始したりデバイスを始動するなどの必要なタスクを実行できる上、/etc/rc.d/init.d/ ディレクトリ内で複雑な初期化スクリプトを書いたりシンボリックリンクを作成したりする必要もありません。

/etc/rc.serialスクリプトは、シリアルポートがブート時にセットしてある必要が有る場合に使用されます。このスクリプトは、setserial コマンドを使用してシステムのシリアルポートを設定します。詳細はsetserial のman ページで御覧下さい。

1.4. SysV Init ランレベル

SysV init ランレベルシステムは、ランレベルの初期化をしている時にどのプログラムをinitが起動又は停止をするかを制御する為の標準プロセスを提供します。SysV initは使用が簡単であることと、伝統的なBSDスタイルのinit プロセスよりも柔軟性があることで選択されています。

SysV init用の設定ファイルは、/etc/rc.d/ディレクトリにあります。このディレクトリに中には、rc, rc.local, rc.sysinitが存在し、オプションとして、rc.serial スクリプトも以下に示すディレクトリと共に存在します：

```
init.d/
rc0.d/
rc1.d/
rc2.d/
rc3.d/
rc4.d/
rc5.d/
rc6.d/
```

init.d/ディレクトリには、サービスの制御時に/sbin/initコマンドによって使用されるスクリプトが含まれています。それぞれ番号付きのディレクトリは、Red Hat Linuxの元でデフォルトとして設定された6つのデフォルトランレベルを代表します。

1.4.1. ランレベル

ランレベルとは、状態又はモードというべきものでSysV/etc/rc.d/rc<*x*>.d/ ディレクトリ内にリストされているサービスにより定義されます。(<*x*>はランレベルの数字です)。

SysV initランレベルの背景にある考え方は、各種システムを異なる方法で使用できるという事実に基づいて展開しています。例えば、サーバはX Window Systemにより増加するシステムリソースの負担なしで、より効率的に動作します。別の例として、システム管理者は診断作業を実行するために低いランレベルでシステムを稼働する必要があるかもしれません。ランレベル1でディスク破損の修理をする場合など、誰も他のユーザーはシステム上で操作は出来ません。

5. *tty*デバイスに関する詳細情報は項5.3.11で御覧下さい

1つのランレベルの特性はinitによってどのサービスを終了し、どれを開始するかを決定します。例えば、ランレベル1(シングルユーザーモード)はネットワークサービスを停止し、ランレベル3ではそのサービスを開始します。ある任意のランレベルで特定のサービスを停止したり、開始したりするように割り当てることで、initはユーザーが手動でサービスを開始したり停止したりすることを必要とせずにマシンのモードを素早く変更することができます。

Red Hat Linuxのデフォルトのランレベルは、以下の様に定義されています：

- 0 — 休止
- 1 — シングルユーザーテキストモード
- 2 — 未使用(ユーザー定義可能)
- 3 — フルマルチユーザーテキストモード
- 4 — 未使用(ユーザー定義可能)
- 5 — フルマルチユーザーグラフィカルモード(X-ベースのログイン画面)
- 6 — 再起動

通常、ユーザーはRed Hat Linux をランレベル3、又はランレベル5で実行します。—両方ともフルマルチユーザーモードです。ユーザーは使用されていないランレベル2と4をカスタマイズして特殊なニーズに対応することもできます。

システムのデフォルトランレベルは、/etc/inittabの中にリストしてあります。システムのデフォルトランレベルを確認するには、次のような行を/etc/inittabの上部で見付けます：

```
id:5:initdefault:
```

上記の例で示されているデフォルトのランレベルは、最初のコロンが表示するように5となっています。それを変更するには、rootとして/etc/inittabを編集します。

警告

/etc/inittabを編集するときには、注意が必要です。簡単なタイプミスがシステムをブート不可能にもします。そうなった場合、ブートディスクを使用するか、シングルユーザーモード、又はレスキューモードでコンピュータを起動してファイルを修理します。

シングルユーザーモードとレスキューモードについての情報は*Red Hat Linux* カスタマイズガイドの中にあるレスキューモードの章をお読み下さい。

デフォルトランレベルの変更は、ブート時にカーネルへブートローダーによって渡される引数を変更することで達成できます。ブート時のランレベル変更に付いての情報は項2.10で御覧下さい。

1.4.2. ランレベルユーティリティ

ランレベルを設定する最良の方法の1つは、*initscript* ユーティリティを使用することです。これらのツールは、SysV initディレクトリの階級の中でファイルを保全するタスクを簡潔化するようにデザインされていますので、システム管理者は/etc/rc.d/のサブディレクトリから数多くのシンボリックリンクを直接操作する必要がなくなります。

Red Hat Linux はそのようなユーティリティを3種類提供します：

- /sbin/chkconfig — /sbin/chkconfig ユーティリティは、シンプルなコマンド行ツールで/etc/rc.d/init.dディレクトリ階級のメンテナンスに使用します。

- **/sbin/ntsysv** — ncurses-ベースの**/sbin/ntsysv**ユーティリティは対話式のテキストベースインターフェイスを提供します。一部のユーザーはchkconfigよりも簡単に使用出来るでしょう。
- **サービス設定ツール** — グラフィカルな**サービス設定ツール** (redhat-config-services) プログラムは、ランレベル設定用の柔軟性のあるGTK2ベースのユーティリティです。

これらのツールに関しての詳細情報には*Red Hat Linux* カスタマイズガイドの中のサービスに対するアクセスの制御の章を御覧下さい。

1.5. シャットダウン

Red Hat Linuxをシャットダウンするには、rootユーザーは/sbin/shutdown コマンドを発行することが出来ます。shutdownのman ページには数多くのオプションの総合一覧があります。しかし最も良く使用されるものは次の2種類です：

```
/sbin/shutdown -h now
/sbin/shutdown -r now
```

全てをシャットダウンした後、-hオプションはマシンを停止し、そして-rオプションは再起動します。

root 以外のユーザーは、ランレベル1 から5のいずれかにいる間、rebootとhaltコマンドを使用してシステムをシャットダウン出来ます。但し、全てのLinuxオペレーティングシステムがこの機能をサポートしているのではありません。

コンピュータが自身で電源を切れない場合、システムが停止したことを示すメッセージが出るまで、コンピュータ(電源)を切断しない様に注意して下さい。

このメッセージを待たずに切断すると、ハードドライブの一部のパーティションがまだアンマウントされておらず、ファイルシステム破損につながる可能性があります。

第2章

ブートローダー

Red Hat Linuxが稼働できるようになる前に、ブートローダーと呼ばれる特殊なプログラムでそれをメモリにロードする必要があります。ブートローダープログラムは通常、システムのプライマリハードドライブ上又は、他のメディアデバイスに存在し、その唯一の任務はLinuxカーネルとその必要とするファイルをロードするか、又は(他のケースでは)他のオペレーティングシステムをメモリにロードすることです。

2.1. ブートローダーとシステムアーキテクチャ

Red Hat Linuxを稼働できる各アーキテクチャはそれぞれ異なったブートローダーを使用します。例えば、Alphaアーキテクチャは`aboot`ブートローダーを使用し、Itaniumアーキテクチャは`ELILO`ブートローダーを使用します。

この章では、x86アーキテクチャ用にRed Hat Linux で供給されている2つのブートローダー(GRUBとLILO)に関するコマンドと設定オプションを説明します。

2.2. GRUB

GNU GRand Unified Boot loader(GRUB)は、ユーザーにシステム起動の時点にロードするインストール済のオペレーティングシステム又はカーネルを選択させてくれるプログラムです。またカーネルに対して引数を渡すこともできるようにします。

2.2.1. GRUB とx86ブートプロセス

このセクションでは、x86システムをブートしている時にGRUBが果たす特別な役割について詳しく説明します。全体的なブートプロセスを知るには項1.2を参照して下さい。

GRUB は次のようなステージでそれ自身をメモリにロードします:

1. ステージ*1*と呼ばれるプライマリブートローダーが*BIOS*によって*MBR*からメモリへ読み込まれます。[1]. プライマリブートローダーはMBR内の512バイト以下のディスク領域に存在し、ステージ1.5又はステージ2 のブートローダーをロードする機能を持ちます。
2. ステージ*1.5*ブートローダーは必要であれば、ステージ*1*ブートローダーによってメモリに読み込まれます。ハードウェアの幾つかでは、ステージ2 ブートローダーに達するために中間ステップを必要とするものがあります。これは、`/boot`パーティションがハードドライブの1024シリンダーヘッド以上にある場合やLBAモードを使用している時に起こります。ステージ1.5ブートローダーは、`/boot`パーティション上か又は、MBRの小さな部分と`/boot`パーティションにあります。
3. ステージ*2*と呼ばれるセカンダリブートローダーがメモリに読み込まれます。 セカンダリブートローダーはGRUBメニューとコマンド環境を表示します。このインターフェイスを通してブートするオペレーティングシステム又は、Linuxカーネルを選択でき、カーネルに引数を渡し、又は使用可能なRAMなどのシステムパラメーターを確認したりすることが出来ます。
4. セカンダリブートローダーはオペレーティングシステム又はカーネルと*`initrd`*をメモリに読み込みます。 GRUBが起動するオペレーティングシステムを決定するとそれをメモリにロードしその後はマシンの制御をオペレーティングシステムに渡します。

1. システムBIOSとMBRに付いての詳しい情報は項1.2.1で御覧下さい。

Red Hat Linuxの起動に使用される起動方法は、ブートローダーがオペレーティングシステムを直接ロードすることからダイレクトロードと呼ばれます。ブートローダーとカーネルの間には仲介はありません。

他のオペレーティングシステムによって使用されるブートプロセスは違うことがあります。例えば、MicrosoftのDOS とWindowsオペレーティングシステムや別の商用オペレーティングシステムはチェーンロード起動法を使用します。この方法では、MBRはオペレーティングシステムを保持しているパーティションの第一セクターを指定するだけです。そこでオペレーティングシステムを実際にブートするのに必要なファイルを見付けます。

GRUBはダイレクトロードとチェーンロードの両方の方法をサポートし、ほとんどどんなオペレーティングシステムでもブートできます。

警告

インストールの間、MicrosoftのDOS とWindowsインストールプログラムは完全にMBRを上書きし、既存のブートローダーを抹消します。デュアルブートシステムを構築している場合は、Microsoftのオペレーティングシステムを最初にインストールするのが適切です。その操作法に関しては*Red Hat Linux* インストールガイド内のデュアルブート環境での*Red Hat Linux*のインストールと題してある付録を参照して下さい。

2.2.2. GRUBの機能

GRUBにはx86アーキテクチャで利用可能で、他のブートローダーと比較してもより好ましい機能を数多く含んでいます。以下に重要な機能の1部を一覧で示します：

- *GRUB*は*x86*マシン上で、真のコマンドベースのプレ-*OS*環境を提供します。このためユーザーは特定のオプションを持ち、システムの情報収集ができるこのオペレーティングシステムのロードに最大の柔軟性を得ることになります。長年、多くの非-x86アーキテクチャはコマンドラインからブートできるプレ-OS環境を起用してきました。幾つかのコマンド機能はLILOや他のx86ブートローダーにも利用できますがGRUBはより豊富な機能を搭載しています。
- *GRUB*は、*Logical Block Addressing (LBA)*モードをサポートします。LBAはハードドライブのファームウェア内のファイルを探す為に使用されるアドレス変換の位置設定をするもので、多くのIDEと全てのSCSIハードドライブで使用されます。LBAの前まではブートローダーは、BIOSがディスク内のそのシリンダーヘッド以降はファイルを探せないと言うBIOSの1024シリンダー限界の問題に遭遇する可能性がありました。LBAサポートはシステムBIOSがLBAモードに対応している限り、GRUBに対して1024シリンダー限界を越えたパーティションからオペレーティングシステムをブートすることができるようにします。ほとんどの現代のBIOS改定はLBAモードに対応しています。
- *GRUB* は*ext2*のパーティションも読み込みます。この機能により、GRUBはその設定ファイル`/boot/grub/grub.conf`にアクセスでき、設定変更がされた時でもシステムがブートする度にMBRにステージ1 ブートローダーの新しいバージョンを書く必要がなくなります。ユーザーがMBR上のGRUBを再インストールする必要が出る唯一の状況は、ディスク上の`/boot` パーティションの物理的な場所が移動された時です。MBRへのGRUBインストールの詳細については項2.3を参照して下さい。

2.3. GRUBのインストール

Red Hat Linuxのインストールプロセスで GRUBがインストールされなかった場合でも、その後にGRUBをインストール出来ます。インストールされると自動的にデフォルトのブートローダーになります。

GRUBをインストールする前に、最新のGRUBパッケージが利用できる事、又はRed Hat Linuxインストール CD-ROMからGRUBパッケージを使用できることを確認してください。パッケージのインストールに関する方法は、*Red Hat Linux* カスタマイズガイドの中の*RPM*によるパッケージ管理の章をお読み下さい。

GRUBパッケージがインストールされると、シェルプロンプトでrootとなり`/sbin/grub-install <location>` コマンドを実行します。ここで、<*location*>とはGRUBステージ1 ブートローダーがインストールされる場所のことです。

次のコマンドを使用すると、GRUBはプライマリーIDEバス上のマスターIDE: `/sbin/grub-install /dev/hda`のMBRにインストールされます。

次にシステムがブートする時には、カーネルがロードする前にGRUBのグラフィカルブートローダーメニューが現れます。

2.4. GRUB 用語

GRUBを使用する前に理解すべき重要な事の1つにハードディスクやパーティションなど、プログラムで使用するデバイスの参照用語です。これらの用語は複数のオペレーティングシステムからブートするようにGRUBを設定する時に特に、重要な情報になります。

2.4.1. デバイスの名前

ハードディスクが複数あるシステムを想定しましょう。システムの1番目のハードドライブは、GRUBによって`(hd0)`と区分されます。そのドライブの1番目のパーティションは、`(hd0,0)`となり、2番目のハードドライブの5番目のパーティションは`(hd1,4)`となります。一般的に、GRUBを使用する時の命名慣習は、以下のような仕組みになります：

(<*type-of-devicegt*><*bios-device-number*>,<*partition-number*>)

名前の中のかっことカンマはデバイス名を認識する上で、重要な意味を持っています。<*type-of-device*>はハードディスク（`hd`）か、フロッピーディスク（`fd`）のどちらが指定されているかを示します。

<*bios-device-number*>は、システムのBIOSにより指定されるデバイスの数を示し、`0`から始まります。プライマリIDEハードディスクドライブの番号は`0`、セカンダリIDEハードディスクドライブの番号は`1`となります。この順序づけは、Linuxカーネルが文字によりデバイスを配列する方法とほぼ一致します。たとえば、`hda`の`a`は`0`、`hdb`の`b`は`1`に当たります。

注意

GRUBのデバイスに対するナンバリングシステムが`1`ではなく`0`から開始することに注意してください。これは初めてGRUBを使用するユーザーが最も間違えやすい点です。

<*partition-number*>はディスクデバイスにある特定のパーティション数を表します。<*bios-device-number*>と同じように、パーティションのナンバリングも`0`から開始します。ほとんどのパーティションは番号で指定されますが、システムがBSDパーティションを使用している場合は、`a`や`c`などの文字が与えられます。

GRUBでは次の規則に従って、デバイスとパーティションを命名します：

- ハードディスクドライブがIDEかSCSIかに関係なく、すべてのハードディスクドライブは`hd`で始まります。フロッピーディスクは`fd`で始まります。

- パーティションに関わらずデバイス全体を指定する場合は、カンマとパーティション番号を除きます。これは、GRUBに特定のディスクの為のMBRを設定させる場合に重要です。たとえば、(hd0)は1番目のデバイス上のMBRを指し、(hd3)は4番目のデバイス上のMBRを指します。
- 複数のハードディスクドライブが存在する場合は、BIOSで設定された順序を知ることが非常に重要です。IDEドライブかSCSIドライブのどちらかしかない場合は簡単ですが、その両方が混在していると混乱しやすくなります。

2.4.2. ファイル名とブロック一覧

複数のOSのブートを可能にするためのメニュー一覧などのファイルに関するコマンドをGRUBに入力する場合、デバイスとパーティションを指定したすぐ後にファイル名を入力する必要があります。

ファイルの絶対名に対するファイル指定は例として次のように構成されます：

```
(<type-of-device><bios-device-number>,<partition-number>)/path/to/file
```

ほとんどの場合、パーティションでのディレクトリパスとファイル名を入力してファイルを指定します。

実際はファイルシステムに表示されないファイルをGRUBに指定することもできます。たとえばパーティションの先頭ブロックのいくつかに表示されるチェーンローダーなどです。これらのファイルを指定するには、パーティションのどこにファイルがあるかをブロックごとにGRUBに教える*blocklist*を与える必要があります。1つのファイルがいくつもの異なるブロックのセットで構成される場合もあるため、決まったブロック一覧の書き方があります。それぞれのファイルのセクションの場所をブロックのオフセット番号で示した後に、そのオフセットポイントからのブロック数、そしてセクションを順番にカンマで区切って表示します。

次にブロック一覧の例を示します：

```
0+50,100+25,200+1
```

このブロック一覧は、パーティションの先頭のブロックから始まってブロック0から49まで、ブロック99から124まで、さらにブロック199を使用するファイルを使用するようにGRUBに指示します。

Microsoft Windowsなどのチェーンロードを使用するOSをGRUBでロードする場合は、ブロック一覧の書き方を知っていると役立ちます。ブロック0から開始する場合はブロックのオフセット番号を省略してもかまいません。例をあげると、1番目のハードディスクドライブの先頭パーティションにあるチェーンロードファイルは次の名前になります：

```
(hd0,0)+1
```

以下に、同じようなブロック一覧を指定する場合に、正しいデバイスとパーティションをルートで設定した後、GRUBのコマンド行で次のようにchainloaderコマンドを示します：

```
chainloader +1
```

2.4.3. GRUBのルートファイルシステム

GRUBで使用する「ルートファイルシステム」という用語に混乱するユーザーもいます。GRUBのルートファイルシステムはLinuxのルートファイルシステムとは無関係であると覚えてくことが重要です。

GRUBのルートファイルシステムは特定のデバイスに関するルートパーティションを意味します。GRUBはこの情報を使用してデバイスをマウントし、そのデバイスからファイルをロードします。

Red Hat Linuxでは、GRUBがルートパーティション(/bootパーティションに相当し、Linuxカーネルを格納する)をロードすると、カーネルファイルの格納場所をオプションとしてkernelコマンドが実行可能になります。Linuxカーネルがブートすると、Linuxユーザーに馴染みのルートファイルシステムを設定します。元のGRUBのルートファイルシステムとそのマウントは無視され、カーネルファイルをブートするためだけに存在します。

詳細については、項2.6のrootコマンドとkernelコマンドを参照してください。

2.5. GRUB インターフェイス

GRUBは異なるレベルの機能を持つ3種類のインターフェイスを備えています。それぞれのインターフェイスによって、linuxカーネル又はオペレーティングシステムのブートが可能になります。

各インターフェイスは以下のようになります：

メニューインターフェイス

‘ GRUBがRed Hat Linuxのインストールプログラムによって自動的に設定される場合、これはデフォルトで示されるインターフェイスとなります。それぞれのブートコマンドで事前に設定されたOS、又はカーネルの一覧を名前の順で示したメニューがこのインターフェイスに表示されます。デフォルト選択以外のオプションをブートするには、矢印キーでそのオプションを選択して[Enter]キーを押します。別の方法として、タイムアウト期間が設定されている為、何もしなければ、GRUBがデフォルトオプションのロードを開始します。

[e]キーを押すと、エントリ編集のインターフェイスになり、[c]キーを押すとコマンドラインインターフェイスをロードします。

このインターフェイスの設定については項2.7を御覧ください。

メニューエントリエディタインターフェイス

‘ メニューエントリーエディタにアクセスするには、ブートローダーメニューで[e]キーを押します。GRUBのこのエントリに関するコマンドが表示されますので、ユーザーはこれらのコマンド行に追加（現在の行の後であれば[o]キー、その後の行であれば[O]キーを使用）、編集（[e]キーを使用）、消去（[d]キーを使用）することにより、OSをブートする前にコマンド行を変更できます。

全ての変更が終ると、[b]キーを押してコマンドを実行し、OSをブートします。[Esc]キーを押せば、変更をキャンセルして標準メニューインターフェイスを再ロードします。[c]キーを押せば、コマンド行インターフェイスをロードします。


ヒント

メニューエディタを使用したGRUBでのランレベルの変更に関する情報は項2.10を参照して下さい。

コマンド行インターフェイス

‘ コマンドラインは最も基本的なGRUBインターフェイスですが、殆んどの制御をできるインターフェイスでもあります。コマンドラインでは、関連するあらゆるGRUBコマンドを入力して[Enter]キーを押して、実行できます。このインターフェイスはシェルに似た高度な機能を備えています。コンテキストに基づく[Tab]キーの補完、コマンド入力時に使用する[Ctrl]キーのコンビネーションなどがあります。たとえば[Ctrl]-[a]で行頭に移動し、[Ctrl]-[e]で行末に移動します。さらに、矢印キー、[Home]キー、[End]キー、[Delete]キーはbashシェルの場合と同じ働きをします。

一般的なコマンドの一覧については、項2.6を御覧下さい。

2.5.1. インターフェイスの使用順序

GRUB環境が第2ステージのブートローダーをロードすると、GRUBはまず設定ファイルを探します。見つかると、それを使用してOSのメニュー一覧を構築して、ブートメニューインターフェイスを表示します。

設定ファイルが見つからない場合や読み込めない場合は、GRUBはコマンド行インターフェイスをロードし、OSのブートの完了に必要なコマンドを入力できるようにします。

設定ファイルが無効の場合は、エラーが出力されて入力が促されます。これによりユーザーは問題の発生個所を正確に知ることができます。どれかキーを押すとメニューインターフェイスに戻り、GRUBによって報告されたエラーに基づいてメニューオプションを編集して問題を修復できます。修復が失敗すると、GRUBはエラーを報告して、メニューインターフェイスに戻ります。

2.6. GRUB コマンド

GRUBには、コマンド行インターフェイスで、便利な多くのコマンドが含まれています。コマンドの中にはコマンド名の後にオプションを受け入れるものもあり、これらのオプションはスペースによって同じ行のコマンドや他のオプションと区別します。

役立つコマンドの一覧を次に示します：

- `boot` —事前に指定されたロード済みのOSやチェーンローダーをブートします。
- `chainloader <file-name>` —指定したファイルをチェーンローダーとしてロードします。指定したパーティションの先頭セクターにあるファイルを取り込むには、ファイル名に`+1`を使用します。
- `displaymem` — BIOSの情報に基づいて現在使用中のメモリを表示します。システムをブートする前にシステムが持つRAM容量がどれだけか見る場合に役立ちます。
- `initrd <file-name>` — ブート時に使用する最初のRAMディスクを指定できるようにします。`initrd`は、ルートパーティションがext3ファイルシステムでフォーマットされる時など、適切にブートするためにカーネルが特定のモジュールを要求する場合に必要です。
- `install <stage-1> <install-disk> <stage-2> p <config-file>` — GRUBをシステムのMBRにインストールします。

  `install`コマンドを使用する時は以下を指定する必要があります：

  - `<stage-1>`— これは、`(hd0,0)/grub/stage1`のように、1番目のブートローダーイメージが存在するデバイス、パーティション、ファイルを表します。
  - `<install-disk>` — `(hd0)`などのステージ1のブートローダーがインストールされるべきディスクを表します。
  - `<stage-2>`— `(hd0,0)/grub/stage2`などのステージ2のブートローダーの場所をステージ1のブートローダーに伝えます。
  - `p<config-file>`— このオプションは、メニュー設定ファイルが`<config-file>`で指定されているメニュー設定ファイルを探すように`install`コマンドに伝えます。設定ファイルへの有効なパスの例として`(hd0,0)/grub/grub.conf`があります。

警告

`install`コマンドはMBRの内容をすべて上書きします。コマンドを実行すると、他のオペレーティングシステムをブートする為に使用されていた(GRUB情報以外の)すべての情報が失われます。

- `kernel <kernel-file-name> <option-1> <option-N>` — OSのブートにダイレクトロード方法を使用する時に、GRUBのルートファイルシステムからロードするカーネルファイルを

指定します。kernelコマンドの後にオプションが続き、それはロードした時にカーネルに渡されます。

Red Hat Linuxの場合、kernelコマンドの例は次のようになります：

```
kernel /vmlinuz root=/dev/hda5
```

この行はvmlinuzファイルを、(hd0,0)のようなGRUBのルートファイルシステムからロードするように指定します。オプションもカーネルに受け渡されて、ロードする時、Linuxカーネルのルートファイルシステムが、最初のIDEハードディスクドライブの5番目のパーティションであるhda5に存在することを指定します。必要に応じて、このオプションの後に複数のオプションを追加することもできます。

- root<*device-and-partition*>— (hd0,0)のように、GRUBのルートパーティションを特定のデバイスとパーティションに設定し、そのパーティションをマウントしてファイルを読み込めるようにします。
- rootnoverify<*device-and-partition*>— rootコマンドと同じ働きをしますが、パーティションをマウントしません。

これら以外のコマンドも利用できます。総合的なコマンド一覧についてはinfo grubと入力して下さい。

2.7. GRUBメニュー設定ファイル

設定ファイル(/boot/grub/grub.conf)は、ブートするOSの一覧を作成するときにGRUBのメニューインターフェイスで使用されますが、基本的に、これによりユーザーは事前設定の実行できるコマンドグループを選択できます。項2.6に示されるコマンドが使用できるほか、設定ファイル内でのみ使用できる特殊コマンドもあります。

2.7.1. 設定ファイルの特殊コマンド

次のコマンドはGRUBのメニュー設定ファイルでのみ使用できます：

- color <*normal-color*><*selected-color*> —メニューで使う特定の色の設定を行えます。フォアグランドとバックグランド用に2色を設定します。red/blackの様に、簡単な色の名前を使います。例えば、以下のようになります：
  ```
  color red/black green/blue
  ```
- default<*title-name*> — メニューインターフェイスがタイムアウトになるとロードされるデフォルトエントリのタイトル名です。
- fallback<*title-name*> — 使用すると、最初の試行が失敗する場合に再試行されるエントリタイトル名を示します。
- hiddenmenu — 使用すると、GRUBメニューインターフェイスを非表示にし、timeout期間の経過後にdefaultエントリをロードします。[Esc]キーを押すと、標準のGRUBメニューを表示できます。
- password<*password*> — 使用すると、パスワードを知らないユーザーによりこのメニューオプションのエントリが編集されるのを防ぎます。

  オプションとして、password<*password*>の後に別のメニュー設定ファイルを指定することができます。この場合に、GRUBが第2ステージのブートローダーを再スタートし、指定した別の設定ファイルを使用してメニューを構成します。この代替ファイルがコマンドから外されている場合、パスワードを知っているユーザーが現在の設定ファイルを編集できます。
- timeout —使用すると、defaultコマンドで指定されたエントリをGRUBがロードするまでの時間間隔を秒単位で設定します。
- splashimage — GRUBのブート時に使用するスプラッシュスクリーン画像の場所を指定します。

- `title` —OSのロードに使用する特定のコマンドグループとともに使用するタイトルを設定します。

(#)記号は、メニュー設定ファイルにコメントを挿入するために、その行の先頭で使用します。

2.7.2. 設定ファイル構成

GRUBのメニューインターフェイスの設定ファイルは`/boot/grub/grub.conf`です。メニューインターフェイス用のグローバル設定を行うコマンドはファイルの一番上に配置されています。その後に、メニューにあるそれぞれのオペレーティングシステム、又はカーネルのエントリが続きます。

Red Hat Linux又はMicrosoft Windows 2000をブートするように設計された最も基本的なGRUBのメニュー設定ファイルは次のように表示されます：

```
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz

# section to load linux
title Red Hat Linux (2.4.18-5.47)
root (hd0,0)
kernel /vmlinuz-2.4.18-5.47 ro root=/dev/sda2
initrd /initrd-2.4.18-5.47.img

# section to load Windows 2000
title windows
rootnoverify (hd0,0)
chainloader +1
```

このファイルは、デフォルトのOSとしてRed Hat Linuxと共にメニューをコンパイルし、10秒後に自動的にブートするようにセットすることをGRUBに指示します。2つのセクションが、各OSエントリに対して1つずつ、このシステムのディスクパーティションテーブル固有のコマンドとともに与えられます。

注意

デフォルトは番号として指定してあることに注意して下さい。これはGRUBが遭遇する最初の`title`の行を意味します。`windows`をデフォルトにしたい場合、`default=0`の値を`default=1`に変更します。

複数のオペレーティングシステムをブートするGRUBのメニュー設定ファイルの設定は、この章の説明範囲を越えるものです。その他のリソースの一覧は項2.11で確認して下さい。

2.8. LILO

LILO は、*LI*nux *LO*aderの略語で、x86 Linuxシステムをブートする為に何年も使用されて来ました。現在ではGRUBがデフォルトのブートローダーですが、幾らかのユーザーはLILOに慣れているからそれを使用したり、また他の人はGRUBがハードウェアによってはブートの問題を持つ可能性がある為、必要上LILOを使用することもあります。

2.8.1. LILO とx86 ブートプロセス

このセクションは、x86システムがブートする時にLILOが果たす特定の役割について詳しく説明します。ブートプロセスの全体的な詳細情報は項1.2で参照して下さい。

LILOは、単なる2ステージローダーと言うことを除いては、殆んどGRUBと同じようにメモリにそれ自身をロードします。

1. ステージ*1*、すなわちプライマリブートローダーは、*BIOS*によって*MBR*からメモリへ読み込まれます。[2]. プライマリブートローダーは、MBR内の512バイト以下のディスク領域に存在します。これが果たす役割は単にステージ2ブートローダーをロードしてそれにディスクジオメトリ情報を渡すことです。
2. ステージ*2*、すなわちセカンダリブートローダーはメモリに読み込まれます。セカンダリブートローダーがRed Hat Linux初期画面を表示します。この画面により、ブートするオペレーティングシステム又はLinuxカーネルを選択することが出来ます。
3. ステージ*2*ブートローダーはオペレーティングシステム又はカーネルと*`initrd`*をメモリに読み込みます。LILOが開始するオペレーティングシステムを決定すると、それをメモリにロードしてマシンの制御をそのオペレーティングシステムに任せます。

ステージ2ブートローダーがメモリに入ると、LILOは、ブートするように設定されている他のオペレーティングシステムやカーネルを初期のRed Hat Linux画面に表示します。デフォルトでは、Red Hat Linuxだけしかインストールしていない場合、利用できるオプションは**linux**のみとなります。システムが複数のプロセッサを持つ場合、単独プロセッサカーネル用に**linux-up**があり、複数プロセッサ(SMP)カーネル用には**linux**があります。他のオペレーティングシステムもブートするようにLILOを設定している場合は、それらのブートエントリも表示されます。

矢印キーを使用すると、目的のオペレーティングシステムを強調表示して、[Enter]キーを押すことでブートプロセスを始めることができます。

`boot:`プロンプトにアクセスするには、[Ctrl]-[X]キー組合せを押します。

2.8.2. LILO 対GRUB

一般的に、3つの大きな相違点を除いてLILO はGRUBと同じように動作します：

- 対話式のコマンドインターフェイスがない。
- MBRにロードするカーネル、又は他のオペレーティングシステムのの場所の情報を保存する。
- ext2パーティションは読み込めない。

この1番目の点はLILO用のコマンドプロンプトが対話式でなく、引数付きのコマンドを1つだけ許可するのみと言う説明です。

最後の2つの点は、LILOの設定ファイルを変更、又は、新しいカーネルをインストールした場合、以下のコマンドを使用することにより、MBRへステージ1のLILOブートローダーを書き換える必要があるとの意味です：

```
/sbin/lilo -v -v
```

これは、間違えた設定のMBRはシステムのブート障害の原因になりますのでGRUBを使用する方法よりもリスクがあります。GRUBでは、設定ファイルが正しくない場合でも、ユーザーが手動でシステムをブート出来るデフォルトのコマンド行インターフェイスに入るだけです。

2. BIOS とMBRに関しては項1.2.1を御覧下さい。

ヒント

Red Hat 更新エージェントを使用してカーネルをアップグレードする場合、MBRも自動的に更新されます。RHNに関する詳細情報は、オンラインのURL：https://rhn.redhat.comで御覧下さい。

2.9. /etc/lilo.confのオプション

LILO 設定ファイルは/etc/lilo.confです。/sbin/liloコマンドはこのファイルを使用してMBRに書き込む情報を決定します。

警告

/etc/lilo.confを編集する前に、そのファイルのバックアップコピーを忘れずに作成して下さい。また、問題がある場合に、MBRへ変更が出来るように正常に機能するブートディスクを用意しておいて下さい。ブートディスクの作成に関する詳細はmkbootdiskのmanページを御覧下さい。

/etc/lilo.confファイルは、ロードするオペレーティングシステム、又はカーネルを決定する目的と、それ自身のインストール先を知る目的で/sbin/liloによって使用されます。

サンプルの/etc/lilo.conf ファイルは以下のような状態です：

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
message=/boot/message
lba32
default=linux

image=/boot/vmlinuz-2.4.0-0.43.6
label=linux
initrd=/boot/initrd-2.4.0-0.43.6.img
read-only
root=/dev/hda5

other=/dev/hda1
label=dos
```

この例は、Red Hat Linux とDOSの2つのオペレーティングシステムをブートするように設定されたシステムを表現しています。このファイルの内容をもう少し詳しく説明しましょう：

- boot=/dev/hda— LILOに1番目のIDEコントローラの1番目のハードディスクにそれ自身をインストールするように指示します。
- map=/boot/map— マップファイルの位置を決定します。通常の使用では、これは変更しないで下さい。
- install=/boot/boot.b— LILOに対して指定されたファイルを新規のブートセクタとしてインストールするように指示します。通常、これは変更すべきではありません。もしinstallの行が欠けている場合、LILOはデフォルトの/boot/boot.bを使用するファイルと判定します。

- `prompt` — LILOに対して`message`の行に案内されているものを表示するという指示を出します。`prompt`を取り除くことは推奨できませんが、除去した場合でもまだ、[Shift]キーを押しながらマシンをブートすることでpromptにアクセスすることができます。
- `timeout=50` — LILOが`default`行のエントリでブートを進める前までにユーザーが入力するのを待つ時間の長さをセットします。この長さは10分の1秒単位で計測され、50をデフォルトにしています。
- `message=/boot/message` — ユーザーが、ブートすべきオペレーティングシステムやカーネルを選択できるようにLILOが表示する画面を示します。
- `lba32` — ハードディスクのジオメトリをLILOに記述します。ここでのもう1つの一般的なエントリは`linear`です。これらの作業の内容を充分に理解していない限り、これらを変更すべきではありません。変更してしまうとシステムをブートできない状態にしてしまう可能性があります。
- `default=linux` — この行以下にリストされているオプションからブートするLILO用のデフォルトオペレーティングシステムを示します。`linux`の名前は、各ブートオプションの下にある`label`行を意味します。
- `image=/boot/vmlinuz-2.4.0-0.43.6` — この特定のブートオプションでブートするlinuxカーネルを指定します。
- `label=linux` — LILO画面のオペレーティングシステムの名前です。この場合、`default`の行でも案内されている名前です。
- `initrd=/boot/initrd-2.4.0-0.43.6.img` — カーネルのブートを可能にするデバイスをブート時に実際に初期化し、スタートするのに使用される初期*ram*ディスクイメージのことです。この初期ramディスクはSCSIカード、ハードディスク、あるいは他の、カーネルをロードするのに必要なデバイスを操作するのに要する一連のマシン固有のドライバです。この初期ramディスクは絶対にマシン同士で共有したりしないで下さい。
- `read-only` — このルートパーティション(以下の`root`の行を参照) 読み込み専用で、ブートプロセス中に変更は出来ないことを指定します。
- `root=/dev/hda5` — どのディスクパーティションをルートパーティションとして使用するかを指定します。
- `other=/dev/hda1` — DOSを格納しているパーティションを指定します。

2.10. ブート時のランレベルの変更

Red Hat Linuxの元では、ブート時にデフォルトのランレベルを変更することが出来ます。

LILOを使用している場合、`boot:`プロンプトにアクセスするには[Ctrl]-[X]キー組合せを押します。その後、以下のように入力します:

`linux <runlevel-number>`

このコマンドでは、*<runlevel-number>*をブートしたいランレベルの数字(**1** から**5**)か、又は、単語**`single`**又は**`emergency`**で入れ換えます。

GRUBを使用している場合は、以下のステップに従います:

- グラフィカルGRUBブートローダーの画面では、**Red Hat Linux**ブートラベルを選択して、[e]キーを押して編集をします。
- カーネルの行まで矢印で移動して、[e]キーを押して編集に入ります。
- プロンプトで、ブートしたいランレベルの数字(**1** から**5**)を入力するか、又は単語**`single`**又は**`emergency`**を入力して[Enter]キーを押します。
- カーネル情報のあるGRUB画面に戻されます。[b]キーを押してシステムをブートします。

ランレベルの詳細については項1.4.1を御覧下さい。

2.11. その他のリソース

この章は単にGRUBとLILOの紹介を目的としています。GRUBとLILOの機能については以下のリソースを参考にしてより多くの情報を習得して下さい。

2.11.1. インストールされているマニュアル

- `/usr/share/doc/grub-<version-number>/` — これはGRUBの使用法とその設定に関する良い情報を含むディレクトリです。このファイルのパス内の`<version-number>`はインストール済みのGRUBパッケージのバージョンを示します。
- `info grub`コマンドを入力してアクセスできるGRUB infoページには、チュートリアル、ユーザー参照マニュアル、プログラマ参照マニュアル、GRUBとその使用法に関するFAQ（よくある質問）が含まれています。
- `/usr/share/doc/lilo-<version-number>/` — このディレクトリにはLILOの使用法とその設定に関する豊富な情報が含まれています。特に、`doc/`サブディレクトリには、有力な情報源である`User_Guide.ps`と呼ばれるポストスクリプトのファイルが含まれています。このディレクトリへのパス内にある`<version-number>`はインストールされているLILOパッケージのバージョンを示します。

2.11.2. 役立つWebサイト

- http://www.gnu.org/software/grub — GNU GRUBプロジェクトのホームページです。このサイトにはGRUBの開発状況に関する情報とFAQが掲載されています。
- http://www.uruk.org/orig-grub—さらなる開発の為にプロジェクトがFree Software Foundationに移行される前のオリジナルのGRUBマニュアルです。
- http://www.redhat.com/mirrors/LDP/HOWTO/mini/Multiboot-with-GRUB.html— Linux以外のOSのブートを含む、GRUBのさまざまな用途を追究します。
- http://www.linuxgazette.com/issue64/kohli.html—システムに初めてGRUBを設定するユーザーのための入門編です。GRUBのコマンド行オプションの概要が含まれます。
- http://www.tldp.org/HOWTO/mini/LILO.html — これは、Linux以外のオペレーティングシステムのブートを含むLILOに関するさまざまな使用法を説明するミニHOWTO情報です。

第3章

ファイルシステム構造

3.1. なぜファイルシステム構造を共有するのか

オペレーティングシステム(OS)のファイルシステム構造は、OSの構造の中でも最も基本的なものです。OSがユーザー、アプリケーション、セキュリティモデルなどと相互作用する方法は、そのほとんどが記憶装置にファイルが保存される方法によって決まります。ユーザーが、さらにはプログラムがファイルの読み書きをする場所を知るための共通のガイドラインに参照できることがさまざまな理由により重要になります。

ファイルシステムは、以下の2つの異なる論理的なカテゴリから見ることができます。

- 共有可能ファイルと共有不可ファイル
- 可変ファイルと静的ファイル

共有可能ファイルは、さまざまなホストによるアクセスが可能なファイルです。共有不可ファイルは、ほかのホストが利用できないファイルです。可変ファイルは、介在なしに何時でも変化できるファイルです。静的ファイルは、読み込み専用ドキュメントやバイナリなどのシステム管理者からの又は、システム管理者がタスクを達成する為に任務を与えているエージェントからの行動なしでは変化しないファイルです。

この様にファイルを見ている理由は、ファイルが置いてあるディレクトリに割り当てられた権限とファイルの機能を関連付けするためです。オペレーティングシステムとそのユーザーが該当ファイルに対してなす相互作用の仕方は、そのファイルのあるディレクトリが読み込み専用か、読み/書きかどうかと、各ユーザーが持つそのファイルへのアクセスレベルを決定します。この構造のトップレベルが重要になります。トップレベルが組織化されないままであるとか、広範に使用される構造を持たない場合などにはディレクトリへのアクセスを制限する必要があり、さもなければ、セキュリティ問題が露呈することになります。

ただし、その構造が標準のものでなければ、単に構造を持つだけでは意味がありません。構造の競合により、構造によって解決される問題より余計に問題が実際に生じる場合があります。このため、Red Hatでは最も広範囲に使用されているファイルシステム構成を選択し、それを少しだけ拡張してRed Hat Linuxで使用される固有のファイルに適用しました。

3.2. Filesystem Hierarchy Standard（FHS）の概要

Red Hatは、多数のファイルやディレクトリの名前と場所を定義した共同制作ドキュメントである*FHS*(*Filesystem Hierarchy Standard*)に従います。

FHSドキュメントは、任意のFHS準拠ファイルシステムに対する権威のあるリファレンスであるものの、この標準には未定義または拡張可能の領域が多く残されています。このセクションでは、この規格の概要を示し、規格で取り扱われていないファイルシステムの部分について説明します。

規格の全文は、以下の場所で参照できます。

http://www.pathname.com/fhs

規格に準拠することが多くのことを意味する中で、最も重要な2つのことは、他の準拠システムとの互換性であり、`/usr/`パーティションを読み込み専用としてマウントできることです（なぜならば、このパーティションは共通の実行可能ファイルを含んでおり、ユーザーによって変更されることを意図していないからです）。`/usr/`を読み込み専用としてマウントしてある、CD-ROMから、または読み込み専用のFNS経由で別のマシンから`/usr/`をマウントすることができます。

3.2.1. FHS標準

ここで示すディレクトリとファイルは、FHSドキュメントで指定されるもののほんの一部です。完全な情報については最新のFHSドキュメントを参照してください。

3.2.1.1. /dev/ディレクトリ

/dev/ディレクトリには、システムに接続されたデバイスを表すファイルシステムエントリが含まれています。システムが正しく機能するためには、これらのファイルが不可欠です。

3.2.1.2. /etc/ディレクトリ

/etc/ディレクトリは、マシンにとってローカルな設定ファイルのために予約されています。/etc/にはバイナリファイルを配置しないことになっています。以前は/etc/に配置されていたすべてのバイナリファイルは、/sbin/または/bin/に移動する必要があります。

X11/ディレクトリとskel/ディレクトリは、/etc/ディレクトリのサブディレクトリです：

```
/etc
 |- X11/
 |- skel/
```

/etc/X11/ディレクトリは、XF86Configなどの X11設定ファイルのためのものです。/etc/skel/ディレクトリは「スケルトン」ユーザーファイルのためのもので、最初にユーザーを作成するときはこれらのファイルを使用してホームディレクトリを埋め込みます。

3.2.1.3. /lib/ディレクトリ

/lib/ディレクトリには、/bin/や/sbin/に含まれるバイナリファイルを実行するために必要なライブラリのみを保存する必要があります。これらの共有ライブラリイメージは、特にシステムをブートしたりルートファイルシステム内でコマンドを実行する場合に重要です。

3.2.1.4. /mnt/ディレクトリ

/mnt/ディレクトリは、CD-ROMやフロッピーディスクなどの一時的にマウントされるファイルシステムです。

3.2.1.5. /opt/ディレクトリ

/opt/ディレクトリは、通常大きな静的アプリケーションソフトウェアパッケージが保存される領域を提供します。

/opt/ディレクトリ内にファイルを配置するパッケージは、同じ名前のディレクトリを作成します。このディレクトリが今度は、ファイルシステム全体に広がる可能性を持つファイルを収納します。これにより、システム管理者は特定パッケージ内の各ファイルの役割を簡単に決定することができます。

例えば、/opt/内にsampleという名前のソフトウェアパッケージがある場合、そのバイナリは/opt/sample/bin/に、manページは/opt/sample/man/に置かれるなど、すべてのファイルを/opt/sample/のディレクトリ内に置くことが出来ます。

それぞれ特定のタスクを実行するサブパッケージを多く含む大きなパッケージもまた、/opt/内に移動し、それによって大きなパッケージは標準化された方法で組織されます。たとえば、sampleパッケージのサブパッケージは/opt/sample/tool1/や/opt/sample/tool2/などの独自のサブディレクトリにそれぞれ移動し、それぞれ独自のbin/、man/、その他の同様のディレクトリを持つことができます。

3.2.1.6. /proc/ディレクトリ

/proc/ディレクトリには、カーネルとの間で情報をやりとりするための特別なファイルが含まれています。

/proc/内ではさまざまなデータが用意されており、このディレクトリを使用してカーネルと通信する方法は数多くあるので、この章ではこの問題を中心に説明しています。詳細は第5章を参照してください。

3.2.1.7. /sbin/ディレクトリ

/sbin/ディレクトリは、rootユーザーのみが使用できる実行可能ファイル群を格納する場所です。/sbin/内の実行可能ファイル群の使用目的は、システムの起動、/usr/のマウント、システム回復操作のみです。ディレクトリに関して、FHSでは次のように説明しています：

> 「通常、/sbinには、/binに含まれるバイナリファイルの他に、システムをブートするために必要なファイルが含まれています。/usrがマウントされたことが認識された後（何も問題がない場合）に実行されるものは、/usr/sbinの中に配置する必要があります。また、ローカル専用のシステム管理バイナリファイルは、/usr/local/sbinの中に配置する必要があります。」

最低でも次のプログラムが/sbin/の中になければなりません：

```
arp, clock,
getty, halt,
init, fdisk,
fsck.*, grub,
ifconfig, lilo,
mkfs.*, mkswap,
reboot, route,
shutdown, swapoff,
swapon, update
```

3.2.1.8. /usr/ディレクトリ

/usr/ディレクトリは、サイト全体にわたって共有することのできるファイルのためのものです。通常、/usr/ディレクトリは独自のパーティションを持っており、読み込み専用でマウント可能とする必要があります。最小限、以下のディレクトリ群を/usr/のサブディレクトリとする必要があります：

```
/usr
 |- bin/
 |- dict/
 |- doc/
 |- etc/
 |- games/
 |- include/
 |- kerberos/
 |- lib/
 |- libexec/
 |- local/
 |- sbin/
 |- share/
 |- src/
 |- tmp -> ../var/tmp/
 |- X11R6/
```

bin/ ディレクトリは実行可能ファイルを含みます。dict/ 非FHS対応ドキュメントページを含みます。etc/ システム全体の設定ファイルを含みます。games ゲームです。include/ C ヘッダーファイルを含みます。kerberos/ Kerberos用にバリナリやその他を含みます。lib/ ユーザーやシェルスクリプトによって直接使用されるように設計されていないオブジェクトファイルやライブラリ。libexec/ディレクトリは、他のプログラムによりコールされる小さなヘルププログラムを含んでいます。sbin/ システム管理バイナリ用(/sbin/ディレクトリに属しない物)。share/ アーキテクチャ固有でないファイルを含みます。src/ ソースコードです。X11R6/ X Window System(**XFree86** on Red Hat Linux)。

3.2.1.9. /usr/local/ディレクトリ

ディレクトリに関して、FHSでは次のように説明しています。

> 「/usr/localは、システム管理者がソフトウェアをローカルにインストールする際に使用するものです。システムソフトウェアの更新時に上書きされないように、この階層を保護する必要があります。マシンのグループの間で共有可能であるプログラムやデータのうち、/usrには含まれないもののために、この階層を使用することができます。」

/usr/local/ディレクトリは、構造に関しては/usr/ディレクトリと似ています。このディレクトリは、以下のサブディレクトリを持っていて、それらのサブディレクトリは、目的に関しては/usr/ディレクトリと似ています：

```
/usr/local
   |- bin/
   |- doc/
   |- etc/
   |- games/
   |- include/
   |- lib/
   |- libexec/
   |- sbin/
   |- share/
   |- src/
```

3.2.1.10. /var/ディレクトリ

FHSはLinuxが/usr/を読み込み専用としてマウントできることを要求しているため、ログファイルを作成するプログラムや、spool/ディレクトリやlock/ ディレクトリを必要とするプログラムは、データを/var/ディレクトリに書き込む必要があります。FHSは/var/の目的を以下のように述べています：

> 「...変数データファイル。ここには、spoolディレクトリとspoolファイル、管理データとログデータ、一時ファイルが含まれます。」

以下のディレクトリ群は/var/のサブディレクトリとする必要があるものです：

```
/var
 |- account/
 |- arpwatch/
 |- cache/
 |- crash/
 |- db/
 |- empty/
 |- ftp/
```

```
|- gdm/
|- kerberos/
|- lib/
|- local/
|- lock/
|- log/
|- mail -> spool/mail/
|- mailman/
|- named/
|- nis/
|- opt/
|- preserve/
|- run/
+- spool/
   |- anacron/
   |- at/
   |- cron/
   |- fax/
   |- lpd/
   |- mail/
   |- mqueue/
   |- news/
   |- rwho/
   |- samba/
   |- slrnpull/
   |- squid/
   |- up2date/
   |- uucp/
   |- uucppublic/
   |- vbox/
   |- voice/
|- tmp/
|- tux/
|- www/
|- yp/
```

messages/やlastlog/などのシステムログファイルは、/var/log/ディレクトリ内に配置されます。/var/lib/rpm/ ディレクトリには、RPMシステムデータベースも含まれています。ロックファイルは/var/lock/に格納され、通常は、そのファイルを使用するプログラム固有のディレクトリに格納されます。/var/spool/ディレクトリは、データファイルを格納する必要がある各種システムのためのサブディレクトリを持っています。

3.2.2. Red Hat Linuxの/usr/local/

Red Hat Linuxの場合、/usr/local/の意図された用途としては、FHSの指定とは多少異なっています。FHSは、システムソフトウェアのアップグレード時には、保護するソフトウェアを/usr/local/に格納すべきであるとしています。Red Hat Linuxからのシステムアップグレードはrpmやグラフィカルな**パッケージ管理ツール** によって安全に行われるため、ソフトウェアを/usr/local/に配置して保護する必要はありません。代わりに、マシンにとってローカルなソフトウェアのために/usr/local/を使用するのがよいでしょう。

例えば、/usr/ディレクトリがリモートホストから読み込み専用NFS共有としてマウントされた場合、/usr/local/ディレクトリの下にパッケージやプログラムをインストールすることが出来ます。

3.3. 特別なファイルの場所

Red Hat Linuxは、特殊ファイルを収納するために、FHSの構造を少々拡張しています。

RPM(*Red Hat Package Manager*)に関するほとんどのファイルは`/var/lib/rpm/`ディレクトリ内に格納されています。詳細情報は*Red Hat Linux* カスタマイズガイドの中の*RPM*によるパッケージ管理と言う章で御覧下さい。

`/var/spool/up2date/`ディレクトリには、システムの為のRPMヘッダ情報など**Red Hat 更新エージェント**で使用されるファイルを含んでいます。この場所は、システムを更新している間にダウンロードされたRPMを一時的に保存するのに使用することが出来ます。Red Hat ネットワークに付いての詳細は、次のwebサイトでRed Hat ネットワークを参照して下さい。https://rhn.redhat.com/。

もう1つのRed Hat Linux特有の場所は、`/etc/sysconfig/`ディレクトリです。このディレクトリは、各種の設定情報を収納しています。ブート時に実行する多くのスクリプトはこのディレクトリ内のファイルを使用します。このディレクトリにある内容とブートプロセスでこれらのファイルが果たす役割についての情報は第4章を御覧下さい。

最後に、説明しなければならないもう一つのディレクトリは、`/initrd/`ディレクトリです。これは空ですが、起動プロセス時にマウントポイントとして使われます。

警告

いかなる理由であれ、`/initrd/`ディレクトリは削除しないで下さい。このディレクトリを削除すると、カーネルパニックエラーメッセージが表示され、システムをブートできなくなる原因となります。

第4章

sysconfig ディレクトリ

/etc/sysconfig/ ディレクトリは、Red Hat Linuxの為のさまざまなシステム設定ファイルが保存されている場所です。

この章では、/etc/sysconfig/にあるファイル、その機能、その内容等の概要を説明していきます。これらのファイルの多くは、特別な、あるいは稀な状況でしか使用しない各種のオプションを含んでいるため、本章の情報は完全性を意図しているものではありません。

4.1. /etc/sysconfig/ディレクトリ内のファイル

次のファイルは通常、/etc/sysconfig/ディレクトリの中で見付けることが出来ます：

- amd
- apmd
- arpwatch
- authconfig
- cipe
- clock
- desktop
- dhcpd
- firstboot
- gpm
- harddisks
- hwconf
- i18n
- identd
- init
- ipchains
- iptables
- irda
- keyboard
- kudzu
- mouse
- named
- netdump
- network
- ntpd
- pcmcia
- radvd

- rawdevices
- redhat-config-securitylevel
- redhat-config-users
- redhat-logviewer
- samba
- sendmail
- soundcard
- spamassassin
- squid
- tux
- ups
- vncservers
- xinetd

注意

上記のファイルの内の幾つかが/etc/sysconfig/ディレクトリにない場合、その関連のプログラムがインストールされていない可能性があります。

4.1.1. /etc/sysconfig/amd

/etc/sysconfig/amd ファイルはamdによって使用される多彩なパラメータを含んでいます。これはファイルシステムの自動マウント/アンマウントを可能にします。

4.1.2. /etc/sysconfig/apmd

/etc/sysconfig/apmdファイルは、サスペンドや復元機能で開始/停止/変更するパワーセッティングの構成をするapmdによって使用されます。このファイルはブート時にapmdをオン又はオフに切替えるように設定してあり、その切替えは、ハードウェアが*APM* (*Advanced Power Management*)をサポートするかどうか、又はユーザーがそれを使用するようにシステムを設定しているかどうかによって左右されます。apmデーモンはLinux カーネル内でパワー管理コードと共に機能するモニタプログラムです。これはラップトップや他のパワー関連の設定でバッテリー低下をユーザーに通知する能力があります。

4.1.3. /etc/sysconfig/arpwatch

/etc/sysconfig/arpwatchファイルは、ブート時に引数をarpwatchデーモンに渡すのに使用されます。arpwatchデーモンはイーサネットのマックアドレスとそのIPアドレスのペアリングのテーブルを保全します。このファイルに利用できるパラメータについての詳細はarpwatchのmanページを御覧下さい。デフォルトでは、このファイルはarpwatch プロセスのオーナーをユーザーpcapに設定します。

4.1.4. /etc/sysconfig/authconfig

/etc/sysconfig/authconfigファイルは、ホスト上で使用される種類の権限を設定します。これには、以下の行の1つ又は複数が含まれます：

- USEMD5=<*value*>, ここで、<*value*>は次のいずれかです：
  - yes — MD5 は認証に使用されます。
  - no — MD5 は認証に使用されません。

- USEKERBEROS=<*value*>, ここで<*value*>は次のいずれかです：
  - yes — Kerberos は認証に使用されます。
  - no — Kerberos は認証に使用されません。

- USELDAPAUTH=<*value*>, ここで<*value*>はつぎのいずれかです：
  - yes — LDAP は認証に使用されます。
  - no — LDAP は認証に使用されません。

4.1.5. /etc/sysconfig/clock

/etc/sysconfig/clockファイルは、システムのハードウェアクロックから読み込んだ値の翻訳を制御します。

その正しい値は次のようになります：

- UTC=<*value*>, ここで<*value*>は、次のブール値のいずれかです：
  - true 又はyes — ハードウェアクロックは世界標準時にセットされます。
  - false 又はno — ハードウェアクロックはローカル時にセットされます。

- ARC=<*value*>, ここで<*value*>は、以下のようになります：
  - true 又はyes — ARCコンソールの42年の時間オフセットが有効になっています。このセッティングは、ARC- 又はAlphaBIOSベースのAlphaシステムのみの為のものです。表示される他の値は、通常のUNIXエポックが有効という意味です。

- SRM=<*value*>, ここで<*value*>は、次のようになります：
  - true 又はyes — SRM コンソールの1900 エポックが有効になっています。この設定はSRM-ベースのAlphaシステムのみのものです。表示される他の値は、通常のUNIXエポックが有効であるとの意味です。

- ZONE=<*filename*> — /etc/localtimeのコピー元である/usr/share/zoneinfoの中のタイムゾーンファイル。このファイルには以下のような情報が含まれます：
  ZONE="America/New York"

以前のリリースのRed Hat Linuxは以下の値を使用していました(現在無視されています)：

- CLOCKMODE=<*value*>, ここで<*value*> は次のいずれかです：
  - GMT — クロックは世界標準時にセットされています。(グリニッジ標準時間)

- ARC — ARCコンソールの42-年のタイムオフセットは有効になっています(Alpha ベースシステムのみ)。

4.1.6. /etc/sysconfig/desktop

/etc/sysconfig/desktopファイルは実行されるべきデスクトップマネージャを以下のように指定します：

```
DESKTOP="GNOME"
```

4.1.7. /etc/sysconfig/dhcpd

/etc/sysconfig/dhcpdファイルは、ブート時に引数をdhcpdデーモンに渡す為に使用されます。dhcpd デーモンはDHCP(Dynamic Host Configuration Protocol)とBOOTP (Internet Bootstrap Protocol)を実装するものです。DHCPとBOOTPはネットワーク上のマシンにホスト名を割り当てます。このファイル用に利用出来るパラメータに関する詳細はdhcpdのmanで御覧下さい。

4.1.8. /etc/sysconfig/firstboot

Red Hat Linux 8.0から開始されたもので、最初にシステムがブートする時、/sbin/initプログラムはetc/rc.d/init.d/firstboot スクリプトをコールし、それが**セットアップエージェント**を開始させます。このアプリケーションによってユーザーは最新の更新のみならず、追加のアプリケーションやドキュメントをインストールすることが出来ます。

/etc/sysconfig/firstbootファイルは**セットアップエージェント** アプリケーションに対しその後の再起動では実行しないよう指示をします。次回システムがブートする時に実行するに、/etc/sysconfig/firstbootを削除して、chkconfig --level 5 firstboot onを実行します。

4.1.9. /etc/sysconfig/gpm

/etc/sysconfig/gpmファイルは、ブート時に引数をgpmデーモンに渡す為に使用されます。gpmデーモンは、マウスの加速と中ボタンクリックの張り付けを可能にするマウスサーバです。このファイルで利用できるパラメータに関する詳細はgpmのmanページで御覧下さい。デフォルトでは、これはマウスデバイスを/dev/mouseにセットします。

4.1.10. /etc/sysconfig/harddisks

/etc/sysconfig/harddisksファイルはハードドライブをチューンします。管理者は/etc/sysconfig/hardiskhd[a-h]を使用して、特定のドライブ用にパラメータを設定することも出来ます。

警告

注意深い計画なしにこのファイルを変更しないようにして下さい。デフォルトの値を変えると、ハードドライブの全てのデータを破損する可能性が有ります。

/etc/sysconfig/harddisksファイルは以下の項目を含むことができます：

- USE_DMA=1,この値を1に設定してある場合、DMAを有効にします。しかし幾つかのチップセットとハードドライブの組合せではDMAがデータ破損の原因ともなり得ます。このオプションを有効にする前にハードドライブのマニュアル、又は製造元でチェックして下さい。
- Multiple_IO=16, 設定が16の場合、I/O 割込み毎に複数のセクターを許可します。有効になっている時には、この機能はオペレーティングシステムの実行負担を30-50%削減します。注意して使用して下さい。
- EIDE_32BIT=3 インターフェースカードに対する(E)IDE 32-bit I/Oサポートを有効にします。
- LOOKAHEAD=1 ドライブの先読みを有効にします。
- EXTRA_PARAMS= 余分のパラメータが追加できる場所を指定します。

4.1.11. /etc/sysconfig/hwconf

/etc/sysconfig/hwconfファイルは、システム上でkudzuが検出するハードウェアの全て、及び使用されるドライバ、ベンダーID、デバイスID情報などの一覧表示します。kudzuプログラムはシステム上の新規、及び変更のあったハードウェアを検出し、設定します。/etc/sysconfig/hwconfファイルは手動で編集されるべきものではありません。もし編集された場合は、デバイスの一部が突然、追加や削除された項目として表示される可能性があります。

4.1.12. /etc/sysconfig/i18n

/etc/sysconfig/i18nファイルは、デフォルトの言語、サポートする言語、及びデフォルトシステムのフォントを設定します。例えば次のような表示になります：

```
LANG="en_US.UTF-8"
SUPPORTED="en_US.UTF-8:en_US:en"
SYSFONT="latarcyrheb-sun16"
```

4.1.13. /etc/sysconfig/identd

/etc/sysconfig/identdファイルは、ブート時に引数をidentdデーモンに渡す為に使用されます。identd デーモンはTCP/IP接続が開いている状態でプロセスのユーザー名を返送します。identdが動作していない場合には、FTP やIRCなどのネットワーク上のサービスは、その注意を出し、遅い反応の原因となります。ただ、通常はidentdは必要なサービスではなく、セキュリティが問題になる場合は、これを起動しないほうが良いでしょう。このファイルで利用できるパラメータに関する情報はidentdのmanページを御覧下さい。デフォルトでは、このファイルはパラメータを含みません。

4.1.14. /etc/sysconfig/init

/etc/sysconfig/initファイルはブートプロセスでシステムの表示法と機能を制御します。

次のような値を使用することが出来ます：

- BOOTUP=<*value*>, ここで<*value*>は次のいずれかになります：
  - BOOTUP=color 標準のカラーブート表示を意味し、これはデバイスの成功か失敗か、そしてサービスが開始しているかを別々のカラーで表示することになります。
  - BOOTUP=verbose 古いスタイルのディスプレイで、単なる成功/失敗のメッセージのみでなく、より多くの情報を提供します。
  - それ以外は、新しいディスプレイとなります。しかしANSI-形式はありません。

- RES_COL=<*value*>, この<*value*> は、ステータスラベルを開始する画面の列の数字です。デフォルトは60です。
- MOVE_TO_COL=<*value*>, この<*value*>はecho -enコマンドを経由してRES_COL行の値までカーソルを動かすという意味です。
- SETCOLOR_SUCCESS=<*value*>, この<*value*>は、echo -enコマンドを経由して成功を示す為のカラーにそのカラーを設定します。デフォルトはグリーンにセットされています。sets
- SETCOLOR_FAILURE=<*value*>, この<*value*>はecho -enコマンドを経由して失敗を示す為のカラーにそのカラーを設定します。デフォルトのカラーは赤にセットされています。
- SETCOLOR_WARNING=<*value*>, この<*value*>はecho -enコマンドを経由して警告のカラーを設定します。デフォルトは黄色にセットされています。
- SETCOLOR_NORMAL=<*value*>, この<*value*>はecho -enコマンドを経由して"ノーマル"のカラーをリセットします。
- LOGLEVEL=<*value*>, この<*value*>は、カーネル用の初期コンソールログインのレベルです。デフォルトは3 ; 8 はすべてを意味します(デバッグを含む) ; 1 はカーネルパニック以外は何も意味しません。syslogdデーモンは一度スタートするとこのセッティングをオーバーライドします。
- PROMPT=<*value*>, ここで<*value*>は次にブール値のいずれかとなります :
  - yes — 対話式モードのキーチェックを有効にします。
  - no — 対話式モードのキーチェックを無効にします。

4.1.15. /etc/sysconfig/ipchains

/etc/sysconfig/ipchains ファイルにはipchainsサービスを設定している時にipchains 初期化スクリプトによって使用される情報が含まれます。

このファイルは、有効なipchains規則が設置してある状態で/sbin/service ipchains saveコマンドを入力することにより変更されます。このファイルは手動で編集しないで下さい。その代わり、コマンド/sbin/ipchainsを使用して必要なパケットフィルター規則を設定して、/sbin/service ipchains saveコマンドを使用してその規則をこのファイルに保存します。

ファイアウォール規則の設定にipchainsを使用することは推薦できません。それは無視されている状態で、将来のリリースのRed Hat Linuxからは無くなるでしょう。ファイアウォールが必要な場合は、代わりにiptablesを使用して下さい。

4.1.16. /etc/sysconfig/iptables

/etc/sysconfig/ipchainsのように、/etc/sysconfig/iptablesファイルは、ブート時又は、サービスが開始された時はいつでも、パケットフィルタサービスを設定する為にカーネルに使用される情報を保存します。

iptables規則を構築する方法を知っている場合以外はこのファイルは手動で編集しないで下さい。最も簡単に規則を追加する方法は**セキュリティレベル設定ツール** (redhat-config-securitylevel)、/usr/sbin/lokkit コマンド、**GNOME Lokkit**アプリケーションのいずれかを使用してファイアウォールを作成することです。これらのアプリケーションを使用するとプロセスの最後にこのファイルが自動的に編集されます。

規則は/sbin/iptablesを使用して手動で作成できます。そして/sbin/service iptables saveと入力して規則を/etc/sysconfig/iptablesファイルに追加します。

このファイルが存在すると、そこに保存されたファイアウォール規則はシステムの再起動やサービスの再スタートの後でも継続されます。

iptablesに関する詳細は第16章で御覧下さい。

4.1.17. /etc/sysconfig/irda

/etc/sysconfig/irdaファイルは、システム上の赤外線デバイスがスタート時に設定される状態を制御します。

次のような値を使用できます：

- IRDA=<*value*>, この<*value*>は次のブール値のいずれかとなります：
  - yes — irattachが実行されて、ネットワークに接続しようとする別のノートブックコンピュータなどが赤外線ポートに接続を試みているかどうかを定期的にチェックします。このシステム上で赤外線デバイスが動作するにはこの行がyesに設定されている必要があります。
  - no — irattachは実行されません。赤外線デバイスの通信は阻止されます。

- DEVICE=<*value*>, ここで<*value*>とは、赤外線接続を担当するデバイス(通常はシリアルポート)です。
- DONGLE=<*value*> ここで<*value*>は、赤外線通信に使用されているドングルのタイプを指定します。この設定は本来の赤外線ポートではなく、シリアルドングルを使用するユーザーの為に存在します。ドングルとは、赤外線経由での通信に使用するために通常のシリアルポートに付けられたデバイスです。このような添付ドングルのコンピュータよりも本来の赤外線ポートを持つノートブックの方か遥かに多いため、デフォルトではこの行はコメントアウトしてあります。
- DISCOVERY=<*value*>, ここで<*value*>は以下のブール値のいずれかとなります：
  - yes —irattachを発見モードでスタートします。これは他の赤外線デバイスを積極的にチェックするという意味です。マシンが赤外線接続を積極的に求めるようにするには、このアプリケーションが稼働している必要があります(ピアは接続を開始しないという意味です)。
  - no —irattachを発見モードでスタートしません。

4.1.18. /etc/sysconfig/keyboard

/etc/sysconfig/keyboardファイルはキーボードの動きを制御します。次の値を使用できます：

- KEYBOARDTYPE=sun|pcこれは、SPARCのみに使用されます。sunと言う意味は、Sunキーボードが/dev/kbdと接続してあることであり、pc が、PS/2キーボードがPS/2ポートに接続してあることを意味します。
- KEYTABLE=<*file*>, ここで<*file*>はキーテーブルファイルの名前です。

  例えば：KEYTABLE="us"。/lib/kbd/keymaps/i386の中でキーテーブルがスタートして、そこから別々のキーボード配列に別れる時にこのファイルが使用されます。すべて<*file*>.kmap.gzのラベルが付きます。/lib/kbd/keymaps/i386の下あり、 KEYTABLEセッティングにマッチする最初のファイルが使用されます。

4.1.19. /etc/sysconfig/kudzu

/etc/sysconfig/kuzduファイルは、ブート時にkudzuによりシステムハードウェアの安全検出を開始します。安全検出ではシリアルポート検出は無効です。

- SAFE=<*value*>, ここで<*value*>は以下のいずれかになります：

- yes — kuzduは安全検出を実行します。
- no — kuzduはノーマル検出を実行します。

4.1.20. /etc/sysconfig/mouse

/etc/sysconfig/mouse ファイルは、利用できるマウスに関する情報を指定するのに使用されます。次の値が使用できます：

- FULLNAME=<*value*>, この<*value*>は使用されているマウスの種類のフルネームを示します。
- MOUSETYPE=<*value*>, この<*value*>は以下のいずれかになります：
  - imps2 — 汎用USB wheel マウス。
  - microsoft — Microsoft™マウス。
  - mouseman — MouseMan™マウス。
  - mousesystems — Mouse Systems™ マウス。
  - ps/2 — PS/2 マウス。
  - msbm — Microsoft™ バスマウス。
  - logibm — Logitech™ バスマウス。
  - atibm — ATI™ バスマウス。
  - logitech — Logitech™ マウス。
  - mmseries — 古いタイプのMouseMan™マウス。
  - mmhittab — mmhittab マウス。
- XEMU3=<*value*>, この<*value*>は次のブール値のいずれかになります：
  - yes — マウスは2つボタンしかありませんが、3つボタンのエミュレーション(模倣)が出来ます。
  - no — マウスはすでに3つボタンを装備しています。
- XMOUSETYPE=<*value*>, この<*value*>は、Xが動作している時に使用されるマウスの種類を示します。ここのオプションは、この同じファイル内でのMOUSETYPEセッティングと同じです。
- DEVICE=<*value*>, この<*value*>は、マウスデバイスです。

さらには、/dev/mouseとは、実際のマウスデバイスを指すシンボリックリンクです。

4.1.21. /etc/sysconfig/named

/etc/sysconfig/namedファイルは、ブート時に引数をnamedデーモンに渡すのに使用されます。named デーモンは、*BIND*(*Berkeley Internet Name Domain*) バージョン9 ディストリビューションを実装する*DNS*(*Domain Name System*)サーバです。このサーバはネットワーク上のIP アドレスと関連しているホスト名のテーブルを管理します。

現在、次の値だけが使用できます：

- ROOTDIR="<*/some/where*>", ここで<*/some/where*>は、namedが実行される設定済みのchroot環境のフルディレクトリのパス(経路)を示します。このchroot環境が最初に設定される必要があります。詳細を得るにはinfo chrootと入力して、info案内を御覧下さい。

- OPTIONS="<*value*>", この<*value*>は、namedman ページにリストされている内、-t以外のオプションです。-tの代わりに、上記のROOTDIRを使用します。

このファイルで利用できるパラメータの詳細情報は、namedのmanページで御覧下さい。BIND DNSサーバを設定する詳しい情報は第12章で御覧下さい。デフォルトでは、このファイルはパラメータを含んでいません。

4.1.22. /etc/sysconfig/netdump

/etc/sysconfig/netdumpファイルは、/etc/init.d/netdump サービスの為の設定ファイルです。netdumpサービスはネットワークを経由してoopsデータとメモリダンプを送信します。一般的にnetdumpは必要のないサービスで、絶対に必要な時にのみ実行します。このファイルで利用できるパラメータに関する詳細情報はnetdumpのmanページで御覧下さい。

4.1.23. /etc/sysconfig/network

/etc/sysconfig/networkファイルは、目的のネットワーク設定に関する情報を指定するのに使用されます。以下の値が使用できます：

- NETWORKING=<*value*>, ここで<*value*>は以下のブール値のいずれかになります：
  - yes — ネットワークを設定する必要があります。
  - no — ネットワークを設定する必要がありません。

- HOSTNAME=<*value*>, ここで<*value*>は、hostname.expample.comなどの完全修飾型ドメイン名(*FQDN*)である必要があります。しかしこれは必要な名前なら何でも結構です。

注意

ユーザーがインストールする可能性のある古いソフトウェア(trnなど)との互換性の為に/etc/HOSTNAMEファイルにはここにある値と同じものを含んでいる必要があります。

- GATEWAY=<*value*>, ここで、<*value*>は、ネットワークゲートウェイのIP アドレスです。
- GATEWAYDEV=<*value*>, ここで、<*value*>はeth0などのゲートウェイデバイスです。
- NISDOMAIN=<*value*>, ここで<*value*>はNIS ドメイン名です。

4.1.24. /etc/sysconfig/ntpd

/etc/sysconfig/ntpdは、ブート時に引数をntpdへ渡す為に使用されます。ntpd デーモンはインターネット標準時間サーバと同期をとる為にシステムクロックを設定して管理します。これはネットワーク時間プロトコル(NTP)のバージョン4 を実装するものです。このファイルで利用できるパラメータについての詳細はブラウザで次のファイルを指定します。:/usr/share/doc/ntp-<*version*>/ntpd.htm(ここで<*version*> とはntpdのバージョン番号です。)。デフォルトでは、このファイルはntpdのオーナーをユーザーntp と設定します。

4.1.25. /etc/sysconfig/pcmcia

/etc/sysconfig/pcmciaファイルは、PCMCIAの設定情報を指定するのに使用されます。次の値が使用されます：

- PCMCIA=<*value*>, この<*value*>は次のいずれかになります：
  - yes — PCMCIA サポートは有効にする必要があります。
  - no — PCMCIA サポートは有効にする必要がありません。

- PCIC=<*value*>, この<*value*>は、以下のいずれかです：
  - i82365 — コンピュータはi82365-スタイルのPCMCIA ソケットチップセットを持ちます。
  - tcic — コンピュータはtcic-スタイルのPCMCIA ソケットチップセットを持ちます。

- PCIC_OPTS=<*value*>, ここで<*value*>は、ソケットドライバ(i82365 又はtcic) タイミングパラメータです。
- CORE_OPTS=<*value*>, ここで<*value*>は、pcmcia_coreオプションの一覧です。
- CARDMGR_OPTS=<*value*>, ここで<*value*>はPCMCIA cardmgrのオプションの一覧です。(例：-qは静かなモードで、-mは指定のディレクトリでロードできるカーネルモジュールを探します。) 詳細情報はcardmgrのmanページを御覧下さい。

4.1.26. /etc/sysconfig/radvd

/etc/sysconfig/radvdファイルは、ブート時に引数をradvdデーモンに渡すために使用されます。radvdデーモンをルーターの要求をリッスンしてIPバージョン6 プロトコル用のルーター広報を送信します。このサービスによってネットワーク上のホストは、これらのルーター広報をベースにして動的にそのデフォルトのルーターを変更することが出来ます。このファイルで利用できるパラメータに付いての詳細はradvdのmanページを御覧下さい。デフォルトでは、このファイルはradvdプロセスのオーナーをユーザーradvdに設定します。

4.1.27. /etc/sysconfig/rawdevices

/etc/sysconfig/rawdevicesファイルは以下のような生デバイスのバインディングの設定に使用されます：

```
/dev/raw/raw1 /dev/sda1
/dev/raw/raw2 8 5
```

4.1.28. /etc/sysconfig/redhat-config-securitylevel

/etc/sysconfig/redhat-config-securitylevelファイルには、最後に**セキュリティレベル設定ツール** (redhat-config-securitylevel)が実行された時にユーザーが選択した全てのオプションが含まれています。ユーザーはこれを手動で変更すべきではありません。**セキュリティレベル設定ツール**の詳細については*Red Hat Linux* カスタマイズガイドの中で、基本的ファイヤーウォールの設定の章を御覧下さい。

4.1.29. /etc/sysconfig/redhat-config-users

/etc/sysconfig/redhat-config-usersファイルは、グラフィカルアプリケーション**ユーザーマネージャ**用の設定ファイルです。Red Hat Linuxの元ではこのファイルはroot、daemon、lpなどのシステムユーザーをフィルターにかける為に使用されます。このファイルは**ユーザーマネージャ**アプリケーション内の**個人設定** => **システムユーザーとグループにフィルター**のプルダウンメニューで編集されますので、手動で編集するものではありません。このアプリケーションの詳細については、*Red Hat Linux* カスタマイズガイドのユーザーとグループの設定の章を御覧下さい。

4.1.30. /etc/sysconfig/redhat-logviewer

/etc/sysconfig/redhat-logviewerファイルはグラフィカルで、対話式のログ表示アプリケーション**ログビューア**用の設定ファイルです。このファイルは**ログビューア**内の**編集** =>**設定** プルダウンメニューにより編集されるため、手動で編集しないで下さい。このアプリケーションに関しての詳細は*Red Hat Linux* カスタマイズガイド内のログファイルの章でお読み下さい。

4.1.31. /etc/sysconfig/samba

/etc/sysconfig/sambaファイルは、ブート時に引数をsmbdデーモンとnmbdデーモンに渡す為に使用されます。smbdデーモンはネットワーク上のWindowsクライアントとのファイル共有の接続を提供します。nmbd デーモンは、IPネーミングサービス上でNetBIOSを提供します。このファイルで利用できるパラメータに関する詳細は、smbdのmanページを御覧下さい。デフォルトでは、このファイルはsmbdとnmbdをデーモンモードで実行するように設定します。

4.1.32. /etc/sysconfig/sendmail

/etc/sysconfig/sendmailファイルによって必要なネットワーク上でメッセージを配送して1人又は複数の受信者にメッセージを送ることが出来ます。このファイルは、Sendmailアプリケーションが実行されるようにデフォルトの値を設定します。そのデフォルトの値は、デーモンをバックグラウンドで動作するようにして、何かがバックアップされた場合の為に1時間毎にキューをチェックするようになっています。

次のような値が使用できます：

- DAEMON=<*value*>, ここで<*value*>は、以下のブール値のいずれかになります：
  - yes —Sendmailは、受信メール用にポート25をリッスンするように設定する必要があります。yesはSendmailの-bdオプションの使用を意味します。
  - no —Sendmailは、受信メール用にポート25をリッスンするように設定する必要がありません。

- QUEUE=1h Sendmailにそれを-q$QUEUEとして与えます。-qオプションは、/etc/sysconfig/sendmailが存在していて、QUEUEが空又は、未定義の場合は、Sendmailに与えられません。

4.1.33. /etc/sysconfig/soundcard

/etc/sysconfig/soundcardファイルは、sndconfigによって生成されるものです。修正すべきではありません。このファイルの唯一の使用目的は、次回にsndconfigが実行された時にデフォルトでメニュー内にポップアップするカードエントリを決定することです。サウンドカード設定情報は/etc/modules.conf ファイルの中に位置しています。

以下のような項目が該当します：

- CARDTYPE=<*value*>, ここで<*value*> は、例としてサウンドブラスター16 サウンドカード用にSB16にセットされます。

4.1.34. /etc/sysconfig/spamassassin

/etc/sysconfig/spamassassinファイルは、ブート時に引数をspamdデーモン(Spamassassinのデーモン版)に渡す為に使用されます。Spamassassinは、電子メールスパム用のフィルタアプリケーションです。利用できるオプションに関しては、spamdのmanページを御覧下さい。デフォルトでは、これはspamdをデーモンモードで実行し、ユーザーの好みを設定し、そして白紙リストを自動作成します。

Spamassassinに関する詳細は、項11.4.2.6を御覧下さい。

4.1.35. /etc/sysconfig/squid

/etc/sysconfig/squidファイルは、ブート時に引数をsquidデーモンに渡すのに使用されます。squid デーモンは、Webクライアントアプリケーション用のプロキシキャッシングサーバです。squidプロキシサーバに関する詳細情報は、Webブラウザを使用して/usr/share/doc/squid-<*version*>/ ディレクトリを開いて御覧下さい(<*version*>の部分はシステムにインストールされているsquidのバージョン番号です)。デフォルトでは、このファイルはsquidをデーモンモードでスタートする様に設定して、停止するまでの時間の値をセットします。

4.1.36. /etc/sysconfig/tux

/etc/sysconfig/tuxファイルは、Red Hat Content Accelerator (旧名 : TUX)、カーネルベースのWebサーバ用の設定ファイルです。Red Hat Content Acceleratorの設定に関する詳細は、Webブラウザを使用して、/usr/share/doc/tux- <*version*>/tux/index.htmlを開いてその内容を確認して下さい(<*version*>は、システムにインストールされているTUXの実際のバージョン番号で入れ換えます)。このファイルで利用出来るパラメータは/usr/share/doc/tux-<*version*>/tux/parameters.htmlに一覧表示してあります。

4.1.37. /etc/sysconfig/ups

/etc/sysconfig/upsファイルは、システムに接続してある*UPS*(*Uninterruptible Power Supplies*)に関する情報を指定するのに使用されます。UPSは、電源が阻止された場合にシステムを正しく停止する時間を与えるため、Red Hat Linuxシステムにとって価値のあるものです。以下のような値が使用されます :

- SERVER=<*value*>, ここで<*value*>は、以下のいずれかになります :
  - yes — UPSデバイスはシステムに接続されています。
  - no — UPSデバイスはシステムに接続されていません。

- MODEL=<*value*>, この<*value*>は、UPSが、システムに接続されていない場合、次のどれかの1つであるか、又はNONEに設定される必要があります :
  - apcsmart — APC SmartUPS™又は、同様のデバイス。
  - fentonups — Fenton UPS™。
  - optiups — OPTI-UPS™デバイス。
  - bestups — Best Power™ UPS。
  - genericups — 汎用ブランドのUPS。

- ups-trust425+625 — Trust™ UPS。

- DEVICE=<*value*>, ここで<*value*>は、/dev/ttyS0など、UPSが接続されている場所を指定します。
- OPTIONS=<*value*>この<*value*>は、UPSに渡す必要のある特別なコマンドです。

4.1.38. /etc/sysconfig/vncservers

/etc/sysconfig/vncserversファイルは*VNC* (*Virtual Network Computing*)サーバがスタートする方法を設定します。

VNC を使用するとユーザーは実行中のマシン上の環境だけでなく、各種アーキテクチャの別のネットワークを通したマシンのデスクトップ環境もリモート表示できます。

以下のようなものが含まれます：

- VNCSERVERS=<*value*> この<*value*>では、VNCサーバがユーザーfred用にディスプレイ:1で開始されることを示すには、"1:fred"のように設定します。ユーザーfredは、リモートリモートVNCサーバに接続する前に、vncpasswdを使用してVNCパスワードを設定しておく必要があります。

VNCサーバを使用している時は、その通信は暗号化されていませんので、信用していないネットワーク上で使用すべきではないことに注意して下さい。VNC通信を安全にするSSHの使用に関しての特別な指示については、以下のサイトにある情報をお読み下さい。http://www.uk.research.att.com/vnc/sshvnc.html SSHについての詳細は第18章、又は*Red Hat Linux* カスタマイズガイドで御覧下さい。

4.1.39. /etc/sysconfig/xinetd

/etc/sysconfig/xinetdファイルはブート時に引数をxinetdデーモンに渡すために使用されます。xinetd デーモンは、ポートへサービスの要求が受信された時にインターネットサービスを提供するプログラムを開始します。このファイル用に利用できるパラメータに関する情報はxinetdのmanページで御覧下さい。xinetdサービスについての詳細は、項15.3を御覧下さい。

4.2. Directories in the /etc/sysconfig/ ディレクトリ

通常、/etc/sysconfig/には、以下のようなディレクトリがあります。

- apm-scripts/ — このディレクトリには、Red Hat APM サスペンド/復帰スクリプトが含まれています。このファイルは直接編集しないで下さい。カスタマイズが必要な場合、/etc/sysconfig/ apm-scripts/apmcontinueと言うファイルを作成すると、スクリプトの最後にコールされます。また、/etc/sysconfig/apmdを編集することでもこのスクリプトを制御できます。
- cbq/ — このディレクトリには、ネットワークインターフェイスのバンド幅管理の為のクラスベースのキュー(Class Based Queuing)を実践する為に必要な設定ファイルが含まれています。
- networking/ — このディレクトリは**ネットワーク管理ツール** (redhat-config-network)によって使用されており、その内容は手動で編集するものではありません。**ネットワーク管理ツール**を使用したネットワークインターフェイスの設定に関する情報は、*Red Hat Linux* カスタマイズガイドにあるネットワーク設定の章をお読み下さい。
- network-scripts/ — このディレクトリには次のネットワーク関連の設定ファイルが含まれています：

- eth0イーサネットインターフェイス用のifcfg-eth0などのようなそれぞれの設定されたネットワークインターフェイスの為のネットワーク設定ファイル。
- ifup とifdownのようなネットワークインターフェイスを起動(up)したり停止(down)したりするのに使用されるスクリプト。
- ifup-isdnやifdown-isdnのようにISDNインターフェイスを起動したり停止したりする為に使用するスクリプト。
- 直接編集すべきではない各種共有のネットワーク機能スクリプト。

network-scriptsディレクトリに関する詳細は、第8章で御覧下さい。

- rhn/ — このディレクトリには、Red Hat ネットワークの設定ファイルとGPGキーが含まれています。このディレクトリ内のファイルは手動で編集すべきではありません。Red Hat Networkに関する詳細は、以下のURLでRed Hat Networkのwebサイトを御覧下さい。URL: https://rhn.redhat.com.

4.3. その他のリソース

この章は、/etc/sysconfig/ディレクトリにあるファイルへの紹介としてだけ意図されています。以下のソースはより総合的な情報を提供することが出来るでしょう。

4.3.1. インストールされているドキュメント

- /usr/share/doc/initscripts-<*version-number*>/sysconfig.txt — このファイルには、/etc/sysconfig/ディレクトリにあるファイルや、そこで利用できる設定オプションなどの権威のある一覧が含まれています。このファイルへのパス内の<*version-number*> は、インストール済みのinitscriptsパッケージのバージョンを示します。

第5章

proc ファイルシステム

Linuxカーネルには2つの主要な機能があります：コンピューターへの物理デバイスのアクセスを制御する事と、いつどのようにこれらのデバイスを相互に作用させるかスケジュールする事です。/procディレクトリには、カーネルの現在の状態を示す特別なファイルの階層が含まれています。— これを利用してアプリケーションやユーザーがカーネルのシステム情報を見ることが出来ます

/procディレクトリでは、システムハードウェアや現在実行中のプロセスの豊富な詳細情報があります。さらには、/procディレクトリツリーの中の幾つかのファイルはユーザーやアプリケーションが操作可能で、カーネルへ設定変更について伝えることが出来ます。

5.1. 仮想ファイルシステム

Linuxでは、すべてがファイルとして保存されます。殆どのユーザーは2つの主要なファイルタイプ：テキストファイルとバイナリファイルに慣れ親しんでいると思われます。しかし、/proc/ ディレクトリには、仮想ファイルと言うもう1つのファイルあります。この理由で、/proc/はよく仮想ファイルシステムと呼ばれます。

これらの仮想ファイルは独特の性質を持ちます。殆どはサイズがゼロバイトでリストされ、1つを表示すると、それは大量の情報を含んでいることがあります。さらには、仮想ファイル上の殆どの時刻と日付の設定が現在の時刻と日付を示しますのでこれが、常時更新されている事実を現しています。

/proc/interrupts、/proc/meminfo、/proc/mounts、/proc/partitionsなどの仮想ファイルは分刻のシステムのハードウェア状況を提供してくれます。その他に、/proc/filesystemsや/proc/sys/ディレクトリの様にシステムの設定ファイルとインターフェイスを提供します。

組織での使用目的の為に、同じような課題の情報を含むファイルは仮想ディレクトリとサブディレクトリにグループ化されます。例えば、/proc/ide/はすべての物理IDEデバイスの情報を含んでいます。同じ様にプロセスディレクトリはシステムの各実行中の情報を含んでいます。

5.1.1. 仮想ファイルの表示

cat、more、lessコマンドを/proc内のファイルと共に使用して、システムについての膨大な情報に即座にアクセスできます。たとえば、コンピュータで使用しているCPUの見る場合は、cat /proc/cpuinfoと入力すると、以下のような表示が得られます：

```
processor : 0
vendor_id : AuthenticAMD
cpu family : 5
model : 9
model name : AMD-K6(tm) 3D+ Processor
stepping : 1
cpu MHz : 400.919
cache size : 256 KB
fdiv_bug : no
hlt_bug : no
f00f_bug : no
coma_bug : no
fpu : yes
fpu_exception : yes
cpuid level : 1
wp : yes
flags : fpu vme de pse tsc msr mce cx8 pge mmx syscall 3dnow k6_mtrr
bogomips : 799.53
```

/procファイルシステム内の様々な仮想ファイルを見ると、情報の一部は意味が分かります。一部は人間が読めないコードになっています。これが、ユーティリティが存在する理由で、それによって、これらの仮想ファイルからデータが抽出され、わかりやすく表示されます。そのようなユーティリティの例としては、lspci、apm、freeそれに、topなどがあります。

注意

/procディレクトリの仮想ファイルの一部はrootユーザーのみ読み取り可能として設定されています。

5.1.2. 仮想ファイルの変更

一般的なルールとして、/proc内の多くの仮想ファイルは読み取り専用です。しかし、カーネルの設定で、調整できるファイルもあります。特に、/proc/sys/ サブディレクトリのファイルが調整可能です。

仮想ファイルの値を変更するためには、echoコマンドと＞記号を使って、ファイルに新しい値をリダイレクトします。例えば、ホスト名を急いで変更するには、以下のように入力します：

```
echo www.example.com > /proc/sys/kernel/hostname
```

他のファイルはバイナリ又はブール値のスイッチとして作用します。例えば、cat /proc/sys/net/ipv4/ip_forwardと入力すると、0または1が返って来ます。0は、カーネルがネットワークパケットをフォワードしないという事を示します。echoコマンドを使って、ip_forwardファイルを1に変更すると、すぐにパケットフォワードが起動します。

ヒント

/proc/sys/サブディレクトリの設定を変更する為に使用するもう1つのコマンドは、/sbin/sysctlです。このコマンドに関する詳細は、項5.4で御覧下さい。

/proc/sys/内の利用できるカーネル設定ファイルのリストは、項5.3.9を参照してください。

5.2. procファイルシステムのトップレベルファイル

以下に、/proc/ディレクトリのトップレベルにあるより役に立つ仮想ファイルの一部のリストを示します。

注意

殆どの場合、このセクションにリストしてあるファイルの内容は、ユーザーのマシンの物と当然、同じではないでしょう。これは、情報がRed Hat Linuxが稼働しているハードウェア限定であることがその理由です。

5.2.1. /proc/apm

このファイルは*APM (Advanced Power Management)*システムの状態に関する情報を提供します。そしてapmコマンドで使用されます。バッテリを装着していないシステムがAC電源に接続されている場合、この仮想ファイルは次のようになります：

```
1.16 1.2 0x07 0x01 0xff 0x80 -1% -1 ?
```

これらのシステムでapm -vコマンドを実行すると、次のような結果が出力されます：

```
APM BIOS 1.2 (kernel driver 1.16)
AC on-line, no system battery
```

電源としてバッテリを使用しないシステムには、apmは、単にマシンをスタンドバイモードにセットする以外はあまり機能しません。ただし、apmコマンドはラップトップではもっと役に立ちます。例えば、次の出力は、電源に接続されRed Hat Linux を稼働しているラップトップ上のcat /proc/apmコマンドからの物です：

```
1.16 1.2 0x03 0x01 0x03 0x09 100% -1 ?
```

同じラップトップを電源から切断し、電池で数分間動作させた場合、apm ファイルの内容は次のように変わります：

```
1.16 1.2 0x03 0x00 0x00 0x01 99% 1792 min
```

ここで、apm -vコマンドは、以下のようなより役に立つデータを作り出します：

```
APM BIOS 1.2 (kernel driver 1.16)
AC off-line, battery status high: 99% (1 day, 5:52)
```

5.2.2. /proc/cmdline

このファイルは、起動した時点でカーネルに渡されたパラメータを示します。サンプルの/proc/cmdlineファイルは以下のようになります：

```
ro root=/dev/hda2
```

これは、1番目のIDEデバイスの2番目のパーティション(/dev/hda2)でカーネルが読み取り専用((ro)で表示)としてマウントされていることを意味します。

5.2.3. /proc/cpuinfo

この仮想ファイルはシステムで使用されているプロセッサのタイプを認識します。次に標準的な/proc/cpuinfoの出力を示します：

```
processor   : 0
vendor_id   : AuthenticAMD
cpu family  : 5
model       : 9
model name  : AMD-K6(tm) 3D+ Processor
stepping    : 1
cpu MHz     : 400.919
cache size  : 256 KB
fdiv_bug    : no
hlt_bug     : no
```

```
f00f_bug    : no
coma_bug    : no
fpu      : yes
fpu_exception  : yes
cpuid level   : 1
wp       : yes
flags     : fpu vme de pse tsc msr mce cx8 pge mmx syscall 3dnow k6_mtrr
bogomips    : 799.53
```

- processor — 各プロセッサに識別番号を提供します。1つのプロセッサしかない場合は、0が表示されます。
- cpu family — システムのプロセッサタイプを正式に表示します。インテルベースのシステムでは、"86"の前に数字を入力して、値を計算します。これは特に586、486、386などの古いアーキテクチャを識別するのに便利です。RPMパッケージが特にこれらの各アーキテクチャ用にコンパイルされている為、この値は、ユーザーがどのパッケージをインストールするか決定するのに役に立ちます。
- model name —プロセッサの一般的な名前とプロジェクト名を表示します。
- cpu MHz — 特定のプロセッサの正確な速度をMHz（千分の1まで）で表示します。
- cache size — プロセッサに利用できるレベル2メモリキャッシュの容量を表示します。
- flags — FPUの有無やMMX指示を処理できるかどうかなど、プロセッサについて多くの機能を定義します。

5.2.4. /proc/devices

このファイルは現在設定されているさまざまなキャラクタデバイスとブロックデバイスを表示します。(モジュールがロードされていないデバイスは除く)。以下にこのファイルのサンプルの出力を示します：

```
Character devices:
  1 mem
  2 pty
  3 ttyp
  4 ttyS
  5 cua
  7 vcs
 10 misc
 14 sound
 29 fb
 36 netlink
128 ptm
129 ptm
136 pts
137 pts
162 raw
254 iscsictl

Block devices:
  1 ramdisk
  2 fd
  3 ide0
  9 md
 22 ide1
```

/proc/devicesからの出力には、デバイスのメジャー番号と名前が含まれています。そして2つの主要なセクションに分かれています。Character devicesとBlock devicesです。

キャラクタデバイスはブロックデバイスと類似していますが、次の2つの相違点があります：

1. ブロックデバイスには利用できるバッファがあり、要求を処理する前に順序付けをすることができます。これは、情報を保存するためのデバイス— 例えばハードディスク—にはとても重要です。というのは、要求をデバイスに書き込む前に順番を付けることができると、効率のよい順番に配置できるためです。キャラクタデバイスにはこの種のバッファリングは必要ではありません。
2. ブロックデバイスは特定サイズのブロック単位で情報を送受信できます。キャラクタデバイスでは、あらかじめ設定されたサイズはなく、適切と判断したサイズでデータを送信します。

デバイスについての情報は/usr/src/linux-2.4/Documentation/devices.txtで御覧下さい。

5.2.5. /proc/dma

このファイルには、使用中の登録済みISA ダイレクトメモリアクセス(DMA)の一覧が含まれています。/proc/dmaファイルの例は以下のようになります：

```
4: cascade
```

5.2.6. /proc/execdomains

このファイルは、このファイルは、Linuxカーネルが現在サポートしている実行ドメインと、サポートしているパーソナリティの範囲の一覧を表示します。

```
0-0 Linux    [kernel]
```

実行ドメインを特定のオペレーティングシステムの「パーソナリティ」と考えます。Solaris、UnixWare、FreeBSDなど他のバイナリフォーマットもLinuxで使用できますので、実行しているタスクのパーソナリティを変更して、プログラマーはオペレーティングシステムが、これらのバイナリからの特定のシステム呼び出しを処理する方法を変更できます。PER_LINUX実行ドメイン以外は、各種のパーソナリティが動的にロード可能なモジュールとして実装できます。

5.2.7. /proc/fb

このファイルには、フレームバッファデバイス、その番号、それを制御するドライバなどの一覧が保存されています。フレームバッファデバイスを搭載したシステムの/proc/fbの典型的な出力例は次のとおりです：

```
0 VESA VGA
```

5.2.8. /proc/filesystems

このファイルは、カーネルが現在サポートしているファイルシステムタイプの一覧を表示します。汎用の/proc/filesystemsファイルの出力例は次のとおりです：

```
nodev rootfs
nodev bdev
nodev proc
```

```
nodev sockfs
nodev tmpfs
nodev shm
nodev pipefs
 ext2
nodev ramfs
 iso9660
nodev devpts
 ext3
nodev autofs
nodev binfmt_misc
```

最初の列は、ファイルシステムがブロックデバイスにマウントされているかどうかを示します。nodevで始まる列は、ファイルシステムがブロックデバイスにマウントされていないことを示します。2番目の列は、サポートされているファイルシステムの名前を表示します。

mountコマンドは、引数として指定されているファイルシステムがない場合に、ここにリストしてあるファイルシステムの中を検索して回ります。

5.2.9. /proc/interrupts

このファイルはx86アーキテクチャ上でIRQごとの割り込み数を記録します。標準的な/proc/interruptsは次のとおりです：

```
    CPU0
 0:  80448940    XT-PIC timer
 1:   174412    XT-PIC keyboard
 2:     0    XT-PIC cascade
 8:     1    XT-PIC rtc
10:   410964    XT-PIC eth0
12:    60330    XT-PIC PS/2 Mouse
14:  1314121    XT-PIC ide0
15:  5195422    XT-PIC ide1
NMI:    0
ERR:    0
```

マルチプロセッサのコンピュータの場合、このファイルは多少異なります：

```
    CPU0   CPU1
 0: 1366814704    0    XT-PIC timer
 1:    128    340  IO-APIC-edge keyboard
 2:     0    0    XT-PIC cascade
 8:     0    1  IO-APIC-edge rtc
12:   5323   5793  IO-APIC-edge PS/2 Mouse
13:     1    0    XT-PIC fpu
16: 11184294  15940594  IO-APIC-level Intel EtherExpress Pro 10/100 Ethernet
20:  8450043  11120093  IO-APIC-level megaraid
30:   10432   10722  IO-APIC-level aic7xxx
31:    23    22  IO-APIC-level aic7xxx
NMI:    0
ERR:    0
```

1番目の列はIRQ番号です。システム内のそれぞれのCPUは独自の列とIRQごとの割り込み数を持ちます。次の列は割込みのタイプを示し、最後の列がそのIRQにあるデバイス名です。

このファイル内の各割り込みタイプはアーキテクチャ固有なので、それぞれ意味することが多少異なります。x86コンピュータの場合、一般的な値は次のとおりです：

- XT-PIC — 従来のATコンピュータの割り込み。
- IO-APIC-edge — この割り込みで電圧信号が「低」から「高」に推移すると、エッジが作成されます。ここで割り込みが生じ、1度だけ信号が生成されます。この種の割り込みは、IO-APIC-level割り込みと同じように、586ファミリ以上のプロセッサを搭載したシステムでのみ生じます。
- IO-APIC-level —電圧信号が「高」になると、その信号が再度「低」になるまで割り込みが生成されます。

5.2.10. /proc/iomem

このファイルは、それぞれのデバイス用に、システムの現在のメモリマップを表示します：

```
00000000-0009fbff : System RAM
0009fc00-0009ffff : reserved
000a0000-000bffff : Video RAM area
000c0000-000c7fff : Video ROM
000f0000-000fffff : System ROM
00100000-07ffffff : System RAM
 00100000-00291ba8 : Kernel code
 00291ba9-002e09cb : Kernel data
e0000000-e3ffffff : VIA Technologies, Inc. VT82C597 [Apollo VP3]
e4000000-e7ffffff : PCI Bus #01
 e4000000-e4003fff : Matrox Graphics, Inc. MGA G200 AGP
 e5000000-e57fffff : Matrox Graphics, Inc. MGA G200 AGP
e8000000-e8ffffff : PCI Bus #01
 e8000000-e8ffffff : Matrox Graphics, Inc. MGA G200 AGP
ea000000-ea00007f : Digital Equipment Corporation DECchip 21140 [FasterNet]
 ea000000-ea00007f : tulip
ffff0000-ffffffff : reserved
```

最初の列は異なる各メモリタイプで使用するメモリレジスタを表示します。2番目の列は、これらのレジスタ内にあるメモリの種類を示します。特に、この列では、システムのRAM内でカーネルが使用するメモリレジスタを表示し、あるいはNICに複数のイーサネットポートがある場合は、各ポートに割り当てられているメモリレジスタも表示します。

5.2.11. /proc/ioports

/proc/ioportsの出力は、デバイスと入出力通信に使用する現在登録されているポート領域の一覧を提供します。このファイルはきわめて長くなることがありますが、開始は次のようになります：

```
0000-001f : dma1
0020-003f : pic1
0040-005f : timer
0060-006f : keyboard
0070-007f : rtc
0080-008f : dma page reg
00a0-00bf : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
02f8-02ff : serial(auto)
0376-0376 : ide1
03c0-03df : vga+
03f6-03f6 : ide0
```

```
03f8-03ff : serial(auto)
0cf8-0cff : PCI conf1
d000-dfff : PCI Bus #01
e000-e00f : VIA Technologies, Inc. Bus Master IDE
  e000-e007 : ide0
  e008-e00f : ide1
e800-e87f : Digital Equipment Corporation DECchip 21140 [FasterNet]
  e800-e87f : tulip
```

1番目の列には2番目の列に表示されているデバイスのために予約されている実際のIOポートアドレス範囲が表示されます。

5.2.12. /proc/isapnp

このファイルはシステム上のISAスロット内のプラグアンドプレイ（*PnP*）カードを表示します。これはサウンドカードの場合が多く、デバイスの数は不定です。/proc/isapnpファイルにサウンドブラスターのエントリがある場合は、次のようになります：

```
Card 1 'CTL0070:Creative ViBRA16C PnP' PnP version 1.0 Product version 1.0
  Logical device 0 'CTL0001:Audio'
    Device is not active
    Active port 0x220,0x330,0x388
    Active IRQ 5 [0x2]
    Active DMA 1,5
    Resources 0
      Priority preferred
      Port 0x220-0x220, align 0x0, size 0x10, 16-bit address decoding
      Port 0x330-0x330, align 0x0, size 0x2, 16-bit address decoding
      Port 0x388-0x3f8, align 0x0, size 0x4, 16-bit address decoding
      IRQ 5 High-Edge
      DMA 1 8-bit byte-count compatible
      DMA 5 16-bit word-count compatible
      Alternate resources 0:1
        Priority acceptable
        Port 0x220-0x280, align 0x1f, size 0x10, 16-bit address decoding
        Port 0x300-0x330, align 0x2f, size 0x2, 16-bit address decoding
        Port 0x388-0x3f8, align 0x0, size 0x4, 16-bit address decoding
        IRQ 5,7,2/9,10 High-Edge
        DMA 1,3 8-bit byte-count compatible
        DMA 5,7 16-bit word-count compatible
```

このファイルは、ここに表示するデバイス数や、あるいはリソース要求によっては、きわめて長くなることがあります。

各カードには、その名前、PnP バージョン番号、製品バージョン番号が表示されています。デバイスがアクティブであり設定されている場合、このファイルはデバイスのポートとIRQ番号も表示します。さらに、互換性の目的で、カードにより多くの異なるパラメータのpreferredとacceptable値も指定します。目的は、PnP カードが相互に正常に動作するようにし、IRQとポートとのコンフリクトを回避することです。

5.2.13. /proc/kcore

このファイルはシステムの物理メモリを表し、コアファイルフォーマットで保存されます。大部分の/procファイルと異なり、kcoreはサイズを表示します。この値はバイト単位で、使用される物理メモリ（RAM）のサイズに4Kバイトを加えたものです。

このファイルの内容は、gdbなどのデバッガで検査されるようにデザインされており人間には読み取れません。

警告

/proc/kcore仮想ファイルは見ないようにしてください。このファイルの内容は、ターミナル上に出力されたテキストを混ぜ合わせたものです。このファイルを誤って見てしまった場合は、[Ctrl]-[C]キーを押して、プロセスを止めて下さい。それから、resetを入力して、コマンドラインのプロンプトを呼び戻してください。

5.2.14. /proc/kmsg

このファイルは、カーネルが生成したメッセージを保持しておくために使用します。これらのメッセージは、/sbin/klogdなど他のプログラムが取り出します。

5.2.15. /proc/ksyms

このファイルには、カーネルモジュールを動的にリンクし結合する為にモジュールツールによって使用されるシンボル定義が含まれています。

```
e003def4 speedo_debug [eepro100]
e003b04c eepro100_init [eepro100]
e00390c0 st_template [st]
e002104c RDINDOOR [megaraid]
e00210a4 callDone [megaraid]
e00226cc megaraid_detect [megaraid]
```

1番目の列には、カーネル内のその機能のメモリアドレスが表示されます。2番目の列にはそのカーネル機能の名前が表示されます。最後の列にはロードされたモジュールの名前が表示されます。

5.2.16. /proc/loadavg

このファイルは、プロセッサーの一定期間のロード平均、uptimeや他のコマンドが使用する追加データを表示します。/proc/loadavgファイルの出力例は次のとおりです：

```
0.20 0.18 0.12 1/80 11206
```

最初の3つの列は、1分、5分、10分間隔で、最新のCPU使用率の測定値を表します。4番目の列は、現在実行されているプロセス数とプロセス総数を表示します。最後の列は、最後に使用したプロセスIDを表示します。

5.2.17. /proc/locks

このファイルは、カーネルが現在ロックしているファイルを表示します。このファイルには、カーネルの内部デバッグデータが保存されており、ファイル内容はシステムの使用によって大きく異なります。負荷の軽いシステムの/proc/locksファイルの出力例は次のとおりです：

```
1: FLOCK ADVISORY WRITE 807 03:05:308731 0 EOF c2a260c0 c025aa48 c2a26120
2: POSIX ADVISORY WRITE 708 03:05:308720 0 EOF c2a2611c c2a260c4 c025aa48
```

各ロックに対して、各行の始めに固有の番号が割り当てられます。2番目の列は、使用ロッククラスを示します。FLOCKは、flockシステム呼び出しからの従来のスタイルのUNIXファイルロックを表し、POSIXは、lockfシステム呼び出しからの新しいPOSIXロックを表します。

3番目の列には2つの値：ADVISORY又はMANDATORYです。ADVISORYは、ロックしても他のユーザーがデータにアクセスできることを意味します。つまり、他のユーザーがロックすることを防止するのみです。MANDATORYは、ロックされている間、他のユーザーがデータにアクセスできないことを意味します。4番目の列は、ロックによってREAD やWRITEのアクセス権を持つユーザーがファイルにアクセスできるようにするかどうかを示します。5番目の列は、ロックを保持するプロセスのIDを示します。6番目の列は、ロックされるファイルのIDを示します。フォーマットは、*MAJOR-DEVICE*:*MINOR-DEVICE* :*INODE-NUMBER*です。7番目の列は、ファイルのロックされた領域の開始と終了を示します。残りの列は、特定のデバッグに使用された内部カーネルデータ構造を示すもので、無視してかまいません。

5.2.18. /proc/mdstat

このファイルには複数ディスクのRAID設定の現在の情報が保存されています。システムにこのような設定がない場合、/proc/mdstatファイルは次のようになります：

```
Personalities :
read_ahead not set
unused devices: <none>
```

ソフトウェアRAIDもしくはmdデバイスを作成しない限り、ファイルは上記の状態のままです。その場合は、/proc/mdstatを使用して、md*X* RAID デバイスでの現在の状況を見ることが出来ます。

/proc/mdstatファイルは、以下のように、システムにRAID 1デバイスとして構成されたmd0を示します。ファイルは、ディスクを現在再同期化しています：

```
Personalities : [linear] [raid1]
read_ahead 1024 sectors
md0: active raid1 sda2[1] sdb2[0] 9940 blocks [2/2] [UU] resync=1% finish=12.3min
algorithm 2 [3/3] [UUU]
unused devices: <none>
```

5.2.19. /proc/meminfo

これは、よく使われる/procディレクトリ内のファイルの1つです。システム上の現在のRAM使用率についての大量の貴重な情報をレポートします。

次の例のような/proc/meminfo仮想ファイルは256MバイトのRAMと384Mバイトのスワップ領域を持つシステムからの例です：

```
   total:  used:  free: shared: buffers: cached:
Mem: 261709824 253407232 8302592    0 120745984 48689152
Swap: 402997248   8192 402989056
MemTotal:   255576 kB
MemFree:     8108 kB
MemShared:      0 kB
Buffers:    117916 kB
Cached:     47548 kB
Active:    135300 kB
Inact_dirty:   29276 kB
Inact_clean:    888 kB
```

```
Inact_target:    0 kB
HighTotal:       0 kB
HighFree:        0 kB
LowTotal:   255576 kB
LowFree:     8108 kB
SwapTotal:   393552 kB
SwapFree:    393544 kB
```

この情報の多くは、free、top、psコマンドで使用します。実際、freeコマンドの出力は/proc/meminfoの内容と構造によく似ています。/proc/meminfoを見ると、メモリの詳細がわかります：

- Mem — システム内の物理RAMの現在の状況を表示します。項目別メモリ総容量、使用メモリ、空きメモリ、共有メモリ、バッファメモリ、キャッシュメモリの使用率をバイト単位で表示します。
- Swap — スワップ領域の総容量、使用容量、空き容量をバイト単位で表示します。
- MemTotal — 物理RAMの総容量。Kバイト単位。
- MemFree — 物理RAMの空き容量。Kバイト単位。
- MemShared — 2.4以降のカーネルでは使用しませんが、以前のカーネルバージョンの互換性のためにあります。
- Buffers — ファイルバッファに使用する物理RAMの容量。Kバイト単位。
- Cached — キャッシュメモリとして使用される物理RAMの容量。Kバイト単位。
- Active — 実際に使用しているバッファメモリか、ページキャッシュメモリの容量。Kバイト単位。
- Inact_dirty — 空きとして利用できる可能性のあるバッファメモリか、キャッシュページメモリの総容量。Kバイト単位。
- Inact_clean — 利用できるバッファかキャッシュページの空き容量合計。Kバイト単位。
- Inact_target — 1秒当たり正味割り当て量。Kバイト単位。1分平均値。
- HighTotalとHighFree — カーネル領域に直接マッピングされないメモリの総容量と空き容量。HighTotalの値は、使用するカーネルタイプによって異なります。Kバイト単位。
- LowTotalとLowFree — カーネル領域に直接マッピングされるメモリの総容量と空き容量。LowTotalの値は使用カーネルタイプによって異なります。Kバイト単位。
- SwapTotal — 利用可能なスワップ総容量。Kバイト単位。
- SwapFree — 空きスワップの総容量。Kバイト単位。

5.2.20. /proc/misc

このファイルは、デバイス番号10のその他のメジャーデバイス上に登録されているその他のドライバを一覧表示します：

```
135 rtc
  1 psaux
134 apm_bios
```

最初の列は各デバイスのマイナー番号で、2番目の列は使用中のドライバです。

5.2.21. /proc/modules

ファイルにはシステムがカーネルにロードしたすべてのモジュールの一覧が表示されます。その内容はシステムの設定と用途によって異なりますが、次の/proc/modulesファイル出力の例と似たような構成になっています：

```
ide-cd       27008  0 (autoclean)
cdrom        28960  0 (autoclean) [ide-cd]
soundcore       4100  0 (autoclean)
agpgart      31072  0 (unused)
binfmt_misc     5956  1
iscsi        32672  0 (unused)
scsi_mod       94424  1 [iscsi]
autofs       10628  0 (autoclean) (unused)
tulip        48608  1
ext3        60352  2
jbd         39192  2 [ext3]
```

最初の列はモジュール名です。2番目の列は、モジュールのメモリサイズをバイト単位で表します。3番目の列は、モジュールが現在ロードされているか（1）、ロードされていないか（0）を示します。最後の列は、使用されていない一定期間の後、モジュールが自動的にアンロードされるか（autoclean）使用されていないか（unused）を示します。名前がかっこ（[または]）内に入っているモジュールは、このモジュールが機能するには別のモジュールが必要であることを意味します。

5.2.22. /proc/mounts

このファイルはシステムが使用しているすべてのマウントの一覧を提供します：

```
rootfs / rootfs rw 0 0
/dev/hda2 / ext3 rw 0 0
/proc /proc proc rw 0 0
/dev/hda1 /boot ext3 rw 0 0
none /dev/pts devpts rw 0 0
none /dev/shm tmpfs rw 0 0
none /proc/sys/fs/binfmt_misc binfmt_misc rw 0 0
```

この出力は/etc/mtabの内容と似ていますが、/proc/mountが最新のものである点が異なります。

最初の列はマウントされているデバイスを示します。2番目の列はマウントポイントを表します。3番目の列はファイルシステムタイプを示します。4番目の列は、読み取り専用でマウントされているか（ro）、読み取り/書き込み（rw）モードでマウントされているかを示します。5番目と6番目の列は、/etc/mtabで使用されるフォーマットと一致させるためのダミー値です。

5.2.23. /proc/mtrr

このファイルは現在システムで使用しているMTRR (Memory Type Range Registers)です。システムのアーキテクチャがMTRRをサポートしている場合は、/proc/mtrr ファイルは、以下に似た表示となります：

```
reg00: base=0x00000000 (  0MB), size= 64MB: write-back, count=1
```

MTRRはIntel P6プロセッサファミリ（Pentium II 以上）で使用します。これは、プロセッサのメモリ範囲へのアクセスを制御するために使用します。PCIバスかAGPバス上のビデオカードを使用する場合、/proc/mtrrファイルを正しく設定すると、パフォーマンスは150%以上向上します。

ほとんどの場合、この値はデフォルトで正しく設定されています。手動でこのファイルを設定する方法については、オンラインでURL: http://web1.linuxhq.com/kernel/v2.3/doc/mtrr.txt.htmlを参照してください。

5.2.24. /proc/partitions

ここに述べる情報の多くは大部分のユーザーにとってはあまり重要ではありません。ただし、次の列は重要です：

- major — このパーティションを持つデバイスのメジャー番号。現在の例のメジャー番号（3）は/proc/devicesのide0デバイスに対応します。
- minor — このパーティションを持つデバイスのマイナー番号。これにより、パーティションを異なる物理デバイスに分け、パーティション名末尾の番号に関連します。
- #blocks — 特定のパーティションに入っている物理ディスクブロックの番号の一覧を表示します。
- name — パーティションの名前。

5.2.25. /proc/pci

このファイルには、システム上の各PCIデバイスの完全な一覧が保存されています。PCIデバイス数によって、/proc/pciはかなり長くなることがあります。基本的なシステムでは次の例に似た表示になります：

```
Bus 0, device  0, function 0:
 Host bridge: Intel Corporation 440BX/ZX - 82443BX/ZX Host bridge (rev 3).
  Master Capable. Latency=64.
  Prefetchable 32 bit memory at 0xe4000000 [0xe7ffffff].
Bus 0, device  1, function 0:
 PCI bridge: Intel Corporation 440BX/ZX - 82443BX/ZX AGP bridge (rev 3).
  Master Capable. Latency=64. Min Gnt=128.
Bus 0, device  4, function 0:
 ISA bridge: Intel Corporation 82371AB PIIX4 ISA (rev 2).
Bus 0, device  4, function 1:
 IDE interface: Intel Corporation 82371AB PIIX4 IDE (rev 1).
  Master Capable. Latency=32.
  I/O at 0xd800 [0xd80f].
Bus 0, device  4, function 2:
 USB Controller: Intel Corporation 82371AB PIIX4 USB (rev 1).
  IRQ 5.
  Master Capable. Latency=32.
  I/O at 0xd400 [0xd41f].
Bus 0, device  4, function 3:
 Bridge: Intel Corporation 82371AB PIIX4 ACPI (rev 2).
  IRQ 9.
Bus 0, device  9, function 0:
 Ethernet controller: Lite-On Communications Inc LNE100TX (rev 33).
  IRQ 5.
  Master Capable. Latency=32.
  I/O at 0xd000 [0xd0ff].
  Non-prefetchable 32 bit memory at 0xe3000000 [0xe30000ff].
Bus 0, device 12, function 0:
 VGA compatible controller: S3 Inc. ViRGE/DX or /GX (rev 1).
  IRQ 11.
  Master Capable. Latency=32. Min Gnt=4.Max Lat=255.
```

```
Non-prefetchable 32 bit memory at 0xdc000000 [0xdfffffff].
```

この出力はすべてのPCIデバイスをバス、デバイス、機能の順にソートした一覧を表しています。デバイスの名前とバージョンのほか、詳細なIRQ情報も得られるので、コンフリクトをすぐに検出できます。

ヒント

より読みやすい情報は次のように入力して得られます：

```
/sbin/lspci -vb
```

5.2.26. /proc/slabinfo

このファイルは、スラブレベルのメモリ使用率についての情報を提供します。2.2以降のLinuxカーネルは、スラブプールを使用してページレベル以上のメモリを管理します。よく使用されるオブジェクトには独自のスラブプールがあります。以下に典型的な/proc/slabinfo 仮想ファイルの一部を表示します：

```
slabinfo - version: 1.1
kmem_cache     64   68  112  2  2  1
nfs_write_data    0   0  384  0  0  1
nfs_read_data     0  160  384  0 16  1
nfs_page       0  200   96  0  5  1
ip_fib_hash     10  113   32  1  1  1
journal_head    51 7020   48  2 90  1
revoke_table     2  253   12  1  1  1
revoke_record    0   0   32  0  0  1
clip_arp_cache    0   0  128  0  0  1
ip_mrt_cache     0   0   96  0  0  1
```

このファイルの値は、キャッシュ名、アクティブなオブジェクト数、オブジェクト総数、オブジェクトサイズ、オブジェクトのアクティブなスラブ（ブロック）数、オブジェクトのスラブ総数、1スラブ当たりのページ数の順序になっています。

この場合、アクティブという言葉は使用中という意味です。

5.2.27. /proc/stat

このファイルには、システムが最後に再起動されたとき以降のシステムについてのさまざまな統計情報が記録されています。/proc/statの内容はきわめて長くなることがありますが、開始は次のようになります：

```
cpu 1139111 3689 234449 84378914
cpu0 1139111 3689 234449 84378914
page 2675248 8567956
swap 10022 19226
intr 93326523 85756163 174412 0 3 3 0 6 0 1 0 428620 0 60330 0 1368304 5538681
disk_io: (3,0):(1408049,445601,5349480,962448,17135856)
ctxt 27269477
btime 886490134
processes 206458
```

よく使用される統計情報は次のとおりです：

- cpu — cpu —システムがユーザーモード、低いプライオリティのユーザーモード（nice）、システムモード、アイドルタスクの状態にあった時間をそれぞれジフィー（1秒の1/100）単位で測定します。すべてのCPUの合計は最上部に表示され、各CPUはその下に個々の統計情報付きで表示されます。
- page — システムがディスクに書き込んだあるいは書き出したメモリページ数。
- swap — システムが入出力したスワップページ数。
- intr — システムへの割り込み数。
- btime — 1970年1月1日以降の起動時間。秒単位。エポック とも言います。

5.2.28. /proc/swaps

このファイルはスワップ領域とその使用率を測定します。スワップパーティションが1つのシステムの場合、/proc/swapの出力は次のようになります：

```
Filename  Type   Size  Used Priority
/dev/hda6  partition 136512 20024 -1
```

この情報の一部は他の/procファイルにもありますが、/proc/swapファイルは、各スワップファイル名のスナップショット、スワップ領域タイプ、合計サイズ、使用サイズ（Kバイト単位）などを提供します。優先列(priority column)は複数のスワップファイルを使用する場合に便利です。優先度が低いほど、そのスワップファイルがより多く使用される可能性が高くなります。

5.2.29. /proc/uptime

このファイルには、システムを最後に再起動してから経過した時間についての情報が保存されています。/proc/uptimeの出力は次のように短いものです：

```
350735.47 234388.90
```

最初の数字は、システムが起動されている時間の総数を秒単位で表しています。2番目の数字はその時間のうちのアイドル時間を秒単位で表しています。

5.2.30. /proc/version

このファイルは使用中のLinuxカーネルとgccのバージョン、及びシステムにインストールされているRed Hat Linuxのバージョンを表示します：

```
Linux version 2.4.20-0.40 (user@foo.redhat.com)(gcc version 3.2.1 20021125
(Red Hat Linux 8.0 3.2.1-1)) #1 Tue Dec 3 20:50:18 EST 2002
```

この情報はユーザーがログインした時のバージョンデータなど、さまざまな目的に使用できます。

5.3. /proc/のディレクトリ

カーネルに関するよく使用される情報のグループは/procディレクトリ内のディレクトリとサブディレクトリにグループ分けされます。

5.3.1. プロセスディレクトリ

各/procディレクトリには番号名の付いた多くのディレクトリがあります。この一覧の開始は次のようになります：

```
dr-xr-xr-x  3 root  root     0 Feb 13 01:28 1
dr-xr-xr-x  3 root  root     0 Feb 13 01:28 1010
dr-xr-xr-x  3 xfs   xfs      0 Feb 13 01:28 1087
dr-xr-xr-x  3 daemon daemon    0 Feb 13 01:28 1123
dr-xr-xr-x  3 root  root     0 Feb 13 01:28 11307
dr-xr-xr-x  3 apache apache    0 Feb 13 01:28 13660
dr-xr-xr-x  3 rpc   rpc      0 Feb 13 01:28 637
dr-xr-xr-x  3 rpcuser rpcuser    0 Feb 13 01:28 666
```

これらのディレクトリは、プロセスのIDを示し、そのプロセス固有の情報を保存しているので、プロセスディレクトリと呼ばれます。各プロセスディレクトリの所有者とグループはプロセスを実行しているユーザーに設定されます。プロセスが終了すると、その/procプロセスディレクトリは消えます。

各プロセスディレクトリには次のファイルがあります:

- cmdline — プロセスを開始する時に発行されるコマンドを含みます。
- cpu — システムの各CPUの使用率についての固有情報を提供します。デュアルCPUシステム上で実行されているプロセスの出力は次のようになります：
  ```
  cpu 11 3
  cpu0 0 0
  cpu1 11 3
  ```
- cwd — そのプロセスで現在動作しているディレクトリへのシンボリックリンク。
- environ — プロセスの環境変数の一覧を提供します。環境変数はすべて大文字で値は小文字です。
- exe — このプロセスの実行可能ファイルへのシンボリックリンク。
- fd — 特定プロセスのファイル記述子すべてが保存されているディレクトリ。これらは番号の付けられたリンクで提示されます：
  ```
  total 0
  lrwx------  1 root  root     64 May  8 11:31 0 -> /dev/null
  lrwx------  1 root  root     64 May  8 11:31 1 -> /dev/null
  lrwx------  1 root  root     64 May  8 11:31 2 -> /dev/null
  lrwx------  1 root  root     64 May  8 11:31 3 -> /dev/ptmx
  lrwx------  1 root  root     64 May  8 11:31 4 -> socket:[7774817]
  lrwx------  1 root  root     64 May  8 11:31 5 -> /dev/ptmx
  lrwx------  1 root  root     64 May  8 11:31 6 -> socket:[7774829]
  lrwx------  1 root  root     64 May  8 11:31 7 -> /dev/ptmx
  ```
- maps — このプロセスに関連するさまざまな実行可能ファイルとライブラリファイルへのメモリマップが保存されています。このファイルはプロセスの複雑度によってかなり長くなる場合があります。sshdプロセスのサンプル出力は次のように開始されます：
  ```
  08048000-08086000 r-xp 00000000 03:03 391479  /usr/sbin/sshd
  08086000-08088000 rw-p 0003e000 03:03 391479  /usr/sbin/sshd
  08088000-08095000 rwxp 00000000 00:00 0
  40000000-40013000 r-xp 00000000 03:03 293205  /lib/ld-2.2.5.so
  40013000-40014000 rw-p 00013000 03:03 293205  /lib/ld-2.2.5.so
  40031000-40038000 r-xp 00000000 03:03 293282  /lib/libpam.so.0.75
  40038000-40039000 rw-p 00006000 03:03 293282  /lib/libpam.so.0.75
  40039000-4003a000 rw-p 00000000 00:00 0
  4003a000-4003c000 r-xp 00000000 03:03 293218  /lib/libdl-2.2.5.so
  4003c000-4003d000 rw-p 00001000 03:03 293218  /lib/libdl-2.2.5.so
  ```

- mem — プロセスが保持しているメモリ。このファイルはユーザーから読み取れません。
- root — プロセスのルートディレクトリへのリンク。
- stat — プロセスのステータス。
- statm — プロセスが使用しているメモリのステータス。/proc/statmファイルの例は次のとおりです：

  ```
  263 210 210 5 0 205 0
  ```

  7つの列はプロセスの異なるメモリ統計情報に関連しています。表示されている順に、左から右に使用メモリの異なる側面をレポートしています：

  1. プログラム合計サイズ。Kバイト単位。
  2. メモリ部分のサイズ。Kバイト単位。
  3. 共有ページ数。
  4. コードのページ数。
  5. データ/スタックのページ数。
  6. ライブラリのページ数。
  7. ダーティなページ数。

- status — statやstatmよりはるかに読みやすい形式でプロセスのステータスを提供します。sshdの出力例は次のとおりです:

  ```
  Name: sshd
  State: S (sleeping)
  Tgid: 797
  Pid: 797
  PPid: 1
  TracerPid: 0
  Uid: 0 0 0 0
  Gid: 0 0 0 0
  FDSize: 32
  Groups:
  VmSize:  3072 kB
  VmLck:     0 kB
  VmRSS:   840 kB
  VmData:   104 kB
  VmStk:    12 kB
  VmExe:   300 kB
  VmLib:  2528 kB
  SigPnd: 0000000000000000
  SigBlk: 0000000000000000
  SigIgn: 8000000000001000
  SigCgt: 0000000000014005
  CapInh: 0000000000000000
  CapPrm: 00000000fffffeff
  CapEff: 00000000fffffeff
  ```

  この出力の中の情報には、プロセスの名前とIDのほか、ステータス（S (sleeping)、R(running)、プロセスなどを実行しているユーザー/グループID、メモリ使用率に関する詳細が含まれています。

5.3.1.1. /proc/self/

/proc/self/ディレクトリは現在実行中のプロセスへのリンクです。これにより、プロセスは、自身のプロセスIDを知らなくても状況を把握できます。

シェル環境では、/proc/self/ディレクトリの一覧は、そのプロセス用のプロセスディレクトリの一覧と同じ内容を生成します。

5.3.2. /proc/bus/

このディレクトリには、システムで利用できるさまざまなバス固有情報が保存されています。したがって、たとえばISA、PCI、USBバスを搭載した標準的なシステムでは、各バスについての現在のデータが/proc/bus/のディレクトリで入手できます。

利用できるサブディレクトリとファイルの内容は、システムの厳密な設定によって大幅に異なります。ただし、各バスタイプの各ディレクトリには、そのタイプの各バスにつき少なくとも1つのディレクトリがあります。これらの個々のバスディレクトリは通常、00 などの数字で表し、そこにはそのバスで使用できるさまざまなデバイスに関するバイナリファイルがあります。

したがって、たとえば、USBバスを搭載しているが、USBデバイスは接続していないシステムには、いくつかのファイルの入った/proc/bus/usbディレクトリがあります：

```
total 0
dr-xr-xr-x  1 root  root     0 May  3 16:25 001
-r--r--r--  1 root  root     0 May  3 16:25 devices
-r--r--r--  1 root  root     0 May  3 16:25 drivers
```

/proc/bus/usbディレクトリには、USBバス上のさまざまなデバイスを追跡するファイルとそれを使用する必要のあるドライバがあります。/proc/bus/usb/001ディレクトリには最初のUSBバス上のすべてのデバイスがあります。devicesファイルの内容を見ると、これはマザーボードのUSBルートハブであることがわかります：

```
T:  Bus=01 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=12  MxCh= 2
B:  Alloc=  0/900 us ( 0%), #Int=  0, #Iso=  0
D:  Ver= 1.00 Cls=09(hub  ) Sub=00 Prot=00 MxPS= 8 #Cfgs=  1
P:  Vendor=0000 ProdID=0000 Rev= 0.00
S:  Product=USB UHCI Root Hub
S:  SerialNumber=d400
C:* #Ifs= 1 Cfg#= 1 Atr=40 MxPwr=  0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub  ) Sub=00 Prot=00 Driver=hub
E:  Ad=81(I) Atr=03(Int.) MxPS=   8 Ivl=255ms
```

5.3.3. /proc/driver/

このディレクトリにはカーネルが使用する特定のドライバについての情報があります。

一般的にここにあるファイルは、rtcで、システムのリアルタイムクロック（*RTC*）用のドライバからの出力を提供します。このRTCというデバイスは、システムがオフになっている間、時計を動かしています。/proc/driver/rtcの出力例は次のとおりです：

```
rtc_time : 01:38:43
rtc_date : 1998-02-13
rtc_epoch : 1900
alarm : 00:00:00
DST_enable : no
BCD : yes
```

```
24hr ：yes
square_wave：no
alarm_IRQ：no
update_IRQ：no
periodic_IRQ：no
periodic_freq：1024
batt_status：okay
```

RTCについての詳細情報は、/usr/src/linux-2.4/Documentation/rtc.txtを参照してください。

5.3.4. /proc/fs

このディレクトリは、エクスポートされるファイルシステムを表示します。NFSサーバを稼働している場合、cat /proc/fs/nfs/exportsとタイプすると、共有されているファイルシステムとそれらのファイルシステムへの権限を表示します。ファイルシステムの共有に関する詳細は第9章で御覧下さい。

5.3.5. /proc/ide/

このディレクトリにはシステム上のIDEデバイスについての情報があります。各IDEチャンネルは、/proc/ide/ide0や/proc/ide/ide1などの個別ディレクトリとして表されます。さらに、driversファイルも利用できます。このファイルは、IDEチャンネル上で使用するさまざまなドライバのバージョン番号を提供します：

```
ide-cdrom version 4.59
ide-floppy version 0.97
ide-disk version 1.10
```

多くのチップセットはまた、さなざまなチャンネルを介して接続されているドライブに関する追加情報をこのディレクトリ内に提供します。たとえば、汎用のIntel PIIX4 Ultra 33チップセットは、/proc/ide/piixファイルを生成しますが、これにより、IDEチャンネル上のデバイス用にDMAか又はUDMAが有効かどうかわかります：

```
                Intel PIIX4 Ultra 33 Chipset.
------------- Primary Channel ---------------- Secondary Channel -------------
        enabled            enabled
------------- drive0 --------- drive1 -------- drive0 ---------- drive1 ------
DMA enabled:  yes       no       yes       no
UDMA enabled: yes       no       no        no
UDMA enabled: 2         X        X         X
UDMA
DMA
PIO
```

ide0など、IDEチャンネルのディレクトリをナビゲーションするとさらに情報が得られます。channelファイルでチャンネル番号が、modelファイルでチャンネルのバスタイプ（pciなど）がわかります。

5.3.5.1. デバイスディレクトリ

各IDEチャンネルディレクトリ内は、デバイスディレクトリがあります。デバイスディレクトリ名は、/devディレクトリ内のドライブ文字に対応しています。例えば、ide0上の最初のIDEドライブは、hdaと名付けられます。

注意

/proc/ide/ディレクトリには、これら各デバイスディレクトリのシンボリックリンクがあります。

各デバイスディレクトリには、一連の情報と統計値があります。これらディレクトリの内容は、接続されているデバイスのタイプにより変わります。多くのデバイスに共通して役に立つファイルには、次のようなものがあります：

- cache — デバイスのキャッシュ。
- capacity — デバイスの容量。512バイトブロック単位。
- driver — デバイスを制御するために使用するドライバとバージョン。
- geometry — デバイスの物理的、かつ論理的ジオメトリ。
- media — diskなどのデバイスのタイプ。
- model — デバイスのモデル名か番号。
- デバイスの現在の一連のパラメータ。このファイルには通常、非常に多くの役に立つ技術情報が保存されています。標準的なIDEハードディスクのsettingsファイル例は次のとおりです：

```
name       value    min    max    mode
----       -----    ---    ---    ----
bios_cyl     784     0     65535   rw
bios_head     255     0     255    rw
bios_sect     63      0     63     rw
breada_readahead  4      0     127    rw
bswap      0      0     1      r
current_speed   66      0     69     rw
file_readahead   0      0     2097151   rw
ide_scsi     0      0     1      rw
init_speed    66      0     69     rw
io_32bit     0      0     3      rw
keepsettings    0      0     1      rw
lun        0      0     7      rw
max_kb_per_request 64      1     127    rw
multcount     8      0     8      rw
nice1       1      0     1      rw
nowerr       0      0     1      rw
number       0      0     3      rw
pio_mode     write-only  0     255    w
slow       0      0     1      rw
unmaskirq     0      0     1      rw
using_dma     1      0     1      rw
```

5.3.6. /proc/irq/

このディレクトリを使用してIRQをCPUアフィニティに設定すると、特定のIRQを1つのCPUにのみ接続できます。また、別の方法としてCPUがどのIRQも処理しないように設定することもできます。

各IRQには独自のディレクトリがあり、各IRQを異なる設定にできます。/proc/irq/prof_cpu_maskファイルはIRQディレクトリのsmp_affinityファイルのデフォルト値を保存したビットマスクです。smp_affinityの値で、特定のIRQを処理するCPUを指定します。

/proc/irq/に関する詳細情報は以下のファイルを参照して下さい：

```
/usr/src/linux-2.4/Documentation/filesystems/proc.txt
```

5.3.7. /proc/net/

このディレクトリでは、さまざまなネットワークのパラメータと統計情報を包括的に表示します。各ファイルには、システムのネットワーキングに関連する情報の特定範囲が保存されています。仮想ファイルの一部は以下のようになります：

- arp — カーネルのARPテーブルが保存されています。このファイルはハードウェアアドレスをシステム上のIPアドレスに接続する際、特に便利です。
- atm — さまざまな非同期転送モード（*ATM*）の設定と統計情報の入ったファイルのあるディレクトリ。このディレクトリはATMネットワーキングとADSLカードでおもに使用します。
- dev — システム上に設定されているさまざまなネットワークデバイスと送受信の統計情報の一覧を表示します。このファイルで、各インターフェイスが送受信したバイト数、入出力パケット数、表示エラー数、損失パケット数などがわかります。
- dev_mcast — 各デバイスがリスニングしている多くのレイヤ2マルチキャストグループを表示します。
- igmp — このシステムが参加しているIPマルチキャストアドレスの一覧を表示します。
- ip_fwchains — ipchainsが使用されている場合、この仮想フィルは現在の規則を表示します。
- ip_fwnames — ipchainsが使われている場合、すべてのファイアウォールチェーン名の一覧を表示します。
- ip_masquerade — ipchainsの下で隠蔽情報のテーブルを提供します。
- ip_mr_cache — マルチキャストルーティングキャッシュの一覧。
- ip_mr_vif — マルチキャスト仮想インターフェイスの一覧。
- netstat — TCPタイムアウト、送受信済みSYNクッキーなどの広範で詳細なネットワーキング統計情報が含まれています。
- psched — グローバルパケットスケジューラパラメータの一覧。
- raw — 生のデバイス統計情報の一覧。
- route — カーネルのルーティングテーブルを表示します。
- rt_cache — 現在のルーティングキャッシュが保存されています。
- snmp — 使用中の各種ネットワーキングプロトコルのSNMP(Simple Network Management Protocol) データの一覧。
- sockstat — ソケット統計情報を提供します。
- tcp — 詳細なTCPソケット情報が保存されています。

- tr_rif — トークンリングRIFルーティングテーブル。
- udp — 詳細なUDPソケット情報が保存されています。
- unix — 現在使用されているUNIXドメインソケットの一覧を表示します。
- wireless — ワイヤレスインターフェイスデータの一覧を表示します。

5.3.8. /proc/scsi/

ディレクトリは、/proc/ide/ディレクトリと同様ですが、これは、SCSIデバイス接続専用のディレクトリです。

このディレクトリの主要なファイルは、/proc/scsi/scsiです。ここには、認識されたSCSIデバイスすべての一覧が保存されます。この一覧からデバイスのタイプ、それと共にモデル名、ベンダー、SCSIチャンネルとIDデータが利用できます。

たとえば、システムにSCSI CD-ROM、テープドライブ、ハードディスクドライブ、RAIDコントローラがある場合、このファイルは次のようになります：

```
Attached devices:
Host: scsi1 Channel: 00 Id: 05 Lun: 00
 Vendor: NEC   Model: CD-ROM DRIVE:466 Rev: 1.06
 Type:  CD-ROM            ANSI SCSI revision: 02
Host: scsi1 Channel: 00 Id: 06 Lun: 00
 Vendor: ARCHIVE Model: Python 04106-XXX Rev: 7350
 Type:  Sequential-Access       ANSI SCSI revision: 02
Host: scsi2 Channel: 00 Id: 06 Lun: 00
 Vendor: DELL   Model: 1x6 U2W SCSI BP  Rev: 5.35
 Type:  Processor           ANSI SCSI revision: 02
Host: scsi2 Channel: 02 Id: 00 Lun: 00
 Vendor: MegaRAID Model: LD0 RAID5 34556R Rev: 1.01
 Type:  Direct-Access         ANSI SCSI revision: 02
```

さらに、システムが使用する各SCSIドライバには、/proc/scsiに独自のディレクトリがあります。ここには、そのドライバを使用する各SCSIコントローラ固有のファイルがあります。したがって、たとえば上記のシステムでは、aic7xxxとmegaraidの2つのドライバが使用されているので、これらのディレクトリが存在します。各ディレクトリ内のファイルには通常IOアドレス範囲、IRQ、そのドライバを使用する特定のSCSIコントローラの統計情報が保存されています。各コントローラがレポートする情報のタイプと量は異なりますが。この例示システムで使用しているAdaptec AIC-7880 Ultra SCSIホストアダプタのファイルは、次のような出力を生成します：

```
Adaptec AIC7xxx driver version: 5.1.20/3.2.4
Compile Options:
 TCQ Enabled By Default : Disabled
 AIC7XXX_PROC_STATS   : Enabled
 AIC7XXX_RESET_DELAY  : 5

Adapter Configuration:
     SCSI Adapter: Adaptec AIC-7880 Ultra SCSI host adapter
             Ultra Narrow Controller
  PCI MMAPed I/O Base: 0xfcffe000
 Adapter SEEPROM Config: SEEPROM found and used.
   Adaptec SCSI BIOS: Enabled
         IRQ: 30
         SCBs: Active 0, Max Active 1,
            Allocated 15, HW 16, Page 255
      Interrupts: 33726
   BIOS Control Word: 0x18a6
```

```
  Adapter Control Word: 0x1c5f
  Extended Translation: Enabled
Disconnect Enable Flags: 0x00ff
   Ultra Enable Flags: 0x0020
 Tag Queue Enable Flags: 0x0000
Ordered Queue Tag Flags: 0x0000
Default Tag Queue Depth: 8
   Tagged Queue By Device array for aic7xxx host instance 1:
    {255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255}
   Actual queue depth per device for aic7xxx host instance 1:
    {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}

Statistics:


(scsi1:0:5:0)
 Device using Narrow/Sync transfers at 20.0 MByte/sec, offset 15
 Transinfo settings: current(12/15/0/0), goal(12/15/0/0), user(12/15/0/0)
 Total transfers 0 (0 reads and 0 writes)
      < 2K   2K+   4K+   8K+  16K+  32K+  64K+ 128K+
 Reads:    0    0    0    0    0    0    0    0
 Writes:   0    0    0    0    0    0    0    0


(scsi1:0:6:0)
 Device using Narrow/Sync transfers at 10.0 MByte/sec, offset 15
 Transinfo settings: current(25/15/0/0), goal(12/15/0/0), user(12/15/0/0)
 Total transfers 132 (0 reads and 132 writes)
      < 2K   2K+   4K+   8K+  16K+  32K+  64K+ 128K+
 Reads:    0    0    0    0    0    0    0    0
 Writes:   0    0    0    1  131    0    0    0
```

この出力では、チャンネルIDに基づいてコントローラに接続されているさまざまなSCSIデバイスへの転送速度、デバイスが読み取り/書き込みするファイルの量とサイズについての詳細な統計情報がわかります。例えば、このコントローラは20Mビット/秒でCD-ROMと通信しており、テープドライブは10Mビット/秒で接続されていることがわかります。

5.3.9. /proc/sys/

/proc/sys/ディレクトリは/proc内の他のディレクトリとは違っています。というのは、システムに関する情報を提供するだけでなく、これによりシステム管理者は、カーネル設定をすぐに有効または無効にする事ができます。

警告

/proc/sysディレクトリ内のさまざまなファイルを使用している生産システム上の設定を変更する際は注意してください。誤った設定変更によりカーネルが不安定になり、システムを再起動しなければならなくなる可能性があります。

従って、/proc/sys内の値を変更する前に、そのファイルの有効なオプションと予想される結果について必ず確認してください。

特定のファイルが設定可能なのか単に情報提供するだけなのかを判断するには、シェルプロンプトで-lを付けて一覧を表示させます。ファイルが書き込み可能であれば、それを使用して一定の方法で

カーネルを設定することができます。たとえば、/proc/sys/fsの一覧（一部）は以下の例のような表示となります：

```
-r--r--r--  1 root  root     0 May 10 16:14 dentry-state
-rw-r--r--  1 root  root     0 May 10 16:14 dir-notify-enable
-r--r--r--  1 root  root     0 May 10 16:14 dquot-nr
-rw-r--r--  1 root  root     0 May 10 16:14 file-max
-r--r--r--  1 root  root     0 May 10 16:14 file-nr
```

この一覧を見ると、dir-notify-enableファイルとfile-maxファイルは書き込み可能なので、カーネル設定に使用することができます。他のファイルは現在の設定に関する情報を提供するだけです。

/proc/sys/ファイル内の値を変更するには、新しい値をファイルにエコーします。たとえば、実行中のカーネル上のシステム要求キーを使用可能にするには、以下のコマンドを入力します：

```
echo 1 > /proc/sys/kernel/sysrq
```

これでsysrqファイルの値は0 (off)から1 (on)に変更されます。

システム要求キーの目的は、単純なキーコンビネーションを使用しすばやくカーネルにインプットすることです。例えば、すぐにシステムをシャットダウンする/再起動する、マウントした全ファイルシステムを同期化する、重要な情報を自分のコンソールにダンプする、などのためにシステム要求キーを使用することが出来ます。この機能は、開発カーネルの使用中やシステムがフリーズした場合に重宝します。しかし、監視されていないコンソールにはセキュリティのリスクがありますので、Red Hat Linuxのデフォルトではこの機能は停止されています。

システム要求キーの詳細については、/usr/src/linux-2.4/Documentation/sysrq.txt を御覧下さい。

幾つかの/proc/sys/設定ファイルには複数の値が含まれていることがあります。その場合、ファイルに新しい値を正確に送るために、下記の例のように、echoコマンドで渡す値の間にスペースを1個挿入します：

```
echo 4 2 45 > /proc/sys/kernel/acct
```

注意

echoを用いて行った設定変更は、システムの再起動時に失われます。設定変更をシステムのブート時にも有効にするには、項5.4を参照してください。

/proc/sys/ディレクトリには、実行カーネルの異なる側面を制御するそれぞれ異なったサブディレクトリが含まれています。

5.3.9.1. /proc/sys/dev/

このディレクトリはシステム上の特定のデバイス用パラメータを提供します。ほとんどのシステムには少なくとも2つのディレクトリ、cdromとraidがありますが、カスタマイズされたカーネルはそれ以外に複数のデバイスドライバ間で1個のパラレルポートを共有できるようにするparportなどのディレクトリを持つことができます。

cdromディレクトリにはinfoと呼ばれるファイルがあります。多くの重要なCD-ROMパラメータを提示します：

```
CD-ROM information, Id: cdrom.c 3.12 2000/10/18
```

```
drive name: hdc
drive speed: 32
drive # of slots: 1
Can close tray: 1
Can open tray: 1
Can lock tray: 1
Can change speed: 1
Can select disk: 0
Can read multisession: 1
Can read MCN: 1
Reports media changed: 1
Can play audio: 1
Can write CD-R: 0
Can write CD-RW: 0
Can read DVD: 0
Can write DVD-R: 0
Can write DVD-RAM: 0
```

このファイルを一瞥すると少なくともカーネルには未知のCD-ROMのクオリティを知ることができます。システム上で複数のCD-ROMが利用できる場合、各デバイスにはそれぞれの情報列が与えられます。

autocloseやcheckmediaなど、/proc/sys/dev/cdrom内のさまざまなファイルは、システムのCD-ROMを制御するために使用できます。これらの機能をオンまたはオフにするには、echoを用います。

RAIDサポートをカーネルにコンパイルした場合、/proc/sys/dev/raid/ディレクトリは少なくとも2つのファイルspeed_limit_minとspeed_limit_maxと共に利用できるようになります。こうした設定は、ディスクの再同期化など特に入出力の激しいタスクでRAIDデバイスが使用される場合の速度の調節に活用できます。

5.3.9.2. /proc/sys/fs/

このディレクトリには、quota、file handle、inode、dentry情報を含むファイルシステムに関するさまざまな側面について多くのオプションと情報が格納されています。

binfmt_miscディレクトリは、さまざまなバイナリフォーマットにカーネルサポートを提供するために使用されます。

/proc/sys/fs内の重要なファイルには、以下のようなものがあります：

- dentry-state — ディレクトリキャッシュのステータスを提供します。このファイルは以下のようになっています：

  ```
  57411 52939 45 0 0 0
  ```

  1番目の数はディレクトリキャッシュエントリの総数を示し、2番目の数は未使用エントリ数を示します。3番目の数はディレクトリの開放から再要求できるまでの秒数を示しています。4番目の数は現在システムが要求しているページ数です。最後の2つの数は未使用で、現在0のみを表示します。
- dquot-nr — キャッシュされたディスククォータ(割り当て)エントリの最大数を示します。
- file-max — カーネルが割り当てるファイルハンドルの最大数を変更することができます。このファイルの値を大きくすると、利用可能なファイルハンドルの不足によるエラーを解消することができます。
- file-nr — 割り当てられたファイルハンドル数、使用されたファイルハンドル数、ファイルハンドルの最大数を表示します。
- overflowgidとoverflowuid — 16ビットグループIDとユーザーIDをサポートするだけのファイルシステムと共に使用するため、それぞれ固定グループIDとユーザーIDを定義します。

- super-max — 利用可能なスーパーブロックの最大数を制御します。
- super-nr — 使用中のスーパーブロックの現在数を表示します。

5.3.9.3. /proc/sys/kernel/

このディレクトリにはカーネルの動作に直接影響するさまざまな異なる設定ファイルが格納されています。最も重要なファイルには以下のようなものがあります：

- acct — ログがあるファイルシステム上で利用可能な空き領域の割合に基づき、プロセスアカウンティングの休止を制御します。デフォルトでは、ファイルは以下のようになっています：
  ```
  4 2 30
  ```
  1番目の値はロギングのリジュームに必要な空き領域の割合を決定し、2番目の値はロギングがサスペンドした場合の空き領域のしきい値の割合を設定します。3番目の値はファイルシステムがロギングをサスペンドするかリジュームするかを確認するためにカーネルがポーリングする間隔を秒単位で設定します。
- cap-bound — ケーパビリティバウンディング設定を制御します。システム上の任意のプロセスが実行可能なケーパビリティの一覧を提供します。ここに表示されないケーパビリティについては、どのような特権が与えられていても、そのプロセスを実行することはできません。ブートプロセス時に少なくともあるポイントから以降は、特定の事項が生じないようにすることでシステムの安全をさらに確保する基本姿勢です。

  この仮想ファイルの値に関する有効な一覧は/usr/src/linux-2.4/include/linux/capability.hで御覧下さい。ケーパビリティバウンディングに付いての詳細は、以下のオンラインURLで確認して下さい：http://lwn.net/1999/1202/kernel.php3。
- ctrl-alt-del — initを使用し[Ctrl]-[Alt]-[Delete]キーでコンピュータをやさしく再起動する（値0）か、ダーティバッファを同期化せず直ちに強制的に再起動する（値1）かを制御します。
- domainname — example.comなどシステムのドメイン名を設定することができます。
- hostname — www.example.comなどシステムのホスト名を設定することができます。
- hotplug — システムが設定変更を検出した場合に使用するユーティリティを設定します。これは主としてUSBとカードバスPCIで使用されます。/sbin/hotplugのデフォルト値は、この役割を果たすために新しいプログラムをテストする場合をのぞいて変更してはいけません。
- modprobe — 必要に応じてカーネルモジュールをロードするために使用されるプログラムのロケーションを設定します。/sbin/modprobeのデフォルト値は、カーネルスレッドがkmodをコールする時に、実際にモジュールをロードするためにkmodがコールを行うことを示します。
- msgmax — プロセス間で送信されるメッセージの最大サイズを設定します。デフォルトは8192バイトです。プロセス間のキューメッセージはスワップできないカーネルメモリに格納されるので、この値を大きくする場合は注意が必要です。msgmaxが増大するとシステムに対するRAMの要求も増大することになります。
- msgmnb — 単一メッセージキューの最大バイト数を設定します。デフォルトは16384です。
- msgmni — メッセージキュー識別子の最大数を設定します。デフォルトは16です。
- osrelease — Linuxカーネルリリース番号を一覧表示します。このファイルを変更するにはカーネルソースを変更し再コンパイルするしかありません。
- ostype — オペレーティングシステムの種類を表示します。デフォルトでは、このファイルはLinuxに設定されています。この値を変更するにはカーネルソースを変更し再コンパイルするしかありません。
- overflowgid とoverflowuid — 16ビットのグループIDとユーザーIDしかサポートしないアーキテクチャ上でシステムコールと共に使用するために、それぞれ固定グループIDとユーザーIDを定義します。

- panic — カーネルパニックが生じたときカーネルがシステムの再起動を延期する秒数を定義します。デフォルトでは、パニック後に自動再起動しないよう0に設定されています。
- printk — このファイルは、印刷かロギングエラーメッセージに関するさまざまな設定を制御します。カーネルがレポートするエラーメッセージにはメッセージの重要度を定義するログレベルが含まれています。ログレベル値の意味は以下のよう順で分類されます :
  - 0 — カーネルエマージェンシー。システムを使用できません。
  - 1 — カーネル通報。直ちに何らかの対策を講じる必要があります。a
  - 2 — カーネルの状態が危機にあるとみなされます。
  - 3 — 一般カーネルエラー状況。
  - 4 — 一般カーネル警告状況。
  - 5 — 正常だが重大な状況にあるというカーネル通知
  - 6 — カーネル情報メッセージ。
  - 7 — カーネルデバッグレベルメッセージ。

  printkファイルには4つの値があります。
  6 4 1 7

  これらの値は、それぞれ異なるエラーメッセージ処理方法を定義します。1番目の値はコンソールログレベルと呼ばれ、コンソールに出力される優先度が最も低いメッセージを定義します（優先度が低いほどログレベル数が大きいことに注意）。2番目の値は、メッセージに添付される明確なログレベルがないデフォルトのログレベルを設定します。3番目の値はコンソールのログレベルでは最低限のログレベル設定をします。最後の値はコンソールログレベルのデフォルト値を設定します。
- rtsig-max — システムが1度にキューに入れるPOSIXリアルタイムシグナルの最大数を設定します。デフォルト値は1024です。
- rtsig-nr — カーネルがキューにしたPOSIXリアルタイムシグナルの現在数です。
- sem — このファイルはカーネル内のセマフォシグナルを設定します。セマフォは特定プロセスの利用を制御するために使用されるSystem V IPCオブジェクトです。
- shmall — システム上で1度に使用可能な共有メモリの合計（単位はバイト）を設定します。この値はデフォルトでは2097152に設定されています。
- shmmax — カーネルが許可する最大共有メモリセグメントのサイズ（単位はバイト）を設定します。この値はデフォルトでは33554432に設定されています。ただし、カーネルはこれ以上の値でもサポートします。
- shmmni — システム全体の共有メモリセグメントの最大数を設定します。(単位はバイト)。この値はデフォルトでは4096です。
- sysrq — デフォルトの0以外の値に設定されている場合、システム要求キーをアクティブにします。システム要求キーに関する詳細は項5.3.9 で御覧下さい。
- threads-max — カーネルが使用するスレッドの最大数を設定します。デフォルト値は2048です。
- version — カーネルが最後にコンパイルされた日付と時間を表示します。#3など、このファイルの最初にあるフィールドはカーネルがソースベースからコンパイルされた回数に関係しています。

The random ディレクトリには、カーネルのための乱数生成に関連した多数の値が格納されます。

5.3.9.4. /proc/sys/net/

このディレクトリには、ネットワーキングトピックに関係するさまざまなディレクトリが含まれています。カーネルのコンパイル時の構成により、appletalk、ethernet、ipv4、ipx、ipv6など、利用可能な異なるディレクトリが作成されます。これらのディレクトリ内で、管理者は実行中のシステム上のネットワーク設定を調節することができます。

Linuxでは多種多様なネットワーキングオプションを利用することができますが、最も一般的な/proc/sys/net/ディレクトリについてのみ説明します。

/proc/sys/net/core/ディレクトリには、カーネルとネットワークレイヤーとの相互作用を制御するさまざまな設定が含まれています。その中で最も重要なファイルは次のようになります：

- message_burst — 新しい警告メッセージを書き込むために必要な時間（10分の1秒単位）。これはDoS（Denial of Service）攻撃を防止するために使用されます。デフォルトでは50に設定されています。
- message_cost — 警告メッセージに費用を課すことでDoS攻撃を防止するために使用されます。このファイルの値（デフォルトは5）が大きいほど、警告メッセージは無視されることが多くなります。

  DoS攻撃の目的は、攻撃対称のシステムに大量の要求をかけて、エラーを発生させ、そのディスクパーティションをログファイルで満杯にするか、又はこのエラーログの処理にシステムリソースの全てを要求することです。message_burstとmessage_costの設定は、使用するシステムが受入可能なリスクとロギング全体のニーズに基づいて変更できるよう設計されています。
- netdev_max_backlog — 特定のインターフェイスがカーネル処理速度以上のパケットを受け取った場合、キューに入れることができるパケットの最大数を設定します。このファイルのデフォルト値は300です。
- optmem_max — ソケットごとに許可された補助バッファの最大サイズを設定します。
- rmem_default — 受け取りソケットバッファのデフォルトサイズ（単位はバイト）を設定します。
- rmem_max — 受け取りソケットバッファの最大サイズ（単位はバイト）を設定します。
- wmem_default — ソケットバッファ送信のデフォルトサイズ（単位はバイト）を設定します。
- wmem_max — ソケットバッファ送信の最大サイズ（単位はバイト）を設定します。

/proc/sys/net/ipv4/ ディレクトリには、追加ネットワーク設定が含まれます。こうした設定の多くを相互に関連させて使用すると、システムへの攻撃を防止するため、あるいは、ルーターとしてシステムを用いる場合に非常に役に立ちます。

用心

これらのファイルで変更にミスをするとシステムへのリモート接続に影響が生じる可能性があります。

/proc/sys/net/ipv4/ディレクトリの最も重要なファイルは、以下のようになっています：

- icmp_destunreach_rate, icmp_echoreply_rate, icmp_paramprob_rate 及びicmp_timeexeed_rate — 特定の条件でホストに対する最大ICMP送信パケットレート（100分の1秒単位）を設定します。0に設定すると遅延がなくなるので、お勧めできません。
- icmp_echo_ignore_all とicmp_echo_ignore_broadcasts — カーネルがあらゆるホストから来るものや、ブロードキャストアドレスとマルチキャストアドレスのみから生じるICMP ECHOパケットを無視できるようにします。カーネルは0でパケットに応答し、1で無視します。
- ip_default_ttl — デフォルトの*TTL*（*Time To Live*）を設定します。これはパケットが目的地に到着する前にホップする回数を制限します。この値を大きくするとシステムパフォーマンスが低下する可能性があります。
- ip_forward — システム上のインターフェイスがパケットを互いに転送することを許可します。デフォルトでは、0に設定されています。このファイルを1にセットすると、ネットワークパケットを転送できます。

- ip_local_port_range — ローカルポートが必要なときTCPかUDPで使用するポートの範囲を指定します。1番目の数は使用する最小ポートで、2番目の数は最大ポートを指定します。デフォルトの1024～4999より多いポートを必要とすることが予想されるシステムでは、このファイルで32768～61000の範囲にします。
- tcp_syn_retries — システムが接続時にSYNパケットを再伝送する回数を制限します。
- tcp_retries1 — 着信接続に応じることが許可された再伝送回数を設定します。デフォルトは3です。
- tcp_retries2 — 許可されたTCPパケットの再伝送回数を設定します。デフォルトは15です。

The /usr/src/linux-2.4/Documentation/networking/ip-sysctl.txt ファイルには、/proc/sys/net/ipv4/ディレクトリで利用できるファイルとオプションの総合的一覧があります。

/proc/sys/net/ipv4/ディレクトリ内にある他の多くのディレクトリは固有のトピックを扱います。/proc/sys/net/ipv4/conf/ディレクトリによって、各システムインターフェイスは各種の設定を行うことが可能になります。これには、(/proc/sys/net/ipv4/conf/default/サブディレクトリの) 未設定デバイスの為のデフォルト設定や(/proc/sys/net/ipv4/conf/all/サブディレクトリの)特別な設定をすべて上書きする設定が含まれます。

/proc/sys/net/ipv4/neigh/ディレクトリには、システムに直接接続されているホスト(隣接ネットワークと呼ぶ)との通信の為の設定があり、少々遠いシステムの為の別の設定も含まれています。

IPV4上のルーティングにも独自のディレクトリ/proc/sys/net/ipv4/route/があります。conf/やneigh/と異なり、/proc/sys/net/ipv4/route/ ディレクトリにはシステム上のインターフェイスにルーティングを適用する仕様が含まれています。max_size、max_delay、min_delayなど設定の多くはルーティングキャッシュのサイズの制御に関係しています。ルーティングキャッシュをクリアするには、flushファイルに任意の値を書き込むだけです。

これらのディレクトリや設定ファイルの値に関する詳細については、/usr/src/linux-2.4/Documentation/filesystems/proc.txtを参照して下さい。

5.3.9.5. /proc/sys/vm/

このディレクトリは、Linuxカーネルの仮想メモリ（VM）サブシステムの設定を援助します。カーネルは仮想メモリを広範囲かつインテリジェントに使用します。これは一般にスワップ領域と呼ばれます。

以下のファイルは、通常/proc/sys/vm/ディレクトリにあるものです：

- bdflush — bdflushカーネルデーモンに関係するさまざまな値を設定します。
- buffermem — バッファメモリに使用する全システムメモリの割合を制御することができます。このファイルの出力は通常、以下のようになっています：

  ```
  2   10   60
  ```

  最初と最後の値は、バッファメモリとして使用するメモリの最小と最大をそれぞれ設定します。中央の値はバッファメモリで使用するシステムメモリの割合を設定しますが、その場合メモリ管理サブシステムはフリーメモリの不足を埋め合わせるため他のメモリ以上にバッファキャッシュのクリアを開始します。
- kswapd — カーネルスワップアウトデーモンkswapdに関係するさまざまな値を設定します。このファイルには3つの値があります：

  ```
  512 32 8
  ```

  1番目の値はkswapdが1度でフリーにしようとするページの最大数を設定します。この値が大きいほど、カーネルは積極的にページをフリーにしようとします。2番目の値はkswapdがページをフリーにしようとする最小回数を設定します。3番目の値はkswapdが1度に書き込もうとするページ数を設定します。この最後の値を適正に調節すると、カーネルにページを大量に書き込みディスク

検索を最小限にするよう命じることで多くのスワップ領域を使用する、システム上のパフォーマンスを改善することができます。

- max_map_count — プロセスが持つメモリマップエリアの最大数を設定します。ほとんどの場合、デフォルト値として65536が適切です。
- overcommit_memory — デフォルトの値0に設定されていると、カーネルは利用できるメモリの容量を推定し、無効な要求には反応しません。残念ながらメモリは精密なアルゴリズムではなく、発見的なアルゴリズムを使用している為、時としてシステムをオーバーロードすることが有ります。

  overcommit_memoryが1に設定してある場合、システムオーバーロードの可能性は上がります。但し、幾つかの科学的ソフトウェアで使用されているメモリ集中型のタスクのパフォーマンスも向上します。

  メモリのオーバーコミットのリスク低減を好むユーザーの為に次の2つのオプションが追加されています。overcommit_memoryを2に設定すると、メモリ要求が物理RAMの半分とスワップの合計を越える場合は、不履行になります。3の設定では、メモリ要求がスワップの対応容量を越えるまで追加された場合、不履行になります。
- pagecache — ページキャッシュが使用するメモリ量を制御します。pagecacheの値はパーセントで、利用可能なページキャッシュメモリの最小と最大を実行するためbuffermemと似た方法で機能します。
- page-cluster — 1度で読み取るページ数を設定します。デフォルト値は4で、これは実際には16ページになりますが、ほとんどのシステムで適切な値です。
- pagetable_cache — プロセッサベース単位でキャッシュされるページテーブル数を制御します。1番目と2番目の値は、それぞれ予備配置されるページテーブルの最小数と最大数になります。

こうしたさまざまなファイルの詳細については、/usr/src/linux-2.4/Documentation/sysctl/vm.txt ファイルを参照してください。

5.3.10. /proc/sysvipc/

このディレクトリにはSystem V IPCリソースに関する情報が含まれています。このディレクトリ内のファイルは、メッセージ（msg）、セマフォ（sem）、共有メモリ（shm）に対するSystem V IPCコールに関係しています。

5.3.11. /proc/tty/

このディレクトリにはシステム上で利用可能な現在使用されている*tty*デバイスに関する情報が格納されています。従来*teletype device*と呼ばれていたもので、キャラクタベースのデータ端末が*tty*デバイスと呼ばれます。

Linuxには3種類のttyデバイスがあります。シリアルデバイスはモデムやシリアルケーブルなどを使用する接続に使用されます。仮想端末は、システムコンソールで[Alt]-[<F>]キーを押すと利用可能な仮想コンソールなど一般のコンソール接続を作成します。疑似端末はXFree86などの、より高いレベルのアプリケーションで使用される双方向コミュニケーションを作成します。ドライバーファイルは使用中の現在のttyデバイス一覧です：

```
serial       /dev/cua    5 64-127 serial:callout
serial       /dev/ttyS   4 64-127 serial
pty_slave      /dev/pts   136  0-255 pty:slave
pty_master     /dev/ptm   128  0-255 pty:master
pty_slave      /dev/ttyp    3  0-255 pty:slave
pty_master     /dev/pty     2  0-255 pty:master
/dev/vc/0      /dev/vc/0    4    0 system:vtmaster
/dev/ptmx      /dev/ptmx    5    2 system
```

```
/dev/console    /dev/console  5   1 system:console
/dev/tty      /dev/tty    5   0 system:/dev/tty
unknown      /dev/vc/%d   4  1-63 console
```

/proc/tty/driver/serial ファイルは各シリアルtty行の使用統計とステータスの一覧を示します。

ttyデバイスをネットワークデバイスと同様の方法で使用できるようにするため、Linuxカーネルはデバイスの回線制御を強化します。これにより、ドライバはデバイス上で伝送されるデータブロック毎に固有のヘッダーを付けることができます。接続のリモートエンドでデータブロックを1本のストリームのように見せることが可能です。SLIPやPPPは一般的な回線制御で、それぞれ一般にシリアルリンクでシステムを接続するために使用されます。

登録した回線制御はldiscディレクトリで利用可能な詳細情報とともにldiscsファイルに格納されます。

5.4. sysctl コマンドの使用

/sbin/sysctlコマンドは/proc/sys/ディレクトリ内のカーネル設定を、閲覧、設定、自動化するために用います。

/proc/sys/ディレクトリ内のすべての可能な設定の概要を知るには、rootとして/sbin/sysctl -aコマンドを入力します。これは大きく包括的な一覧を作成します。その一部は以下のようになっています：

```
net.ipv4.route.min_delay = 2
kernel.sysrq = 0
kernel.sem = 250  32000  32  128
```

これはファイルを個別に見た場合と同じ基本情報です。唯一の違いはファイルロケーションです。/proc/sys/net/ipv4/route/min_delay/はnet.ipv4.route.min_delayで示されます。ディレクトリのスラッシュはドットと仮定したproc.sysに置き換えられています。

sysctlコマンドは、echoの代わりに、/proc/sys/ ディレクトリの書き換え可能ファイルに値を割り当てるために使います。例えば、以下のコマンドを用いる代わりに使います：

```
echo 1 > /proc/sys/kernel/sysrq
```

代わりにsysctlコマンドは以下の様に使います：

```
sysctl -w kernel.sysrq="1"
kernel.sysrq = 1
```

/proc/sys/にこうした単一の値をすぐに設定できるのでテスト中は重宝しますが、/proc/sys/の特別な設定はすべてマシンの再起動時に失われるので生産システム上では機能しません。カーネルに対しずっと有効にしておきたい設定を保存するには、その設定を/etc/sysctl.confファイルに追加します。

システムがブートするたびに、/etc/rc.d/rc.sysinitスクリプトがinitによって実行されます。このスクリプトには/etc/sysctl.confを使用してカーネルに渡す値を設定する為にsysctlを実行するコマンドが含まれています。ですから、/etc/sysctl.confに追加された値は、毎回システムブート後に有効となります。

5.5. その他のリソース

以下にprocファイルシステムに関する他の情報源を示します。

5.5.1. インストールされているドキュメント

/procに関する最良のマニュアルのほとんどは、使用しているシステム上で利用できます。

- /usr/src/linux-2.4/Documentation/filesystems/proc.txt — /procディレクトリのすべての側面に関する限定された関連情報が含まれています。
- /usr/src/linux-2.4/Documentation/sysrq.txt — システム要求キーオプションの概要。
- /usr/src/linux-2.4/Documentation/sysctl/ — sysctlのさまざまなヒントが含まれているディレクトリです。カーネル（kernel.txt）、ファイルシステムへのアクセス（fs.txt）、仮想メモリ使用（vm.txt）に関する値の変更が含まれています。
- /usr/src/linux-2.4/Documentation/networking/ip-sysctl.txt — さまざまなIPネットワーキングオプションの閲覧。
- /usr/src/linux-2.4 — /procに関する最も信頼できる情報については、カーネルソースコードをお読みください。kernel-source RPMがシステムにインストールされていることを確認し、ソースコードについては/usr/src/linux-2.4ディレクトリをご覧ください。

5.5.2. 役に立つWebサイト

- http://www.linuxhq.com — このサイトにはLinuxカーネルの各バージョンに対するソース、パッチ、ドキュメンテーションの完全なデータベースが用意されています。

第6章

ユーザーとグループ

ユーザーとグループの管理は、Red Hat Linuxシステム管理の中核をなします。

ユーザーは、実在の使用者に連結したアカウントの意味で人であるか、又は、使用する特殊なアプリケーション用に存在するアカウントでも有り得ます。

グループは、共通の目的の為にユーザーを統合して組織を論理的に表したものです。同一グループのユーザーはグループ所有のファイルを読み込み、書き込み、又は実行をすることが出来ます。

各ユーザーとグループには、ユーザー*id*(*UID*)とグループ*id*(*GID*)という独特の識別数字がそれぞれ割り当てられます。

ファイルが作成されると、それはそのユーザーとグループの所有者に割り当てられます。また、ファイルには所有者、グループ、その他用に個別の読み込み権限、書き込み権限、実行権限が割り当てられます。ファイルが属するユーザーとグループ、及びそのファイルのアクセス権限はrootユーザーあるいは、殆どの場合、ファイルの作成者によって変更が可能です。

適切なユーザーとグループの管理と、ファイル使用権限の効率の良い管理はシステム管理者の最も重要な業務の1つです。ユーザーとグループの管理に関する設定要領の詳細を知るには*Red Hat Linux* システムアドミニストレーションプレミアの中のアカウントとグループの管理の章を参照して下さい。

6.1. ユーザーとグループの管理ツール

ユーザーやグループの管理は退屈な作業で有り得ますが、Red Hat Linux はユーザーとグループの管理を簡潔化するツールと取り決めを提供します。

ユーザーとグループを管理する最も簡単な方法は、グラフィカルアプリケーション**ユーザーマネージャ**(`redhat-config-users`)を使用することです。**ユーザーマネージャ**の詳細については*Red Hat Linux* カスタマイズガイドにあるユーザーとグループの設定の章を参照して下さい。

次のコマンドラインツールを使用してユーザーとグループを管理することもできます：

- `useradd`, `usermod`, 及び`userdel` — ユーザーアカウントの追加、削除、及び修正をする業界標準の方法。
- `groupadd`, `groupmod`, 及び`groupdel` — ユーザーグループの追加、削除、及び修正をする業界標準の方法。
- `gpasswd` — `/etc/group`ファイルを管理するための業界標準の方法。
- `pwck`, `grpck` — パスワード、グループ、及び、関連のシャドウファイルの確証の為のツール。
- `pwconv`, `pwunconv` — シャドウパスワードへの変換と標準パスワードへの復帰の為のツール。

ユーザーとグループの管理に関する概要は、*Red Hat Linux* システムアドミニストレーションプレミアで御覧下さい。ユーザーとグループの管理の為のコマンドラインツールの詳細を確認するには、*Red Hat Linux* カスタマイズガイドのユーザーとグループの設定の章を御覧下さい。

6.2. 標準的なユーザー

表6-1は、「すべて」をインストールした場合の`/etc/passwd`ファイルの中で設定されている標準ユーザーの一覧を示します。この表のGID(グループid)はユーザーのプライマリグループです。標準のグループの一覧に関しては、項6.3を参照して下さい。

ユーザー	UID	GID	ホームディレクトリ	シェル
ユーザー	UID	GID	ホームディレクトリ	シェル
root	0	0	/root	/bin/bash
bin	1	1	/bin	/sbin/nologin
daemon	2	2	/sbin	/sbin/nologin
adm	3	4	/var/adm	/sbin/nologin
lp	4	7	/var/spool/lpd	/sbin/nologin
sync	5	0	/sbin	/bin/sync
shutdown	6	0	/sbin	/sbin/shutdown
halt	7	0	/sbin	/sbin/halt
mail	8	12	/var/spool/mail	/sbin/nologin
news	9	13	/var/spool/news	
uucp	10	14	/var/spool/uucp	/sbin/nologin
operator	11	0	/root	/sbin/nologin
games	12	100	/usr/games	/sbin/nologin
gopher	13	30	/usr/lib/gopher-data	/sbin/nologin
ftp	14	50	/var/ftp	/sbin/nologin
nobody	99	99	/	/sbin/nologin
rpm	37	37	/var/lib/rpm	/bin/bash
vcsa	69	69	/dev	/sbin/nologin
ntp	38	38	/etc/ntp	/sbin/nologin
canna	39	39	/var/lib/canna	/sbin/nologin
nscd	28	28	/	/bin/false
rpc	32	32	/	/sbin/nologin
postfix	89	89	/var/spool/postfix	/bin/true
named	25	25	/var/named	/bin/false
amanda	33	6	var/lib/amanda/	/bin/bash
postgres	26	26	/var/lib/pgsql	/bin/bash
sshd	74	74	/var/empty/sshd	/sbin/nologin
rpcuser	29	29	/var/lib/nfs	/sbin/nologin
nsfnobody	65534	65534	/var/lib/nfs	/sbin/nologin
pvm	24	24	/usr/share/pvm3	/bin/bash
apache	48	48	/var/www	/bin/false
xfs	43	43	/etc/X11/fs	/sbin/nologin
desktop	80	80	/var/lib/menu/kde	/sbin/nologin

ユーザー	UID	GID	ホームディレクトリ	シェル
gdm	42	42	/var/gdm	/sbin/nologin
mysql	27	27	/var/lib/mysql	/bin/bash
webalizer	67	67	/var/www/html/usage	/sbin/nologin
mailman	41	41	/var/mailman	/bin/false
mailnull	47	47	/var/spool/mqueue	/sbin/nologin
smmsp	51	51	/var/spool/mqueue	/sbin/nologin
squid	23	23	/var/spool/squid	/dev/null
ldap	55	55	/var/lib/ldap	/bin/false
netdump	34	34	/var/crash	/bin/bash
pcap	77	77	/var/arpwatch	/sbin/nologin
ident	98	98	/	/sbin/nologin
privoxy	100	101	/etc/privoxy	
radvd	75	75	/	/bin/false
fax	78	78	/var/spool/fax	/sbin/nologin
wnn	49	49	/var/lib/wnn	/bin/bash

表6-1. 標準的なユーザー

6.3. 標準的なグループ

表6-2は、「すべて」をインストールした場合に設定された標準のグループの一覧です。グループはRed Hat Linux内の/etc/groupファイルに保存されています。

グループ	GID	メンバー
root	0	root
bin	1	root, bin, daemon
daemon	2	root, bin, daemon
sys	3	root, bin, adm
adm	4	root, adm, daemon
tty	5	
disk	6	root
lp	7	daemon, lp
mem	8	
kmem	9	
wheel	10	root
mail	12	mail
news	13	news

グループ	GID	メンバー
uucp	14	uucp
man	15	
games	20	
gopher	30	
dip	40	
ftp	50	
lock	54	
nobody	99	
users	100	
rpm	37	rpm
utmp	22	
floppy	19	
vcsa	69	
ntp	38	
canna	39	
nscd	28	
rpc	32	
postdrop	90	
postfix	89	
named	25	
postgres	26	
sshd	74	
rpcuser	29	
nfsnobody	65534	
pvm	24	
apache	48	
xfs	43	
desktop	80	
gdm	42	
mysql	27	
webalizer	67	
mailman	41	
mailnull	47	
smmsp	51	
squid	23	

グループ	GID	メンバー
ldap	55	
netdump	34	
pcap	77	
ident	98	
privoxy	101	
radvd	75	
fax	78	
slocate	21	
wnn	49	

表6-2. 標準的なグループ

6.4. ユーザープライベートグループ

Red Hat Linuxはユーザープライベートグループ（*UPG*）体系を使用してUNIXのグループを使いやすくしています。

UPGは新規のユーザーがシステムに追加される度に、生成されます。UPGはそれが生成される元であるユーザーと同名を持っており、そのユーザーのみがUPGのメンバーです。

UPGの使用により、新規のファイルやディレクトリに対し安全にデフォルトの権限を設定することが可能なため、ユーザーとそのユーザーのグループはそれらのファイルやディレクトリを自由に修正出来るようになります。

新規に作成されたファイルやディレクトリに対してどの権限を与えるかを決定する設定は*umask*と呼ばれ、`/etc/bashrc`ファイル内に設定されています。伝統的にUNIXシステムでは、そのumaskは022に設定されています。この設定では、ファイル又はディレクトリを作成したユーザー本人のみが変更できます。この体系下では、他のユーザーとユーザーグループのメンバーでもそのユーザーのファイルは変更出来ません。しかしUPG体系の中では、各ユーザーが自己のプライベートグループを持つことから、このグループ保護は必要ではありません。

6.4.1. グループディレクトリ

ほとんどのIT組織は、主要プロジェクトごとにグループを作成し、そのグループのファイルにアクセスする必要のある人をグループに割り当てることを好みます。このような伝統的な体系では、誰かがファイルを作成した場合に、作成者の属するプライマリグループがそのファイルの所有者になるため、ファイルの管理が困難でした。1人の人間が複数のプロジェクトに従事する場合、正しいファイルを正しいグループと関連付けるのは難しくなります。UPG体系では、グループはsetgidビットセットを持つディレクトリで作成されたファイルに自動的に割り当てられるため、ディレクトリを共有するグループプロジェクトの管理が非常に簡単になります。

例としてあげると、あるグループが`/usr/lib/emacs/site-lisp/`ディレクトリ内のファイルで作業をしている場合、幾らかの人々はディレクトリの修正をさせる信頼がありますが、全ての人がそうではありません。そこで先ず、以下のようなコマンドを使用して`emacs` グループを作成します：

```
/usr/sbin/groupadd emacs
```

そのディレクトリの内容を`emacs`グループと関連づけるには次のように入力します：

```
chown -R root.emacs /usr/lib/emacs/site-lisp
```

ここでgpasswdコマンドを使用してこのグループに正式なユーザーを追加することが可能になります：

```
/usr/bin/gpasswd -a <username> emacs
```

以下のコマンドで、このディレクトリ内に実際にファイルを作成する権限をユーザーに与えます。

```
chmod 775 /usr/lib/emacs/site-lisp
```

ユーザーが新しいファイルを作成すると、そのファイルのグループとしてユーザーのデフォルトであるプライベートグループが割り当てられます。次にsetgidビットを設定して、そのディレクトリに作成された全てにディレクトリ自身(emacs)と同じグループ権限を割り当てます。次のコマンドを使用します：

```
chmod 2775 /usr/lib/emacs/site-lisp
```

この時点で各ユーザーのデフォルトumaskが002である為に、ユーザーが新しいファイルを書き込む度に管理者がファイルの権限を変更することなく、emacsグループの全てのメンバーは/usr/lib/emacs/site-lisp/ディレクトリ内でファイルを作成及び編集することができます。

6.5. シャドウパスワード

マルチユーザー環境では、シャドウパスワード(シャドウユーティリティパッケージで提供) の使用がかなり重要になってきます。その使用によりファイルのシステム認証ファイルのセキュリティが強化されます。この理由でRed Hat Linuxインストールプログラムはデフォルトでシャドウパスワードを有効にしています。

従来のUNIXベースのシステムでパスワードを保存する方法より優れたシャドウパスワードの利点を以下の一覧で示します：

- 暗号化されたパスワードハッシュを全て読み取り可能な/etc/passwdかrootにしか読み取れない/etc/shadowに移動することでシステムセキュリティを向上。
- パスワードの老朽化に関連する情報を保存。
- /etc/login.defsファイルを使用してセキュリティポリシーを執行。

shadow-utilsパッケージユーティリティによって提供される殆どのユーティリティはシャドウパスワードが有効であることに関係なく、正しく動作します。ただし、パスワード老朽化情報が専用の/etc/shadow ディレクトリに収納されている為、パスワード老朽化情報の作成や編集をするコマンドは動作しません。

シャドウパスワードを最初に有効にしないと動作しないコマンドを以下の一覧でしめします：

- chage
- gpasswd
- /usr/sbin/usermod -e or -f options
- /usr/sbin/useradd -e or -f options

第7章

X Window System

Red Hat Linuxのハートは、カーネルでしょうが、多くのユーザーにとっては、オペレーティングシステムの顔は、やはり*X Window System*によって提供される*X*と呼ばれるグラフィカル環境です。

UNIX™の世界では各種のウィンドウ環境が、数十年存在して来ました。現在の多くの主流オペレーティングシステムより歴史があります。これらの年月を通して、UNIXライクなオペレーティングシステムでは、X がグラフィカル環境の優越性を保持しています。

Red Hat Linuxのグラフィカル環境はXのオープンソース実装であるXFree86™によって供給されています。XFree86は、世界中で何百人もの開発者がいる急速発展中のオープンソースソフトウェアプロジェクトです。各種ハードウェアデバイスとアーキテクチャを幅広くサポートできる、異なるオペレーティングシステムとプラットホームで実行できるといった特徴があります。

X Window Systemは、クライアント/サーバーアーキテクチャーを使用します。*X* サーバーは、ネットワーク又はローカルループバックインターフェイスを経由した*X* クライアントからの接続を監視します。サーバーは、ビデオカード、モニター、キーボード、マウスなどハードウェアとの連携があります。X クライアントアプリケーションは、ユーザースペース内にあり、ユーザーとそのユーザーの要求をXサーバーに渡すためのグラフィカルユーザーインターフェイス(*GUI*) を構成します。

7.1. XFree86

Red Hat Linux 9 ではXFree86のバージョン4.xをX Window Systemのベースとして使用し、これは3Dハードウェアアクセラレータサポート、anti-aliasedフォント用のXRender拡張、モジュラードライバーベースのデザイン、そして最新のビデオハードウェアと入力デバイス用サポートなどの最先端のXFree86技術の増強を含んでいます。

重要

Red Hat Linux は、XFree86バージョン3のサーバーパッケージのサポートはしておりません。Red Hat Linuxの最新のバージョンにアップグレードする前に、ビデオカードがXFree86 のバージョン4と互換性があるかどうかを、Red Hatハードウェア互換一覧のサイトで確認してください。http://hardware.redhat.com.

XFree86関連のファイルは、主に次の2つの場所にあります：

`/usr/X11R6/`

‘ これは、X サーバーと幾つかのクライアントアプリケーション、さらにX ヘッダファイル、ライブラリ、モジュール、ドキュメントなどを収納しています。

`/etc/X11/`

‘ これには、X クライアントとサーバーのアプリケーション用の設定ファイルが含まれています。この中には、X サーバー自身の設定ファイル、古い`xfs` フォントサーバー、X ディスプレーマネージャ、その他多くのベースコンポーネントがあります。

新しいFontconfigベースのフォントアーキテクチャーは`/etc/fonts/fonts.conf` (`/etc/X11/XftConfig`ファイルの改訂版となる)であることに注意して下さい。フォントの設定と追加に関しては項7.4を御覧下さい。

XFree86サーバーは、幅広い種類のハードウェア上で高度なタスクを実行しますので、詳細な設定を必要とします。Red Hat Linuxインストールプログラムは、XFree86パッケージがインストールの選択項目から外れていることがない限り、XFree86を自動的にインストールして設定をします。しかし、

モニターやビデオカードが変更される場合、XFree86 は再設定される必要があります。これを実行する一番簡単な方法は**X 設定ツール**

Xのセッションがアクティブな時に、**X 設定ツール**をスタートするには、パネル上の**メインメニューボタン**=> **システム設定** =>**ディスプレー**と進んで行きます。X セッションの間に**X 設定ツール**を使用した後は、一度ログオフをして、再度ログオンするとその変更が有効になります。**X 設定ツール**の使用に関する詳細は*Red Hat Linux* 入門ガイド内のオーディオ、ビデオ、その他の遊びの機能の章で確認して下さい。

場合によっては、XFree86サーバーの再設定は、その設定ファイル`/etc/X11/XF86Config` の手動編集が必要になるかも知れません。このファイルの構造に関しては項7.3を御覧下さい。

7.2. デスクトップ環境とウィンドウマネージャ

XFree86サーバーが実行している状態になると、Xクライアントアプリケーションはそれに接続して、ユーザー用のGUIを作成することが出来ます。Red Hat Linuxでは、ごく基本的なタブウィンドウマネージャから高度に発達した対話式の*GNOME*デスクトップ環境まで、多くのRed Hat Linuxユーザーにお馴染みの幅広いGUIが利用できます。

より発達したGUIである、*GNOME*デスクトップ環境を構成するには、X クライアントアプリケーションの2つの主要クラス；デスクトップ環境、及びウィンドウマネージャがXFree86サーバーに接続される必要があります。

7.2.1. デスクトップ環境

デスクトップ環境は、一緒に使用されると共通のグラフィカルユーザー環境と開発プラットフォームを構築する各種のXクライアントを収束します。

デスクトップ環境は、幾つかの高度な機能を持ち、その使用によりXクライアントと他の実行中プロセスがお互いに交信できるようになり、またその環境の中で動作する様に書き込まれている全てのアプリケーションが、ドラッグアンドドロップなどの高度タスクを実行できるようになります。

Red Hat Linux は以下の2種類のデスクトップ環境を提供します：

- *GNOME* — GTK+ 2 グラフィカルツールキットをベースにしたRed Hat Linux用のデフォルトデスクトップ環境。
- *KDE* — Qt 3 グラフィカルツールキットをベースにした代用のデスクトップ環境。

GNOME とKDEは両方とも、ワープロ、スプレッドシート、Webブラウザなどの高度な作業効率のアプリケーションを持っており、またGUIのルックとフィールをカスタマイズするためのツールも提供します。さらには、GTK+ 2とQt の両方のライブラリが揃っている場合、KDEアプリケーションは、GNOMEの中で実行可能でまたその逆も可能になります。

GNOME とKDE デスクトップ環境のカスタマイズ法についての情報については、*Red Hat Linux* 入門ガイドを参照してください。

7.2.2. ウィンドウマネージャ

ウィンドウマネージャ は、デスクトップ環境に1部であるか、又は、場合によってはスタンドアローンのこともあります。その主要目的はグラフィカルウィンドウがどのように配置され、サイズ変更され、そして移動されるかを制御します。ウィンドウマネージャは、またタイトルバー、ウィンドウの焦点調節、そしてユーザー設定のキーとマウスボタンの連携なども制御します。

5種類のウィンドウマネージャがRed Hat Linuxに収納されています：

- `kwin` — *KWin*ウィンドウマネージャは、KDEデスクトップ環境用のデフォルトウィンドウマネージャです。カスタムテーマをサポートする効率の良いウィンドウマネージャです。

- metacity — *Metacity* ウィンドウマネージャは、GNOMEデスクトップ環境のデフォルトウィンドウマネージャです。カスタムテーマをサポートする簡単で効率の良いウィンドウマネージャです。
- mwm — *Motif* ウィンドウマネージャは、基本的なスタンドアローンのウィンドウマネージャです。単独で機能するように設計されているため、GNOME やKDEと一緒に使用すべきではありません。
- sawfish — *Sawfish* ウィンドウマネージャは、フル機能をもったウィンドウマネージャで、Red Hat Linux 8.0のリリースまでは、GNOMEデスクトップ環境用のデフォルトでした。これは単独でもデスクトップ環境との併用でも使用できます。
- twm — 最小のタブウィンドウマネージャで、これはすべてのウィンドウマネージャの中で最も基本的なツールセットを提供し、単独又はデスクトップ環境との併用でも使用できます。XFree86の一部としてインストールされます。

ウィンドウマネージャは、その違いを明確に知る為にデスクトップ環境なしで単独で実行することも出来ます。これを実行するには、コマンドxinit -e <*path-to-window-manager*>を入力します。ここで<*path-to-window-manager*>はウィンドウマネージャのバイナリファイルのある場所です。そのバイナリファイルはwhich <*window-manager-name*>と入力して見付けることが出来ます。

7.3. XFree86サーバー設定ファイル

XFree86サーバーは、シングルバイナリ実行可能ファイル(/usr/X11R6/bin/XFree86)、これは/usr/X11R6/lib/modules/ディレクトリからランタイムにおいて、必要なX サーバーモジュールを動的にロードします。これらのモジュールの一部はサーバーによって自動的にロードされますが、それ以外は選択肢となり、XFree86サーバー設定ファイルの中で指定しなければなりません。

XFree86サーバーとその関連設定ファイルは/etc/X11/ディレクトリ内に保存されています。XFree86サーバー用の設定ファイルは/etc/X11/XF86Configです。Red Hat Linuxがインストールされると、XFree86の設定ファイルが、インストールプロセスの間にシステムハードウェアについての情報を使用して作成されます。

7.3.1. XF86Config

/etc/X11/XF86Configを手動で編集が必要なことはあまりありませんが、トラブルシューティングの時などにさまざまなセクションとオプションパラメータについて知っておくと便利です。

7.3.1.1. 構造

/etc/X11/XF86Configのファイルはセクションの集まりで構成されており、それぞれのセクションはシステムハードウェアの特定の動作を担当します。

各セクションはSection "<*section-name*>" (ここで<*section-name*>とはセクションのタイトルです)行で始まり、EndSection行で終了します。各行の中には、オプション名を含む行があり、少なくとも1つのオプション値がときには引用符で囲まれています。

[#]マークで始まる行は、人間が読むためのコメントとして使用され、XFree86サーバーには読み込まれない行です。

/etc/X11/XF86Configファイルの幾つかはブール値スィッチを取り、これが機能のオンとオフの切替えをします。有効なブール値は以下のようになります :

- 1、on、true、yes — これらはいずれも、オプションをオンにします。
- 0、off、false、no — これらはいずれもオプションをオフにします。

以下に、標準的な/etc/X11/XF86Configファイルに表示されている順序のセクションの一部を示します。XFree86ファイルサーバーの設定ファイルに関する詳細情報はXF86Configのmanページで確認することができます。

7.3.1.2. ServerFlags

オプションのServerFlagsセクションには、さまざまなグローバルXFree86サーバーの設定が含まれています。このセクションの設定はServerLayoutセクションに配置されているオプションで上書きされてしまいます。(詳細は項7.3.1.3で御覧下さい)。

ServerFlagsセクション内のエントリーはそれぞれ独自の行にあり、Optionという表示で始まる2重引用符["]で囲まれたオプションを持ちます

以下にServerFlagsセクションのサンプルを示します：

```
Section "ServerFlags"
   Option "DontZap" "true"
EndSection
```

役に立つオプションの幾つかを以下に示します：

- "DontZap" "<*boolean*>" — <*boolean*>の値が、「true」の場合、XFree86サーバーを直ちに停止するような[Ctrl]-[Alt]-[Backspace]キーの使用を防止します。
- "DontZoom" "<*boolean*>" — <*boolean*>の値が、「true」の場合、[Ctrl]-[Alt]-[Keypad-Plus]キーの使用と、[Ctrl]-[Alt]-[Keypad-Minus]キーを使用した、設定済のビデオ解像度を切替える操作を防止します。

7.3.1.3. ServerLayout

ServerLayoutセクションは、XFree86サーバーによって制御されている入力/出力用のデバイスを組み合わせます。最低でもこのセクションは1つの出力デバイスと2つの入力デバイス(キーボードとマウス)を指定する必要があります。

次の例では、標準的なServerLayoutセクションを示しています：

```
Section "ServerLayout"
    Identifier  "Default Layout"
    Screen   0 "Screen0" 0 0
    InputDevice  "Mouse0" "CorePointer"
    InputDevice  "Keyboard0" "CoreKeyboard"
EndSection
```

以下にServerLayoutセクションで一般的に使用されるエントリーを示します：

- Identifier — このServerLayoutセクション用の独自の名前を指定します。
- Screen — XFree86サーバーで使用されるScreenセクションの名前を指定します。複数のScreenオプションが存在することができます。

  以下に標準的なScreenエントリーの例を示します：
  ```
  Screen   0 "Screen0" 0 0
  ```

  この例のScreenエントリー(0)は、最初のモニターコネクター、あるいはビデオカード上の*head*がScreenセクションの指定した設定を識別子"Screen0"で使用することを示しています。

  ビデオカードが複数のヘッドを持っている場合、別の番号と別のScreenセクション識別子を持つもう1つのScreenエントリーが必要になります。

  "Screen0"の右の番号は、画面の左上隅に使うXとYの絶対座標です(デフォルトは0 0)。

- InputDevice — XFree86サーバーと併用するInputDeviceセクションの名前を指定します。

  少なくとも2つのInputDeviceエントリーが必要です：1つは、デフォルトのマウス用で、もう1つはデフォルトのキーボード用です。オプションのCorePointerとCoreKeyboardはこれらが主要なマウスとキーボードであることを示します。

- Option "<*option-name*>" — このセクションのエクストラパラメーターを指定するオプションのエントリーです。ここにリストされているエントリーはServerFlagsセクションにリストされているものを上書きします。

  この<*option-name*>は、XF86Configのmanページのこのセクションにリストしてある有効なオプションで入れ換えます。

複数のServerLayoutセクションを作成することが出来ます。しかし、サーバーは、コマンドラインの引数として代用のServerLayout セクションが指定されている場合以外は、最初に表示される物を読み込みます。

7.3.1.4. Files

Filesセクションは、フォントパスのように、XFree86サーバーへの重要なサービス用のパスを設定します。

以下の例で、標準的なFilesセクションを示します：

```
Section "Files"
    RgbPath   "/usr/X11R6/lib/X11/rgb"
    FontPath  "unix/:7100"
EndSection
```

以下に一般的にFilesセクションで使用されるエントリーを示します：

- RgbPath — RGBカラーデータベースの場所を指定します。このデータベースはXFree86の全てのカラー名を定義し、それを特定のRGB値と結合させます。
- FontPath — XFree86 サーバーが、xfsフォントサーバーからフォントを取り出す為に接続する必要のある場所を指定します。

  デフォルトでは、FontPathはunix/:7100です。これは、XFree86 サーバーに対して、ポート7100にあるIPC(inter-process communication)用のUNIXドメインソケットを使用してフォント情報を取得するように指示します。

  XFree86やフォントの詳細については、項7.4を参照してください。

- ModulePath — オプションのパラメーターです。これはXFree86サーバーモジュールを保存する代替のディレクトリを指定します。

7.3.1.5. Module

Moduleは、XFree86サーバーがロードする予定の/usr/X11R6/lib/modules/ディレクトリ内のモジュールを指定します。モジュールはXFree86サーバーに追加の機能を与えます。

以下の例で標準的なModuleセクションを示します：

```
Section "Module"
 Load "dbe"
 Load "extmod"
 Load "fbdevhw"
 Load "glx"
 Load "record"
 Load "freetype"
 Load "type1"
```

```
 Load "dri"
EndSection
```

7.3.1.6. InputDevice

各InputDeviceセクションはXFree86サーバーに対して1つの入力デバイスを設定します。システムは標準レベルで、最低でも2つのInputDeviceセクション(キーボードとマウス) を持ちます。

以下の例は、マウス用の標準的なInputDevice セクションを示します：

```
Section "InputDevice"
 Identifier "Mouse0"
 Driver   "mouse"
 Option   "Protocol" "IMPS/2"
 Option   "Device" "/dev/input/mice"
 Option   "Emulate3Buttons" "no"
EndSection
```

一般にInputDeviceセクションで使用されるエントリーは以下のようになります：

- Identifier — InputDeviceセクションの独自の名前を指定します。これは必須のエントリーです。
- Driver — デバイス用にXFree86がロードする必要のあるデバイスドライバーの名前を指定します。
- Option — そのデバイスに関係する必要なオプションを指定します。

  マウスには、以下のオプションがあります：

  - Protocol — IMPS/2など、マウスで使用するプロトコルを指定します。
  - Device — 物理デバイスの場所を指定します。
  - Emulate3Buttons — 2つのマウスボタンを同時に押した時に3ボタンマウスのように動作させるかどうか指定します。

  このセクションの有効なオプションのリストについては、XF86Configのmanページを参照してください。

デフォルトでは、InputDeviceセクションには、ユーザーが追加のオプションを設定できるようにするコメントがあります。

7.3.1.7. Monitorセクション

各Monitorセクションは、システムによって使用されるモニターのタイプを1つ設定します。1つのMonitorセクションは最低限必要ですが、マシンによって使用される各モニターの追加分の設定も有り得ます。

モニターの設定で最善の方法は、インストールのプロセス中にXを設定するか、又は**X 設定ツール**を使用することです。この**X 設定ツール**の使用に関する詳細は*Red Hat Linux* 入門ガイドの中にあるオーディオ、ビデオ、その他の遊びの機能という章でお読み下さい。

次の例は、モニター用の標準的なMonitorセクションを示しています：

```
Section "Monitor"
 Identifier  "Monitor0"
 VendorName  "Monitor Vendor"
 ModelName  "DDC Probed Monitor - ViewSonic G773-2"
 DisplaySize 320 240
```

```
 HorizSync  30.0 - 70.0
 VertRefresh 50.0 - 180.0
EndSection
```

警告

/etc/X11/XF86ConfigのMonitorセクション内で値を手動で編集することには注意が必要です。不適切な値はモニターを損傷したり、破損したりする可能性があります。モニターのマニュアル等で安全な操作用のパラメータ一覧を確認して下さい。

以下にMonitorセクションで一般的に使用されるエントリーを示します：

- Identifier — Monitorセクション用の独自に名前を指定します。これは必須のエントリーです。
- VendorName — モニターのベンダーを指定するオプションのパラメータです。
- ModelName — モニターのモデル名を指定するオプションのパラメータです。
- DisplaySize — モニターの表示面積をミリメーターで指定するオプションのパラメータです。
- HorizSync — モニターで互換性のある水平同期周波数の幅をkHz単位で指定します。この値はXFree86サーバーが、モニター用の組込みの、あるいは指定済のModelineエントリーの有効性を判定するのに役立ちます。
- VertRefresh — モニターでサポートされている垂直同期周波数の幅をkHz単位で指定します。これらの値は、XFree86サーバーが、モニター用の組込みの、あるいは指定済のModelineエントリーの有効性を判定するのに役立ちます。
- Modeline — 特定の水平同期周波数と垂直同期周波数を固定した、ある解像度でのモニター用の追加のビデオモードを指定するパラメータです。Modelineエントリーに関する詳細はXF86Configのmanページで御覧下さい。
- Option "<*option-name*>" — セクションへのエクストラパラメータを指定するオプションのエントリーです。<*option-name*>は、XF86Configのmanページ内のこのセクション用にリストしてある有効なオプションで入れ換えます。

7.3.1.8. Device

各Deviceセクションは、システム上のビデオカード1つを設定します。1つのDeviceセクションが最低限ですが、マシン上にインストールされている各ビデオカードの為に追加分の設定も有り得ます。

ビデオカードを設定する最善の方法は、インストールプロセス中にXを設定するか、又は**X 設定ツール**を使用することです。この**X 設定ツール**の使用についての詳細は*Red Hat Linux* 入門ガイド の中のオーディオ、ビデオ、その他の遊びの機能の章を御覧下さい。

次の例は、ビデオカード用の標準的なDeviceセクションを示しています：

```
Section "Device"
 Identifier "Videocard0"
 Driver   "mga"
 VendorName "Videocard vendor"
 BoardName  "Matrox Millennium G200"
 VideoRam  8192
    Option   "dpms"
EndSection
```

以下にDeviceセクションで一般的に使用されるエントリーを示します：

- `Identifier` — このDeviceセクション用の独自の名前を指定します。これは必須のエントリーです。
- `Driver` — ビデオカードを使用するためにXFree86サーバーがロードする必要のあるドライバーを指定します。hwdataパッケージでインストールしてある/usr/X11R6/lib/X11/Cardsの中にドライバーのリストがあります。
- `VendorName` — ビデオカードのベンダーを指定するオプションのパラメータです。
- `BoardName` — ビデオカードの名前を指定するオプションのパラメータです。
- `VideoRam` — ビデオカード上で利用できるRAMの容量をキロバイトで指定するオプションのパラメータです。この設定はXFree86 サーバーがビデオRAMの容量を検出できなかった時にのみ必要です。
- `BusID` — ビデオカードバスの位置を指定するオプションのパラメータです。このオプションは、システムに複数のカードがある時にのみ必要となります。
- `Screen` — Deviceセクションが設定するビデオカード上のモニターコネクター又はヘッドを指定するオプションのエントリーです。このオプションは複数ヘッドを持つビデオカードだけに役立ちます。

  複数のモニターが同一のビデオカードの異なるヘッドに接続されている場合、別々のDevice セクションが必要となり、各セクションは異なるScreen 値を持つ必要があります。

  Screen エントリーの値は、整数である必要があります。ビデオカード上の最初のヘッドは、値0を持ちます。各追加のヘッドは追加の度に値を1つずつ加算していきます。
- `Option "<option-name>"` — このセクションのエクストラパラメータを指定するオプションのエントリーです。<option-name>はXF86Configのmanページのこのセクション用にリストしてある有効なオプションで入れ換えます。

  一般的なオプションの1つは"dpms"で、これはモニターのService Star 省電力規定機能を起動します。

7.3.1.9. Screen

各Screenセクションは、1つのビデオカード(又はビデオカードヘッド)を、Deviceセクションと Monitorセクションをそれぞれ参照することにより、1つのモニターに結合します。1つのScreenは最低限ですが、マシン上にあるビデオカードとモニターのそれぞれの組み合わせの為に、追加の構成も有り得ます。

以下の例は、標準的なScreenセクションを示しています：

```
Section "Screen"
 Identifier "Screen0"
 Device  "Videocard0"
 Monitor  "Monitor0"
 DefaultDepth  16
 SubSection "Display"
  Depth  24
  Modes  "1280x1024" "1280x960" "1152x864" "1024x768" "800x600" "640x480"
 EndSubSection
 SubSection "Display"
  Depth  16
  Modes  "1152x864" "1024x768" "800x600" "640x480"
 EndSubSection
EndSection
```

以下に一般的に使用されるScreenセクションのエントリーを示します：

- `Identifier` — このScreenセクションの独自の名前を指定します。これは必須のエントリーです。
- `Device` — Deviceセクションの独自の名前を指定します。これは必須のエントリーです。
- `Monitor` — Monitorセクションの独自の名前を指定します。これは必須のエントリーです。
- `DefaultDepth` — デフォルトの色の深さをビットで指定します。上記の例では16であり、数千の色を提供するデフォルトです。また最低限1つの指定が必ず必要ですが、複数のDefaultDepthエントリーも許可されます。
- `SubSection "Display"` — 特定の色の深さで利用できるスクリーンモードを指定します。Screenセクションは複数のDisplayサブセクションを使用できますが、必ず、DefaultDepthエントリーで指定してある色の深さの為の1つのDisplayサブセクションが必要です。
- `Option "<option-name>"` — このセクションのエクストラパラメータを指定するオプションのエントリーです。<option-name>はXF86Configのmanページの中にあるこのセクション用リスト内の有効なオプションで入れ換えます。

7.3.1.10. DRI

オプションのDRI セクションは、*DRI* (*Direct Rendering Infrastructure*)用のパラメータを指定します。DRI は、最近のビデオハードウェアに組み込まれている3Dソフトウェアアプリケーションを利用可能にするインターフェイスです。さらには、DRIはビデオカードドライバーによりサポートされている限り、ハードウェアアクセラレーションを経由して2Dのパフォーマンスも向上させることが出来ます。

ModuleセクションでDRIが有効にあっている場合以外はこのセクションは無視されます。

以下の例は、標準的なDRIセクションを示します：

```
Section "DRI"
   Group   0
   Mode    0666
EndSection
```

異なるビデオカードは、DRIを異なる方法で使用しますので、最初に/usr/X11R6/lib/X11/doc/README.DRIファイルを参照するまでは、このセクションの値を変更しないで下さい。

7.4. フォント

Red Hat Linux は2つの方法を使用してXFree86下のフォントとディスプレイを管理します。より新しいFontconfigフォントサブシステムがフォント管理をより簡単にし、anti-aliasingなどの高度なディスプレイ機能を提供します。Qt 3 又はGTK+ 2 グラフィカルツールキットを使用するようにプログラムされているアプリケーション用にはこのシステムが自動的に利用されます。

互換性の為にRed Hat Linuxには、コアX フォントサブシステムと呼ばれるオリジナルのフォントサブシステムが含まれています。このシステムは15年の歴史を持ち、*xfs*(*X Font Server*)をベースにして構成されています。

このセクションでは、上記の両方のシステムを使用してXの為のフォントの設定法を説明して行きます。

7.4.1. Fontconfig

Fontconfig フォントサブシステムを使用すると、アプリケーションがシステム上のフォントに直接アクセスできるようになり、Xft 又は他のレンダリング機構を使って高度なanti-aliasingでのFontconfigフォントを描写できます。グラフィカルアプリケーションはFontconfigと共にXft ライブラリを使用してテキストを画面に描くことが出来ます。

時期が来れば、Fontconfig/Xftフォントシステムがコアフォントシステムの入れ換えになるでしょう。

重要

Fontconfig フォントサブシステムは、いまのところ**OpenOffice.org**と**Abiword**では、機能しません。この2つのアプリケーションは独自のフォントレンダリング技術を使用しています。

Fontconfigは、`/etc/fonts/fonts.conf`設定ファイルを共有することに注意して下さい。これは`/etc/X11/XftConfig`から交替したものです。Fontconfigの設定ファイルは手動で編集しないで下さい。

ヒント

新しいフォントシステムへの移動の為、GTK+ 1.2アプリケーションは、**フォント設定**のダイアログ(パネル上から**メインメニューボタン => 個人設定 => フォント**と入る)での変更には影響されません。これらのアプリケーションには、以下の行をファイル`~/.gtkrc.mine`に追加することでフォントが設定できます:

```
style "user-font" {

fontset = "<font-specification>"

}

widget_class "*" style "user-font"
```

*<font-specification>*は`-adobe-helvetica-medium-r-normal--*-120-*-*-*-*-*-*`のような伝統的なX アプリケーションで使用されるスタイルでのフォント指定に入れ換えます。コアフォントの総合リストは、`xlsfonts` コマンドの実行で取得するか、又は`xfontsel`を対話式に使用して作成できます。

7.4.1.1. Fontconfigへのフォントの追加

新しいフォントをFontconfigサブシステムに追加することは簡単明快なプロセスです。

1. システム全体にフォントを追加するには、その新しいフォントを`/usr/share/fonts/local/`ディレクトリにコピーします。

   フォントを個人のユーザー用に追加するには、その新しいフォントをユーザーのホームディレクトリ内の`.fonts/`ディレクトリにコピーします。

2. `fc-cache`コマンドを使用して、以下の例に示すようにフォント情報のキャッシュを更新します:

   ```
   fc-cache <path-to-font-directory>
   ```

   このコマンドでは、*<path-to-font-directory>*はその新しいフォントを収納しているディレクトリ(`/usr/share/fonts/local/`か、`~/.fonts/`)で入れ換えます。

ヒント

個人ユーザーは、フォントをグラフィック的にインストールするためにNautilusの`fonts:///`を閲覧して新しいフォントをそこへドラッグすることが出来ます。

重要

フォント名が拡張子`.gz`で終っている場合、それは圧縮してあり、解凍するまで使用できません。これを実行するには、`gunzip`コマンドを使用するか、又はファイルをダブルクリックして、フォントを**Nautilus**内のあるディレクトリにドラッグします。

7.4.2. コアX フォントシステム

互換性の為にRed Hat Linuxは今でもコアX フォントサブシステムを提供しています。これは、X フォントサーバー(`xfs`)を使用してフォントをX クライアントアプリケーションに提供します。

XFree86サーバーは、`/etc/X11/XF86Config`設定ファイルの`Files`セクション下の`FontPath`エントリーで指定してあるフォントサーバーを探します。`FontPath`エントリーの詳細情報に関しては項7.3.1.4を御覧下さい。

XFree86サーバーは、指定のポート上で`xfs`サーバーに接続し、フォント情報を取得します。この理由で、Xがスタートできるように`xfs`サービスは実行中でなければなりません。特定のランレベル用の設定サービスの情報は*Red Hat Linux カスタマイズガイド*内のサービスに対するアクセスの制御と言う章を御覧下さい。

7.4.2.1. `xfs`設定

`/etc/rc.d/init.d/xfs`スクリプトは`xfs`サーバーを開始します。`/etc/X11/fs/config`ファイルの中では数種のオプションが設定できます。

次に一般的なオプションの一覧を示します：

- `alternate-servers` — フォントサーバーが利用できない場合に、使用予定の代替用のフォントサーバーの一覧を指定します。リスト内の各サーバーはカンマで区切る必要があります。
- `catalogue` — 使用するフォントパスの順番のリストを指定します。リスト内で1つのフォントパスと次のフォントパスの間にカンマが存在する必要があります。

  フォントパスの直後に文字列`:unscaled`を使用して、そのパス内の無倍率のフォントを最初にロードさせます。その後全体のパスを指定そして他の倍率付きのフォントもロードされるようにします。
- `client-limit` — そのフォントサーバーが面倒を見るクライアントの最大数を指定します。デフォルトは`10`です。
- `clone-self` — `client-limit`が打ち込まれた時、フォントサーバーにそれ自身の新しいバージョンをクローンできる様にします。デフォルトでは、このオプションは`on`です。
- `default-point-size` — この値を指定しないフォント用にデフォルトのフォントを指定します。このオプションの値はデシポイントで設定してあります。デフォルトの`120`は、12ポイントのフォントに相当します。
- `default-resolutions` — XFree86サーバーによりサポートされている解像度のリストを指定します。リスト内の各解像度はカンマで区切る必要があります。

- `deferglyphs` — *glyphs* (フォントを可視的に表示する為に使用されるグラフィックス)を遅延させるかどうか指定します。この機能を無効にするには`none`を使用し、全てのフォント用にこの機能を有効にするには`all`を使用し、またこの機能を16ビットフォント用にのみ有効にするには`16`を使用します。
- `error-file` — `xfs`のエラーが記録された場所のパスとファイル名を指定します。
- `no-listen` — `xfs`が特定のプロトコルをリッスンしないように防止します。デフォルトではこのオプションは`tcp`に設定されており、セキュリティの目的で`xfs`がTCPポートをリッスンすることを止めています。もし`xfs`をネットワーク上でフォントサービスとして使用する場合はこの行は削除する必要があります。
- `port` — `no-listen`が存在しない、又はそれがコメントアウトしてある場合、`xfs`がリッスンするポートとしてTCPポートを指定します。
- `use-syslog` — システムエラーログを使用するかどうかを指定します。

7.4.2.2. xfsへのフォントの追加

コアX フォントサブシステム(`xfs`)にフォントを追加するには、次のステップに従います：

1. すでに存在していない場合は、ルートで次のコマンドを使用して`/usr/share/fonts/local/`というディレクトリを作成します：
   ```
   mkdir /usr/share/fonts/local/
   ```
   `/usr/share/fonts/local/`ディレクトリの作成が必要な場合、次のコマンドをルートで入力してディレクトリを`xfs`のパスに追加します：
   ```
   chkfontpath --add /usr/share/fonts/local/
   ```
2. 新しいフォントファイルを`/usr/share/fonts/local/` ディレクトリにコピーします。
3. ルートとして次のコマンドを発行して、フォント情報を更新します：
   ```
   ttmkfdir -d /usr/share/fonts/local/ -o /usr/share/fonts/local/fonts.scale
   ```
4. ルートとして次のコマンドを使用して`xfs`フォントサーバーを再起動します：
   ```
   service xfs reload
   ```

7.5. ランレベルとXFree86

殆どの場合、Red Hat Linuxのデフォルトインストールは、ランレベル5 として知られるグラフィカルログイン環境で起動するようにマシンを設定します。しかし、ランレベル3 と呼ばれる、テキストのみの複数ユーザーモードで起動してそこからX のセッションを開始することも可能です。

ランレベルに関する詳細は項1.4で御覧下さい。

このセクションは、XFree86がどのようにランレベル3 とランレベル5 でスタートするかを説明していきます。

7.5.1. ランレベル3

ランレベル3にいる時、X のセッションを開始する最善の方法は、ログインして`startx`と入力することです。`startx`コマンドは`xinit`コマンドへのフロントエンドであり、XFree86サーバーを起動してそれをX クライアントアプリケーションと繋ます。ユーザーはすでにシステムにランレベル3でログインしている為、`startx`はディスプレイマネージャを起動したり、ユーザーを認証したりしません。ディスプレイマネージャに関する詳細は項7.5.2でお読み下さい。

`startx`コマンドが実行される時、それは、ユーザーのホームディレクトリ内の`.xinitrc`ファイルを検索してデスクトップ環境と、そして他のX クライアントアプリケーションを実行するように定義し

ます。.xinitrcファイルがない場合、システムデフォルトの/etc/X11/xinit/xinitrcファイルを代わりに使用します。

デフォルトのxinitrcスクリプトは、その後ユーザーのホームディレクトリ内で.Xresources、.Xmodmap、及び.Xkbmapや、さらに/etc/X11/ディレクトリ内のXresources、Xmodmap、及びXkbmapを含むユーザー定義のファイルやデフォルトのシステムファイルを探します。存在していれば、XmodmapとXkbmapは、xmodmapユーティリティに使用されて、キーボードを設定します。Xresourcesファイルが、アプリケーションへの特定のユーザー設定値を割り当てる為に読み込まれます。

これらのオプションの設定のあとは、xinitrcスクリプトが/etc/X11/xinit/xinitrc.d/ディレクトリに置かれている全てのスクリプトを実行します。このディレクトリの重要なスクリプトの1つはデフォルト言語の設定などの構成をするxinputです。

次にxinitrcスクリプトは、ユーザーのホームディレクトリ内の.Xclientsを実行しようとして、それがない場合には/etc/X11/xinit/Xclientsへ向かいます。Xclientsの目的は、デスクトップ環境を、または単に基本のウィンドウマネージャを起動するためです。ユーザーのホームディレクトリ内の.Xclients スクリプトは.Xclients-defaultファイルにあるユーザー指定のデスクトップ環境を起動します。ユーザーのホームディレクトリ内に.Xclientsがなければ、標準の/etc/X11/init/Xclientsスクリプトがもう1つのデスクトップ環境を開始する試みをします。GNOMEを最初に試し、次にKDE、そしてtwmと続きます。

ランレベル3でのX のログアウトをした後では、ユーザーはテキストモードのユーザーセッションに戻ります。

7.5.2. ランレベル5

システムがランレベル5で起動する時、ディスプレイマネージャと呼ばれる特殊なX クライアントアプリケーションが起動します。ユーザーは、デスクトップ環境、又はウィンドウマネージャが起動する前に、ディスプレイマネージャを使用して認証する必要があります。

システム上にインストールされているデスクトップ環境によっては3種類のディスプレイマネージャがユーザー認証に利用できます。

- gdm — Red Hat Linux用のデフォルトのディスプレイマネージャであるgdmによって、ユーザーは言語の選択、シャットダウン、再起動、及びシステムへのログイン等が出来ます。
- kdm — KDEのディスプレイマネージャによりユーザーはシャットダウン、再起動、及びシステムへのログインができます。
- xdm — 非常に基本的なディスプレイマネージャでこれにより、ユーザーがシステムにログインできるようになります。

ランレベル5 でブートする時、prefdmスクリプトは/etc/sysconfig/desktopファイルを参照することにより、好みのディスプレイマネージャを決定します。このファイルに利用できるオプションのリストに付いては、/usr/share/doc/initscripts-<*version-number*> /sysconfig.txtファイル(<*version-number*>にはinitscriptsパッケージのバージョン番号が入ります。)を参照して下さい。

それぞれのディスプレイマネージャは/etc/X11/xdm/Xsetup_0を参照して、ログイン画面をセットします。ユーザーがシステムにログインすると、/etc/X11/xdm/GiveConsoleスクリプトが実行され、コンソールの所有者をユーザーに割り当てます。その後、/etc/X11/xdm/Xsessionスクリプトが実行されて、通常はランレベル3 からX をスタートする時にxinitrc スクリプトにより実践される多くのタスクが達成されます。これにはシステムとユーザーリソースの設定、及び/etc/X11/xinit/xinitrc.d/ディレクトリのスクリプトの実行が含まれます。

ユーザーは、gdm又はkdmのディスプレイマネージャを使って認証をする時、**Session**メニューから(パネル上の**メインメニューボタン** =>**個人設定** => **More Preferences** => **セッション**と進んでアクセスできます。)選択して使用するデスクトップ環境を指定することができます。ディスプレイマネージャ内でデスクトップ環境が指定されない場合、/etc/X11/xdm/Xsessionスクリプト

がユーザーのホームディレクトリにある.xsessionファイルと.Xclients ファイルをチェックして、どちらのデスクトップ環境をロードするか判定します。最後の手段としてランレベル3 と同様に、/etc/X11/xinit/Xclientsファイルを使用して1つのデスクトップ環境、またはウィンドウマネージャが選択されます。

デフォルトのディスプレイ（:0）でXセッションを終了し、ログアウトするときは、/etc/X11/xdm/TakeConsoleスクリプトが実行し、コンソールの所有権をルートユーザーに再度割り当てます。ユーザーがログインした後、実行を続けていたオリジナルのディスプレイユーザーは、新しいディスプレイマネージャを生成します。これにより、XFree86サーバーが再起動し、新規ログインウィンドウを表示して再度プロセス全体を起動します。

ユーザーがランレベル5からX のログアウトをすると、ディスプレイマネージャに戻ります。

ディスプレイマネージャがユーザー認証を行う方法にかんしては/usr/share/doc/gdm-<*version-number*>/README (この<*version-number*>にはインストールされているgdmパッケージのバージョン番号が入ります)とxdmのmanページを参照して下さい。

7.6. その他のリソース

XFree86サーバー、このサーバーに接続されているクライアント、さまざまなデスクトップ環境とウィンドウマネージャについては他にも多量の情報があります。

7.6.1. インストールされているドキュメント

- /usr/X11R6/lib/X11/doc/README — XFree86アーキテクチャについて、また新規ユーザーとしてXFree86プロジェクトに関するその他の情報の入手方法について簡単に説明しています。
- /usr/X11R6/lib/X11/doc/RELNOTES — 高度な知識を持つユーザーを対象としており、XFree86で使用可能な最新機能について説明しています。
- man XF86Config — XFree86設定ファイルに関する情報が入っています。ファイル内のさまざまなセクションの構文の意味も網羅しています。
- man XFree86 — すべてのXFree86情報に関する基本的なmanページであり、ローカルとネットワークのXサーバー接続の違いを説明し、共通環境変数を検証し、コマンドラインオプションを一覧表示し、有効な管理用のコントロールキーの組み合わせを示します。
- man Xserver — X ディスプレイサーバーを説明しています。

7.6.2. 役に立つWebサイト

- http://www.xfree86.org —XFree86プロジェクトのホームページ。X Window SystemのXFree86オープンソースバージョンを紹介します。必須ハードウェアを制御し、GUI環境を提供するために、XFree86はRed Hat Linuxにバンドルされています。
- http://dri.sourceforge.net — DRI（Direct Rendering Infrastructure）プロジェクトのホームページ。DRIはXFree86のコアハードウェア3Dアクセラレーションコンポーネントです。
- http://www.redhat.com/mirrors/LDP/HOWTO/XFree86-HOWTO — XFree86のマニュアルインストールとカスタム設定について詳述しているHOWTOドキュメント。
- http://www.gnome.org/ — GNOMEプロジェクトのホームページ。
- http://www.kde.org/ — KDEデスクトップ環境のホームページ。
- http://nexp.cs.pdx.edu/fontconfig/ — XFree86用のFontconfigフォントサブシステムのホームページです。

7.6.3. 参考書籍

- *The Concise Guide to XFree86 for Linux* Aron Hsiao著/Que刊— LinuxシステムにおけるXFree86のオペレーションについて専門家の視点を紹介します。
- *The New XFree86* Bill Ball著/Prima Publishing刊— XFree86の全体像と、GNOMEやKDEなどの一般的なデスクトップ環境とXFree86との関係について詳しく説明します。
- *Beginning GTK+ and GNOME*（Peter Wright著/Wrox Press, Inc.刊）—プログラマにGNOMEアーキテクチャを紹介し、GTK+で開始する方法を示します。
- *GTK+/GNOME Application Development* Havoc Pennington著/New Riders Publishing刊— GTK+プログラミングの核心について高度な解説をしています。サンプルコードと使用可能なAPIの全体像に重点を置いています。
- *KDE 2.0 Development*（David Sweet、Matthias Ettrich著/ Sams Publishing刊）—開発を担当する初心者と熟練者に対して、KDE用に作成されたQTアプリケーションに必要な多数の環境ガイドラインの使用法について説明します。

II. ネットワークサービスへの参照

Red Hat Linux では、幅広いさまざまなネットワークサービスを活用することができます。このセクションでは、ネットワークインターフェイスを設定する方法と共に、NFS、Apache HTTP サーバー、Sendmail、Fetchmail、Procmail、BIND、LDAPのような重要なネットワークサービスについての詳細情報を提供しています。

目次

第8章
ネットワークインターフェイス

Red Hat Linux を使用したすべてのネットワーク通信は、設定したソフトウェアインターフェイスとシステムに接続された物理的なネットワークデバイス間で行われます。

各種ネットワークインターフェイス用の設定ファイルとインターフェイスをアクティブ/非アクティブにするスクリプトは`/etc/sysconfig/network-scripts`ディレクトリにあります。インターフェイスファイルの数量と種類はシステムごとに異なりますが、3種類のカテゴリのファイルがこのディレクトリに存在します：

- インターフェイス設定ファイル
- インターフェイス制御スクリプト
- ネットワーク機能ファイル

これらの各カテゴリのファイルは、Red Hat Linuxの元で合同で機能し各種ネットワークデバイスを動作させます。

この章ではこれらのファイル間での関係とそれらの使用方法について説明していきます。

8.1. ネットワーク設定ファイル

インターフェイスの設定ファイルについて探求する前に、先ずネットワーク設定で使用される主要な設定ファイルを項目別に分けていきます。ネットワークスタックの設定の内でのこれらのファイルの役割を理解すると、Red Hat Linuxシステムをカスタマイズする時に役に立ちます。

主要なネットワーク設定ファイルは、次のようになります：

- `/etc/hosts` — このファイルの主な目的は、他の方法では解決できないホスト名を解決することです。またDNSサーバのない小規模のネットワーク上のホスト名の解決にも使用できます。コンピュータが加入しているネットワークの種類に関係なく、このファイルはループバックデバイスのIPアドレス(`127.0.0.1`)を指定する行を`localhost.localdomain`として含む必要があります。詳細はhostsのmanページを御覧下さい。
- `/etc/resolv.conf` — このファイルは、DNSサーバと検索ドメインのIPアドレスを指定します。他の設定がない限りこのファイルはネットワーク初期化スクリプトだけで構成されます。このファイルに関する詳細は`resolv.conf`のmanページを御覧下さい。
- `/etc/sysconfig/network` — すべてのネットワークインターフェイス用のルーティングとホストの情報を指定します。このファイルとそれが受け付けるディレクティブに関する詳細情報は項4.1.23で御覧下さい。
- `/etc/sysconfig/network-scripts/ifcfg-<`*`interface-name`*`>` — Red Hat Linuxシステム上のそれぞれのネットワークインターフェイスの為に、それに対応するそれぞれのインターフェイス設定スクリプトがあります。これらの各ファイルは、そのネットワークインターフェイス特定の情報を提供します。このタイプのファイルとそれが受け付けるディレクティブに関する情報は項8.2を参照して下さい。

用心

`/etc/sysconfig/networking/`ディレクトリは**ネットワーク管理ツール** (`redhat-config-network`)で使用されており、その内容は手動で編集されるべきではありません。**ネットワーク管理ツール**を使用したネットワークインターフェイス設定の詳細については*Red Hat Linux* カスタマイズガイドの中に掲載されているネットワーク設定の章を御覧下さい。

8.2. インターフェイス設定ファイル

インターフェイス設定ファイルは、個々のネットワークデバイス用のソフトウェアインターフェイスを制御します。システムはブートする時、これらのファイルを使用してどのインターフェイスを立ち上げるか、及びそれらをどのように構成するかを決定します。これらのファイルは通常、ifcfg-<*name*> と名付けられ、<*name*>の部分には設定ファイルが制御するデバイスの名前が入ります。

8.2.1. イーサネットインターフェイス

最も一般的なインターフェイスファイルの1つはifcfg-eth0で、これはシステム内の最初のイーサネットネットワークインターフェースカード すなわちNICを制御します。システム内に複数のNICがある場合は、複数のifcfg-eth<*X*>ファイル(<*X*>は特定のインターフェイスに対する独自の番号) を用意します。各デバイスには独自の設定ファイルがあるので、管理者はそれぞれのインターフェイス機能を別々に制御できます。

以下に固定IPアドレスを使用するifcfg-eth0ファイルのサンプルを示します：

```
DEVICE=eth0
BOOTPROTO=none
ONBOOT=yes
NETWORK=10.0.1.0
NETMASK=255.255.255.0
IPADDR=10.0.1.27
USERCTL=no
```

インターフェイス設定ファイルで要求される値はほかの値によって変わることがあります。たとえば、DHCPを利用するインターフェイスのifcfg-eth0ファイルは、IP情報がDHCPサーバーより供給されるため、次のように少し異なっています：

```
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
```

ネットワーク管理ツール(redhat-config-network)を使用すると各種ネットワークインターフェイスの変更が簡単に実行できます。(このツールの使用法に関する詳細は*Red Hat Linux* カスタマイズガイドの中のネットワーク設定の章を御覧下さい。

但し、任意のネットワークインターフェイスの設定ファイルは手動で編集することも出来ます。

イーサネットインターフェイスの設定ファイル内の設定可能なパラメータを一覧で以下に示します：

- BOOTPROTO=<*protocol*>, ここで<*protocol*>は次のいずれかです：
  - none — 起動時プロトコルを使用してはいけません。
  - bootp — BOOTPプロトコルを使用しなければいけません。
  - dhcp — DHCPプロトコルを使用しなければいけません。

- BROADCAST=<*address*>, ここで<*address*>はブロードキャストアドレスです。このディレクティブは古くなりました。
- DEVICE=<*name*>, ここで<*name*>は、物理デバイスの名前です(論理名である動的割り当てPPPデバイスを除く)。

- DNS*{1,2}*=<*address*>, ここで<*address*>はネームサーバアドレスで、もしPEERDNSディレクティブがyesにセットされている場合は、/etc/resolv.confに配置されます。
- IPADDR=<*address*>, ここで<*address*>はIPアドレスです。
- NETMASK=<*mask*>, ここで<*mask*>はネットマスク値です。
- NETWORK=<*address*>, ここで<*address*>はネットワークアドレスです。このディレクティブは古くなっています。
- ONBOOT=<*answer*>, ここで<*answer*>は以下のいずれかです：
  - yes — このデバイスは起動時に有効にする必要があります。
  - no — このデバイスは起動時に有効にしてはいけません。

- PEERDNS=<*answer*>, ここで<*answer*>は以下のいずれかです：
  - yes — DNSディレクティブがセットしてある場合は、/etc/resolv.confを変更します。DCHPを使用する場合、yesがデフォルトです。
  - no — /etc/resolv.confを変更しません。
  - SRCADDR=<*address*>, ここで<*address*>は送信パケット用の指定されたソースIPアドレスです。
  - USERCTL=<*answer*>, ここで<*answer*>は以下のいずれかです：
    - yes — rootでないユーザーは、このデバイスを制御できます。
    - no — rootでないユーザーは、このデバイスを制御できません。

8.2.2. ダイヤルアップインターフェイス

ダイヤルアップ接続を通してインターネットに接続する場合、インターフェイスの設定ファイルが必要です。

PPP インターフェイスファイルは次の形式、ifcfg-ppp<*X*> を使用して名前が付いています。(<*X*> は特定のインターフェイスに対する独自の番号です。)

PPPインターフェイス設定ファイルは、wvdialか、**ネットワーク管理ツール**か、又は**Kppp**がダイヤルアップアカウントの作成に使用された時に、自動的に作成されます。*Red Hat Linux* 入門ガイドには、これらのGUIベースのダイヤルアップ接続ツールの使用法に関する案内が含まれています。このファイルも手動で作成と編集ができます。

以下に標準的なifcfg-ppp0ファイルを示します：

```
DEVICE=ppp0
NAME=test
WVDIALSECT=test
MODEMPORT=/dev/modem
LINESPEED=115200
PAPNAME=test
USERCTL=true
ONBOOT=no
PERSIST=no
DEFROUTE=yes
PEERDNS=yes
DEMAND=no
IDLETIMEOUT=600
```

SLIP(Serial Line Internet Protocol) はもう1つのダイアルアップインターフェイスですが、一般には使用されなくなっています。SLIPファイルのインターフェイス設定ファイル名には、`ifcfg-sl0` などがあります。

まだ説明されていない他のオプションの中で、これらのファイルで使用できるものを以下に示します：

- `DEFROUTE=`<*answer*>, ここで<*answer*>は以下のいずれかです：
  - `yes` — このインターフェイスをデフォルトルートとして設定します。
  - `no` — このインターフェイスをデフォルトルートとして設定しません。

- `DEMAND=`<*answer*>, ここで<*answer*>は以下のいずれかです：
  - `yes` — このインターフェイスにより、接続を試みられたときに`pppd`はその接続を開始します。
  - `no` — このインターフェイスの接続は手動で確立する必要があります。

- `IDLETIMEOUT=`<*value*>, ここで<*value*>はインターフェイスが自己切断するまで活動停止している秒数です。
- `INITSTRING=`<*string*>, ここで<*string*>はモデムデバイスに渡されるinit文字列です。このオプションは主にSLIPインターフェイスと併用されます。
- `LINESPEED=`<*value*>, ここで<*value*>はデバイスのボーレートです。取り得る標準値には`57600`, `38400`, `19200`, `9600` があります。
- `MODEMPORT=`<*device*>, ここで<*device*>はこのインターフェイスの接続を確立するために使用するシリアルデバイスの名前です。
- `MTU=`<*value*>, ここで<*value*>はこのインターフェイスの*MTU*（*Maximum Transfer Unit*）設定値です。MTUはヘッダー情報を除いた、1フレームが転送できるデータの最大バイト数を表します。ダイアルアップの場合、MTUの値を`576`に設定すると、脱落するパケットが減り、接続のスループットが少し改善されます。
- `NAME=`<*name*>, ここで<*name*>は一連のダイアルアップ接続設定に与えられているタイトルの合同参照名です。
- `PAPNAME=`<*name*>, ここで<*name*>はリモートシステムに接続できるようにするために行われる*PAP*（*Password Authentication Protocol*）交換時に与えられるユーザー名です。
- `PEERDNS=`<*answer*>, ここで<*answer*>は次のいずれかになります：
  - `yes` — 接続が確立したときにリモートシステムによって提供されるDNSサーバーを使用するために、このインターフェイスはシステムの`/etc/resolv.conf`ファイルエントリを変更します。
  - `no` — `/etc/resolv.conf` ファイルは変更されません。

- `PERSIST=`<*answer*>, ここで<*answer*>は次のいずれかになります：
  - `yes` — このインターフェイスは、たとえモデムがハングアップした後に停止されても、常にアクティブのままにする必要があります。
  - `no` — このインターフェイスは常にアクティブのままにしてはいけません。

- `REMIP=`<*address*>, ここで<*address*>はリモートシステムのIPアドレスです。これは、通常、指定しないでおきます。

- WVDIALSECT=<*name*>, ここで<*name*>は/etc/wvdial.confのダイアラ設定とこのインターフェイスを関連付けます。ダイアラ設定には、ダイアルする電話番号、インターフェイスの重要情報などが含まれています。

8.2.3. 他のインターフェイス

これらのオプションを使用する他の一般的なインターフェイス設定には次の項目が含まれます：

- ifcfg-lo — ローカルのループバックインターフェイスはよくテストで使用されるだけでなく、同じシステムを指定し直すIPアドレスを必要とするさまざまなアプリケーションでも使用されます。ループバックデバイスに送信されたデータはすぐにホストのネットワーク層に戻されます。


警告

決してループバックインターフェイスのスクリプトである/etc/sysconfig/network-scripts/ifcfg-loは手動で編集しないで下さい。編集するとシステムの正常な動作が妨害される可能性があります。

- ifcfg-irlan0 —赤外線インターフェイスによって、ラップトップとプリンタなどデバイス間の情報を赤外線リンク上で流すことができます。これは、通常ピアツーピア接続で可能という事以外はイーサネットと同じような方法で動作します。
- ifcfg-plip0 —*PLIP (Parallel Line Interface Protocol)*接続も、これがパラレルポートを使用すること以外は殆んど同様な方法で動作します。
- ifcfg-tr0 — トークンリングトポロジーは以前程*LAN* (*Local Area Networks*)上で一般的ではありません。イーサネットのよって取り残されています。

8.2.4. エイリアスファイルとクローンファイル

使用頻度の少ない2種類のインターフェイス設定ファイルが/etc/sysconfig/network-scriptsディレクトリにあり、それらはエイリアスファイルとクローンファイルです。

エイリアスインターフェイス設定ファイルにはifcfg-<*if-name*>: <*alias-value*>の書式の名前を使用することで、エイリアスがインターフェイスを指すようになります。たとえば、ifcfg-eth0:0ファイルならば、DEVICE=eth0:0と静的IPアドレス10.0.0.2を指定するように設定でき、ifcfg-eth0のDHCPよりIP情報を受け取るようにすでに設定されているイーサネットインターフェイスのエイリアスとして機能します。そのとき、eth0デバイスは動的IPアドレスにバインドされますが、そのシステム上では固定IPアドレス10.0.0.2を介していつも参照できます。

クローンインターフェイス設定ファイルにはifcfg-<*if-name*> -<*clone-name*>の命名慣習に従います。エイリアスファイルは既存のインターフェイス設定ファイルを参照する別の方法ですが、クローンファイルはインターフェイスを指定する際に追加オプションの指定に使用します。たとえば、eth0という標準のDHCPイーサネットインターフェイスの場合は、次のようなものになります：

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
```

設定されていない場合、USERCTLはnoにセットされますので、ユーザーはこのインターフェイスをアップ/ダウンすることはできません。この操作をユーザーができるようにするには、ifcfg-eth0をifcfg-eth0-userにコピーしてクローンを作成し、以下の行を追加します：

```
USERCTL=yes
```

ifup eth0-userコマンドを使用してユーザーがeth0 インターフェイスをアップする措置を講じると、ifcfg-eth0とifcfg-eth0-userからの設定オプションが一緒に使用されます。これは非常に基本的な例ですが、この方法はさまざまなオプションとインターフェイスで使用できます。

インターフェイス設定のエイリアスファイルとクローンファイルを作成する最も簡単な方法はグラフィカルな**ネットワーク管理ツール**を使用することです。このツールの使用に関する情報は*Red Hat Linux* カスタマイズガイドの中のネットワーク設定の章を御覧下さい。

8.3. インターフェイス制御スクリプト

インターフェイス制御スクリプトは、インターフェイス接続のアクティブ化と非アクティブ化を制御します。おもなインターフェイス制御スクリプトとしては`/sbin/ifdown`と`/sbin/ifup`の2つがあり、`/etc/sysconfig/network-scripts` ディレクトリにあるさまざまな種類の制御スクリプトをコールします。

`ifdown`と`ifup`のインターフェイスは、`/sbin/`ディレクトリにあるスクリプトへのシンボリックリンクです。どちらかのスクリプトが呼び出されると、次のような、指定されるべきインターフェイスの値を要求します：

```
ifup eth0
Determining IP information for eth0... done.
```

その時点で、`/etc/rc.d/init.d/functions`ファイルと`/etc/sysconfig/network-scripts/network-functions`ファイルは多彩なタスクを実行するのに使用されます。詳細については項8.4 を御覧下さい。

インターフェイスが1つ指定済みであることと、要求を実行中のユーザーがそのインターフェイスの制御をする許可があることを確認した後、正しいタイプのスクリプトが呼び出されて、インターフェイスをアップ/ダウンします。以下に一般的なインターフェイス制御スクリプトを示します：

- `ifup-aliases` — 複数のIPアドレスが1つのインターフェイスに関連付けられているとき、インターフェイス設定ファイルからIPエイリアスを設定します。
- `ifdown-cipcb`と`ifup-cipcb` — *CIPE*（*Crypto IP Encapsulation*） 接続のアップ/ダウンに使用します。
- `ifdown-ipv6` と`ifup-ipv6` — さまざまなインターフェイス設定ファイルと`/etc/sysconfig/network`の環境変数を使用するIPv6関連の機能呼び出しを含みます。
- `ifup-ipx` — IPXインターフェイスのアップに使用します。
- `ifup-plip` — PLIPインターフェイスのアップに使用します。
- `ifup-plusb` — ネットワーク接続用USBインターフェイスのアップに使用します。
- `ifdown-post` と`ifup-post` — ある特定のインターフェイスをアップ/ダウンした後に実行するコマンドを含みます。
- `ifdown-ppp` と`ifup-ppp` — PPPインターフェイスをアップ/ダウンするために使用します。
- `ifup-routes` — デバイスの静的ルートを、そのインターフェイスがアップするときに追加します。
- `ifdown-sit` と`ifup-sit` — IPv4接続内にあるIPv6トンネルのアップ/ダウンに関連した機能呼び出しを含みます。
- `ifdown-sl` と`ifup-sl` — SLIPインターフェイスのアップ/ダウンに使用します。

警告

`/etc/sysconfig/network-scripts/`ディレクトリのスクリプトを削除や変更すると、各種のインターフェイス接続のおかしな動作や、その停止等の原因となることに注意してください。経験豊富な上級ユーザーのみ、ネットワークインターフェイス関連のスクリプトを変更すべきです。

すべてのネットワークスクリプトを同時に操作する最も簡単な方法は、ネットワークサービス(/etc/rc.d/init.d/network)上の/sbin/serviceコマンドを使用することです。以下のコマンドの例のようにします：

```
/sbin/service network <action>
```

この例では、<*action*>ではstartか, stopか, restartのいずれかを使用します。

設定したデバイスと現在アクティブになっているネットワークインターフェイスの一覧を表示するには、次のコマンドを使用します：

```
/sbin/service/network status
```

8.4. ネットワーク機能ファイル

Red Hat Linuxは、さまざまな方法で使用される重要な機能を含む複数のファイルを使用して、インターフェイスをアップ/ダウンします。各インターフェイス制御ファイルに同じ機能を別々に含める代わりに、これらの機能は使いやすいようにグループ化して数ファイルにまとめ、必要なときに使えるようにします。

/etc/sysconfig/network-scripts/network-functionsファイルには多くのインターフェイス制御スクリプトに便利で最も一般に使用されるIPv4機能が含まれています。これらの機能にはインターフェイスのステータスの変化に関する情報を要求したプログラムの実行、ホスト名の設定、ゲートウェイデバイスの検索、ある特定デバイスのアップ/ダウンの確認、デフォルトルートの追加などがあります。

IPv6インターフェイスに必要な機能はIPv4インターフェイスのものとは異なりますので、この情報を保持するためにnetwork-functions-ipv6ファイルが特別に存在します。IPv6サポートは、そのプロトコルによって通信するためにカーネルで有効になっていなければいけません。IPv6サポートの有無を調べる機能がnetwork-functions ファイルにあります。さらに、静的IPv6ルートの作成/削除、トンネルの作成/削除、インターフェイスへのIPv6アドレスの追加/削除、あるインターフェイス上でIPv6アドレスの存在をテストする機能もこのファイルにあります。

8.5. その他のリソース

以下のリソースには、ネットワークインターフェイスに関する情報が含まれており、以下のような場所で入手できます。

8.5.1. インストールされているドキュメント

- /usr/share/doc/initscripts-<*version*>/sysconfig.txt — この章でカバーされていないIPv6オプションを含む、ネットワーク設定ファイル用に利用可能なオプションへの総合ガイド。
- /usr/share/doc/iproute-<*version*>/ip-cref.ps — このPostscript™ファイルには、ルーティングテーブルの操作や他の操作に使用できるipコマンドについての豊富な情報が含まれています。このファイルを表示するには、**ghostview**又は、**kghostview**アプリケーションを使用して下さい。

redhat.

第9章

NFS（Network File System）

*NFS (Network File System)*を利用すると、ホストはあるリモートシステムのパーティションをマウントし、それらのパーティションがローカルファイルシステムであるかのように使用できます。これにより、システム管理者はネットワーク上の中央にリソースを設置して、許可を持つユーザーがそこにあるファイルにいつでもアクセスすることができます。

2つのバージョンのNFSが現在使用されています。NFSバージョン2(NFSv2)は、ここ数年間にわたる実績があり、各種オペレーティングシステムによって広くサポートされています。NFSバージョン3(NFSv3)には機能がさらに追加されており、可変ファイル処理サイズやエラー報告機能の向上が含まれています。Red Hat Linuxは、NFSv2とNFSv3の両方をサポートし、NFSv3をサポートしているサーバーと接続するときはデフォルトでNFSv3を使用します。

この章では、NFSバージョン2を主に取り扱いますが、紹介する概念の多くはバージョン3にも適用されます。ここでは、基本的なNFSの概念と補足情報だけ示します。クライアントマシンとサーバーマシン上でのNFSの設定と動作に関する詳しい説明については、*Red Hat Linux* カスタマイズガイドの*NFS (*ネットワークファイルシステム*)*の章を参照してください。

9.1. 方法論

Linuxはカーネルレベルのサポートと常時稼働しているデーモンプロセスの組み合わせを使用してNFSファイル共有機能を提供します。しかし、LinuxカーネルのNFSサポートが機能するように有効になっていなければいけません。NFSは*RPC*（*Remote Procedure Call*）を使用してクライアント、サーバー間で要求をルーティングします。つまり、`portmap`サービスが有効で、NFS通信を行うために適正なランレベルでアクティブでなければならないということを意味します。以下のさまざまなプロセスが`portmap`と共に動作して、既知のNFS接続が許可をうけエラーなしに処理されることを確実にします：

- `rpc.mountd` — NFSクライアントからマウント要求を受け取り、現在エクスポートされているファイルシステムと合致するかどうかをチェックする稼働中のプロセス。
- `rpc.nfsd` — ユーザーレベル部分のNFSサービスを実装したプロセス。Linuxカーネルと一緒に動作して、NFSクライアントが使用するサーバースレッドの追加など、NFSクライアントの動的要求に応えます。
- `rpc.lockd` — 最近のカーネルでは不要のデーモン。NFSファイルロックは現在ではカーネルによって行われています。デフォルトではこの機能が含まれていない旧カーネルを利用するユーザー向けに`nfs-utils`パッケージに含まれています。
- `rpc.statd` — *NSM*（*Network Status Monitor*） RPCプロトコルを実装します。NFSサーバーが正常に停止しないで再起動されたときにリブート通知をします。
- `rpc.rquotad` —ユーザー割り当て量をリモートユーザーに提供するRPCサーバー。

これらのプログラムのすべてがNFSサービスに必要なわけではありません。有効にしなければならないサービスは、`rpc.mountd`、`rpc.nfsd`、`portmap`だけです。他のデーモンは追加の機能を提供しますが、サーバー環境がそれらを要求する場合にのみ使用すべきものです。

NFSバージョン2は*UDP*（*User Datagram Protocol*）を使用して、クライアントとサーバーの間でstatelessネットワーク接続を提供します。NFSバージョン3はUDP又はIP上で動作するTCPを使用します。クライアントが共有ボリュームにアクセスすることを許可された後にNFSサーバーはクライアントにクッキーを送信する為、stateless UDP接続はネットワークトラフィックを最小限に抑えます。このクッキー、つまりサーバー側に格納される乱数値はクライアントからサーバーへのRPC要求と共に渡されます。NFSサーバーはクライアントに影響を与えることなく再起動でき、クッキーはそのまま残ります。

NFSは、クライアントシステムがリモートファイルシステムをマウントしようとする時にのみ、認証を実行します。アクセスを制限する為に、NFSサーバーは先ずTCPラッパーを使用します。TCPラッパーは、/etc/hosts.allowファイルと/etc/hosts.denyファイルを読み込み、ある特定のクライアントがNFSサーバーへのアクセスを許可されるか拒否されるかを決定します。TCPラッパーのアクセス制御の設定に関する情報は第15章で確認して下さい。

TCPラッパーによってクライアントがアクセスを認可された後は、NFSサーバーがその設定ファイル/etc/exportsを参照して、このクライアントがエクスポートのファイルシステムをマウント出来るかどうかを決定します。認可が出た後は、どんなファイルやディレクトリの操作もRPC(Remote Procedure Call)を使用してサーバーに送ることができます。

警告

NFSマウント権限はユーザーではなく、クライアントに限定して許可しています。エクスポートされたファイルシステムは、リモートシステム上のどのユーザーからでもアクセスが可能です。

/etc/exportsを設定する時には、エクスポートしたファイルシステムの読み込み/書き込み権限の許可には充分に注意して下さい。

9.1.1. NFSとportmap

NFSは、RPC(remote procedure calls)に依存しながら機能します。portmapサービスはRPC要求を正しいサービスにマッピングするのに必要となります。RPCプロセスはスタートしたことをportmapに対し通知し、モニターしているポート番号とサービスする予定のRPCプログラムを表します。クライアントシステムは、その後、特定のRPCプログラム番号を持つサーバー上のportmapに連絡を取ります。そして、portmapは、目的のサービスと通信をする為に、クライアントを正しいポート番号にリダイレクトします。

RPCベースのサービスは着信するクライアント要求とすべての接続をするportmapに依存しているので、portmapはこれらのサービスが開始される前に利用可能になっていなければいけません。何らかの理由でportmapサービスが不意に停止したときは、portmapと、起動したときに稼働していたすべてのサービスを再起動してください。

portmapサービスはTCPラッパーのホストアクセスファイル(/etc/hosts.allow及び/etc/hosts.deny)と共に使用することが出来て、どのリモートシステムがサーバー上のRPCベースのサービスを使用する許可をもつか制御します。詳細は第15章を御覧下さい。portmapの為のアクセス制御ルールは全てのRPCベースのサービスに影響します。又は、特定のアクセス制御ルールが各NFS RPCデーモンを影響するように指定することも可能です。rpc.mountdとrpc.statdのmanページには、これらのルールの為の正確な構文に関する情報が含まれています。

9.1.1.1. portmapでNFSのトラブルシューティング

portmapは、RPCサービスとそれらの間の通信で使用されるポート番号の調整を提供しますので、トラブルシューティングをする時点でportmapを使用して現在のRPCサービスのステータスを表示するために役に立ちます。rpcinfoコマンドは、各RPCベースのサービスをポート番号、RPCプログラム番号、バージョン、及びIPプロトコルタイプ(TCP又はUDP)と共に表示します。

適正なNFS RPCベースのサービスがportmapに有効になっているかを確かめるには、rpcinfo -pを使用します：

```
program vers proto  port
 100000  2 tcp  111 portmapper
 100000  2 udp  111 portmapper
 100024  1 udp 1024 status
 100024  1 tcp 1024 status
 100011  1 udp  819 rquotad
```

```
  100011  2 udp  819 rquotad
  100005  1 udp 1027 mountd
  100005  1 tcp 1106 mountd
  100005  2 udp 1027 mountd
  100005  2 tcp 1106 mountd
  100005  3 udp 1027 mountd
  100005  3 tcp 1106 mountd
  100003  2 udp 2049 nfs
  100003  3 udp 2049 nfs
  100021  1 udp 1028 nlockmgr
  100021  3 udp 1028 nlockmgr
  100021  4 udp 1028 nlockmgr
```

-pオプションは、指定されたホストのポートマッパーを見つけ出します。特定のホストが一覧表示されていなければローカルホストをデフォルトとします。ほかのオプションはrpcinfoのmanページに記載されています。

上記の出力から、さまざまなNFSサービスが稼働していることがわかります。NFSの1つが正常に起動していなければ、portmapはそのサービスのクライアントからのRPC要求を正しいポートにマッピングできなくなります。多くの場合、rootとしてNFSを再起動する（/sbin/service nfs restart）と、これらのサービスはportmapに正しく登録され、稼働し始めます。

9.2. NFSサーバー設定ファイル

NFSを使用してファイルとディレクトリを共有するシステムを作成することは簡単です。NFSによりリモートユーザーにエクスポートされるすべてのファイルシステムと、これらのファイルシステムに関連付けられているアクセス権は、/etc/exportsファイルにあります。このファイルはexportfsコマンドによって読み出され、rpc.mountdとrpc.nfsdに、許可ホストがファイルシステムのリモートマウントをするために必要な情報を提供します。

exportfsコマンドを使用すれば、rootユーザーはさまざまなNFSサービスを再起動しなくてもディレクトリを選択してエクスポート/アンエクスポートできます。exportfsに適切なオプションが渡されると、エクスポート対象ファイルシステムが/var/lib/nfs/xtabに書き込まれます。ファイルシステムに対するアクセス権を決定するときにrpc.mountdはxtabファイルを参照するので、エクスポートファイルシステムの一覧に対する変更はただちに有効になります。

exportfsを使用するときは、次のような各種のオプションが利用できます：

- -r — /etc/exportsに一覧表示されているすべてのディレクトリは、/etc/lib/nfs/xtabに新しいエクスポート一覧を作成することによってエクスポートされます。このオプションは/etc/exportsになされた変更を使って事実上このエクスポート一覧を更新します。
- -a — exportfsに渡されるほかのオプションに基づいて、すべてのディレクトリはエクスポート/アンエクスポートされます。
- -o *options* —この使用によりユーザーは/etc/exportsに一覧表示されていないディレクトリをエクスポート対象として指定できます。これらの追加のファイルシステム共有は、/etc/exportsの中に指定してあるものと同じように書き込まなければいけません。このオプションは、エクスポートファイルシステムを、エクスポート対象ファイルシステムの一覧に永続的に追加する前にテストする目的で使用されます。
- -i — /etc/exportsを無視します。コマンドラインから出されるオプションのみがエクスポートファイルシステムの定義に使用されます。
- -u — ディレクトリをリモートユーザーがマウントしないようにアンエクスポートします。コマンドexportfs -uaは、さまざまなNFSデーモンが稼働している間、NFSファイル共有を事実上サスペンドします。NFS共有を継続できるようにするには、exportfs -rを入力します。

- -v — 冗長動作。exportfsコマンドを実行する時、エクスポート/アンエクスポートするファイルシステムは、より詳しい内容表示をします。

exportfsコマンドにオプションが渡されないときは、現在エクスポートされているファイルシステムの一覧が表示されます。

/etc/exportsへの変更内容も、service nfs reloadコマンドを使ってNFSサービスをリロードすることで読み込まれます。これによって、/etc/exportsファイルを再エクスポートする間もNFSデーモンは稼働し続けます。

9.2.1. /etc/exports

/etc/exportsファイルはどのファイルシステムがリモートホストにエクスポートされるかを制御して、オプションを指定します。空白行は無視され、コメントは「#」マークを使用して作成でき、長い行はバックスラッシュ(\)を使用してラップできます。各エクスポートファイルシステムは、それぞれ独自の行上になければいけません。。エクスポートファイルシステムの後ろに置かれた許可ホストの一覧は、空白文字で区切られていなければいけません。ホストそれぞれのオプションはホスト識別子の直後にあるかっこ( )内に配置し、ホスト識別子と最初のかっこ( )の間に空白があってはいけません。

その最も単純な書式で/etc/exportsが必要とするのは、エクスポート対象ディレクトリと、そのディレクトリの使用を許可されているホストだけです：

```
/some/directory bob.example.com
/another/exported/directory 192.168.0.3
```

/sbin/service nfs reloadコマンドを使用して/etc/exportsを再エクスポートした後、bob.example.comホストは/some/directoryをマウントでき、192.168.0.3は/another/exported/directoryをマウントできます。この例ではオプションを指定していないので、次のようなデフォルトのNFSプリファレンスが有効になります：

- ro — 読み取り専用。このファイルシステムをマウントしているホストはファイルシステムを変更できなくなります。ホストがファイルシステムの変更をできるようにするには、rw（読み書き）を指定しなければいけません。
- async — サーバーは適切と判断したときにデータをディスクに書き込むことができます。ホストがデータを読み取り専用としてアクセスしていればこれは重要ではありませんが、ホストが読み書きファイルシステムに変更を加えているときにサーバーがクラッシュした場合、データは消失する可能性があります。sync オプションを指定することによって、クライアントの書き込み要求が実際に完了するまではすべてのファイル書き込みをディスクにコミットしなければいけません。これは、パフォーマンスを低下させる可能性があります。
- wdelay — 別の書き込み要求が発生するおそれがあるとき、NFSサーバーはディスクへの書き込みを遅らせます。この結果、別々の書き込みコマンドによってディスクにアクセスする回数が減り、書き込みオーバーヘッドが少なくなることでパフォーマンスが向上します。no_wdelayはこの機能をオフにしますが、利用できるのは、syncオプションを使用している場合だけです。
- root_squash — nobodyのユーザーIDを与えることにより、リモート接続しているrootユーザーからrootの権限を取り上げます。これは事実上、リモートrootユーザーの能力を最低のローカルユーザーまで「押し下げ」て、リモートrootユーザーがローカルシステムのrootユーザーであるかのように振る舞うのを阻止します。一方、no_root_squashオプションはrootの押し下げをオフにします。rootを含めてすべてのリモートユーザーを押し下げるには、all_squashオプションを使用します。ユーザーIDとグループIDを指定してある特定ホストからのリモートユーザーと一緒に使用するには、anonuidオプションとanongid オプションをそれぞれ使用します。この場合、リモートNFSユーザーの為に特別なユーザーアカウントを作成して、（anonuid=<*uid-value*>,anongid=<*gid-value*>）の共有と指定をすることができます。<*uid-value*>はユーザーID番号で、<*gid-value*>はグループID番号です。

これらのデフォルトに上書きするには、それを行うオプションを指定しなければいけません。たとえば、rwを指定しなければ、そのエクスポートが共有されるのは読み取り専用だけになります。すべてのエクスポートファイルシステムの個々のデフォルトは明示的に上書きしなければいけません。また、ほかのオプションはデフォルト値がないところでは使用できます。これには、サブツリーのチェックを無効にする能力、安全でないポートからのアクセス許可、及び安全でないファイルロックの許可(特定の初期NFSクライアント実装には必要)などが含まれます。使用頻度の少ないこれらのオプションについての詳細は、exports manページを参照してください。

ホスト名を指定するときには、次のような方法を使用します：

- *single host* —ある1つのホストを完全修飾ドメイン名/ホスト名/IPアドレスで指定しています。
- *wildcards* — *文字と?文字を使用して、特定の文字列に一致する完全修飾ドメイン名のグループ化に使用されます。但し、DNSの逆引き検索が失敗した場合、偶然機能する可能性のあるIPアドレスはワイルドカードでは使用しないで下さい。

  しかし、ワイルドカードは思いの外厳密であるので、完全修飾ドメイン名に使用するときは注意してください。たとえば、ワイルドカードとして*.example.comを使用すると、sales.example.comはエクスポートファイルシステムにアクセスできますが、bob.sales.example.comはできません。この両方に合致させ、さらにsam.corp.example.comにも合致させるには、*.example.com *.*.example.comを指定する必要があります。
- *IP networks* — より規模が大きいネットワーク内ではIPアドレスを元にホストを合致させることができます。たとえば、192.168.0.0/28は、192.168.0.0から192.168.0.15までの最初の16個のIPアドレスはエクスポートファイルシステムにアクセスできますが、192.168.0.16以上のものはできません。
- *netgroups* — @<*group-name*>として作成されたNISネットグループ名を使用できます。これは、事実上、このエクスポートファイルシステムのアクセス制御の下にNISサーバーを置くことになります。そこでは、/etc/exportsに影響させずにユーザーをNISグループに追加したり、NISグループから削除したりできます。

警告

/etc/exportsファイルをフォーマットする方法は非常に厳密で、特に空白文字の使用について重要です。エクスポートファイルシステムをホストから離し、ホスト同士を空白文字で離すことを忘れないでください。ただし、コメント行で使用する場合を除き、ファイル内にこれ以外の空白文字がないようにしてください。

たとえば、次の2行は同じ内容ではありません。

```
/home bob.example.com(rw)
/home bob.example.com (rw)
```

第1行では、bob.example.comのユーザーだけが/homeディレクトリに読み書きアクセスができます。第2行では、bob.example.comのユーザーはそのディレクトリを読み取り専用（デフォルト）のみでマウントできるだけですが、他のユーザーは読み書きでマウントできてしまいます。

9.3. NFSクライアント設定ファイル

サーバーによって利用可能になっているNFS共有は、すべて多様な方法を使用してマウントできます。共有は、mountコマンドを使用して手動でマウント出来ますが、この方法ではシステムが再起動する度にrootユーザーがmountコマンドを入力することになります。起動時に、自動的にNFS共有がマウントされる設定は/etc/fstabの修正、又はautofsサービスの使用を含む2種類があります。

9.3.1. /etc/fstab

/etc/fstabファイルに適切にフォーマットした行を配置すると、エクスポートファイルシステムを手動でマウントするのと同じ効果が得られます。/etc/fstabファイルはシステムの起動時に/etc/rc.d/init.d/netfsスクリプトによって読み取られます。そこに指定してあるNFS共有はどれもマウントされます。

NFSエクスポートをマウントする/etc/fstab行は、次のようなものになります：

```
<server>:</path/of/dir> </local/mnt/point> nfs <options> 0 0
```

<*server-host*>は、ファイルシステムをエクスポートするサーバーのホスト名、IPアドレス、完全修飾ドメイン名のいずれかです。

</*path/of/directory*>はエクスポートディレクトリへのパス(経路)です。

</*local/mount/point*>はエクスポートディレクトリをマウントするローカルファイルシステム上の場所を指定します。このマウントポイントは、/etc/fstabが読み込まれる前に存在する必要があります。そうしないとマウントはできません。

nfsオプションはマウントされるファイルシステムのタイプを指定します。

<*options*>の部分には、ファイルシステムのマウントする方法を指定します。例えば、オプションの部分がrw,suidを示す場合、エクスポートファイルシステムは読み書きでマウントされ、サーバーによって設定されたユーザーIDとグループIDが使用されます。ここでは、かっこ( )を使用しないことに注意して下さい。マウントオプションについての詳細は、項9.3.3を参照してください。

9.3.2. autofs

/etc/fstabを使用する上での難点の1つは、そのマウントされたファイルシステムをどれほど使用しているかに関係なく、そのマウントを所定の状態に保持するためにシステムが専用にリソースを提供しなければならないことです。これは1つや2つのマウントでは問題になりませんが、1度に多数のシステムへのマウントを保持するときは、システム全体のパフォーマンスが低下します。/etc/fstabに代わる方法はカーネルベースのautomount ユーティリティを使用することです。これを使用すると、NFSファイルシステムのマウント/アンマウントが自動的に行われ、リソースを節約できます。

/etc/rc.d/init.dディレクトリにあるautofsスクリプトを使用して、/etc/auto.master主設定ファイルによりautomountを制御します。automountはコマンド行で指定できますが、手ですべてを入力するよりも、ファイルのセットにマウントポイント、ホスト名、エクスポートディレクトリ、オプションを指定する方が便利です。指定のランレベルで起動、停止するサービスとしてautofsを実行することで、さまざまなファイルのマウント設定を自動的に実装できます。

autofs設定ファイルは親子関係に配列されています。主設定ファイル（/etc/auto.master）は、ある特定のマップタイプにリンクされているシステム上のマウントポイントを参照します。マップタイプは、そのほかの設定ファイル、プログラム、NISマップ、一般的でないマウント方法などの書式をとります。auto.masterファイルには、次のように編成されたマウントポイントのそれぞれを参照する行が含まれています：

```
<mount-point>  <map-type>
```

この行の<*mount-point*>要素はローカルファイルシステム上のマウントする場所を示しています。<*map-type*>は、マウントされるマウントポイントの方法に関連しています。NFSエクスポートの自動マウントの最も一般的な手段は、1つのファイルを特定のマウントポイント用のマップタイプとして使用することです。マップファイルは通常、auto. <*mount-point*>という名前が付けられ、この<*mount-point*> とはauto.masterの中で指定されているマウントポイントで、以下のような行が含まれています：

```
<directory> <mount-options> <host>:<exported-file-system>
```

<*directory*>は、エクスポートファイルシステムをマウントするマウントポイント内のディレクトリです。標準のmountコマンドとほぼ同じで、ファイルシステムをエクスポートするホストとエクスポート対象のファイルシステムは、<*host*>: <*exported-filesystem*>セクションに存在する必要があります。エクスポートファイルシステムをマウントするときに使用する特定のオプションを指定するには、<*mount-options*> セクションにカンマで区切って配置します。autofsを使用するNFSマウントの場合は、<*mount-options*>セクションに-fstype=nfsを配置する必要があります。

autofs設定ファイルは多数のタイプのデバイスやファイルシステムのさまざまなマウントに使用できますが、NFSマウントを作成する際に特に役立ちます。たとえば、一部の組織はユーザーの/home/ディレクトリをNFS共有により中央のサーバーに格納しています。次に、ワークステーションのそれぞれでauto.master ファイルを設定して、NFSにより/home/ディレクトリをマウントする方法の詳細を含むauto.home ファイルをポイントさせています。これにより、ユーザーは、内部ネットワークのどこでログインしても /home/ディレクトリの個人データと設定ファイルにアクセスできます。この場合のauto.master ファイルは、次のようなものになります：

```
/home /etc/auto.home
```

この結果、/etc/auto.homeファイルによって設定されるローカルシステムに/home/マウントポイントがセットアップされます。このファイルの内容は、おそらく次のようなものです：

```
* -fstype=nfs,soft,intr,rsize=8192,wsize=8192,nosuid server.example.com:/home
```

この行の内容では、ユーザーがアクセスを試みる、ローカルの/home/ディレクトリ下のすべてのディレクトリ（アスタリスク文字により）は、そのエクスポート/home/ファイルシステム内のserver.example.comシステム上でNFSマウントが行われることを示しています。このマウントオプションは、NFSがマウントする/home/ディレクトリそれぞれがセッティングの特定の集まりを使用することを指定しています。マウントここで使用されているの例も含めて、オプションの詳細については、項9.3.3を参照して下さい。

9.3.3. 共通のNFSマウントオプション

NFSによりファイルシステムをリモートホストにマウントする以外に、多種多様なオプションをマウント時に指定して使いやすくすることができます。これらのオプションは、手動のmountコマンド、/etc/fstab設定値、autofs、その他のマウント方法などを併用できます。

以下のオプションは、NFSマウントに最も使われているものです：

- hard 又はsoft — エクスポートファイルシステムをサービスするホストが使用不能になった場合に、NFS接続経由のファイルを使用するプログラムが停止して、サーバーがオンライン復帰するのを待つかどうか（hard）、あるいはエラーを報告するかどうか（soft）を指定します。

  hardを指定した場合は、intrオプションを一緒に指定していない限り、NFS通信が再開するのを待つプロセスを終了することはできません。

  softを指定した場合は、追加のtimeo=<*value*> オプションを設定できます。ここで、<*value*>はエラーが報告されるまでの経過秒数を指定します。
- intr — これを使用すると、サーバーがダウンしたか、途絶えたときにNFS要求を割り込ませることができます。
- nolock — 旧NFSサーバーに接続するときに必要になる場合があります。ロックを要求するには、lockオプションを使用します。
- noexec — マウントされたファイルシステム上でバイナリの実行を許可しません。これが役立つのは、NFS経由で使用しているシステム上に互換性のないバイナリを含むLinux以外のファイルシステムをマウントしている場合です。
- nosuid — set-user-identifier/set-group-identifierビットを有効にすることを許可しません。

- `rsize=8192` 及び`wsize=8192` — 1度に伝送するデータのブロックサイズ（バイト単位）の設定値を大きくすることで、NFS通信の読み出し（`rsize`）、書き込み（`wsize`）をスピードアップします。これらの値を変更するときは注意してください。ブロックサイズを大きくすると、一部の旧Linuxカーネルとネットワークカードが正常に機能しないおそれがあります。
- `nfsvers=2` 及び`nfsvers=3` — 使用するNFSプロトコルのバージョンを指定します。

`mount` manページには利用できるオプションがさらに数多くあり、中には非NFSファイルシステムをマウントするときに使用するオプションなどがあります。

9.4. NFSのセキュリティ

NFSは、多数の既知のホストがあるファイルシステム全体をおおむね透過方式で共有することにおいてうまく機能します。NFSマウントを介してファイルにアクセスする多くのユーザーは、使用しているファイルシステムが自分のシステムにローカルではないことに気が付かないかもしれません。しかし、その使いやすさと共にさまざまなセキュリティ問題が内在しています。

あるサーバー上のNFSファイルシステムをエクスポートしたり、それらをクライアントにマウントしたりするときは、以下の点について考慮する必要があります。それを実行すれば、NFSセキュリティリスクを最小限に抑え、サーバーのデータの保護を改善できます。

9.4.1. ホストアクセス

NFSが管理するのは、マウント要求するホストをベースにしたエクスポートファイルをマウントできる人で、ファイルシステムを実際に利用するユーザーではありません。ホストはエクスポートファイルシステムをマウントする権利を明示的に与えられていなければいけません。ファイルとディレクトリのアクセス権以外のアクセス制御は、ユーザーにはできません。言い換えると、ファイルシステムがNFSを介してエクスポートされると、NFSサーバーに接続されているどのリモートホストのどのユーザーでも共有データにアクセスできます。この内在するリスクを制限するには、管理者が読み取り専用のアクセスだけを許可するか、あるいは、ユーザーを一般的なユーザーidとグループidに「押し下げ」ます。ただし、この解決法は、本来の目的であるNFS共有の使用を阻止することにもなりえます。

さらに、NFSファイルシステムをエクスポートしているシステムが使用するDNSサーバーの制御権を侵入者が取得すると、ある特定のホスト名や完全修飾ドメイン名に関連付けられているシステムは無許可マシンへとポイントされる可能性があります。ここで、NFSマウントの追加セキュリティを用意する為のユーザー名又はパスワード情報が交換されていませんので、この無許可マシンがNFS共有をマウントするように許可されたシステムそのものになります。また、あるホストがNFS共有をマウントできるようにするためにNISネットグループを使用している場合は、同様のリスクがNISサーバーの侵害にも当てはまります。`/etc/exports`にIPアドレスを使用することによって、この種のアタックをより困難にすることはできます。

ワイルドカードは、NFS共有のエクスポートを許可しているときに控えめに使用すべきです。ワイルドカードが有効な範囲は意図した以上のシステムに達する可能性があります。

NFSの安全な使用に関する詳細は*Red Hat Linux* セキュリティガイドの中のサーバーセキュリティの章を参照して下さい。

9.4.2. ファイルアクセス権

NFSファイルシステムはリモートホストによって読み書きモードでいったんマウントされると、各共有ファイルが持つ唯一の保護はその権限だけです。仮に同じuseridの値を持つ2人のユーザーが、同じNFSファイルシステムをマウントした場合、彼らは互いに相手のデータを変更することができます。さらにはクライアントシステムでrootとしてログインする人は誰でも`su -`コマンドを使用して、NFS共有を介して特定のファイルにアクセスできるユーザーになることが出来ま

す。NFSとuseridの衝突に関する詳細は、*Red Hat Linux* システムアドミニストレーションプレミアの中のアカウントとグループの管理を参照して下さい。

NFSによりファイルシステムをエクスポートするときのデフォルト動作は、*root squashing*を使用することです。この結果、ローカルマシンのrootユーザーとしてNFS共有を利用する人のユーザーIDは、サーバーのnobodyアカウント値に設定されます。root squashingは停止してはいけません。

NFS共有を読み取り専用でエクスポートする場合、`all_squash`オプションの使用を検討して下さい。これは、エクスポートファイルシステムにアクセスするすべてのユーザーにnobodyユーザーのユーザーIDを取得させます。

9.5. その他のリソース

NFSサーバーを管理することは1つの挑戦です。この章で紹介していない多くのオプションがNFSファイルシステムのエクスポートやマウント用に用意されています。詳細については、以下のリソースを参照してください。

9.5.1. インストールされているドキュメント

- `/usr/share/doc/nfs-utils-<version-number>/` — `<version-number>`の部分にはNFSパッケージのバージョン番号を入れます。このディレクトリには、NFSをLinuxにインストールする方法を説明しており、さまざまなNFS設定例や、ファイル転送のパフォーマンスに与える影響なども記載しています。
- `man mount` — NFSサーバーとNFSクライアントの両方の設定用マウントオプションについて総合的に解説しています。
- `man fstab` — システム起動時にファイルシステムをマウントするために使用する`/etc/fstab`ファイルの書式について詳細に説明しています。
- `man nfs` — NFS固有のファイルシステムエクスポートとマウントオプションについて詳細に説明しています。
- `man exports` — NFSファイルシステムをエクスポートするときに、`/etc/exports`ファイルに使用する一般的なオプションを示しています。

9.5.2. 関連書籍

- *Managing NFS and NIS*（Hal Stern、Mike Eisler、Ricardo Labiaga共著、O'Reilly & Associates刊）—利用できるさまざまなNFSエクスポートとマウントオプションについての優れたリファレンスガイドになっています。
- *NFS Illustrated*（Brent Callaghan著、Addison-Wesley Publishing Company刊）— NFSとほかのネットワークファイルシステムを比較し、NFS通信の仕組みついて詳細に示しています。

第10章

Apache HTTP サーバー

Apache HTTP サーバーは、Apache Software Foundation(http://www.apache.org)によって開発された強健で、商用製品の匹敵するオープンソースのWebサーバーです。Red Hat Linuxには、Apache HTTP サーバーバージョン2とさらにその機能を強化するように設計されている数多くのサーバーモジュールが含まれています。

Apache HTTP サーバーにインストールされているデフォルトの設定は、殆どの状況で変更せずに利用できるはずです。この章では、カスタム設定を必要とする、又は古いApache HTTP サーバー1.3から設定ファイルを変換する必要のあるユーザーを支援する為に、多くのApache HTTP サーバー設定ファイルについてその概要を説明しています。

警告

グラフィカルな**HTTP 設定ツール**(`redhat-config-httpd`)を使用している場合、**HTTP 設定ツール**はこのファイルを使用する度に再生成しますので、Apache HTTP サーバーの設定ファイルは手動で編集しないでください。

HTTP 設定ツールに関する詳細情報は*Red Hat Linux* カスタマイズガイドの中の*Apache HTTP* サーバーの設定の章で御覧下さい。

10.1. Apache HTTP サーバー2.0

Apache HTTP サーバーバージョン2とそのバージョン1.3(バージョン1.3 はRed Hat Linux 7.3 とそれ以前で使用)には重要な差異が有ります。このセクションは、Apache HTTP サーバー2.0の機能をいくつかを検証して、重要な変更点の概要を説明します。バージョン1.3の設定ファイルを2.0形式に移行する方法の案内は項10.2で御覧下さい。

10.1.1. Apache HTTP サーバー2.0の機能

Apache HTTP サーバー2.0の到来により、多くの機能が利用できます。以下にその主なものを示します:

- 新しい*Apache API* — モジュールはより強力な新しいAPI(Application Programming Interfaces)のセットを活用します。

  

  重要

  Apache HTTP サーバー1.3 用に構成されたモジュールは、新しいAPIに移動しない限り機能できません。特定のモジュールが移動されたかどうか不明な場合は、アップグレードする前に開発者に確認して下さい。

- フィルタリング — モジュールはコンテントフィルタとして機能できます。フィルタリングの作用の仕方に付いての詳細は項10.2.4で確認できます。
- *IPv6* サポート — 次世代のIPアドレス形式がサポートされます。
- 簡潔化されたディレクティブ — 多くの混乱しやすいディレクティブが排除され、他のディレクティブも簡潔化されています。特定のディレクティブについての情報は項10.5で御覧下さい。
- 複数言語のエラー対応 — *SSI*(*Server Side Include*)ドキュメントを使用する時、カスタマイズしたエラー対応ページが複数言語で供給されます。

- 複数プロトコルサポート — 複数のプロトコルがサポートされます。

より総括的な変更リストは次のサイトでオンライン検索ができます。http://httpd.apache.org/docs-2.0/.

10.1.2. Apache HTTP サーバー2.0のパッケージ変更

Red Hat Linux 8.0から開始して、Apache HTTP サーバーパッケージは名前が変更されています。さらにその関連のパッケージにも名前変更、使用停止、又は他のパッケージとの合併などがあります。

以下にパッケージ変更のリストを示します：

- `apache`、`apache-devel`、`apache-manual`は名前変更になり`httpd`、`httpd-devel`、`httpd-manual`とそれぞれ変わりました。
- `mod_dav`パッケージは`httpd` パッケージに吸収されています。
- `mod_put`パッケージと`mod_roaming` パッケージは、その機能が`mod_dav`で提供される機能のサブセットである為、削除されました。(`mod_dav`は`httpd`に吸収されています。)
- `mod_auth_any`と`mod_bandwidth`パッケージは削除されました。
- `mod_ssl`パッケージ用のバージョン番号は、今回`httpd`パッケージと同期化されています。これはApache HTTP サーバー2.0用の`mod_ssl`パッケージは、Apache HTTP サーバー1.3用の`mod_ssl`パッケージよりも低い(mod_sslの)バージョン番号を持つということになります。

10.1.3. Apache HTTP サーバー2.0のファイルシステムの変更

Apache HTTP サーバー2.0へアップグレードをすると、ファイルシステムのレイアウトに以下のような変更が起こります：

- 新しい設定ディレクトリ*`/etc/httpd/conf.d/`*が追加されました。 — この新しいディレクトリは、`mod_ssl`、`mod_perl`、`php`等の個別のパッケージモジュール用の設定ファイルを保存する為に使用されます。サーバーはこの場所から設定ファイルをロードするように、Apache HTTP サーバーの設定ファイルである`/etc/httpd/conf/httpd.conf`にあるディレクティブ`Include conf.d/*.conf`によって指示されます。

  
  重要

  既存の設定を移行する時には、この行が挿入されていることが重要になります。

- *`ab`*と*`logresolve`*プログラムは移動しました。 — これらのユーティリティプログラムは、`/usr/sbin/`ディレクトリから`/usr/bin/`へ移動されました。これにより、これらのバイナリの絶対パスを持つスクリプトは機能しません。
- *`dbmmanage`*コマンドは入れ換えられました。 — `dbmmanage`コマンドは、`htdbm`により入れ換えられています。詳細は項10.2.4.4で御覧下さい。
- *`logrotate`*の設定ファイルは名前が変更されています。 — `logrotate`設定ファイルは`/etc/logrotate.d/apache`から`/etc/logrotate.d/httpd`へと名前が変更されました。

次のセクションでは、Apache HTTP サーバー1.3設定から2.0形式への移行の仕方を簡単に説明しています。

10.2. Apache HTTP サーバー1.3 の設定ファイルの移行

Apache HTTP サーバーがすでにインストールされていたRed Hat Linux 7.3又はそれ以前のバージョンからアップグレードする場合、新しいApache HTTP サーバー2.0 パッケージの標準設定ファイルは/etc/httpd/conf/httpd.conf.rpmnewとしてインストールされており、オリジナルのバージョン1.3 httpd.confはそのまま残っています。勿論、これはユーザー任意の計画で、新しい設定ファイルを使用して、古い設定をそれに移行させるか、又は、既存のファイルをベースとして使用して、必要な分を編集するかを決定することになります。ただし、ファイルの幾らかの部分では他の部分よりも多く変更されており、全般的に両方を混ぜたアプローチが適切でしょう。バージョン1.3とバージョン2.0の標準の設定ファイルは両方とも、3つのセクションに判れています。このガイドの目的はどのようなコースが最適であるかを提案することにあります。

/etc/httpd/conf/httpd.confが、デフォルトRed Hat Linuxバージョンの変更されたバージョンであり、オリジナルのコピーが入手できる場合は、以下の例のようにdiffコマンドを執行するのが最も簡単でしょう：

```
diff -u httpd.conf.orig httpd.conf | less
```

このコマンドは、実行された変更をすべて拾い出します。もし、オリジナルファイルのコピーがない場合は、rpm2cpioとcpioコマンドを使用してそれをRPMパッケージから取り出すことも出来ます。以下のようにします：

```
rpm2cpio apache-<version-number>.i386.rpm | cpio -i --make
```

上記のコマンドでは、*<version-number>*を、apache パッケージ用のバージョン番号で入れ換えます。

最後に、Apache HTTP サーバーが設定エラーをチェックする為のテストモードを持っていることを覚えていると便利です。これにアクセスするには、次のようなコマンドを入力します：

```
apachectl configtest
```

10.2.1. グローバル環境の設定

設定ファイルのグローバル環境のセクションには、処理できる同時要求の数量や各種ファイルの場所などApache HTTP サーバーの全体的な運営に影響するディレクティブが含まれています。このセクションは、他の部分に比べて大量の変更がありますのでApache HTTP サーバー2.0設定ファイルのこのセクションをベースとして、そして古い設定をここに移行することが推奨されます。

10.2.1.1. バインドするインターフェイスとポートの選択

BindAddressとPortのディレクティブはもう存在しません。それらの機能は現在、より柔軟なListen ディレクティブにより提供されます。

Port 80が、1.3バージョンの設定ファイルでセットされていた場合、2.0設定ファイルではそれをListen 80に変更します。Portに*80*以外の値がセットしてあった場合、、そのポート番号をServerNameディレクティブの内容に付加します。

例として、以下にApache HTTP サーバー1.3 ディレクティブのサンプルを示します：

```
Port 123
ServerName www.example.com
```

この設定をApache HTTP サーバー2.0に移行するには、次の構成を使用します：

```
Listen 123
ServerName www.example.com:123
```

この課題についての情報は、以下のApache Software Foundationのサイトにあるドキュメントでお読み下さい：

- `http://httpd.apache.org/docs-2.0/mod/mpm_common.html#listen`
- `http://httpd.apache.org/docs-2.0/mod/core.html#servername`

10.2.1.2. サーバープールサイズの規定

Apache HTTP サーバー2.0では、要求を受理し、それを処理する為に子プロセスを送る責任は、*MPM*(*Multi-Processing Modules*)と呼ばれるモジュールのグループへと分離されていました。他のモジュールとは異なり、MPMグループからはモジュール1つだけがApache HTTP サーバーによってロードされます。バージョン2.0には3つのMPM モジュールが同梱されています：`prefork`、`worker`、及び`perchild`です。

オリジナルのApache HTTP サーバー1.3 の動作は、`prefork`MPMの中に移動され、Red Hat Linux上では、現在`prefork` MPMだけが利用できます。但し、他のMPMも後日、利用できるようになるでしょう。

`prefork` MPMはApache HTTP サーバー1.3と同じディレクティブを受け付けますので、以下のディレクティブは直接移行が出来ます：

- `StartServers`
- `MinSpareServers`
- `MaxSpareServers`
- `MaxClients`
- `MaxRequestsPerChild`

この課題に関する情報はApache Software Foundationのサイトで次のドキュメントでお読み下さい：

- http://httpd.apache.org/docs-2.0/mpm.html

10.2.1.3. DSO (Dynamic Shared Object)のサポート

ここでは多くの変更が必要です。そしてバージョン2.0に合わせるためにApache HTTP サーバー1.3の設定ファイルを修正(変更をバージョン2.0設定に移行することとは別)しようとしているユーザーには、標準のRed Hat Linux Apache HTTP サーバー2.0 設定ファイルからこのセクションをコピーすることが推奨されます。

標準のApache HTTP サーバー2.0設定からセクションをコピーしたくない人は以下の点を注意して下さい：

- `AddModule`と`ClearModuleList`のディレクティブはもう存在しません。これらのディレクティブはモジュールが正しい順序で有効になることを確証すのに使用されていました。Apache HTTP サーバー2.0 APIにより、モジュールはその順序を指定することが出来ますので、以前のディレクティブの必要性がなくなりました。
- `LoadModule` 行の順序はもう関係ありません。
- 多くのモジュールに対し、追加、削除、名前変更、分離、合併などが実行されています。
- 自身のRPM(`mod_ssl`、`php`、`mod_perl`、その他)にパッケージされているモジュール用`LoadModule`の行はもう必要ありません。それらは現在、`/etc/httpd/conf.d/`ディレクトリ内の該当するファイルの中に存在します。
- 各種`HAVE_XXX`の定義はもう定義付けされません。

重要

オリジナルファイルを編集している場合、httpd.confには以下のディレクティブが含まれることが最大の重要点であることに注意して下さい：

```
Include conf.d/*.conf
```

このディレクティブが欠如すると、それ自身のRPM内にパッケージしてあるすべてのモジュール(mod_perl、php、mod_sslなど)が機能しなくなります。

10.2.1.4. 他のグローバル環境の変更

以下のディレクティブはApache HTTP サーバー2.0設定から削除されました：

- *ServerType* — Apache HTTP サーバーはServerType standaloneとしてのみ実行できますので、このディレクティブは不要物となります。
- *AccessConfig*及び*ResourceConfig* — これらのディレクティブは、Includeディレクティブの機能のミラーである為、削除されました。もし、AccessConfigとResourceConfigのディレクティブがセットされている場合は、それらをIncludeディレクティブで入れ換えてください。

  ファイルが、古いディレクティブにより示されている順序で確実に読み込まれるようにするには、Includeディレクティブがhttpd.confの末尾に位置していて、さらにResourceConfigに相当するものと、その後にAccessConfigに相当するものが添付されている必要があります。デフォルトの値を使用している場合、それらを明確にconf/srm.conf ファイルとconf/access.confファイルと指定して含めます。

10.2.2. メインサーバーの設定

設定ファイルのメインサーバー設定セクションはメインサーバーをセットし、<VirtualHost>のコンテナで定義した仮想ホストで処理されない全ての要求に応答します。ここでの値は<VirtualHost>のコンテナに定義されるデフォルトを提供するものです。

このセクションのディレクティブは、Apache HTTP サーバー1.3 と2.0のバージョン間で少ししか変更されていません。メインサーバーの設定が大幅にカスタマイズされている場合、既存の設定をApache HTTP サーバー2.0に合うように編集したほうが簡単である可能性があります。軽いカスタマイズしかしていないユーザーはそれらの変更をデフォルトの2.0設定へ移動すべきです。

10.2.2.1. UserDir マッピング

UserDirディレクティブは、http://example.com/~bob/などのURLを有効にして、/home/bob/public_htmlなどのユーザーbob のホームディレクトリ内のサブディレクトリへマップするために使用されます。この機能の副作用として侵入者にシステム上にあるユーザー名の存在を判定をさせてしまうことがあります。この理由のため、Apache HTTP サーバー2.0のデフォルト設定では、このディレクティブが無効になっています。

UserDirマッピングを有効にするには、httpd.confから以下の状態のディレクティブを変更します：

```
UserDir disable
```

次のように変えます：

```
UserDir public_html
```

この課題についての情報は、Apache Software Foundationのサイト内にある次のドキュメントを参考にして下さい。http://httpd.apache.org/docs-2.0/mod/mod_userdir.html#userdir.

10.2.2.2. ロギング

以下のロギングのディレクティブは削除されています：

- `AgentLog`
- `RefererLog`
- `RefererIgnore`

しかし、エージェントログと参照ログはまだ、`CustomLog`ディレクティブと`LogFormat`ディレクティブで利用できます。

この課題についての情報はApache Software Foundationのサイトで、次のドキュメントをお読み下さい：

- http://httpd.apache.org/docs-2.0/mod/mod_log_config.html#customlog
- http://httpd.apache.org/docs-2.0/mod/mod_log_config.html#logformat

10.2.2.3. ディレクトリの索引

無用になった`FancyIndexing`ディレクティブは、すでに削除されています。同じ機能を`IndexOptions`ディレクティブ内の`FancyIndexing` *option*で利用出来ます。

`IndexOptions`ディレクティブへの新しい`VersionSort`オプションは、バージョン番号を含んでいるファイルをより自然に分類するようになります。例えば、`httpd-2.0.6.tar`はディレクトリ索引のページで`httpd-2.0.36.tar`の前に来ます。

`ReadmeName`と`HeaderName`のデフォルトのディレクティブは`README` と`HEADER`から`README.html`と`HEADER.html`へ変更されています。

この課題についての情報はApache Software Foundationのサイトで次のドキュメントから入手してください：

- http://httpd.apache.org/docs-2.0/mod/mod_autoindex.html#indexoptions
- http://httpd.apache.org/docs-2.0/mod/mod_autoindex.html#readmename
- http://httpd.apache.org/docs-2.0/mod/mod_autoindex.html#headername

10.2.2.4. コンテンツ交渉

`CacheNegotiatedDocs`ディレクティブは、改訂で引数に`on` 又は`off`を使うようになりました。`CacheNegotiatedDocs`の既存のインスタンスは`CacheNegotiatedDocs on`で入れ換える必要があります。

この課題に関する情報はApache Software Foundationのサイトで以下のドキュメントから入手して下さい：

- http://httpd.apache.org/docs-2.0/mod/mod_negotiation.html#cachenegotiateddocs

10.2.2.5. エラードキュメント

ErrorDocumentディレクティブでhard-codedメッセージを使用するには、メッセージはApache HTTP サーバー1.3で要求されていた2重引用符を前に付けるのではなく、前後の2重引用符["]で囲まれなければなりません。

ErrorDocumentの設定をApache HTTP サーバー2.0に移行するには以下の構造を使用します：

```
ErrorDocument 404 "The document was not found"
```

上記のErrorDocumentディエクティブに付いて来る2重引用符に注意して下さい。

この課題に関する情報についてはApache Software Foundationのサイトで次のドキュメントから入手することが出来ます：

- http://httpd.apache.org/docs-2.0/mod/core.html#errordocument

10.2.3. 仮想ホスト設定

<VirtualHost>コンテナの内容のすべては、項10.2.2に示してあるようにメインサーバーと同様な方法で移行すべきです。

重要

SSL/TLS仮想ホスト設定は、メインサーバー設定ファイルから/etc/httpd/conf.d/ssl.confへ移動されたことに注意して下さい。

この課題についての情報は*Red Hat Linux* カスタマイズガイド内の*Apache HTTP*セキュアサーバーの設定という章、及び以下のサイトにあるオンラインドキュメントで御覧下さい：

- http://httpd.apache.org/docs-2.0/vhosts/

10.2.4. モジュールとApache HTTP サーバー2.0

Apache HTTP サーバー2.0では、モジュールシステムはモジュールが一緒に連結されるように、又は新しい、興味深い方法で結合されるように変更されました。*CGI* (*Common Gateway Interface*)スクリプトは、例として、サーバー構文解析付きHTMLドキュメントを生成でき、これはmod_includeでプロセスされます。これにより、特定の目的を達成するためのモジュールを組み合わせる方法に膨大な可能性が出て来ます。

この機能の仕組みは、各要求が正確に1つのハンドラモジュールによりサービスされて、0又はそれ以上のフィルタモジュールが続くようになっています。

Apache HTTP サーバー1.3の例として、PHPスクリプトはすべて、そのPHPモジュールによって処理されています。Apache HTTP サーバー2.0では、そのような供給はコアモジュールによって処理 されます。—これは固定ファイルをサービスします。— その後、PHPモジュールによってフィルタされます。

厳密には、この使用法及びApache HTTP サーバー2.0のすべての他の新機能については、このマニュアルの担当範囲を越えるものです。ただ、PATH_INFOディレクティブが、フィルタとして実装されている、モジュールにより処理されるドキュメントで使用される場合は、変更は分岐されます。これは、それぞれ本来のパス名の後にパス情報を含んでいるためです。初期段階で要求を処理するコアモジュールは、デフォルトでPATH_INFOを理解できず、その情報を持つ要求に対し404 Not Foundエ

ラーを送り返します。代替としてAcceptPathInfoディレクティブを使用して、コアモジュールがPATH_INFOの要求を受け付けるように強制します。

以下に、このディレクティブの例を示します：

```
AcceptPathInfo on
```

この課題の追加情報は、Apache Software Foundationのサイトの以下のドキュメント内で御覧になれます：

- http://httpd.apache.org/docs-2.0/mod/core.html#acceptpathinfo
- http://httpd.apache.org/docs-2.0/handler.html
- http://httpd.apache.org/docs-2.0/filter.html

10.2.4.1. mod_sslモジュール

mod_sslの設定はhttpd.confから/etc/httpd/conf.d/ssl.confファイルへ移動されました。このファイルをロードするため、そしてmod_sslが機能する為には、項10.2.1.3で表示されているように、Include conf.d/*.confステートメントがhttpd.confに含まれる必要があります。

SSL仮想ホストのServerNameは明確にそのポート番号を指定する必要があります。

例えば、次のApache HTTP サーバー1.3 ディレクティブの例のようになります：

```
<VirtualHost _default_:443>
  # General setup for the virtual host
  ServerName ssl.example.name
  ...
</VirtualHost>
```

この設定をApache HTTP サーバー2.0に移行するには、次の構成を使用します：

```
<VirtualHost _default_:443>
  # General setup for the virtual host
  ServerName ssl.host.name:443
  ...
</VirtualHost>
```

この課題についての情報は、Apache Software Foundationのサイトにある以下のドキュメントでお読み下さい：

- http://httpd.apache.org/docs-2.0/mod/mod_ssl.html
- http://httpd.apache.org/docs-2.0/vhosts/

10.2.4.2. mod_proxyモジュール

プロキシーアクセス制御ステートメントは、今回の設定では、<Directory proxy:>ではなく、<Proxy>ブロック内にあります。

古いmod_proxyのキャシング機能は、次の3つのモジュールに分割されました：

- mod_cache
- mod_disk_cache
- mod_file_cache

これらのモジュールは、全般的にmod_proxyモジュールの古いバージョンと同じ、又は似たようなディレクティブを使用します。

この課題に関する追加情報は、Apache Software Foundationのサイトの中にある以下のドキュメントでお読みください：

- http://httpd.apache.org/docs-2.0/mod/mod_proxy.html

10.2.4.3. mod_includeモジュール

mod_includeモジュールは、今回のリリースでフィルタとして実装されており、このため、異なる方法で有効になります。フィルタに関する情報は項10.2.4で確認して下さい。

例えば、Apache HTTP サーバー1.3ディレクティブのサンプルは以下のようになります：

```
AddType text/html .shtml
AddHandler server-parsed .shtml
```

この設定をApache HTTP サーバー2.0に移行するには、次の構成を使用します：

```
AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
```

以前と同様に、Options +Includesディレクティブは、この改訂でも<Directory>コンテナ、あるいは.htaccess ファイルによって必要とされていることに注意して下さい。

この課題に関する追加情報は、Apache Software Foundationのサイトにある以下のドキュメント内で読むことが出来ます：

- http://httpd.apache.org/docs-2.0/mod/mod_include.html

10.2.4.4. mod_auth_dbmモジュールとmod_auth_dbモジュール

Apache HTTP サーバー1.3 は、Berkeley DatabasesとDBMデータベースをそれぞれ使用したmod_auth_dbとmod_auth_dbmと言う2つの認証モジュールをサポートしていました。これらのモジュールは今、Apache HTTP サーバー2.0では、mod_auth_dbmとして1つのモジュールに結合されています。これは数種の異なるデータベース形式にアクセスできます。mod_auth_dbから移行するには、AuthDBUserFileとAuthDBGroupFileをmod_auth_dbmの同種であるAuthDBMUserFileとAuthDBMGroupFileで入れ換えることで、設定ファイルを調節する必要があります。また、AuthDBMType DB ディレクティブを付け加えて使用中のデータベースファイルのタイプを表示する必要があります。

次の例では、Apache HTTP サーバー1.3用のmod_auth_db設定のサンプルを示しています：

```
<Location /private/>
 AuthType Basic
 AuthName "My Private Files"
 AuthDBUserFile /var/www/authdb
 require valid-user
</Location>
```

この設定をApache HTTP サーバーのバージョン2.0に移行するには、次の構成を使用します：

```
<Location /private/>
 AuthType Basic
 AuthName "My Private Files"
 AuthDBMUserFile /var/www/authdb
 AuthDBMType DB
```

```
 require valid-user
</Location>
```

AuthDBMUserFileディレクティブは.htaccessファイルでも使用することができることに注意して下さい。

ユーザー名とパスワードデータベースの処理に使用されるdbmmanage Perl スクリプトは、Apache HTTP サーバー2.0内では、htdbmに入れ替わっています。htdbmプログラムは同様の機能を持ち、mod_auth_dbmの様に、各種のデータベース形式を運営することが出来ます。-Tオプションはコマンドライン上で使用するフォーマットを指定することが出来ます。

表10-1は、dbmmanageを使用してDBM-formatデータベースからhtdbmフォーマットへ移行する方法を示しています。

操作	dbmmanage コマンド(1.3)	対応するhtdbm コマンド(2.0)
データベースにユーザーを追加(設定したパスワード使用)	`dbmmanage authdb add username password`	`htdbm -b -TDB authdb username password`
データベースにユーザーを追加(パスワードを要求)	`dbmmanage authdb adduser username`	`htdbm -TDB authdb username`
データベースからユーザーを削除	`dbmmanage authdb delete username`	`htdbm -x -TDB authdb username`
データベースにユーザーをリストする	`dbmmanage authdb view`	`htdbm -l -TDB authdb`
パスワードを確認	`dbmmanage authdb check username`	`htdbm -v -TDB authdb username`

表10-1. dbmmanageからhtdbmへの移行

-mオプションと-sオプションはdbmmanage とhtdbmの両方で機能し、ハッシュパスワード用のMD5又はSHA1アルゴリズムの使用をそれぞれ有効にします。

htdbmで新規のデータベースを作成する場合は、-cオプションを使用しなければなりません。

この課題についての追加情報は、Apache Software Foundationのサイトにある以下のドキュメントで御覧下さい：

- http://httpd.apache.org/docs-2.0/mod/mod_auth_dbm.html

10.2.4.5. mod_perlモジュール

mod_perlの設定は、httpd.confから/etc/httpd/conf.d/perl.confファイルへ移動されました。このファイルをロードするには、そしてmod_perlが機能するようにするには、Include conf.d/*.confステートメントが項10.2.1.3に示してあるようにhttpd.conf内に含まれる必要があります。

httpd.conf内でのApache::の存在は、ModPerl::に入れ換えなければなりません。さらに、ハンドラが登録される方法は変更されました。

以下にApache HTTP サーバー1.3 mod_perl設定のサンプルを示します：

```
<Directory /var/www/perl>
  SetHandler perl-script
  PerlHandler Apache::Registry
  Options +ExecCGI
</Directory>
```

これはApache HTTP サーバー2.0用のmod_perlと同等のものです：

```
<Directory /var/www/perl>
  SetHandler perl-script
  PerlModule ModPerl::Registry
  PerlHandler ModPerl::Registry::handler
  Options +ExecCGI
</Directory>
```

mod_perl 1.x 用の殆どのモジュールは、修正なしでmod_perl 2.xと一緒に機能するはずです。XSモジュールはリコンパイルが必要で、少々、Makefileの修正が必要になるかも知れません。

10.2.4.6. mod_pythonモジュール

mod_pythonの設定はhttpd.confから/etc/httpd/conf.d/python.confファイルへ移動しました。このファイルをロードする為に、またmod_pythonが機能する為には、Include conf.d/*.confステートメントが項10.2.1.3に示してあるように、httpd.confに含まれる必要があります。

10.2.4.7. PHP

PHPの設定ファイルは、httpd.confから/etc/httpd/conf.d/php.confへ移動されました。このファイルをロードするためには、項10.2.1.3に示してあるように、Include conf.d/*.confステートメントがhttpd.confの中になければなりません。

PHP は、フィルタとして実装されています。そのため、異なる方法で有効にする必要があります。フィルタに関する情報は項10.2.4 で御覧下さい。

Apache HTTP サーバー1.3では、PHPは以下のディレクティブを使用して実装されていました：

```
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
```

Apache HTTP サーバー2.0では、その代わりに次のようなディレクティブを使用します：

```
<Files *.php>
  SetOutputFilter PHP
  SetInputFilter PHP
</Files>
```

PHP 4.2.0 とそれ移行のバージョンでは、グローバルスコープで利用できた定義済変数のデフォルトセットは変更されています。個別の入力とサーバーの変数はデフォルトでは、もうグローバルスコープの中に直接配置はされません。この変更が、スクリプト分裂の原因になるかも知れません。register_globalsの設定で/etc/php.iniファイル内をOnにして以前の動作に戻して下さい。

この課題に関する追加情報は、以下のURLでグローバルスコープの変更について参照して下さい：

- http://www.php.net/release_4_1_0.php

10.3. インストールの後

httpdパッケージをインストールした後は、httpd-manual パッケージをインストールしてWebブラウザをhttp://localhost/manual/にポイントするか、又はhttp://httpd.apache.org/docs-2.0/のサイトにあるApache ドキュメントを閲覧することで、Apache HTTP サーバードキュメントを使用できるようになります。

Apache HTTP サーバーのドキュメントには、全ての設定オプションの総合リストと総括的な説明が収納されています。便宜上、この章ではApache HTTP サーバー2.0で使用される設定ディレクティブの簡単な説明を用意しています。

Red Hat Linuxに収納されているApache HTTP サーバーのバージョンは、mod_sslパッケージとopensslパッケージで提供されている強力なSSL 暗号化法を使用してセキュアWebサーバーをセットアップする能力を持っています。設定ファイルを見ている時、それがセキュアWebサーバーと、そうでないWebサーバーを両方含んでいることに気が付くでしょう。セキュアWebサーバーは仮想ホストとして、動作することになり、/etc/httpd/conf.d/ssl.confファイルの中に設定されています。仮想ホストに関する詳細は、項10.8で御覧下さい。セキュアサーバーの仮想ホストの設定に関する情報は、項10.8.1で御覧になれます。Apache HTTP セキュアサーバーの設定に関する情報は*Red Hat Linux* カスタマイズガイドにある*Apache HTTP* セキュアサーバーの設定の章で御覧になれます。

注意

Red Hat, Inc. は、FrontPageの拡張版を同梱していません。これは、Microsoft™社のライセンスが、サードパーティへのこの拡張版の同梱を禁止しているからです。FrontPageの拡張版とApache HTTP サーバーに関する情報は、オンラインで以下のURLで見ることが出来ます：http://www.rtr.com/fpsupport/

10.4. httpdの開始と停止

httpd RPMで/etc/rc.d/init.d/httpdスクリプトをインストールできます。そして、それが/sbin/serviceコマンドを使用して、アクセス出来るようになります。

サーバーを開始するには、ルートで次のように入力します：

```
/sbin/service httpd start
```

サーバーを停止するには、ルートで次のように入力します：

```
/sbin/service httpd stop
```

restartオプションは、Apache HTTP サーバーを停止して、再開始する手順の短縮形です。

サーバーを再開始するには、ルートで次のように入力します：

```
/sbin/service httpd restart
```

注意

Apache HTTP サーバーをセキュアサーバーとして実行している場合、startやrestartオプションを使用する時はサーバーパスワードを入力する必要があります。

しかし、httpd.confファイルを編集した後は、サーバーをその度に停止と開始をする必要がなく、その代わりにreloadのオプションを使用します。

サーバーの設定ファイルをリロードするには、ルートで次のように入力します：

```
/sbin/service httpd reload
```

注意

Apache HTTP サーバーをセキュアサーバーとして実行している場合、reloadオプションを使用する時にはサーバーパスワードは要求されません。

デフォルトでは、httpdサービスは、起動時に自動的には開始されません。起動時にhttpdサービスが開始されるように設定するには、/sbin/chkconfigや/**sbin/ntsysv**などのinitscript ユーティリティを使用するか、あるいは、**サービス設定ツール**プログラムを使用します。これらのツールに関する情報は*Red Hat Linux* カスタマイズガイドの中の サービスに対するアクセスの制御の章でお読み下さい。

注意

Apache HTTP サーバーをセキュアサーバーとして実行している時、特定タイプのサーバー鍵のファイルがある場合以外は、マシンが起動した後にパスワードが要求されます。

Apache HTTP セキュアサーバーに設定に関する情報は、*Red Hat Linux* カスタマイズガイドの*Apache HTTP* セキュアサーバーの設定の章で御覧下さい。

10.5. httpd.confの設定ディレクティブ

Apache HTTP サーバーの設定ファイルは、/etc/httpd/conf/httpd.confです。httpd.confファイルには充分なコメントがあり、殆ど自然に理解出来る内容です。デフォルトの設定で殆どの状態に対応します。但し、他の重要な設定オプションに馴染んでおくことも大切です。

警告

Apache HTTP サーバー2.0のリリースで、多くの設定オプションが変更されました。バージョン1.3の設定ファイルから2.0形式へ移行する場合は、項10.2を参照して下さい。

10.5.1. 一般的な設定のヒント

Apache HTTP サーバーを設定している場合、/etc/httpd/conf/httpd.confを編集してそれから、リロードするか、又は再開始するか、又は項10.4で示してあるようにhttpdのプロセスを停止して開始し直します。

httpd.confを編集する前に、最初にオリジナルファイルのコピーを作成します。バックアップを作成しておくと、設定ファイルを編集している時にミスをしても簡単に復元できます。

ミスがあり、Webサーバーが正常に動作しない場合、先ずhttpd.confの中で最近の編集した場所を見てそれがミスタイプでないかどうか確認します。

次に、Webサーバーのエラーログ、/var/log/httpd/error_logを見ます。エラーログはユーザーの経験次第では、簡単に解釈できないかも知れません。但し、経験者にはエラーログの最後のエントリーが、何が起こったかについて役に立つ情報を提供してくれるはずです。

次は、httpd.confの中に含まれている多くのディレクティブの短い説明のリストです。これらの説明は、全てを表現するものではありません。この詳細については、次のサイトでHTML形式で提供されているApacheのドキュメントを御覧下さい。URL: http://httpd.apache.org/docs-2.0/.

また、mod_sslディレクティブについては、次のサイトでHTML形式で提供されているドキュメントを御覧下さい。URL: http://httpd.apache.org/docs-2.0/mod/mod_ssl.html.

10.5.2. ServerRoot

ServerRootは、webサイトの内容を含んでいるトップレベルのディレクトリです。デフォルトで、ServerRootはセキュアサーバー用も非セキュアサーバー用も両方とも"/etc/httpd"へ設定されています。

10.5.3. ScoreBoardFile

ScoreBoardFileは、内部のサーバープロセス情報を保存して、その情報は親サーバープロセスと子プロセスの間で使用されます。Red Hat LinuxはScoreBoardFileを保存するために共有メモリを使用しておりデフォルトの/etc/httpd/logs/apache_runtime_statusは単にフォールバック(予備)として使用されるだけです。

10.5.4. PidFile

PidFileは、サーバーがそのプロセスID (PID)を記録するファイルの名前を示します。デフォルトでは、PIDは/var/run/httpd.pid内にリストされています。

10.5.5. Timeout

Timeoutは、通信中に送信と受信を待つ時間の長さ秒単位で示します。特にTimeoutは、GET要求を受信するのにサーバーが待つ長さ、POST 又はPUT要求でTCPパケットを受信するまでの待つ長さ、また、ACKがTCPパケットに反応するまでの長さなどを示します。デフォルトではTimeoutは300秒にセットしてあり、ほとんどの状況には適切です。

10.5.6. KeepAlive

KeepAliveは、サーバーが接続1つに対して複数の要求を許可するかどうかセットし、1人のクライアントがサーバーリソースを過度に使用することを防止します。

デフォルトで、Keepaliveはoffにセットしてあります。それがonにセットしてある場合に、サーバーが混雑してくると、サーバーはすばやく最大数の子プロセスを分配します。この状態では、サーバーは顕著に遅くなります。もしKeepaliveを有効にしておく場合はKeepAliveTimeoutを「低」(KeepAliveTimeoutディレクティブの詳細については、項10.5.8を参照して下さい。)にセットしサーバー上の/var/log/httpd/error_logログファイルをモニタしておくと良いでしょう。このログはサーバーが子プロセスが無くなる時に報告してきます。

10.5.7. MaxKeepAliveRequests

このディレクティブは、固定接続1つに対して許可される要求の最大数をセットします。Apache Projectは、サーバーのパフォーマンスが向上する高い設定を推奨します。MaxKeepAliveRequestsは、デフォルトで100にセットしてありますが、これでほとんどの状況には適切なはずです。

10.5.8. KeepAliveTimeout

KeepAliveTimeoutは要求が対応された後、接続を閉じるまでのサーバーが待つ時間を秒数で設定します。サーバーが要求を受信すると、Timeout ディレクティブが代わりに適用されます。KeepAliveTimeoutはデフォルトで15 秒に設定されています。

10.5.9. MinSpareServers 及びMaxSpareServers

Apache HTTP サーバーは、トラフィックをベースにして適切な数のスペアサーバープロセスを管理することにより、動的に認識されるロードを受け入れます。サーバーは要求待ちのサーバーの数をチェックし、もしMaxSpareServers 以上にある場合は幾つかをキルして、もしその数がMaxSpareServersよりも少ない場合は、幾つか作成します。

デフォルトのMinSpareServers値は5で、デフォルトのMaxSpareServers値は20です。これらのデフォルト設定は、状況に合わせて適切にする必要があります。MinSpareServersをあまり大きな数字にしない様に気を付ける必要があります。大きくし過ぎると、トラフィックが軽い状態の時でもサーバー上に大きなプロセッシング負荷をかけてしまいます。

10.5.10. StartServers

StartServersセットアップの時点で、作成されるサーバープロセスの数を設定します。Webサーバーは、トラフィックロードをベースにしてサーバープロセスを動的にキルしたり、作成したりしますので、このパラメータを変更する必要があります。Webサーバーは、開始時点で8つのサーバープロセスを始動するようにセットしてあります。

10.5.11. MaxClients

MaxClientsは、同時に動作できるサーバープロセスの合計数、又は同時に接続できるクライアントの合計数の限度を設定します。このディレクティブの主な理由は暴走するApache HTTP サーバーがオペレーティングシステムをクラッシュさせる可能性を避けるためです。忙しいサーバーにとっては、この値は高めに設定すべきです。サーバーのデフォルトは150です。この値がMaxClients用の設定で256を越えないようにすることが推奨されます。

10.5.12. MaxRequestsPerChild

MaxRequestsPerChildは、各子サーバープロセスが消滅する前に対応する要求の最大数合計を設定します。MaxRequestsPerChildを設定する主な理由は、長く残存するプロセスにより誘導されたメモリリークを避けるためです。サーバー用のデフォルトMaxRequestsPerChildは1000になっています。

10.5.13. Listen

Listenコマンドは、Webサーバーが要求の来信を受け付けるポートを識別します。デフォルトでApache HTTP サーバーは、非セキュアなWeb 通信をポート80で、そして(セキュアサーバーを定義

する/etc/httpd/conf.d/ssl.conf内の) セキュアなWeb 通信をポート443でリッスン(監視)します。

Apache HTTP サーバーが1024以下のポートでリッスンするように設定されている場合は、ルートユーザーが開始することになります。1024及びそれ以上のポートでは通常のユーザーがhttpdを開始することが出来ます。

Listenディレクティブは、サーバーが受理する接続の為の特定のIP アドレスも指定することが出来ます。

10.5.14. Include

Includeを使用するとランタイム時に他の設定を含む様にすることができます。

これらの設定ファイルのパスはServerRootに対して絶対パスでも相対パスでも使用できます。

重要

mod_ssl、mod_perl、phpなどの個別にパッケージされたモジュールを使用するサーバーには、httpd.conf内のSection 1: Global Environmentの中に次のようなディレクティブがなければなりません：

```
Include conf.d/*.conf
```

10.5.15. LoadModule

LoadModuleは、DSO(Dynamic Shared Object)をロードする為に使用されます。LoadModuleの使い方を含めたApache HTTP サーバーのDSOサポートに関する情報は項10.7で参照できます。モジュールのロード順序はApache HTTP サーバー2.0 の中では重要ではありません。Apache HTTP サーバー2.0のDSOサポートに関しては項10.2.1.3を御覧下さい。

10.5.16. ExtendedStatus

ExtendedStatusディレクティブは、server-statusハンドラがコールされる時点で、Apacheが基本(off)又は詳細(on)のどちらのサーバーステータス情報を生成するのかを制御します。server-statusハンドラはLocationタグを使用してコールされます。server-statusのコールに関する詳細は項10.5.63に含まれています。

10.5.17. IfDefine

<IfDefine>及び、</IfDefine>のタグは、<IfDefine>タグ内にあるテストが「true」であれば、適用される設定ディレクティブを囲みます。テストが「false」であれば、ディレクティブは無視されます。

<IfDefine>タグ内のテストは、パラメータ名(例えばHAVE_PERL)です。パラメータが定義されているとそれは、サーバーのスタートコマンドへの引数として提供されていると言う意味であり、テストは「true」です。この場合、Webサーバーがスタートするとテストは「true」で、IfDefineタグ内に含まれるディレクティブが適用されます。

デフォルトで、<IfDefine HAVE_SSL>タグはセキュアサーバー用の仮想ホストタグを囲みます。<IfDefine HAVE_SSL>タグは、ssl_module用にLoadModuleタグとAddModuleタグも囲みます。

10.5.18. User

Userディレクティブは、サーバープロセスのユーザー名をセットしてどのファイルにそのサーバーがアクセスを許可されるかを決定します。このユーザーにアクセスできないファイルは、Apache HTTP サーバーに接続しているクライアントにもアクセス出来ません。

デフォルトで、Userは、apacheにセットされています。

注意

セキュリティの目的でApache HTTP サーバーは、ルートユーザーとしての実行を拒否します。

10.5.19. Group

これは、Apache HTTP サーバープロセスのグループ名を指定します。

デフォルトでは、Groupはapacheにセットされています。

10.5.20. ServerAdmin

ServerAdminディレクティブをWebサーバー管理者の電子メールアドレスにセットします。この電子メールアドレスがサーバーが生成するWebページのエラーメッセージ内に表示されますので、ユーザーは電子メールをサーバー管理者に送ることにより問題の報告をすることが出来ます。

デフォルトでは、ServerAdminはroot@localhostにセットされています。

ServerAdminをセットする一般的な方法は、それをwebmaster@example.comにセットすることです。そしてwebmasterを、/etc/aliasesのWebサーバーの責任者にエイリアス設定し、/usr/bin/newaliasesを実行します。

10.5.21. ServerName

サーバー用にServerNameを使用してホスト名とポート番号(Listenディレクティブと一致すること)を設定します。ServerNameはマシンの実際のホスト名に一致する必要はありません。例えば、Webサーバーがwww.example.comである時、サーバーのホスト名は実際foo.example.comであることが可能です。ServerNameに指定してある値は、システムにより解決できる有効なDNS(Domain Name Service)でなければなりません。これは勝手に作りあげることは出来ません。

以下に、ServerNameディレクティブのサンプルを示します：

```
ServerName www.example.com:80
```

ServerNameを指定する時は、確実にIPアドレスとサーバー名の対が/etc/hostsファイル内に含まれるようにして下さい。

10.5.22. UseCanonicalName

このディレクティブはonにセットされている時、Apache HTTP サーバーが、ServerNameとPortの両ディレクティブで指定された値を使用して自身を参照するように設定します。UseCanonicalNameがoffにセットされている場合、サーバーは、自身を参照する時、要求しているクライアントで使用された値をその代わりに使います。

UseCanonicalNameは、デフォルトでoffにセットされています。

10.5.23. DocumentRoot

DocumentRootは、要求に対応してサービスされる殆どのHTMLファイルを含んでいるディレクトリです。セキュアと非セキュアの両方ののWebサーバー用のデフォルトDocumentRootは/var/www/htmlディレクトリです。例えば、サーバーは次のようなドキュメントの要求を受信するかも知れません：

```
http://example.com/foo.html
```

この場合、サーバーはデフォルトディレクトリ内で以下のファイルを検索します：

```
/var/www/html/foo.html
```

DocumentRootがセキュアと非セキュアの両タイプのWebサーバーで共有されないようにする為には、DocumentRootを変更します。項10.8を参照して下さい。

10.5.24. Directory

<Directory /path/to/directory>タグ及び</Directory>タグは、*container*と呼ばれる物を作成し、特定のディレクトリやサブディレクトリにのみ適用されることになっている設定ディレクティブのグループを囲むために使用されます。ディレクトリに適用されるディレクティブはいずれも<Directory>タグ内で使用されます。

デフォルトでは、Options(項10.5.25を参照)とAllowOverride(項10.5.26を参照)のディレクティブを使用して、非常に限定的なパラメータがルートディレクトリに適用されます。この設定では、より許可を必要とするシステム上のディレクトリは、これらの設定が明確に与えられる必要があります。

デフォルトの設定では、もう1つのDirectoryコンテナがDocumentRoot用に設定され、これはディレクトリツリーにより穏やかなパラメータを割り当てますので、Apache HTTP サーバーはそこに存在するファイルにアクセスすることが出来ます。

Directoryコンテナは、ScriptAlias ディレクティブに指定してあるディレクトリの外部にある、サーバーサイドアプリケーションの為の追加のcgi-binディレクトリを設定するのにも使用できます。(ScriptAliasディレクティブについての詳細は項10.5.44で御覧下さい。

これを達成するには、DirectoryコンテナはExecCGIオプションをそのディレクトリ用にセットする必要があります。

例えば、CGIスクリプトが/home/my_cgi_directoryに配置してある場合、次のDirectoryコンテナをhttpd.confファイルに追加して下さい：

```
<Directory /home/my_cgi_directory>
  Options +ExecCGI
</Directory>
```

次に、AddHandlerディレクティブを、アンコメントして.cgi拡張付のファイルをCGIスクリプトとして識別する必要があります。AddHandlerの設定についての案内は項10.5.59で御覧下さい。

これが機能するためには、CGIスクリプトとそのスクリプトへのパス全体用の権限は0755にセットされなければなりません。

10.5.25. Options

Optionsディレクティブは、特定のディレクトリでどのサーバー機能が利用できるかを制御します。例えば、ルートディレクトリに指定してある限定パラメータの元でOptionsは、FollowSymLinksにのみセットされます。サーバーはルートディレクトリ内のシンボリックリンクに追従する許可がある以外はどの機能も有効になりません。

デフォルトでは、DocumentRootディレクトリ内でOptionsはIndexesとFollowSymLinksを含むようにセットされています。Indexesは、DirectoryIndex (例えば；index.html)が指定されていない場合、サーバーに1つのディレクトリ用にディレクトリリストを生成する許可をします。そしてFollowSymLinksはサーバーにそのディレクトリ内でシンボリックリンクに追従するよう許可をします。

注意

メインサーバー設定セクションからのOptionsステートメントは各VirtualHostコンテナに個別に反復する必要があります。VirtualHostコンテナに関する詳細は項10.5.69で御覧下さい。

10.5.26. AllowOverride

AllowOverrideディレクティブは、.htaccessファイル内の宣言でOptionsが上書きされるかどうかをセットします。デフォルトでは、ルートディレクトリも、DocumentRootも.htaccessの上書きを許可しないようにセットされています。

10.5.27. Order

Orderディレクティブは、allowとdenyのディレクティブが評価をされる順序を制御します。DocumentRootディレクトリ用には、サーバーはDenyディレクティブの前にAllowディレクトリを評価するように設定されています。

10.5.28. Allow

Allowは、どの要求者が該当するディレクトリへアクセスできるかを指定します。要求者には、all、ドメイン名、IPアドレス、部分IPアドレス、ネットワーク/ネットマスクの対、その他などがあります。DocumentRoot ディレクトリはallからの要求をAllowするように設定されており、これは全てがアクセスを持つことを意味します。

10.5.29. Deny

Denyは、誰がアクセスを拒否されるかを指定すること以外はAllowと同様に機能します。DocumentRootは、デフォルトでは、誰の要求にもDenyしないように設定されています。

10.5.30. UserDir

UserDirは、各ユーザーのホームディレクトリ内のサブディレクトリの名前です。ここにWebサーバーでサービスされる個人のHTMLファイルを配置します。このディレクティブは、デフォルトでdisableにセットされています。

サブディレクトリの名前は、デフォルトの設定でpublic_htmlにセットされています。例えば、サーバーは次のような要求を受信するかも知れません：

```
http://example.com/~username/foo.html
```

サーバーは次のファイルを見付けようとします：

```
/home/username/public_html/foo.html
```

上記の例では、/home/username/はユーザーのホームディレクトリです。(ユーザーのホームディレクトリへのデフォルトパスは固定ではないことに注意して下さい)。

ユーザーのホームディレクトリへの権限が正しく設定されていることを確認して下さい。ユーザーのホームディレクトリは0711へセットする必要があります。読み(r)と実行(x)のビットがユーザーのpublic_html ディレクトリ(0755 も可能です)にセットする必要があります。ユーザーのpublic_htmlディレクトリでサービスされるファイルは最低でも0644にセットする必要があります。

10.5.31. DirectoryIndex

DirectoryIndexは、サーバーによりサービスされるデフォルトページで、ユーザーがディレクトリ名の後ろに順スラッシュ(/)を指定してディレクトリのインデックスを要求した時のサービスです。

ユーザーが、ページhttp://*example*/*this_directory*/, を要求すると、存在する場合はDirectoryIndexページが、そうでなければ、サーバー生成のディレクトリリストが提供されます。DirectoryIndex用のデフォルトはindex.htmlとindex.html.varのタイプマップです。サーバーはそれらのファイルのいずれかを探し、最初に見付かった物を表示します。ファイルを見付けることが出来ない場合は、Options Indexesがそのディレクトリにセットされているため、ディレクトリリストの機能が停止されていなければ、サーバーはHTML形式でそのディレクトリ内のサブディレクトリとファイルのリストを生成して表示します。

10.5.32. AccessFileName

AccessFileNameは、各ディレクトリ内のアクセス制御用にサーバーが使用すべきファイルの名前を示します。デフォルトでは.htaccessとなっています。

AccessFileNameディレクティブの直後に、Filesタグのセットが.htで始まる全てのファイルのアクセス制御を適用します。これらのディレクティブは、セキュリティの目的で.htaccessファイル(又は、.htで始まる他のファイル)へのいかなるWebアクセスも拒否します。

10.5.33. CacheNegotiatedDocs

デフォルトでは、Webサーバーはプロキシサーバーに対して、内容を基準にして交渉したドキュメントはいずれもキャッシュしないように注文をつけます。(これらの内容は、要求者からの入力により時間経過と共に変化するものです)。CacheNegotiatedDocsが、onにセットしてある場合、その機能を無効にしてプロキシサーバーにドキュメントキャッシュを許可します。

10.5.34. TypesConfig

TypesConfigは、デフォルトのMIMEタイプのマッピングのリスト(ファイル名拡張子からコンテンツタイプへ)を設定するファイルの名前を指定します。デフォルトのTypesConfigファイルは/etc/mime.typesです。/etc/mime.typesを編集する代わりに、MIMEタイプマッピングを追加する方法はAddTypeディレクティブを使用することです。

AddTypeについての詳細は項10.5.58で御覧下さい。

10.5.35. DefaultType

DefaultTypeはWebサーバーの為にデフォルトのコンテンツタイプをセットしてMIMEタイプが決定できないドキュメント用に使用します。デフォルトでは、text/plainです。

10.5.36. IfModule

<IfModule>タグと</IfModule>タグは特定のモジュールがロードされた時にのみ始動される条件付のコンテナを作成します。IfModule内に含まれているディレクティブは、1つ又は2つの条件でプロセスされます。開始用の<IfModule>タグに含まれるモジュールがロードされるとディレクティブはプロセスされます。あるいは、感嘆符[!]がモジュール名の前にある場合、<IfModule>タグ内に指定してあるモジュールがロードされない時にのみ、ディレクティブはプロセスされます。

Apache HTTP サーバーモジュールに関する詳細は項10.7で御覧下さい。

10.5.37. HostnameLookups

HostnameLookupsは 、on、off、doubleのいずれかにセットされます。HostnameLookupsがonにセットしてある場合、サーバーは自動的に各接続のIPアドレスを解決します。IPアドレスを解決するとは、サーバーがプロセッシングオーバーヘッドを追加することにより、DNSサーバーに1つ、又は、複数の接続をすることを意味します。もしdoubleにHostnameLookupsがセットされている場合、サーバーはダブルのDNSルックアップをしてより多くのプロセッシングオーバーヘッドを追加します。

サーバー上のリソースを節約する為に、デフォルトではHostnameLookupsはoffにセットしてあります。

ホスト名がサーバーログファイル内に要求される場合、サーバーログファイルを巡回している時に、より効率良く大量のDNSルックアップを達成する複数のログアナライザーの中から1つを実行して見て下さい。

10.5.38. ErrorLog

ErrorLogは、サーバーエラーがログしてあるファイルを指定します。デフォルトでは、このディエクティブは/var/log/httpd/error_logにセットしてあります。

10.5.39. LogLevel

LogLevelは、エラーログ内のエラーメッセージの詳細レベルをセットします。LogLevelは、(低詳細度から高詳細度まで) emerg、alert、crit、error、warn、notice、info、debug等を設定できます。デフォルトのLogLevelはwarnです。

10.5.40. LogFormat

LogFormatディレクティブは、各種のWebサーバーログファイルの形式を設定できます。実際に使用されるLogFormatは、CustomLogディレクティブにある設定に左右されます。(項10.5.41を参照して下さい)。

以下に、CustomLogディレクティブがcombinedに設定してある場合のフォーマットオプションを示します：

%h (リモートホストのIPアドレス又はホスト名)

‘ 要求を出しているクライアントのリモートIPアドレスをリストします。HostnameLookupsがonに設定してある場合、DNSで入手できない場合を除いて、クライアントホスト名が記録されます。

%l (rfc931)

‘ 使用されていません。このフィールドのログファイルにはハイフン[-]が表示されます。

%u (認証されたユーザー)

‘ 認証が必要な場合、ユーザーの記録があるユーザー名をリストします。通常、これは使用しません。このフィールドのログファイルにはハイフン[-]が表示されます。

%t (日付)

‘ 要求の日付と時刻をリストします。

%r (要求の文字列)

‘ ブラウザ又はクライアントから来たままの要求の文字列をリストします。

%s (ステータス)

‘ クライアントホストに返されたHTTPステータスコードをリストします。

%b (バイト)

‘ ドキュメントのサイズをリストします。

%\"%{Referer}i\" (参照元)

‘ クライアントホストをWebサーバーに照会したwebページをURLでリストします。

%\"%{User-Agent}i\" (ユーザー/エージェント)

‘ 要求を出しているWebブラウザのタイプをリストします。

10.5.41. CustomLog

CustomLogは、ログファイルとログファイル形式を識別します。デフォルトでは、ログは/var/log/httpd/access_logファイルに記録されます。

デフォルトのCustomLog形式はcombinedです。以下にcombinedログファイルの形式を示します：

remotehost rfc931 user date "request" status bytes referrer user-agent

10.5.42. ServerSignature

ServerSignatureディレクティブは、Apache HTTP サーバーサーバーやServerNameを含む行を、クライアントに戻されたエラーメッセージなどのサーバー生成ドキュメントに追加します。デフォルトでは、ServerSignatureはonにセットされています。

これは、offやEMailにも設定できます。EMailは、mailto:ServerAdminHTMLタグを応答自動生成の著名行に追加します。

10.5.43. Alias

Alias設定により、DocumentRootディレクトリ外部のディレクトリがアクセスできるようになります。エイリアスで終了しているURLはすべてそのエイリアスのパスに名前解決をします。デフォルトでは、1つのiconsディレクトリ用に1つのエイリアスが既に設定してあります。iconsディレクトリはWebサーバーによりアクセスできますが、そのディレクトリはDocumentRootの中ではありません。

10.5.44. ScriptAlias

ScriptAliasディレクティブは、CGIスクリプトがロードされる場所を定義します。一般には、CGIスクリプトをDocumentRootの中に残すのはよくありません。そこでは、テキストドキュメントとして覗かれてしまう可能性があります。この理由で、DocumentRootディレクトリの外にサーバーサイドの実行可能ファイルとスクリプトを収納した特別なディレクトリがScriptAliasディレクティブによって指定されます。このディレクトリはcgi-binとして知られ、デフォルトでは/var/www/cgi-bin/にセットされています。

cgi-binディレクトリの外部に実行可能ファイルを保存する為のディレクトリを確立することは可能です。その方法の案内は項10.5.59と項10.5.24を御覧下さい。

10.5.45. Redirect

webページが移動すると、新しいURLへファイルのロケーションをマップするためにRedirectが使用されます：

```
Redirect /<old-path>/<file-name> http://<current-domain>/<current-path>/<file-name>
```

この例では、古いパス情報<*old-path*>の<*file-name*>と<*current-domain*>と<*current-path*>の部分を、<*file-name*>の為の現在のドメインとパス情報で入れ換えます。

この例では、古い場所での<*file-name*>への要求は自動的に新しい場所に転送されます。

より高度な転送技術として、Apache HTTP サーバーに収納されているmod_rewriteモジュールを使用する方法があります。mod_rewriteの設定法に関する情報は、以下のApache Software Foundationのサイトでオンラインドキュメントをお読み下さい。http://httpd.apache.org/docs-2.0/mod/mod_rewrite.html.

10.5.46. IndexOptions

IndexOptionsは、アイコンの追加、ファイルの説明、その他により、サーバー生成のディレクトリ一覧の表示を制御します。Options Indexes(項10.5.25を参照)がセットされている場合、Web サーバーがインデックスのないディレクトリの為のHTTP要求を受信した時、Web サーバーは、ディレクトリ一覧を生成します。

先ず、WebサーバーはDirectoryIndexディレクティブ内のリストしてある名前と一致するファイルを求めて、要求されたディレクトリを探索します。(通常index.html)。index.html ファイルが見付からない場合、Apache HTTP サーバーは要求されたディレクトリのHTMLディレクトリ一覧を作成します。このディレクトリ一覧の表示は一部、IndexOptionsディレクティブによって制御されます。

デフォルト設定では、FancyIndexingが起動されます。これは、ユーザーはコラムヘッダをクリックすることで、ディレクトリ一覧を再構成することができるという意味です。そのヘッダをもう1度クリックすると、上昇順から下降順に転換出来ます。FancyIndexingは、またファイル拡張子に応じて異なるファイルには異なるアイコンを表示します。

AddDescriptionオプションは、FancyIndexingと併用された時、サーバー生成のディレクトリ一覧内にあるファイル用に短い説明を提供します。

IndexOptionsは、サーバー生成のディレクトリの表示を制御する為にセットされた他のパラメータを数多く持っています。これらのパラメータにはIconHeightとIconWidthが含まれており、サーバーが、その生成したwebページ内のアイコンの為にHTMLHEIGHTタグとHTMLWIDTHタグを含むようにします。アイコンを作成する為のIconsAreLinksは、ファイル名、その他と一緒にHTMLリンクアンカーの1部として機能します。

10.5.47. AddIconByEncoding

このディレクティブは、サーバー生成のディレクトリ一覧内のMIMEエンコーディングと共にファイルによって表示されるアイコンの名前を示します。たとえば、デフォルトでWebサーバーはcompressed.gifのアイコンを、サーバー生成のディレクトリ一覧内にMIMEでエンコードしたx-compress とx-gzip ファイルの次に表示します。

10.5.48. AddIconByType

このディレクティブは、サーバー生成のディレクトリ一覧内のMIMEタイプと一緒にあるファイルの次に表示されているアイコンの名前を示します。例えば、サーバーは、サーバー生成のディレクトリ一覧内で、mime-タイプのtextと共にあるファイルの次のtext.gifアイコンを表示します。

10.5.49. AddIcon

AddIconは、特定の拡張子を持つファイル用にサーバー生成のディレクトリ一覧で表示するアイコンを指定します。例えば、Webサーバーは.bin 又は.exeの拡張子を持つファイル用のbinary.gifアイコンを表示するようにセットされています。

10.5.50. DefaultIcon

DefaultIconは、他にアイコンの指定がないファイルの為にサーバー生成のディレクトリ一覧内に表示するアイコンを指定します。デフォルトはunknown.gifイメージファイルです。

10.5.51. AddDescription

FancyIndexingをIndexOptionsパラメータとして使用する時、サーバー生成したディレクトリ一覧内の特定のファイル又はファイルタイプ用にAddDescriptionディレクティブを使用してユーザー指定の説明を表示することができます。AddDescriptionディレクティブは特定のファイル、ワイルドカード表現、ファイル拡張子などの一覧をサポートします。

10.5.52. ReadmeName

ReadmeNameは、ディレクトリ内にそれが存在する場合、サーバー生成のディレクトリ一覧の末尾に追加されているファイルの名前を指定します。Webサーバーは、最初にそのファイルをHTML 文書としてインクルードしようとします。その後、それを平文としてインクルードしようとします。デフォルトでは、ReadmeNameはREADME.htmlにセットされています。

10.5.53. HeaderName

HeaderNameは、それがディレクトリに存在する場合、サーバー生成のディレクトリ一覧の前に追加されているファイルの名前を指定します。ReadmeNameと同様にサーバーはそれをHTML文書としてインクルードを試み、それが出来なければ平文としてインクルードします。

10.5.54. IndexIgnore

IndexIgnoreは、ファイル拡張子、部分ファイル名、ワイルドカード表現、完全ファイル名等を一覧表示します。Webサーバーは、サーバー生成のディレクトリ一覧にあるそれらのパラメータのいずれかに一致するファイルはインクルードしません。

10.5.55. AddEncoding

AddEncoding は、特定のエンコードタイプを指定するファイル名拡張子の名前を指定します。AddEncodingは、また幾らかのブラウザに対してダウンロードされる特定のファイルを展開するように指示することにも使用できます。

10.5.56. AddLanguage

AddLanguageは、ファイル名拡張子を特定の言語を関連付けします。このディレクティブは、クライアント上のWebブラウザの言語設定を基にした複数言語の内容をサービスするApache HTTP サーバーの為に役に立ちます。

10.5.57. LanguagePriority

LanguagePriorityは、クライアントのWebブラウザが言語の設定をしていない場合、別の言語の為の前例をセットします。

10.5.58. AddType

MIMEタイプとファイル拡張子の対を定義するには、AddTypeディレクティブを使用します。例えば、PHP4を使用する場合、AddTypeディレクティブを使用してWebサーバーにPHPファイル拡張子(.php4、.php3、.phtml、.php)をPHP MIMEタイプとして認識させます。次のディレクティブは、Apache HTTP サーバー対して.shtmlファイル拡張子を認識するように指示しています：

```
AddType text/html .shtml
AddHandler server-parsed .shtml
```

10.5.59. AddHandler

AddHandlerは、ファイル拡張子を特定のハンドラにマップします。例えば、cgi-scriptハンドラは拡張子.cgiと適合し、自動的に末尾に.cgiが付いているファイルをCGIスクリプトとして取り扱います。以下のAddHandlerディレクティブのサンプルは.cgi拡張子用の物です。

```
AddHandler cgi-script .cgi
```

このディレクティブは、cgi-binの外部にあるCGIを有効にしてディレクトリコンテナ内のExecCGIオプションを持つ、サーバー上のいずれかのディレクトリ内で機能します。あるディレクトリ用にExecCGI オプションを設定する方法に関しては項10.5.24を御覧下さい。

CGIスクリプトに加えて、AddHandlerディレクティブはサーバー構文解析されたHTMLとイメージマップファイルをプロセスする為に使用されます。

10.5.60. Action

Actionは、MIMEのコンテンツタイプとCGIのスクリプトペアを指定しますので、そのメディアタイプのファイルが要求された時には、特定のCGIのスクリプトが実行されます。

10.5.61. ErrorDocument

ErrorDocumentディレクティブは、HTTP応答コードとクライアントから返送されるメッセージ又はURLを関連付けます。デフォルトでは、エラーが発生した時lにWebサーバーが簡単な、暗号的なエラーメッセージを出力します。ErrorDocumentディレクティブは強制的に、Webサーバーがカスタマイズのメッセージを出力するか、クライアントをローカル又は、外部のURLへ転送するようにします。


重要

有効であるためには、メッセージは1対の2重引用符["]で囲まれている必要があります。

10.5.62. BrowserMatch

BrowserMatchディレクティブにより、サーバーは環境変数を定義してユーザー/エージェントHTTPヘッダフィールド(クライアントのWebブラウザタイプを識別します) を基にして適切な操作をします。デフォルトでは、WebサーバーはBrowserMatchを使用して問題が判別している特定のブラウザには接続を否定し、またその動作で問題があると知られているブラウザ用のkeepalives とHTTPヘッダフラッシュを無効にします。

10.5.63. Location

<Location>タグと</Location>タグはURLを基にしたアクセス制御がその中で指定できるコンテナを作成します。

例えば、ステータス報告を見るためにサーバーのドメインから接続している人々には、以下のようなディレクティブを使用します :

```
<Location /server-status>
  SetHandler server-status
  Order deny,allow Deny from all
  Allow from <.example.com>
</Location>
```

<.example.com>を、Webサーバー用の第2レベルのドメイン名で入れ換えます。

サーバー設定の報告(インストール済のモジュールと設定ディレクティブを含む)をドメインの内部からの要求へ提供するには、次のようなディレクティブを使用します :

```
<Location /server-info>
  SetHandler server-info
  Order deny,allow
  Deny from all
  Allow from <.example.com>
</Location>
```

同様に、<.example.com>を、Webサーバー用の第2レベルのドメイン名で入れ換えます。

10.5.64. ProxyRequests

Apache HTTP サーバーとプロキシサーバーとして機能するように設定するには、<IfModule mod_proxy.c>行の先頭にあるハッシュマーク(#)を取り除いて、mod_proxy モジュールをロードしてから、ProxyRequestsディレクティブをOnにセットします。

10.5.65. Proxy

<Proxy *>タグと</Proxy>タグはプロキシサーバーのみに適用されることになっている設定ディレクティブのグループを囲むコンテナを作成します。<Directory>コンテナ内に許可されている多くのディレクティブは<Proxy>コンテナの中でも使用することができます。

10.5.66. ProxyVia

ProxyViaコマンドはHTTP Via:ヘッダの行が、プロキシサーバーを通過していく要求又は応答と共に送信されるかどうかを制御します。Via:ヘッダは、ProxyViaがOnにセットされている場合、ホスト名を表示します。Fullの場合にはホスト名とApache HTTP サーバーバージョンを表示し、Offの場合には、Via:の行を変更なしにパスして、Blockの場合には、Via:の行は削除されます。

10.5.67. キャッシュディレクティブ

多くのコメント化されたキャッシュディレクティブがデフォルトのApache HTTP サーバー設定ファイルで供給されています。殆どの場合、ハッシュマーク「[#]」をその行の先頭から削除してアンコメント化することで充分です。以下により重要なキャッシュ関連のディレクティブの幾つかのリストを示します。

- CacheRoot — キャッシュファイルを含んでいるディレクトリ名を指定します。デフォルトCacheRootは/var/httpd/proxy/ディレクトリです。
- CacheSize — キャッシュが使用できる容量をキロバイトで指定します。デフォルトCacheSizeは5 KBです。
- CacheGcInterval — キャッシュ内のファイルが削除されるまでに経過すべき期間を時間数で指定します。デフォルトのCacheGcIntervalは4時間です。
- CacheMaxExpire — キャッシュ内に文書が保存される(オリジナルWebサーバーからのリロードなしで)長さを指定します。デフォルトでは24時間です。
- CacheLastModifiedFactor — 文書の内、送信元のサーバーからその期限が設定されて来なかった文書の為に期限の作成を指定します。デフォルトでは、CacheLastModifiedFactorは0.1にセットされており、これはその文書の期限は最後にその文書が編集されてから経過した時間の10分の1に等しいという意味です。
- CacheDefaultExpire — 期限時間をサポートしないプロトコルを使用して受信された文書の為に期限を時間で指定します。デフォルトでは1時間にセットしてあります。
- NoCache — 内容がキャッシュ化されていないホストのリストを指定します。

10.5.68. NameVirtualHost

NameVirtualHostディレクティブは、必要であれば、名前ベースの仮想ホストの為にIPアドレスとポート番号を関連付けします。名前ベースの仮想ホストにより、Apache HTTP サーバーの1つは複数IPアドレスを使用せずに異なるドメインにサービスすることが出来ます。

注意

名前ベースの仮想ホストは、非セキュアなHTTP接続でのみ 機能します。セキュアサーバーを使用して仮想ホストを使う場合は、代わりにIPアドレスベースの仮想ホストを使用して下さい。

名前ベースの仮想ホストを有効にするには、NameVirtualHostの設定ディレクティブをアンコメントして、正しいIPアドレスを追加します。その後、各仮想ホストに他のVirtualHostコンテナを追加します。

10.5.69. VirtualHost

<VirtualHost>タグと</VirtualHost>タグは仮想ホストの性格を形成するコンテナを作成します。<VirtualHost> コンテナは殆どの設定ディレクティブを受け付けます。

コメント化されたVirtualHostコンテナのセットが、httpd.confによって提供されます。これは各仮想ホストに必要な設定ディレクティブのミニマムセットを表現しています。仮想ホストに関する情報は項10.8で御覧下さい。

注意

SSL仮想ホストのコンテナはすべて、/etc/httpd/conf.d/ssl.conf ファイルに移動されています。

10.5.70. SSL 設定ディレクティブ

/etc/httpd/conf.d/ssl.confファイル内のSSL ディレクティブは、SSL とTLSを使用してセキュアなWeb通信を有効するための設定ができます。

10.5.70.1. SetEnvIf

SetEnvIfは、着信するセキュアな接続のヘッダを基にした環境変数の設定をします。供給された/etc/httpd/conf.d/ssl.conf ファイルで、これはHTTP keepaliveを無効にして、SSLを許可するのに使用され、クライアントブラウザからの閉じる為のアラート通知なしで接続を閉じることが出来ます。この設定は確実にSSL接続を閉じることが出来ない特定のブラウザの為に必要です。

SSLディレクティブに関する情報は、以下のアドレスのサイトで確認できます：

- http://localhost/manual/mod/mod_ssl.html
- http://httpd.apache.org/docs-2.0/mod/mod_ssl.html

Apache HTTPセキュアサーバーの設定に関する案内は、*Red Hat Linux* カスタマイズガイドの中の*Apache HTTP* セキュアサーバーの設定の章でお読み下さい。

注意

殆どの場合、SSLディレクティブはインストールされている状態で適切に設定されています。設定ミスは、セキュリティの弱点の原因になる可能性がありますのでApache HTTPセキュアサーバーディレクティブの変更には充分に気を付けて下さい。

10.6. デフォルトのモジュール

Apache HTTP サーバーは多くのモジュールを収納した状態で配布されています。デフォルトの状態で、以下のモジュールが、インストール済でRed Hat Linuxの`httpd`パッケージと共に有効になっています：

```
mod_access
mod_auth
mod_auth_anon
mod_auth_dbm
mod_auth_digest
mod_include
mod_log_config
mod_env
mod_mime_magic
mod_cern_meta
mod_expires
mod_headers
mod_usertrack
mod_unique_id
mod_setenvif
mod_mime
mod_dav
mod_status
mod_autoindex
mod_asis
mod_info
mod_cgi
mod_dav_fs
mod_vhost_alias
mod_negotiation
mod_dir
mod_imap
mod_actions
mod_speling
mod_userdir
mod_alias
mod_rewrite
mod_proxy
mod_proxy_ftp
mod_proxy_http
mod_proxy_connect
```

さらには、以下のモジュールは、追加のパッケージをインストールすることにより利用できるようになります：

```
mod_auth_mysql
mod_auth_pgsql
mod_perl
mod_python
mod_ssl
php
squirrelmail
```

10.7. モジュールの追加

Apache HTTP サーバーは*DSO*(*Dynamically Shared Objects*) と言うモジュールをサポートします。これは必要に応じてランタイムに簡単にロードすることができます。

Apache Project は、総括的なDSOドキュメントを次のサイトで提供しています。http://httpd.apache.org/docs-2.0/dso.html. また、`http-manual`パッケージがインストールされている場合、DSO関連のドキュメントはhttp://localhost/manual/mod/ で御覧下さい。

Apache HTTP サーバーで、DSOを使用するには、`/etc/httpd/conf/httpd.conf`の中の`LoadModule`ディレクティブで、DSOが指定される必要があります。もし、このモジュールが別のパッケージで提供されている場合、`/etc/httpd/conf.d/` ディレクトリ内のモジュール設定ファイルの中にその行が表示されなければなりません。`LoadModule`ディレクティブに関する詳細は項10.5.15で御覧下さい。

`http.conf`からモジュールの追加や削除をする場合、Apache HTTP サーバーは、項10.4で示してあるようにリロード又は、再スタートされる必要があります。

新しいモジュールを作成している場合、インクルードファイル、ヘッダファイル、*APache eXtenSion*(`/usr/sbin/apxs`) アプリケーション(これはDSOのコンパイルにインクルードファイルとヘッダファイルを使用します。)を収納している`httpd-devel`パッケージを先にインストールします。

モジュールを書き終ると、`/usr/sbin/apxs`を使用して、Apacheのソースツリーの外でモジュールソースをコンパイルします。`/usr/sbin/apxs` コマンドの使用法に付いてはオンラインのApache ドキュメントを以下のサイトで確認してください。http://httpd.apache.org/docs-2.0/dso.html `apxs`については、そのmanページを御覧下さい。

コンパイルが終了すると、そのモジュールを`/usr/lib/httpd/` ディレクトリの中にいれます。その後以下の構成を使用して、`LoadModule`の行を`httpd.conf`に追加します：

```
LoadModule <module-name> <path/to/module.so>
```

上記の例では、*<module-name>*をモジュール名で入れ換え、*<path/to/module.so>*をDSOへのパス名で入れ換えます。

10.8. 仮想ホスト

Apache HTTP サーバーの組み込み型仮想ホストの使用で、サーバーは、IPアドレス、ホスト名、ポートのどれが要求されているかにより、異なる情報を提供することができます。仮想ホストの総括的なガイドは以下のサイトで御覧になれます。http://httpd.apache.org/docs-2.0/vhosts/.

10.8.1. 仮想ホストの開始

名前ベースの仮想ホストを作成するには、例に示しているように`httpd.conf`で用意されている仮想ホストコンテナを使用するのが最適です。

仮想ホストの例を以下に示します：

```
#NameVirtualHost *
#
#<VirtualHost *>
#  ServerAdmin webmaster@dummy-host.example.com
#  DocumentRoot /www/docs/dummy-host.example.com
#  ServerName dummy-host.example.com
#  ErrorLog logs/dummy-host.example.com-error_log
#  CustomLog logs/dummy-host.example.com-access_log common
#</VirtualHost>
```

名前ベースの仮想ホストを有効にするには、ハッシュマーク(#)を取り除くことでアンコメント化し、アスターリスク(*)をマシンに割り当てられたIPアドレスで入れ換えます。

次に<VirtualHost>コンテナをアンコメント化してカスタマイズして、仮想ホストを設定します。

<VirtualHost>の行でアスタリスク(*)をサーバーのIPアドレスに入れ換えます。そしてServerNameの部分をマシンに割り当てられた有効なDNS名で入れ換えて、必要に応じて他のディレクティブを設定します。

<VirtualHost>コンテナは、カスタマイズしやすく、メインサーバー設定内で利用できる殆どのディレクティブを受け付けます。

ヒント

デフォルト以外のポート上でリッスンする仮想ホストを設定する場合、そのポートは、/etc/httpd/conf/http.confファイルのグローバル設定セクションの中のListenディレクティブに追加される必要があります。

新しく作成された仮想ホストを有効にするには、Apache HTTP サーバーをリロード、又は再スタートする必要があります。この方法に付いての案内は項10.4で御覧下さい。

名前ベースと、IPアドレスベースの両方の仮想ホストの作成とその設定に関する情報は、以下のサイトで御覧下さい。http://httpd.apache.org/docs-2.0/vhosts/.

10.8.2. セキュアWeb サーバー仮想ホスト

デフォルトで、Apache HTTP サーバーは非セキュアサーバーとセキュアサーバーの両方用に設定されています。その非セキュアサーバーとセキュアサーバーは両方共同じ、IPアドレスとホスト名を使用しますが、異なるポートをリッスンします：それぞれ80と443です。この為非セキュアとセキュアな通信は両方共同時に実行することが出来ます。

SSL強化のHTTP発信の1つの特徴は、それが標準のHTTPプロトコルに比べてリソース集中型であり、セキュアサーバーは、1秒間に多くのページをサービス出来ません。この理由で、特にトラフィックの激しいWebサイトでは、利用できる情報を最低限に保つことが得策となります。

重要

HTTP要求が適切な名前ベースの仮想ホストを識別する前にSSLハンドシェイクが起こるため、名前ベースの仮想ホストは、セキュアWebサーバーと一緒に使用しないで下さい。名前ベースの仮想ホストは、非セキュアなWebサーバーでのみ機能します。

セキュアサーバー用の設定ディレクティブは、/etc/httpd/conf.d/ssl.conf ファイルの中の仮想ホストタグの中に収納されています。

デフォルトでは、セキュアも非セキュアもWebサーバーは同じ、DocumentRootを共有しています。ただセキュアWebサーバー用のDocumentRootは異なるものが推奨されます。

非セキュアなWebサーバーが接続を受け付けるのを止めるには、httpd.conf内のListen 80を読む行の先頭にハッシュマーク(#)を付けることによって、その行をコメント化します。それが終了すると以下の例のようになります：

```
#Listen 80
```

SSL強化のWebサーバーの設定に関する情報は、*Red Hat Linux* カスタマイズガイドの中に有る*Apache HTTP* セキュアサーバーの設定という章で御覧下さい。高度な設定のヒントに関しては、Apache Software Foundationの以下のサイトで利用できるオンラインドキュメントを御覧下さい。

- http://httpd.apache.org/docs-2.0/ssl/.
- http://httpd.apache.org/docs-2.0/vhosts/

10.9. その他のリソース

Apache HTTP サーバーに付いてもっと知りたい場合は、以下のリソースを参照して下さい。

10.9.1. 役に立つWebサイト

- http://httpd.apache.org — Apache HTTP サーバーの中央ホームページです。ディレクティブやデフォルトモジュールの総合的な情報があります。
- http://www.modssl.org — `mod_ssl`のホームページです。
- http://www.apacheweek.com — Apacheのすべてに関する週刊オンライン情報です。

10.9.2. 関連書籍

- *Apache Desktop Reference* by Ralf S. Engelschall; Addison Wesley — ASFのメンバーで`mod_ssl`の著者であるRalf Engelschall氏により書かれた*Apache Desktop Reference*は、コンパイル時、設定、そしてラインタイムにApache HTTP サーバーを使用する為の総合的な参照ガイドです。この本はまた、オンラインで次のサイトから入手できます。http://www.apacheref.com/.
- *Professional Apache* by Peter Wainwright; Wrox Press Ltd — *Professional Apache* は経験豊富なあるいは新たなWebサーバー管理者の両方を対象にしたWrox Press Ltdの"Programmer to Programmer"シリーズの書籍です。
- *Administering Apache* by Mark Allan Arnold; Osborne Media Group — さらに安全なサービスを提供することを目的としたインターネットサービスプロバイダーを対象に書かれてします。
- *Apache Server Unleashed* by Richard Bowen, et al; SAMS BOOKS — Apache HTTP サーバーの百科辞典的な情報源です。
- *Apache Pocket Reference* by Andrew Ford, Gigi Estabrook; O'Reilly — O'Reilly Pocket Referenceシリーズに最近追加されたものです。

第11章

電子メール

電子メール(*email*)の誕生は1960年代の初期でした。メールボックスはユーザーにのみ読み込み可能なユーザーのホームディレクトリにあるファイルでした。原始的なメールアプリケーションは、そのファイルの下にテキストメッセージを付けるため、ユーザーは常時増加するファイルを押し退けて目的のメッセージを探す必要がありました。このようなシステムでは、同じシステム上のユーザーにだけメッセージを送信できた状態でした。

実際の電子メールメッセージファイルの最初のネットワーク送信は、コンピュータエンジニアであるRay Tomlinsonがテストメッセージを ARPANETを経由した2つのマシン間で送信した1971年に始まりました。(インターネットの前身です)。電子メールでの通信はすぐに有名になり、2年以内にARPANETのトラフィックの75%を占めていました。

今日、標準化したネットワークプロトコル上の電子メールシステムは、インターネット上で最も使用されるサービスの1部になるまで発展しています。Red Hat Linuxでは、電子メールのサービスとアクセスをする為の多くの高度なアプリケーションを提供しています。

この章では、今日使用されている最近の電子メールプロトコルと電子メールで送受信ができるように設計されているプログラムを説明します。

11.1. 電子メールプロトコル

現在の電子メールはクライアント/サーバーアーキテクチャーを使用して配送されます。電子メールのメッセージはメールクライアントプログラムを使用して作成します。このプログラムがメッセージをサーバーに送り、そのサーバーは受信者側の電子メールサーバーにメッセージを転送します。そこから最終的にメッセージが受信者の電子メールクライアントに供給されます。

このプロセスを有効にするために、多種多様の標準ネットワークプロトコルが異なるマシンを、殆どの場合、異なるオペレーティングシステムと異なる電子メールプログラムで電子メールの送受信を可能にします。

以下で説明してあるプロトコルは、電子メール転送で最も一般に使用されているプロトコルです

11.1.1. Mail Transport Protocols

クライアントアプリケーションからサーバーへと、その送信側のサーバーから受信側のサーバーまでのメールの配送は*SMTP* (*Simple Mail Transfer Protocol*)により処理されます。

11.1.1.1. SMTP

SMTPの主要目的は、メールサーバー間でメールを転送することです。しかし、それが電子メールクライアントにとっても重要になります。メールを送るためには、クライアントはメッセージを配送元のサーバーに送り、そのサーバーが配送先のメールサーバーに配達の連絡をします。この理由で、電子メールクライアントを設定する時に、SMTPサーバーを指定する必要がある訳です。

Red Hat Linuxでは、ユーザーはSMTPサーバーをローカルマシン上で設定してメール配送を処理できます。そして、さらに発信メール用のリモートサーバーの設定もすることができます。

SMTPプロトコルで重要なポイントの1つは、これが認証を必要としないことです。この為、インターネット上の誰でも他の誰かに、あるいは大規模な団体にさえも電子メールを送信することが出来ます。実はこれがゴミメール、すなわち*spam*を可能にするSMTPの性格なのです。最新のSMTP サーバーは、そのサーバーにアクセスできる既知のホストのみに許可をすることでこの様な行為を最小限に抑えています。このような規制をしていないサーバーはオープンリレーサーバーと呼ばれます。

Red Hat Linuxは、Sendmail (`/usr/sbin/sendmail`)をそのデフォルトのSMTPプログラムとして使用します。しかし、より簡単なメールサーバーアプリケーション、Postfix (`/usr/sbin/postfix`)も利用できます。

11.1.2. Mail Access Protocols

電子メールをメールサーバーから取り出すために、電子メールアプリケーションで使用される2つの主要プロトコルがあります。*POP* (*Post Office Protocol*)と*IMAP* (*Internet Message Access Protocol*)です。

SMTPとは異なり、これらのプロトコルは両方とも、ユーザー名とそのパスワードを使用して認証する接続を要求します。デフォルトでは、この両方のプロトコルはネットワーク上で暗号化なしで渡されます。

11.1.2.1. POP

Red Hat Linuxでは、デフォルトのPOPサーバーは`/usr/sbin/ipop3d`であり、`imap`パッケージによって用意されています。POPサーバーを使用する時、電子メールメッセージはメールクライアントアプリケーションによりダウンロードされます。デフォルトで、殆どの電子メールクライアントはメールサーバーからメッセージが正常に転送された後には、自動的にメッセージを削除するように設定されています。しかし、通常、この設定は変更できます。

POPは、*MIME*(*Multipurpose Internet Mail Extensions*) などの重要なインターネットメッセージング標準にも互換性があり、これでメールの添付が可能になります。

POPは、電子メールの読み取りに使用するシステムが1台しかないユーザーに最適です。また、インターネットへの固定接続がない場合やメールサーバーを含むネットワークがない場合にも機能します。POPは認証した時点でクライアントプログラムに各メッセージの内容すべてをダウンロードするように要求しますので、遅いネットワークに接続しているユーザーにとっては大変です。これは、特にメッセージが大きいサイズの添付ファイルを持っている時には長い時間がかかります。

最新の標準バージョンのPOPプロトコルはPOP3です。

但し、使用頻度の低い、他のPOPプロトコルの変種は多種存在します :

- *APOP* — MDS 認証付きのPOP3です。暗号化なしでパスワードを送るのではなくユーザーパスワードの暗号化されたハッシュ(語群)が電子メールクライアントからサーバーに送ります。
- *KPOP* — Kerbero認証付きのPOP3です。この詳細については第17章を御覧下さい。
- *RPOP* — RPOP認証付きのPOP3です。これは、パスワードに似た、ユーザー毎のIDを使用してPOP要求を認証します。しかし、IDは暗号化されていないので、RPOPが通常のPOPより安全であることはありません。

追加のセキュリティとして、クライアント認証とデータ転送のセッション用に*SSL*(*Secure Socket Layer*) 暗号化を使用することも出来ます。これは、`ipop3s`サービス、又は、`/usr/sbin/stunnel`プログラムを使用して有効にすることができます。その詳細は項11.5.1で御覧下さい。

11.1.2.2. IMAP

Red Hat LinuxのデフォルトIMAPサーバーは、`/usr/sbin/imapd`であり、これは`imap`パッケージで用意されています。IMAPメールパッケージを使用すると電子メールのメッセージはサーバーに残りますので、ユーザーはそこで読み取ったり削除したりすることが出来ます。IMAPにより、クライアントアプリケーションはサーバー上のメールディレクトリを作成、名前変更、あるいは削除して電子メールの編成や保存ができます。

IMAPは、複数のマシンを使用して電子メールにアクセスするユーザーに特に便利です。このプロトコルは、また遅い回線経由でメールサーバーに接続しているユーザーにも便利です。それは電子メール

のヘッダ(頭書き)だけがメッセージの代理でダウンロードされますので、それを開くまでは回線も節約できるからです。ユーザーはさらにメッセージを表示あるいはダウンロードせずに削除することも出来ます。

また、便利なようにIMAPアプリケーションは、メッセージのコピーをローカルにキャッシュすることが可能で、これによりIMAPサーバーに直接接続されていない時も保存しているメッセージを閲覧することができます。

IMAPは、POPと同様にMIMEなどの重要なインターネットメッセージング基準に互換性をもつため、電子メールの添付も可能です。

セキュリティの補強の為に、クライアント認証とデータ転送セッションの為の*SSL*暗号法を使用することができます。これは`imaps`サービス、又は`/usr/sbin/stunnel`プログラムの使用をすることで有効にすることが出来ます。詳細については項11.5.1を参照して下さい。

他にもフリータイプと商用タイプのIMAPクライアントとサーバーが利用できます。その殆どはIMAPプロトコルを拡張して、追加の機能を提供しています。総合的な一覧はオンラインで、以下のサイトで確認できます。http://www.imap.org/products/longlist.htm.

11.2. 電子メールプログラム分類

一般的に、全ての電子メールプログラムには3つの分類のうちのひとつに分けられます。それらはすべて電子メールメッセージの移動と管理のプロセスで特定の役割を果たします。大半のユーザーは、メッセージを送受信するための特定の電子メールプログラムしか意識しません。これらのタイプはそれぞれ、電子メールが正しい宛先に着信するかどうかを確認するために重要です。

11.2.1. Mail Transfer Agent

MTA(*Mail Transfer Agent*)はSMTPを使用してホスト間で電子メールを転送します。1つのメッセージが目的地まで移動する間に幾つかのMTAが関与することもあります。

マシン間のメッセージの配信はかなり単純なものに見えますが、特定のMTAがリモートホストに配信するためのメッセージを受け入れることができるか、あるいは受け入れなければならないかを決定するプロセス全体は非常に複雑です。また、スパムから問題が発生するため、特定のMTAを使用することは通常、MTA自体の設定あるいはMTAネットワークアドレスへのアクセス欠如のいずれかで制限されます。

最新の電子メールクライアントプログラムの多くは、メールを送信する時に、MTAとして動作します。しかし、この動作は実際のMTAの役目と混同しないで下さい。電子メールクライアントプログラムが電子メールを(MTAのように)発信できる唯一の理由はアプリケーションを実行しているホストが自分自身のMTAを所有していないからです。これは、特に非Unix ベースのオペレーティングシステム上の電子メールクライアントで明確です。しかし、これらのクライアントプログラムは、使用許可のあるMTAにのみ発信メッセージを送信するだけで、受信者の電子メールサーバーにメッセージを直接配達することはありません。

Red Hat Linuxが2つのMTA (Sendmail とPostfix)をインストールするため、電子メールクライアントプログラムは多くの場合、MTAとして動作する必要はありません。Red Hat LinuxにはFetchmailと言う特殊目的用のMTAも含まれています。

SendmailとFetchmailの詳細については、項11.3を参照して下さい。

11.2.2. Mail Delivery Agent

MDA(*Mail Delivery Agent*)は、MTAによって喚起され着信メールを正式なユーザーのメールボックスにファイルします。多くの場合、MDAが実際に`mail`又はProcmailのように*LDA*(*Local Delivery Agent*) (ローカル配送エージェント)となります。

電子メールクライアントによって読まれる場所まで配達するメッセージを扱うプログラムはどれもMDAと考えられます。この理由で、幾つかのMTA(SendmailやPostfix)は、それらが新規メールのメッセージをローカルユーザーのメールスプールファイルの追加する時、MDAの役目を果たすと言えます。一般的にMDA はシステム間でメッセージを配送しませんし、ユーザーインターフェイスも提供することはありません。MDAは電子メールクライアントアプリケーションがアクセスできるようにローカルマシン上のメッセージを分配したり分類したりします。

11.2.3. Mail User Agent

*MUA(Mail User Agent)*は電子メールクライアントアプリケーションと同義語です。MUAは少なくともユーザーが電子メールメッセージを読み書きできるようにするプログラムです。多くのMUAはPOPプロトコルやIMAPプロトコルを通じてメッセージを検索したり、メッセージを保存するためのメールボックスを設定したり、発信メッセージをMTAに送り付けたりすることが出来ます。

MUAには**Mozilla Mail**などのグラフィカルなものや、`mutt`や`pine`のようなテキストベースの非常に簡単なインターフェイスもあります。

11.3. Mail Transport Agents

Red Hat Linux には2つの主要なMTA、SendmailとPostfixが含まれています。Sendmailはデフォルトの MTAとして設定されていますが、デフォルト設定をMTAからPostfixに切替えることは簡単です。

ヒント

MTAのデフォルト設定をSendmailからPostfixに切替える方法に関する情報は*Red Hat Linux* カスタマイズガイド内にある*Mail Transport Agent (MTA)*の設定の章を御覧下さい。

Red Hat Linuxには、またFetchmailと呼ばれる特殊目的のMTAが含まれています。これは、電子メールをリモートMTAからローカルMTAへ配達するのに使用されます。

このセクションでは、SendmailとFetchmailの詳細に触れています。

11.3.1. Sendmail

Sendmailの基本目的は、他のMTAのようにホスト間の電子メールを、通常はSMTPプロトコルを使用して転送することです。しかし、Sendmailは高度な設定柔軟度を持つことから、使用されるプロトコルを含めてどのように電子メールを取り扱うかの側面ほとんどすべてを制御できます。このMTAが持つパワーと拡張性のため、多くのシステム管理者によってSendmailの使用が選択されています。

11.3.1.1. 目的と制限

重要なことは、Sendmailが出来ないことを考えるのでなく、Sendmailが何であるか、及び何ができるかを知ることです。複数の役割を果たすモノリシックアプリケーションの時代では、Sendmailが組織内で電子メールサーバーを実行するために必要な唯一のアプリケーションであると考えることがあり得ます。技術的には事実で、、Sendmailがメールをユーザーのディレクトリにスプールし、コマンドラインを通じて新しい電子メールをユーザーの為に発信することができます。しかしほとんどのユーザーは実際に簡単なメールの配達だけを求めているのではありません。通常、ユーザーはPOPかIMAPを利用するMUAを使ってローカルマシンにメッセージをダウンロードして電子メールによる交流を望んでいます。または、メールボックスにアクセスする為にWebインターフェイスを好む場合もあるでしょう。これらの他のアプリケーションはSendmailとの併用で動作することができますが実際には別の理由で存在し、当然独立して稼働することが出来ます。

Sendmailがすべき、又は出来る設定の全てを言及することはこのセクションの担当範囲を越えてしまいます。文字通りに数百の異なるオプションと規則のセットがある中で、このマニュアル全項目では実行できるすべてと、物事がうまく行かない時の修正法を説明することに従事しています。Sendmailのリソースに関する情報は項11.6でお読み下さい。

このセクションでは、デフォルトでSendmailと共にインストールされているファイルの説明をして、さらに迷惑メール(spam)停止の仕方及び(LDAP)*Lightweight Directory Access Protocol* を使ったSendmailの拡張法などの基本的設定変更を説明していきます。

11.3.1.2. Sendmailのデフォルトインストール

Sendmailの実行ファイルは`/usr/sbin/sendmail`です。

Sendmailの長くて詳細に渡る設定ファイルは`/etc/mail/sendmail.cf`です。直接`sendmail.cf`を編集することは避けて下さい。Sendmailに対し設定の変更をするには、`/etc/mail/sendmail.mc`ファイルを代わりに編集します。オリジナルの`/etc/mail/sendmail.cf`をバックアップして、その後`m4`マクロプロセッサーを使用して、新しいバージョンの`/etc/mail/sendmail.cf`を作成します。Sendmailの設定に関する詳細は項11.3.1.3で御覧下さい。

さまざまなSendmail設定ファイルは、次のような`/etc/mail/`ディレクトリにインストールされます。

- `access` — 発信電子メール用のSendmailを使用するシステムを指定します。
- `domaintable` — ドメイン名マッピングを指定します。
- `local-host-names` — ホストのエイリアスを指定します。
- `mailertable` — 特定ドメインのルーティングを無効にする命令を指定します。
- `virtusertable` — エイリアスのドメイン特有の形式を指定します。これにより、複数の仮想ドメインが1つのマシン上でホストされます。

`access`、`domaintable`、`mailertable`、`virtusertable`などの`/etc/mail/`内の設定ファイルのいくつかは、実際に、設定変更を行う前にデータベースファイルに情報を保存する必要があります。データベースファイルの設定の中にそのような変更を含めるためには、次のコマンドを実行します：

```
makemap hash /etc/mail/<name> < /etc/mail/<name>
```

ここで、*<name>*は、変換する設定ファイルの名前で入れ換えます。

例えば、`example.com`ドメインに宛てられた全てのメールを`<bob@other-example.com>`に転送してもらう場合、次の行を`virtusertable`ファイルに追加します：

```
@example.com  bob@other-example.com
```

この変更を完結するには、次のコマンドをrootとして使用して`virtusertable.db`ファイルを更新する必要があります：

```
makemap hash /etc/mail/virtusertable < /etc/mail/virtusertable
```

これにより、新しい設定を持つ新規の`virtusertable.db`ファイルが作成できます。

11.3.1.3. 一般的なSendmail設定変更

Sendmailsの設定ファイルを変更する時は、既存のファイルを編集するのではなく、全く新しい`/etc/mail/sendmail.cf`ファイルを生成することが推奨されます。

重要

sendmail.cfファイルを変更する前に、そのファイルの作業用のバックアップを作成するのが良いでしょう。

Sendmailに必要な機能を追加するには、/etc/mail/sendmail.mcファイルを編集します。編集が終ったら、m4/etc/mail/sendmail.mc > /etc/sendmail.cfコマンドを実行することにより、m4マクロプロセッサを使って新しいsendmail.cfを生成します。新しい/etc/sendmail.cfを作成した後、Sendmailを再起動して有効にする必要があります。これを行う最も簡単な方法として、ルートとして/sbin/service sendmail restartコマンドを入力して下さい。

デフォルトで、Sendmailにm4マクロプロセッサーがインストールされています。m4パッケージの一部となっています。

重要

デフォルトのsendmail.cfは、ローカルコンピュータ以外のいかなるホストからも、ネットワーク接続を受け入れることをSendmailに許可しません。他のクライアントのサーバーとしてSendmailを設定したい場合は、/etc/mail/sendmail.mcを編集し、ネットワークデバイスをリッスンするようにDAEMON_OPTIONSを変更するか、このオプション全体をコメントアウトして下さい。その後、以下を実行して/etc/mail/sendmail.cfを再生成して下さい：

```
m4 /etc/mail/sendmail.mc > /etc/mail/sendmail.cf
```

この設定は、大半のSMTP専用サイトには有効であるはずです。UUCP(UNIX to UNIX Copy)サイトについては無効です。UUCPメール転送を使用したい場合は、新しいsendmail.cfを生成する必要があります。

/usr/share/sendmail-cfディレクトリの下のディレクトリ内にあるファイルを編集する前に/usr/share/sendmail-cf/README ファイルを参照してください。ディレクトリ内のファイルが将来の/etc/mail/sendmail.cf ファイルの設定に影響を与える可能性があります。

11.3.1.4. マスカレード

一般的なSendmailの設定では、ネットワーク上にあるすべてのマシンのためのメールゲートウェイとしての役割を1つのマシンに果たさせます。たとえば、ある会社はすべての電子メールを処理し、発信メールに返信用アドレスを添付するmail.bigcorp.comと呼ぶマシンを持ちたいと想定しましょう。

この様な状況では、Sendmailサーバーは会社のネットワーク上のマシンをマスカレードしてその返信用アドレスが、user@devel.bigcorp.com.ではなく、user@bigcorp.comとなるようにする必要があります。

そうするには、以下の行を/etc/mail/sendmail.mcに追加します：

```
FEATURE(always_add_domain)dnl
FEATURE(‘masquerade_entire_domain’)
FEATURE(‘masquerade_envelope’)
FEATURE(‘allmasquerade’)
MASQUERADE_AS(‘bigcorp.com.’)
MASQUERADE_DOMAIN(‘bigcorp.com.’)
MASQUERADE_AS(bigcorp.com)
```

m4を使用して新しいsendmail.cfを生成した後はこの設定が、このネットワーク内部からのメールがすべてbigcorp.comから送信されたように見えます。

11.3.1.5. スパムの停止

電子メールスパムとは、通信を要求していないユーザーが受け取る、不要で欲しくもない電子メールと定義出来ます。それは、非常に破壊的でコストのかかる広範囲なインターネット通信標準の乱用です。

Sendmailは、ジャンクメールを送信するための新しいスパミング手法が採用されないように阻止することを比較的容易にしました。デフォルトでさらに一般的なスパミング手法の多くを阻止します。

たとえば、中継とも呼ばれているSMTPメッセージの転送は、Sendmailバージョン8.9以降にデフォルトで無効にされました。この変更が行われる前であれば、Sendmailはある部署(`y.com`)からメッセージを受け入れて別の部署(`z.net`)に送るようにメールホスト(`x.org`)に指示できました。しかし、現在では、こちらのサーバーを通じてメールを中継することをドメインに許可するようにSendmailに具体的に設定する必要があります。ドメインへの中継を設定するには、`/etc/mail/relay-domains`ファイルを編集し、Sendmailを再起動します。

しかし、多くの場合、ユーザーはインターネットを通じて制御できないような他のサーバーからのスパムの砲撃を受けるおそれがあります。そのような場合、`/etc/mail/access`ファイルから提供されているSendmailのアクセス制御機能を使用して、歓迎できないホストからの接続を防止出来ます。次の例では、ファイルが阻止、及びSendmailサーバーへのアクセスの許可との両方に使用されています：

```
badspammer.com    ERROR:550 "Go away and do not spam us anymore"
tux.badspammer.com  OK
10.0        RELAY
```

この例は、`badspammer.com`から送られたすべての電子メールが、スパマーに戻されるメッセージ付きで550 RFC-821対応エラーコードによりブロックされることを示しています。ただし、受け入れられると思われる`tux.badspammer.com`サブドメインから送られた電子メールは受理されます。最後の行は、10.0.*.*ネットワークから送られたすべての電子メールがメールサーバーを通じて中継できることを示します。

`/etc/mail/access.db`はデータベースであるため、変更をするには`makemap`を起動します。これには、rootで次のコマンドを使用します：

```
makemap hash /etc/mail/access < /etc/mail/access
```

アクセスの許可とその阻止に関して、この例はSendmailが出来ることのほんの一部しか示していません。詳細と他の例については`/usr/share/doc/sendmail/README.cf`を御覧下さい。

Sendmailは、メールを配送する時に、Procmail MDAをコールしますのでSpamAssassinなどのスパムフィルタを使用してユーザーはスパムを認識し、ファイルすることも出来ます。SpamAssassinの使用については項11.4.2.6を参照して下さい。

11.3.1.6. LDAPでのSendmailの使用

LDAP(*Lightweight Directory Access Protocol*)を使用すると非常に大きいグループから特定のユーザーに関する特定情報を非常に高速かつ強力に検索できます。たとえば、LDAPサーバーを使用して、ユーザーのラストネームで一般的な法人ディレクトリから特定の電子メールアドレスを調べることができます。このような実践形態では、LDAPはSendmailと大きく異なり、LDAPは階層的なユーザー情報を保存し、Sendmailにはすでにアドレス指定された電子メールメッセージ内のLDAPクエリの結果が与えられるだけです。

しかし、SendmailはLDAPとの非常に大きな統合化をサポートします。この場合、SendmailはLDAPを使用して、中型から企業レベルの組織をサポートするために協調動作する各種メールサーバー上で`aliases`や`virtusertables`などの個別に保守されるファイルを置き換えます。簡単に言えば、Sendmailとその個別の設定ファイルから、多数の異なるアプリケーションでサポートされている強力なLDAPクラスタへ、メールルーティングレベルを抽出することができます。

Sendmailの現在のバージョンには、LDAPに対するサポートが含まれています。LDAPを使用してSendmailサーバーを拡張するには、まず、**OpenLDAP** などのLDAPサーバーを動作させ正しく設定します。その後、次をインクルードできるように、/etc/mail/sendmail.mcを編集します：

```
LDAPROUTE_DOMAIN('yourdomain.com')dnl
FEATURE('ldap_routing')dnl
```

注意

これは、LDAPによるSendmailの非常に基本的な設定のためだけのものです。特に共通のLDAPサーバーを使用するように数台のSendmailマシンを設定する場合、LDAPの実装の仕方によってはユーザーの設定はこの基本的な設定とは非常に異なるはずです。

詳細なLDAPルーティング設定の手順と例については、/usr/share/doc/sendmail/README.cfを参照してください。

次に、m4を実行してSendmailを再起動することによって、/etc/mail/sendmail.cfファイルを作成しなおします。これを行う手順については、項11.3.1.3 を参照して下さい。

LDAPの詳細については、第13章を参照してください。

11.3.2. Fetchmail

Fetchmailは、リモートサーバーから電子メールを呼び込んでローカルのMTAへそれを配送します。多くのユーザーがメッセージをリモートサーバーよりダウンロードするプロセスを、MUAの中で電子メールの読み込みと編成から分離できる能力を評価しています。ダイヤルアップをするユーザーのニーズを考慮してデザインされており、FetchmailはPOP3やIMAPなどのプロトコルを使用して接続して、電子メールスプールファイルにすべての電子メールメッセージを高速にダウンロードします。Fetchmailは、必要に応じて、SMTPサーバーに電子メールメッセージを転送することもできます。

Fetchmailは、ユーザーのホームディレクトリ内の.fetchmailrcファイルを使用してユーザーごとに設定されます。

Fetchmailは.fetchmailrcファイルの設定内容を使用して、リモートサーバー上の電子メールの有無をチェックして抜き出し、電子メールを正しいユーザーのスプールファイルに配置するためにローカルMTAを使用して電子メールをローカルマシンのポート25に配信しようとします。Procmailが使用できる場合は、それを使用して電子メールをフィルタ処理し、MUAで読み取れるようにメールボックスに設定します。

11.3.2.1. Fetchmail設定オプション

Fetchmailを実行するときにリモートサーバー上の電子メールの有無をチェックするのに必要なコマンドライン上のすべてのオプションをパスすることはできますが、.fetchmailrcファイルを使用したほうがはるかに簡単です。すべての設定オプションは.fetchmailrcファイル内にありますが、コマンドラインでそのオプションを指定してFetchmailを実行するときにそのオプションを上書きすることができます。

ユーザーの.fetchmailrcファイルは、3つの特定タイプの設定オプションに分けられます。

- グローバルオプション —プログラムの動作を制御したり、電子メールの有無をチェックするすべての接続に設定を与えるための手順をFetchmailに示します。

- サーバーオプション — ポーリングされるサーバーに関するホスト名などの必要な情報を指定したり、特定の電子メールサーバーで設定されている、チェックするポートやタイムアウトまで待つ秒数などの表示したい個人設定を指定したりします。これらのオプションは、そのサーバーで使用されるすべてのユーザーオプションに影響を与えます。
- ユーザーオプション —特定の電子メールサーバーを使用して電子メールの認証や有無のチェックを行うのに必要なユーザー名やパスワードなどの情報が含まれています。

グローバルオプションは.fetchmailrcファイルの一番上にあり、その後に1つ又は複数のサーバーオプションがあり、各サーバーオプションはFetchmailがチェックしなければならない異なる電子メールサーバーを指定します。その電子メールサーバー上でチェックしたいユーザーアカウントごとに、サーバーオプションの後にユーザーオプションがあります。サーバーオプションと同様に、特定のサーバー上にある複数の電子メールアカウントをチェックしたいときなど、そのサーバーで使用する複数のユーザーオプションを指定できます。

サーバーオプションは、サーバー情報の前にある特別なオプションの動詞、すなわち、pollやskipを使用して.fetchmailrcファイル内のサービスに呼び出されます。pollアクションはこのサーバーオプションの実行時にそのサーバーオプションを使用するようにFetchmailに指示し、そのサーバーオプションは実際に各種のユーザーオプションを使用して電子メールの有無をチェックします。しかし、Fetchmailを呼び出すときにこのサーバーのホスト名を指定しないと、skipアクションの後のサーバーオプションはチェックされません。skipオプションを使用すると、.fetchmailrc内にテスト設定を設定し、現在機能している設定に影響を与えずに特に必要なときだけそのサーバーを使用してチェックします。

.fetchmailrcファイルのサンプルは、次のように表示されます。

```
set postmaster "user1"
set bouncemail

poll pop.domain.com proto pop3
  user 'user1' there with password 'secret' is user1 here

poll mail.domain2.com
  user 'user5' there with password 'secret2' is user1 here
  user 'user7' there with password 'secret3' is user1 here
```

この例では、グローバルはオプションセットなので、最終手段としてユーザーに電子メールが送られ(postmasterオプション)、すべての電子メールエラーは、送信者ではなく、ポストマスターに送られます(bouncemailオプション)。setアクションは、この行にグローバルオプションが含まれていることをFetchmailに伝えます。その後、2つのメールサーバーが指定され、その1つはPOP3を使用してチェックするようにセットされ、もう1つは実際に機能するものを検索するために各種のプロトコルを試行するようにセットされます。2つ目のサーバーオプションを使用して2人のユーザーがチェックされますが、ユーザーのいずれかの為のメールはすべてユーザー1のメールスプールに送られます。これにより、複数のサーバー上で複数のメールボックスがチェックできるようになり、1つのMUAボックスに表示されます。各ユーザー特有の情報は、userアクションで始まります。

注意

.fetchmailrcファイルにパスワードを設定する必要はありません。with password '*<password>*'を省略すると、Fetchmailが起動された時にパスワードを要求するようになります。

Fetchmailには、多くの異なるグローバル、サーバー、及びローカルオプションが含まれています。これらのオプションの多くは、稀にしか使用されないか、又は非常に特別な場合のみの使用となります。fetchmailのmanページでは、各オプションを詳細に説明していますが、殆どの一般的なものはここでリストしてあります。

11.3.2.2. グローバルオプション

各グローバルオプションは、setアクションの後の1行に設定してください。

- daemon <*seconds*> — Fetchmailがバックグランドにいて、指定した間隔でメールを取り込むデーモンを指定します。
- postmaster — 配達の問題がある場合、メールを送るためのローカルユーザーを指定します。
- syslog — エラーとステータスメッセージ用のログファイルを指定します。デフォルトでこれは、/var/log/maillogです。

11.3.2.3. サーバーオプション

pollかskipのいずれかのアクションの後にサーバーオプションを.fetchmailrc内のそれらの独自の行に設定します。

- auth <*auth-type*> — 使用する認証のタイプを指定します。デフォルトでは、password認証が使用されますが、プロトコルによってはkerberos_v5、kerberos_v4、sshなどの他のタイプの認証もサポートするものがあります。any 認証タイプを使用すると、Fetchmailはまず、パスワードを必要としない方法を試し、次にパスワードをマスクする方法を試し、最後にサーバーに対して認証するためのパスワードを平文でサーバーに送ろうとします。
- interval <*number*> — すべての設定済みサーバー上で電子メールの有無をチェックする<*number*>回ごとに指定したサーバーをポーリングします。このオプションは通常、めったにメッセージを受信しない電子メールサーバーで使用します。
- port <*port-number*> —指定されたプロトコルのデフォルトポート番号を無効にします。
- proto <*protocol*> — pop3やimapなどの特定のプロトコルを使用してこのサーバー上でメッセージの有無をチェックするようにFetchmailに指示します。
- timeout <*seconds*> — Fetchmailが接続試行を諦めるまでのサーバーの不活動期間を指定します。この値が指定されていない場合、デフォルトの300秒が採用されます。

11.3.2.4. ユーザーオプション

ユーザーオプションは、サーバーオプションの下の独自の行か、サーバーオプションと同じ行に設定できます。いずれの場合も、定義したオプションはuserオプション（以下で定義している）の後に表示されます。

- fetchall —すでに表示されているメッセージを含むキュー内のすべてのメッセージをダウンロードするようにFetchmailに指示します。デフォルトでは、Fetchmailは新しいメッセージだけをプルダウンします。
- fetchlimit <*number*> —停止する前にある個数のメッセージだけを回収できます。
- flush —新しいメッセージを回収する前にキュー内のすでに表示されているすべてのメッセージを削除するようにFetchmailに指示します。
- limit <*max-number-bytes*> — 特定サイズ以下のメッセージだけを検索できることを指定できます。このオプションは、大きいメッセージをダウンロードするのに時間がかかりすぎるときの低速ネットワークリンクで便利です。
- password '<*password*>' —このユーザーに使用するパスワードを指定します。
- preconnect "<*command*>" — このユーザーに対するメッセージを検索する前に指定されたコマンドを実行するようにFetchmailに指示します。
- postconnect "<*command*>" — このユーザーに対するメッセージを検索した後に指定されたコマンドを実行するようにFetchmailに指示します。

- `ssl` — SSL暗号化を有効にします。
- `user "<username>"` — メッセージを検索するためにFetchmailで使用するユーザー名を設定します。このオプションは、ほかのユーザーオプションの前に表示する必要があります。

11.3.2.5. Fetchmailコマンドオプション

コマンドラインで使用できるFetchmailオプションの大半は、`fetchmail`コマンドを実行するときに、`.fetchmailrc`設定オプションをミラー化します。このミラー化が行われるのは、設定ファイルがあってもなくてもFetchmailを使用できるようにするためです。大半のユーザーは、コマンドラインでこれらのオプションを使用しません。それは、Fetchmailを実行するたびに使用される`.fetchmailrc`ファイルにこれらのオプションを残すほうが簡単であるからです。

しかし、特定目的のために他のオプションを付けて`fetchmail`コマンドを実行したい場合があります。コマンドラインで指定されたすべてのオプションは設定ファイルオプションを無効にするので、コマンドオプションを発行して、エラーを発生させている`.fetchmailrc`の設定を一時的に無効にすることもできます。

11.3.2.6. 情報オプション、あるいはデバッグオプション

`fetchmail`コマンドの後に使用されるある種のオプションは、重要な情報を与える可能性があります。

- `--configdump` — `.fetchmailrc`とFetchmailのデフォルトからの情報に基づいてすべての可能なオプションを表示します。このオプションを使用すると、ユーザーに対する電子メールは検索されません。
- `-s` — Fetchmailをサイレントモードで実行し、エラー以外のメッセージが`fetchmail`コマンドの後に表示されないようにします。
- `-v` — Fetchmailを冗長モードで実行し、Fetchmailとリモート電子メールサーバーの間のすべての通信を表示します。
- `-V` — このオプションを選択すると、Fetchmailは詳細バージョン情報を表示し、そのグローバルオプションを一覧し、電子メールプロトコルや認証方法などの各ユーザーで使用される設定を示します。このオプションを使用すると、ユーザーに対する電子メールは検索されません。

11.3.2.7. 特別なオプション

これらのオプションは、`.fetchmailrc`ファイルでよく見られるデフォルトを無効にする場合に便利です。

- `-a` — リモート電子メールサーバーからすべてのメッセージを（新しいメッセージかすでに表示されたメッセージかに関係なく）ダウンロードするようにFetchmailに指示します。デフォルトでは、Fetchmailは、新しいメッセージだけをダウンロードします。
- `-k` —このオプションを選択すると、Fetchmailはメッセージをダウンロードした後にリモート電子メールサーバー上にメッセージを残します。このオプションは、メッセージをダウンロードした後にメッセージを削除するデフォルト動作を無効にします。
- `-l <max-number-bytes>` — 特定サイズ以上のすべてのメッセージをダウンロードせず、リモート電子メールサーバー上にメッセージを残すようにFetchmailに指示します。
- `--quit` — Fetchmailデーモンプロセスを終了します。

`fetchmail`マニュアルページには、以上のコマンド以外のコマンドや`.fetchmailrc`オプションが表示されています。

11.4. Mail Delivery Agents

Red Hat Linux には2種類のMDA (Procmail と`mail`)が含まれています。これらのアプリケーションは両方とも、Local Delivery Agent と考えられ、その両方が電子メールをMTAのスプールファイルからユーザーのメールボックスへ転送します。しかし、Procmailはさらに強健なフィルターシステムです。

このセクションは、Procmailのみについて詳細を説明します。`mail`コマンドに付いての情報は、そのmanページを御覧下さい。

Procmailを使用すると、ローカルホストのメールスプールファイルにある電子メールのフィルターと配送をします。Procmailは強力で、システムリソースにやさしく、広範囲で使用されています。これは、電子メールクライアントアプリケーションで読み込まれる予定の電子メールを配送する時点で重要な役割りを果たします。

Procmailは幾つかの方法で喚起されます。MTAが電子メールをメールスプールファイルに配置するとProcmailが起動します。その後Procmail はMUAが検索できるように、電子メールをフィルタし、ファイルします。別の方法では、メッセージが受信された時にProcmailを実行するように、MUAを設定してメッセージが正しいメールボックスに移動されるようにします。デフォルトでは、ユーザーのホームディレクトリ内の`.procmailrc`ファイルの存在は、MTAが新しいメッセージを受信する度にProcmailを喚起します。

Procmailが電子メールでとるアクションは、メッセージがプログラムで照合されるという特定のレシピ、すなわち規則、からの指示により異なります。メッセージがレシピに一致すると、電子メールはある種のファイル内に設定されるか、削除されるか、またはそれ以外の方法で処理されます。

Procmailが起動すると、電子メールメッセージを読み取り、ヘッダー情報から本体を分離します。次に、Procmailはデフォルトのシステム全体のProcmail環境変数とレシピ用の`/etc/procmailrcs`ディレクトリ内の`/etc/procmailrc`ファイルと`rc`ファイルを探します。次に、Procmailはユーザーのホームディレクトリ内の`.procmailrc`ファイルを探し、そのユーザーに固有の規則を見つけます。多くのユーザーは、`.procmailrc`で参照されるProcmail用の独自の追加`rc`ファイルも作成します。ただし、これらのファイルはメールフィルタ処理の問題が発生した場合にただちにオン/オフすることができます。

デフォルトでは、システム全体の`rc`ファイルが`/etc`ディレクトリに存在せず、ユーザー`.procmailrc`ファイルも存在しません。Procmailの使用を開始するには、特定の環境変数と、特定のメッセージタイプ用の規則をもって、`.procmailrc`ファイルを作成する必要があります。

ほとんどの設定では、Procmailが起動して電子メールのフィルタ処理を試行するかどうかの決定は、ユーザーの`.procmailrc`ファイルの有無に基づきます。Procmailを無効にして、`.procmailrc`ファイルに作業内容は保存したい場合、`mv ~/.procmailrc ~/.procmailrcSAVE`などのコマンドで類似するファイルの名前にそのファイルを移動します。Procmailのテストを再開できる状態のときは、ファイル名を`.procmailrc`に戻します。ただちに、Procmailが再度機能し始めます。

11.4.1. Procmailの設定

Procmail設定ファイル、特にユーザーの`.procmailrc`には、ある種の重要な環境変数が含まれています。これらの変数は、どのメッセージをソートするか、レシピに一致しないメッセージをどう処理するかなどをProcmailに指示します。

これらの環境変数は通常、`.procmailrc`の先頭に表示されます。以下のような形式になります：

<env-variable>="*<value>*"

この例では、*<env-variable>*は変数の名前であり、*<value>*は、変数を定義します。

大半の環境変数はほとんどのProcmailユーザーに使用されず、それより重要な環境変数の多くはすでにデフォルト値が定義されています。ほとんどの場合、次のような変数が使用されます：

- `DEFAULT` —レシピに一致しないメッセージが設定されるデフォルトのメールボックスを設定します。

デフォルトのDEFAULT値は、$ORGMAILと同じです。

- INCLUDERC — 再度チェックするメッセージのためのレシピをさらに含む追加rcファイルを指定します。このため、スパムのブロッキングや電子メールリストの管理などのさまざまな役割を満たす個々のファイルにProcmailレシピを分離し、次にユーザーの.procmailrcファイル内のコメント文字を使用してそれらの役割をオン/オフに切り替えることができます。

  例えば、ユーザーの.procmailrcファイルの行は次のようになります：

```
MAILDIR=$HOME/Msgs
INCLUDERC=$MAILDIR/lists.rc
INCLUDERC=$MAILDIR/spam.rc
```

  ユーザーが電子メールリストのProcmailフィルタ処理をオフに切り替えて、スパム制御は所定の位置に残したい場合、#文字で最初のINCLUDERC行を簡単にコメント化することができます。

- LOCKSLEEP — Procmailが特定のロックファイルを使おうとする試行と試行の間の時間数を秒単位で設定します。デフォルトは8秒です。
- LOCKTIMEOUT — ロックファイルが古くて削除できるとProcmailが判断するまでに、ロックファイルが最後に変更されてから経過しなければならない時間数を秒単位で設定します。デフォルトは1,024秒です。
- LOGFILE — Procmailの情報メッセージかエラーメッセージのいずれかを含むロケーションとファイル。
- MAILDIR — Procmailにカレント作業ディレクトリを設定します。設定すると、他のProcmailパスはすべてこのディレクトリを基準にします。
- ORGMAIL —オリジナルメールボックスを指定するか、メッセージをデフォルトロケーションかレシピが要求するロケーションに設定できない場合にメッセージを設定する別の場所を指定します。

  デフォルトでは、/var/spool/mail/$LOGNAMEの値が使用されます。
- SUSPEND —スワップスペースなどの必要なリソースがない場合にProcmailが休止する時間数を秒単位で設定します。
- SWITCHRC —このオプションを使用すると、追加のProcmailレシピを含む外部ファイルを指定できます。レシピチェックが実際に参照用設定ファイル上で停止し、SWITCHRCで指定されたファイル上のレシピだけが使用されること以外、INCLUDERCオプションと非常に似ています。
- VERBOSE — このオプションの使用で、Procmailは非常に多くの詳細情報をログにとることができます。このオプションはデバッグに便利です。

他の重要な環境変数は、ログイン名であるLOGNAME、ホームディレクトリのロケーションであるHOME、デフォルトシェルであるSHELLなどのシェルから抜き出されます。

すべての環境変数とそれらのデフォルト値に関する総合的な説明は、procmailrcmanページで御覧下さい。

11.4.2. Procmailレシピ

新規ユーザーには、多くの場合、レシピの作成がProcmailを使用するための最も困難な学習領域と思われるかも知れません。ある程度理解できるものです。それは、レシピが照合用文字列に修飾を指定するための特別なフォーマットである正規表現を使用してメッセージの照合を行うからです。ただし、正規表現はそれほど設定しにくかったり、読み取るときに理解しにくかったりするものではありません。また、Procmailレシピを書く方法の一貫性は、正規表現とは無関係に、何が行われているかを検出することを容易にします。

正規表現についての完全な説明は、このセクションの担当範囲を越えています。Procmailレシピの構成は、より重要でありサンプルのProcmailレシピがインターネットのさまざまなサイトで参照できます。(例えば：http://www.iki.fi/era/procmail/links.html) これらのレシピの例に中で見られる正規表現の正しい使用とその応用は、Procmailレシピの構成の理解のレベルにより左右されます。基本的な正規表現規則の入門情報は、grepのmanページで参照することが出来ます。

Procmailレシピは、次の形式をとります。

```
:0<flags>: <lockfile-name>

* <special-condition-character> <condition-1>
* <special-condition-character> <condition-2>
* <special-condition-character> <condition-N>

<special-action-character><action-to-perform>
```

Procmailレシピの最初の2文字はコロン（:）と0です。このレシピを処理するときにProcmailが何をするかを制御するには、オプションとして0の後に各種フラグを設定できます。<*flags*> セクションの後のコロンは、このメッセージのためのロックファイルを作成することを指定します。ロックファイルを作成する場合は、<*lockfile-name*>スペースでその名前を指定します。

レシピには、メッセージと照合するいくつかの条件を含めることができます。条件がない場合、すべてのメッセージはレシピに一致します。正規表現にはメッセージとの照合を実施するために、いくつかの条件が設定されます。複数の条件を使用する場合、アクションを実行するためにこれらの条件がすべて一致しなければなりません。条件は、レシピの最初の行で設定されたフラグに基づいてチェックされます。*文字の後に設定されたオプションの特別な文字は、さらに条件を制御できます。

<*action-to-perform*>は、条件のうちの1つに一致する場合にメッセージに対して何が発生するかを指定します。レシピごとにアクションは1つしか指定できません。多くの場合、ここではメールボックスの名前を使用して照合用のメッセージをそのファイルに送り、電子メールを有効にソートします。アクションを指定する前にも、特別なアクション文字を使用できます。

11.4.2.1. 配信レシピと非配信

レシピが特定メッセージに一致する場合に使用されるアクションは、レシピが配信と非配信のいずれかであるかを決定します。配信レシピには、ファイルにメッセージを書き込んだり、別のプログラムにメッセージを送ったり、別の電子メールアドレスにメッセージを転送したりするアクションが含まれています。非配信レシピは、ネスト用ブロックを使用するときなどの他のアクションをカバーします。ネスト用ブロックは、レシピの条件に一致する追加アクションを指定する中かっこ{ }に囲まれたアクションです。ネスト用ブロックをネストして、メッセージ上のアクションを識別して実行するためのさらに大きな制御を与えることができます。

メッセージを照合する配信レシピを使用すると、Procmailは指定されたアクションを実行し、他のレシピに対するメッセージの比較を停止します。非配信レシピで条件に一致するメッセージは、現行と以降のrcファイル内の他のレシピと比較され続けます。つまり、非配信レシピを使用すると、メッセージに対して指定されたアクションがとられた後にメッセージがレシピ内を通り続けます。

11.4.2.2. フラグ

フラグは、どのようにレシピの条件をメッセージと比較するか、あるいはレシピの条件をメッセージと比較するかどうかを決定する際に非常に重要です。次のフラグが一般に使用されます。

- A — このレシピがAフラグやaフラグのない以前のレシピもこのメッセージに一致した場合のみ使用されることを指定します。

  現行のレシピでの照合を行う前に、この直前の照合用レシピでのアクションが正常終了したかどうかを確認するには、その代わりにaフラグを使用します。
- B — メッセージの本文を構文解析し、一致する条件を探します。
- b — メッセージをファイルに書き込んだり転送したりするなどの結果的なアクションで本文を使用します。これはデフォルトの動作です。

- c — 電子メールのカーボンコピーを作成します。これが配信レシピで便利なのは、必要なアクションをメッセージ上で実行でき、メッセージのコピーをrc ファイルで処理し続けることができるからです。
- D — egrepの比較を大文字と小文字を区別するものにします。デフォルトでは、比較プロセスは大文字と小文字を区別しません。
- E — Aフラグと似ていますが、Eフラグのない直前のレシピが一致しなかった場合にこのレシピ内の条件がメッセージと比較されます。これは、*else*アクションに匹敵します。

  直前のレシピは一致したがアクションが失敗したときだけこのレシピをチェックしたい場合は、eフラグを使用します。
- f — パイプをフィルタとして使用します。
- H — メッセージのヘッダーを構文解析し、一致する条件を探します。これは、デフォルトで行われます。
- h — 結果のアクションでヘッダーを使用します。これはデフォルトの動作です。
- w — 指定されたフィルタか、プログラムが終了するまで待機し、フィルタ処理されたメッセージを考慮する前にそのフィルタか、プログラムが成功したかどうかを報告するようにProcmailに指示します。

  フィルタかアクションが成功したかどうかを判断するときに"Program failure"メッセージを無視したいときは、その代わりにWオプションを使用します。

追加フラグは、procmailrcマニュアルページで見つけることができます。

11.4.2.3. ローカルロックファイルの指定

ロックファイルは、Procmailで複数のプロセスが同時に一定のメッセージを変更することを止めるのに非常に便利です。レシピの最初の行のフラグの後にコロン(:)を設定することによって、ローカルロックファイルを指定できます。このため、宛先のファイル名とLOCKEXT グローバル環境変数で設定されたすべてのものに基づいて、ローカルロックファイルが作成されます。

別の方法として、コロンの後にこのレシピで使用するローカルロックファイルの名前を指定します。

11.4.2.4. 特別な条件とアクション

Procmailレシピの条件とアクションの前に使用される特別な文字は、それらの条件とアクションを解釈する方法を変更します。

レシピの条件行の先頭にある*文字の後に、次の文字を使用できます。

- ! — 条件の行でこの文字は条件を反転するため、条件がメッセージと一致しない場合のみ照合が行われます。
- < — メッセージが指定数のバイトより少ないかどうかを確認します。
- > — メッセージが指定数のバイトより多いかどうかを確認します。

特別なアクションを実行するには、次の文字を使用します。

- ! — アクションの行では、指定された電子メールアドレスにメッセージを転送するようにProcmailに指示します。
- $ — rcファイル内で以前に設定された変数を参照します。これは通常、各種レシピで参照される一般的なメールボックスを設定する場合に使用します。
- | — パイプ文字は、特定のプログラムを起動してこのメッセージを処理するようにProcmailに指示します。

- { and } ── 追加レシピを組み込んで照合用メッセージに適用するためのネスト用ブロックを作成します。

アクション行で特別な文字を使用しない場合、Procmailではメッセージを書かなければならないメールボックスをそのアクション行が指定していると判定します。

11.4.2.5. レシピの例

Procmailは非常に柔軟性のあるプログラムなので、メッセージをとても具体的な条件と照合し、次にそれらのメッセージに対して詳細なアクションを実行できます。しかし、この柔軟性の結果、始めからProcmailレシピを作成してある種の目的を実現することは、新規ユーザーにとって困難である可能性があります。

Procmailレシピ条件を構築するためのスキルを開発する最善の方法は、他の人が構築した多数の例を見ることで組み合わされた正規式をよく理解することから生まれます。Procmailレシピの構成のデモンストレーションの役割を果たす次の非常に基本的な例があり、それはさらに複雑な構成の基礎を与えることができます。

基本的なレシピには、次の例で示すように条件が付いていないものさえあります：

```
:0:
new-mail.spool
```

第1行では、ローカルロックファイルを作成することを指定してレシピを開始しても、名前を指定しないので、Procmailは宛先のファイル名とLOCKEXTを使用してそれに名前を付けます。条件は指定されないので、すべてのメッセージはこのレシピに一致するため、MAILDIR環境変数で指定されたディレクトリ内にあるnew-mail.spoolと呼ぶ単一のスプールファイルに設定されます。この場合、MUAはこのファイル内のメッセージを見ることができます。

この基本的なレシピはすべてのrcファイルの終わりに配置して、メッセージをデフォルトロケーションに送ることができます。さらに複雑な例では、メッセージを特定の電子メールアドレスから取り出して、次の例のように破棄することであります。

```
:0
* ^From: spammer@domain.com
/dev/null
```

この例では、spammer@domain.comで送られたメッセージはすべて、ただちに/dev/nullに移動され、削除されます。

重要

規則と一致するメッセージを/dev/nullに移動する（永久削除）前に規則が正しく機能していることに注意してください。レシピ条件で不注意に意図していないメッセージを取り込んだ場合、それは跡形もなく消滅し、規則をトラブルシュートすることが困難になります。

改善案としてはレシピのアクションを特定のメールボックスにポイントして、それが、定期的に*false positives*、すなわち偶然に条件と一致したメッセージを探すためにチェックするようにします。偶然に一致したメッセージがないと確認できた後は、メールボックスを削除してアクションに対しメッセージを/dev/nullに送るように指示します。

Procmailは主に電子メールのフィルタとして使用され、電子メールが手動で保存されることを防止するために自動的に正しい場所に配置します。以下のレシピは特定のメーリングリストから送られた電子メールを取り込んでそれを正しいフォルダーに置きます。

```
:0:
* ^(From|CC|To).*tux-lug
```

```
tuxlug
```

tux-lug@domain.comメーリングリストから送られたメッセージはすべて、MUAのためにtuxlugメールボックスに自動的に配置されます。ここで、メールのFrom行、CC行、To行のどれかにメーリングリストの電子メールアドレスがあれば、この例の条件はメッセージに一致することに注意してください。

より強力なレシピをより詳細に調べるには項11.6で確認できる、多くのProcmailオンラインリソースを参照して下さい。

11.4.2.6. スパムフィルタ

新しい電子メールの受信時にSendmail、Postfix、Fetchmailによってコールされますので、Procmailはスパムと戦う強力なツールとして使用されます。

これは特にProcmailがSpamAssassinと併用される時に明確になります。一緒に使用するとこれらの2つのアプリケーションは素早くスパムを認識して、分類するか又は破壊します。

SpamAssassin はヘッダ解析、テキスト解析、ブラックリスト、スパム追跡データベースなどを使用し、スパムを識別してタグを付けます。

ローカルユーザーにとって最も簡単なSpamAssassinの使用法は、以下の行を~/.procmailrc fileの先頭近くに置くことです：

```
INCLUDERC=/etc/mail/spamassassin/spamassassin-default.rc
```

The /etc/mail/spamassassin/spamassassin-default.rc には、入信の電子メールすべての為にSpamAssassinを起動する簡単なProcmail 規則が含まれています。電子メールがスパムだと判定された場合、そのようにヘッダにタグが付けられ、タイトルは次のパターンでその前に付けられます：

```
*****SPAM*****
```

電子メールのメッセージ本文もまた、何が原因してスパムと診断されたのかという流動符号が前付けされます。

スパムとしてタグの付く電子メールをファイルするには、次の規則と良く似たものが使用されます：

```
:0 Hw
* ^X-Spam-Status: Yes
spam
```

この規則はヘッダ内でスパムとタグの付いた電子メールの全てをspam呼ばれるメールボックスにファイルします。

SpamAssassinはPerlスクリプトなので、負荷の大きいサーバー上ではバイナリSpamAssassinデーモン(spamd)およびクライアントアプリケーション(spamc)を使用する必要があるかも知れません。このようなSpamAssassinの設定にはルートでホストにアクセスしなければいけません。

spamdデーモンをスタートするには、ルートで以下のように入力します：

```
/sbin/service spamassassin start
```

システムが起動するときにSpamAssassinデーモンをスタートさせるには、**サービス設定ツール**(redhat-config-services)などのinitscriptユーティリティを使用してspamassassinサービスを始動します。initscriptユーティリティに関する情報は項1.4.2 で御覧になれます。

Perlスクリプトの代わりに、Procmailを設定してSpamAssassinクライアントアプリケーションを使用する場合、以下の行を~/.procmailrcファイルの上部に置くか、又はシステム全体の設定には、その行を/etc/procmailrcの中に入れます：

```
INCLUDERC=/etc/mail/spamassassin/spamassassin-spamc.rc
```

11.5. Mail User Agents

Red Hat Linuxには、数十のメールプログラムが揃っています。その中には**Mozilla Mail**や**Ximian Evolution**などの機能満載でグラフィカル電子メールクライアントプログラムと、さらには**mutt**や**pine**のようなテキストベースの電子メールプログラムも含まれています。

これらのアプリケーションの使用法に関しては、*Red Hat Linux* 入門ガイドの中にある*E*メールアプリケーションの章を参照して下さい。

このセクションの後の部分では、クライアントとサーバー間の通信のセキュリティに焦点をおいて説明していきます。

11.5.1. 通信のセキュリティ

MozillaMail、**mutt**、**pine**など, Red Hat Linuxに組み込まれた人気のあるMUAは、SSLで暗号化された電子メールセッションを提供します。

暗号化されずにネットワーク上を流れる他のサービスと同様に、ユーザー名、パスワード、メッセージ全体などの重要な電子メール情報はすべて、傍受して見られるおそれがあります。標準のPOPプロトコルとIMAPプロトコルでは、すべての認証情報は「平文」(暗号化なし)で送られますので侵入者が、ネットワーク上を通過するユーザー名とパスワードを取得することにより、ユーザーのアカウントにアクセスする可能性があります。

11.5.1.1. 安全な電子メールクライアント

リモートサーバー上の電子メールをチェックするような設計のほとんどのLinux MUAは、暗号化するためのSSLをサポートします。電子メールを検索するときにSSLを使用するには、電子メールを電子メールクライアントとサーバー上で有効にする必要があります。

SSLはクライアント側で簡単に有効にできます。これは多くの場合、MUAの設定領域でボタンをクリックするか、MUA設定ファイル内でオプションを使用します。安全なIMAPとPOPは、MUAがメッセージの認証とダウンロードに使用する既知のポート番号（それぞれ993と995）を持っています。

11.5.1.2. 安全な電子クライアント通信

電子メールサーバー上のIMAPユーザーとPOPユーザーにSSL暗号化を提供するのは、簡単にできます。

先ず、SSL証明書を作成します。これは2つの方法で達成できます：SSL証明書を取得できるように*CA*(*Certificate Authority*)へ申請する方法と、自己署名付き証明書を作成する方法です。

重要

自己署名付き証明書は、テスト目的の為にのみ使用すべきものです。生産環境で使用するサーバーはすべてCAにより認可されたSSL証明書を使用すべきです。

IMAP用に自己署名付き証明書を作成するには、`/usr/share/ssl/certs/` ディレクトリに入り、次のコマンドをルートとして入力します：

```
make imapd.pem
```

すべての質問に答えるとプロセスを完了します。

POP用に自己署名付き証明書を作成するには、/usr/share/ssl/certs/ ディレクトリに入り、ルートとして次のコマンドを入力します：

```
make ipop3d.pem
```

これも、すべての質問に答えればプロセスは完了します。

上記のプロセスが終了すると、/sbin/serviceコマンドを終了して該当のデーモン(imaps又はpop3s)を開始させます。その後、**サービス設定ツール** (redhat-config-services)などのinitscript ユーティリティを使用して正しいランレベルで開始するようにimaps サービスあるいはpop3sサービスを設定します。initscript ユーティリティについての詳細情報は項1.4.2で御覧下さい。

別の方法としては、stunnelコマンドを、標準の安全でないimapdデーモン又はpop3dデーモンの周りを囲むSSL 暗号化ラッパーとして使用することも出来ます。

stunnelプログラムはRed Hat Linuxに収納されている外部OpenSSLライブラリを使用して強力な暗号化法と接続保護を提供します。これには、SSL証明書を取るために*Certificate Authority*(*CA*)に申請するのが適切です。但し、自己署名付き証明書を作成することも可能です。

自己署名付きSSL証明書を作成するには、/usr/share/ssl/certs/ディレクトリに入り、以下のコマンドを入力します：

```
make stunnel.pem
```

ここでも全ての質問に答えてプロセスを完了します。

証明書が生成されると、stunnelコマンドを使用して、imapdメールデーモンがスタートできるようになります。次のコマンドを入力します：

```
/usr/sbin/stunnel -d 993 -l /usr/sbin/imapd imapd
```

このコマンドが発行されると、IMAP電子メールクライアントを開くことが出来て、SSL暗号を使用した電子メールサーバーへの接続ができます。

stunnelコマンドを使用してpop3dをスタートするには、以下のコマンドを入力します：

```
/usr/sbin/stunnel -d 993 -l /usr/sbin/pop3d pop3d
```

stunnelの使用法に関する詳細はstunnelのmanページを参照するか、又は次のディレクトリにあるドキュメントを参照して下さい。/usr/share/doc/stunnel-<*version-number*>/

11.6. その他のリソース

以下に電子メールアプリケーションに関するその他のドキュメントの一覧を示します。

11.6.1. インストールされたドキュメント

- Sendmailの設定に関する情報は、sendmailとsendmail-cf のパッケージの中に収納されています。
  - /usr/share/doc/sendmail/README.cf — m4、Sendmailのファイルロケーション、サポートされるメイラー、高度な機能にアクセスする方法などに関する情報を示します。

- `/usr/share/doc/sendmail/README` — Sendmailディレクトリ構成、IDENTプロトコルサポート、ディレクトリパーミッションの詳細、これらのパーミッションが正しく設定されないときに発生する可能性のある一般的な問題などに関する情報を示します。

その他に`sendmail`のmanページ及び`aliases`のmanページには、それぞれ各種Sendmailオプションと`Sendmail /etc/mail/aliases`ファイルの正しい設定に関する役立つ情報が含まれています。

- `/usr/share/doc/fetchmail-<version-number>` — `FEATURES`ファイルと導入用FAQドキュメント内のFetchmail機能を網羅した一覧を示します。
- `/usr/share/doc/procmail-<version-number>` — Procmailの概要を示す`README`ファイル、すべてのプログラム機能を探索するための`FEATURES`ファイル、多数の一般的な設定の質問に対する回答を示すFAQファイルが含まれています。

Procmailの機能を学習したり新しいレシピを作成する場合、以下のProcmailマニュアルページは非常に役に立ちます。

- `procmail` — Procmailの機能の概要と電子メールのフィルタ処理に関連するステップを示します。
- `procmailrc` — レシピの作成に使用する`rc`ファイルフォーマットについて説明しています。
- `procmailex` — 多数の有効な現実世界のProcmailレシピの例を示します。
- `procmailsc` — 特定のレシピがある種のメッセージに合致しているかどうかを確認するために、Procmailで使用されるウェイト付きスコア計算手法について説明しています。
- `/usr/share/doc/spamassassin-<version-number>/` — このディレクトリには、SpamAssassinに関する大量の情報が含まれています。ここで`<version-number>`の部分は`spamassassin`パッケージのバージョン番号で入れ換えます。

11.6.2. 役に立つWebサイト

- http://www.redhat.com/mirrors/LDP/HOWTO/Mail-Administrator-HOWTO.html — 電子メールの機能の概要を示し、クライアント側とサーバー側の可能な電子メールのソリューションと設定を検証しています。
- http://www.redhat.com/mirrors/LDP/HOWTO/Mail-User-HOWTO/ — ユーザーの観点から電子メールを見て、各種の一般的な電子メールクライアントアプリケーションを調べ、別名、転送、自動応答、メーリングリスト、メールフィルタ、スパムなどのトピックへの案内を提供します。
- http://www.redhat.com/mirrors/LDP/HOWTO/mini/Secure-POP+SSH.html — ポート転送機能を持つSSHを使用してPOP電子メールを検索する方法した、電子メールのパスワードとメッセージが安全に転送出来る方法を説明しています。
- http://www.sendmail.net/ — 提供されている多数のオプションの拡大ビューなど、Sendmailに関するニュース、インタビュー、記事などを含んでいます。
- http://www.sendmail.org/ — Sendmailの機能と設定の例の徹底的な技術分析を提供しています。
- http://tuxedo.org/~esr/fetchmail —オンラインマニュアルと徹底的なFAQを特徴とするFetchmailのホームページ。
- http://www.procmail.org/ — Procmail専用のメーリングリストと各種FAQドキュメントへのリンクを持つProcmailのホームページ。
- http://www.ling.helsinki.fi/users/reriksso/procmail/mini-faq.html — トラブルシューティングのヒントや、ファイルロッキングとワイルドカード文字の使い方などに関する詳細を示す、優れたProcmail FAQ。

- http://www.uwasa.fi/~ts/info/proctips.html — `.procmailrc`ファイルをテストし、Procmailスコア機能を使用して特定のアクションをとる必要があるかどうかを決定するなど、さまざまな状況でのProcmailの使用を非常に簡単にする数10のヒントを提供しています。
- http://www.spamassassin.org/ — SpamAssassin プロジェクト本部のサイトです。

11.6.3. 関連書籍

- *Sendmail* Bryan Costales、Eric Allman、その他共著。O'Reilly & Associates社—DelivermailとSendmailの最初の作成者の支援を受けて書かれた優れたSendmailリファレンス。
- *Removing the Spam: Email Processing and Filtering* Geoff Mulligan著; Addison-Wesley Publishing Company — SendmailやProcmailなどの定評のあるツールを使用してスパムの問題を管理する電子メール管理者が使用するさまざまな方法を考察した本。
- *Internet Email Protocols: A Developer's Guide* by Kevin Johnson; Addison-Wesley Publishing Company — 主要な電子メールプロトコルやそれらのプロトコルが提供するセキュリティについて徹底的に検討しています。
- *Managing IMAP* by Dianna Mullet and Kevin Mullet; O'Reilly & Associates — IMAPサーバーを設定するのに必要なステップについて詳述しています。

第12章
BIND

インターネットを含む、殆どの最近のネットワーク上で、ユーザーは他のコンピュータを名前で探します。これによりユーザーは、ネットワークリソースのネットワークアドレス番号を記憶する煩わしさから免れます。そんな名前ベースの接続を許可するネットワークを効率的に設定するには、*DNS*(*Domain Name Service*)あるいは、ネームサーバーを設定します。これによりネットワーク上のホスト名を数値アドレスに、又はその逆方向に解決するものです。

この章ではRed Hat Linuxに収納されているネームサーバーである*BIND* (*Berkeley Internet Name Domain*)DNSサーバーの設定ファイルの構造に焦点を置きながら、それがローカル及びリモートで管理される方法を説明します。

Bind 設定ツール(`redhat-config-bind`)を使用したBIND設定の仕方については、*Red Hat Linux* カスタマイズガイド内の*BIND*の設定の章を参照して下さい。

警告

Bind 設定ツールを使用する場合は、手動で編集しないで下さい。どんな変更もすべて**Bind 設定ツール**を再使用するときに上書きされてしまいます。

12.1. DNSについて

ネットワーク上のホストがホスト名(完全修飾ドメイン名（*FQDN*）とも呼びます)を使用して別のホストに接続する時、そのホストのマシンの名前をそのIPアドレスに関連付ける為にDNSが使用されます。

DNS とFQDNを使用すると、システム管理者はマシンに対する名前ベースのクエリに影響することなくIPアドレスをホストに変換する柔軟性を持てる利点があります。また、管理者は名前ベースのクエリを処理するマシンを入れ換えることが出来ます。

DNSは、通常幾つかのドメインに対し権限を所有し、他のドメイン用のDNSサーバーを参照する中央設置のサーバーを使用して実装されます。

クライアントホストがネームサーバーからの情報を要求すると、通常それはポート53に接続されます。それからネームサーバーはリゾルバライブラリをベースにしてFQDNを解決しようとします。このリゾルバライブラリには、要求されたホストに関して又は以前のクエリのキャッシュデータの権限情報を収納している可能性があります。もし、ネームサーバーがリゾルバライブラリに解答を持っていない場合は、ルートネームサーバーと呼ばれる他のネームサーバーにクエリをして、課題となっているFQDN用の権限を持つネームサーバーを判定します。その後、その情報を使用してその権限ネームサーバーにクエリし、要求のあったホストのIPアドレスを決定します。逆引きのクエリをしている場合、名前ではなく不明なIPアドレスでクエリされること以外は同じ手順が使用されます。

12.1.1. ネームサーバーゾーン

インターネット上では、ホストのFQDNは異なるセクションの分割することが出来ます。これらのセクションはツリーのように階級として構成され、その内容は主幹、1次分岐、2次分岐、などとなります。次のようなFQDNを考えてみましょう：

```
bob.sales.example.com
```

特定のシステムに関連しているIPアドレスを取得する為にFQDNを解決する方法を考える場合、名前は右から左へ読む必要があります。それぞれの階級はピリオド(`.`)で区切られています。この例で

は、comがこのFQDN用のトップレベルドメインを示します。exampleと言う名前は、comの下のサブドメインで、salesはまたexampleの下のサブドメインです。最も左側の名前bobはそのマシンを識別するホスト名です。

ホスト名を除く、それぞれのセクションはゾーンと呼ばれ、特定のネーム空間を定義します。ネーム空間はそのサブドメインの左側の名前を制御します。この例では2つしかサブドメインを含んでいませんが、FQDNは最低限の1つのサブドメインが設定されている限り、ネーム空間の構成に応じてそれ以上含むことが出来ます。

ゾーンは、ゾーンファイルの使用を通じて権限のあるネームサーバー上で定義されます。これがそのゾーンのネーム空間、その特定のドメイン又はサブドメインで使用するメールサーバー、その他を記述します。ゾーンファイルは、本来の権限が所在しファイルが変更される場所であるプライマリネームサーバー(別名：マスターネームサーバー)と、そのプライマリネームサーバーからそれらのゾーンファイルを受け取る セカンダリネームサーバー(別名：スレーブネームサーバー)に保存されます。どのネームサーバーでも同時に異なるゾーン用のプライマリとセカンダリネームサーバーになることが可能で、それらもまた、複数ゾーン用の権限として考慮できます。それはネームサーバーの構成の仕方により決定されるものです。

12.1.2. ネームサーバーのタイプ

プライマリネームサーバー設定には4つのタイプがあります。

- *master* —あるネーム空間に対するオリジナルの権限あるゾーンレコードを保存し、そのネーム空間に関する回答を探している他のネームサーバーからの質問に回答します。
- *slave* — このサーバーに権限があると考えられているネーム空間に関して、他のネームサーバーからのクエリに回答します。しかし、スレーブネームサーバーは、マスターネームサーバーからネーム空間情報を入手します。
- *caching-only* — 名前からIPへの解決を行うサービスを提供しますが、どのゾーンにも権限を持っていません。すべての解決に対して回答は一定期間メモリ内のデータベースにキャッシュされます。キャッシュ期間は通常検索されたゾーンレコードによって指定されます。
- *forwarding* — 解決を行うべきネームサーバーの一覧に要求を転送します。指定されたネームサーバーのうちどのネームサーバーも解決することができない場合、処理は停止し、解決は失敗します。

ネームサーバーは、上記のタイプのうち1つのタイプであるか、複数のタイプであることが可能です。たとえば、あるネームサーバーはあるゾーンではマスターであり、他のゾーンではスレーブであってもかまいませんし、解決転送だけを提供するものであってもかまいません。

12.1.3. ネームサーバーとしてのBIND

BINDネームは/usr/sbin/namedを通して名前解決サービスをします。BINDはまた、/usr/sbin/rndcと言う管理ユーティリティも含んでいます。rndcに関する詳細は、項12.4の中で御覧になれます。

BINDでは、以下の2ヶ所にその設定ファイルが格納されています：

- /etc/named.conf — namedデーモン用の設定ファイル。
- /var/named/ディレクトリ— namedの作業ディレクトリで、ゾーン、静的、キャッシュのファイルをそれぞれ格納します。

以下の数ヶ所のセクションで、BIND設定ファイルを詳細に説明します。

12.2. /etc/named.conf

named.confファイルは、左右の大括弧{ }で囲まれた組み込みオプションを使用した一連のステートメントです。多くの小さなエラーがnamedの開始を阻止しますので、管理者はnamed.confを編集する時に構文上のエラーを避けるように注意する必要があります。

警告

Bind 設定ツールを使用している場合は、/etc/named.confファイルと/var/namedディレクトリ内のファイルを手動で編集しないでください。これらのファイルへの手動変更はすべて、次回に**Bind 設定ツール**が使用される時点に上書きされてしまいます。

標準的なnamed.confは、次の例に様に構成されています：

```
<statement-1> ["<statement-1-name>"] [<statement-1-class>] {
  <option-1>;
  <option-2>;
  <option-N>;
};

<statement-2> ["<statement-2-name>"] [<statement-2-class>] {
  <option-1>;
  <option-2>;
  <option-N>;
};

<statement-N> ["<statement-N-name>"] [<statement-N-class>] {
  <option-1>;
  <option-2>;
  <option-N>;
};
```

12.2.1. 一般的なステートメントのタイプ

以下のタイプのステートメントが一般的に/etc/named.confで使用できます：

12.2.1.1. aclステートメント

aclステートメント(access control statement)はネームサーバーへ許可又は拒否されるホストのグループを定義します。

aclステートメントは次の形をとります：

```
acl <acl-name> {
  <match-element>;
  [<match-element>; ...]
};
```

このステートメント内では、*<acl-name>*をアクセス制御リストの名前で入れ換え、*<match-element>*をセミコロンで隔離されたIPアドレスで入れ換えます。殆どの場合、個々のIPアドレス、又はIPネットワーク表記(10.0.1.0/24など)を使用してaclステートメント内のIPアドレスを識別します。

次のアクセス制御リストは、設定を簡素がする為にキーワードとして既に定義されています：

- any — すべてのIPアドレスと一致。
- localhost — ローカルシステムによって使用されているIPアドレスと一致。

- localnets — ローカルシステムが接続しているネットワークのIPアドレスと一致。
- none — どのIPアドレスとも一致しない。

他のステートメント(optionsステートメントなど)と一緒に使用した場合、aclステートメントはBINDネームサーバーの誤用を防止するのに役に立ちます。

次の例は、2つのアクセス制御リストを定義し、optionsステートメントを使用してネームサーバーによるそれらの取扱方法を指定しています：

```
acl black-hats {
 10.0.2.0/24;
 192.168.0.0/24;
};

acl red-hats {
 10.0.1.0/24;
};

options {
 blackhole { black-hats; };
 allow-query { red-hats; };
 allow-recursion { red-hats; };
}
```

上記の例は、black-hatsとred-hatsの2つのアクセス制御リストを示していますが、black-hatsリスト内のホストは、ネームサーバーへのアクセスを拒否されて、red-hatsリスト内のホストは通常のアクセスが許可されています。

12.2.1.2. includeステートメント

includeステートメントはファイルをnamed.confにインクルードできるようにします。この方法で貴重な設定データ(keysなど)は制限付き権限と共に個別のファイルに保管することが可能です。

includeステートメントは次のような形を取ります：

```
include "<file-name>"
```

このステートメントでは、<*file-name*>はファイルへの絶対パスで入れ換えます。

12.2.1.3. optionsステートメント

optionsステートメントはグローバルサーバーの設定オプションを定義して他のステートメントのデフォルトをセットします。これはnamedの作業ディレクトリの場所、許可できるクエリのタイプ、その他を指定するのに使用されます。

optionsステートメントは以下の形をとります：

```
options {
   <option>;
[<option>; ...]
};
```

このステートメントでは、<*option*>ディレクティブは有効なオプションで入れ換えます。

次のようなオプションが一般的に使用されます：

- allow-query — どのホストにこのネームサーバーへのクエリを許可するかを指定します。デフォルトでは、すべてのホストのクエリが許可されます。ここでアクセス制御リストや一連のIPアドレス又はネットワークを使用して、特定のホストだけにネームサーバーへのクエリを許可することができます。
- allow-recursion — allow-queryと似ていますが、これは繰り返しクエリに適用されます。デフォルトでは、すべてのホストにネームサーバーへの繰り返しクエリを行うことが許可されます。
- blackhole — どのホストがサーバーへのクエリを許可されないかを指定します。
- directory — named作業ディレクトリをデフォルトの/var/named以外のディレクトリに変更します。
- forward — forwardersディレクティブの転送方法を制御します。

  以下のようなオプションが許可されます：

  - first — forwardersディレクティブに指定されているネームサーバーにクエリが実行されると、その後namedがその名前を自分で解決しようとするように指定します。
  - only — forwardersディレクティブに指定されているネームサーバーへのクエリが失敗した場合、namedが自分で名前解決しないように指定します。

- forwarders — 解決を求めて要求が転送されるネームサーバーの有効なIPアドレスの一覧を指定します。
- listen-on — namedがクエリの監視をする場所であるネットワークインターフェイスを指定します。デフォルトではすべてのインターフェイスが使用されます。

  このようにして、DNSサーバーがゲートウェイでもある場合、BINDは、そのネットワークの1つから届くクエリのみに回答するように設定できます。

  listen-onディレクティブは以下のように見えます：

```
options {
 listen-on { 10.0.1.1; };
};
```

  このようにすると、プライベートネットワーク用ネットワークインターフェイスから到着した要求(10.0.1.1)だけが受け付けられます。
- notify — ゾーンが更新されたときに、namedがスレーブサーバーに更新を通知するかどうかを制御します。これは以下のようなオプションを受け付けます：
  - yes — スレーブサーバーに通知します。
  - no — スレーブサーバーに通知しません。
  - explicit — ゾーンステートメント内のalso-notifyリストに指定されているスレーブサーバーにのみ通知します。

- pid-file — namedによって作成されたプロセスIDファイルの場所を指定します。
- statistics-file — 統計ファイル用に別の場所を指定することができます。デフォルトでは、named統計は、/var/named/named.statsファイルに保存されています。

他にも数十のオプションが利用できます。その多くは正常に機能するには他の1つ依存します。詳細に関しては項12.7.1内の*BIND 9* アドミニストレータ参照マニュアル、及びbind.confのmanページを御覧下さい。

12.2.1.4. zoneステートメント

zoneステートメントはその設定ファイルの場所やゾーン独自のオプションなどゾーンの特徴を指定します。このステートメントは、グローバルoptionsステートメントを上書きするのに使用できます。

zoneステートメントは以下のような形をとります：

```
zone <zone-name> <zone-class> {
  <zone-options>;
  [<zone-options>; ...]
};
```

このステートメントでは、*<zone-name>*がゾーンの名前であり、*<zone-class>*がゾーンのオプションクラスで、*<zone-options>*がゾーンの特徴となるオプションの一覧です。

*<zone-name>*の属性はゾーンステートメントにとって重要です。ゾーンの名前が、/var/named/に位置している対応のゾーンファイルで使用される$ORIGINディレクティブに、割り当てられるデフォルト値であるからです。namedデーモンはゾーンの名前をゾーンファイル内にリストしてあるいずれかの非FQDNに付け加えます。

例えば、zoneステートメントがexample.com用にネーム空間を定義する場合、example.comを*<zone-name>*として使用すると、それはexample.comゾーンファイルの中でホスト名の末尾に配置されます。

ゾーンファイルに関する詳細は項12.3で御覧下さい。

最も一般的なzoneステートメントオプションには次のようなものがあります：

- allow-query — このゾーンについての情報を要求することのできるクライアントを指定します。デフォルトでは、すべてのクエリ要求を許可します。
- allow-transfer — ゾーン情報の転送を要求することが許可されたスレーブサーバーを指定します。デフォルトでは、すべての転送要求を許可します。
- allow-update — ゾーン内の情報を動的に更新することのできるホストを指定します。デフォルトでは、すべての動的更新要求を拒否します。

  ホストがゾーン情報を更新するのを許可する場合には十分注意が必要です。指定されたホストが完全に信頼されていない場合には、このオプションを有効にしないでください。もし可能であれば管理者に手動でゾーンのレコードを更新してもらい、namedサービスをリロードするのがよいでしょう。
- file — ゾーンの設定データが記載されたnamed作業ディレクトリの中のファイル名を指定します。
- masters — mastersオプションは権限のあるゾーン情報の要求先のIPアドレスをリストします。ゾーンがslave typeとして定義されている場合にのみ使用します。
- notify — ゾーンが更新された時にnamedからスレーブサーバーにnamedから通知を出すかどうかを制御します。次のようなオプションが受け付けられます：
  - yes — スレーブサーバーに通知します。
  - no — スレーブサーバーに通知しません。
  - explicit — ゾーンステートメント内にあるalso-notifyリストに指定してあるスレーブサーバーにのみ通知します。
- type — ゾーンのタイプを定義します。

  以下に有効なオプションを示します：
  - forward — 他のネームサーバーにこのゾーンの情報を求めるすべての要求を転送します。

- hint — ルートネームサーバーをポイントするのに使用される特別なタイプのゾーンです。ルートネームサーバーは、他の方法ではあるゾーンのことがわからない場合に、クエリを解決します。hint ゾーンでは、デフォルトを越えた設定は必要ありません。
- master — このゾーン用の権限としてのネームサーバーを示します。システムにそのゾーンの設定ファイルがある場合、そのゾーンはmasterとして設定しなくてはなりません。
- slave — このネームサーバーをこのゾーンのスレーブサーバーと指名します。さらに、このゾーン用にマスターネームサーバーのIPアドレスを指定します。

- zone-statistics — namedにこのゾーンについての統計を保持するよう命令し、デフォルト位置（/var/named/named.stats）か、あるいはserverステートメントのstatistics-fileオプションによって指定された場所にこれを書きこみます。serverのステートメントに関する詳細は項12.2.2で御覧下さい。

12.2.1.5. zoneステートメントのサンプル

マスターネームサーバーやスレーブネームサーバーの/etc/named.confファイルに対する変更は、zoneステートメントの追加、変更、削除などに関わるものです。これらのzoneステートメントには、数多くのオプションを含めることができまが、ほとんどのネームサーバーは、そのうちのほんのわずかしか利用しません。以下のzoneステートメントは、マスター/スレーブネームサーバー関係で利用することのできる非常に基本的な例です。

以下にexample.comをホストするプライマリネームサーバー(192.168.0.1)用のzoneステートメントの1例を示します：

```
zone "example.com" IN {
 type master;
 file "example.com.zone";
 allow-update { none; };
};
```

上記ステートメントでは、ゾーンをexample.comと名づけ、namedをmasterとして設定し、namedに/var/named/example.com.zone ファイルを読み込むように指示しています。またnamedに対して他のホストによる更新を受け付けないように指示しています。

example.com用のスレーブサーバーのzoneステートメントは以前の例とは少々違ってみえます。スレーブサーバー用には、タイプがslaveとセットしてあり、allow-update行の場所には、namedに対してマスターサーバーのIPアドレスを伝えるディレクティブがあります。

example.comの為のスレーブサーバーのzoneステートメントは以下のようになります：

```
zone "example.com" {
 type slave;
 file "example.com.zone";
 masters { 192.168.0.1; };
};
```

このzoneステートメントは、スレーブサーバー上のnamedを設定して、example.comゾーンに関する情報を得るために192.168.0.1のIPアドレスでマスターサーバーを見付けます。スレーブサーバーがマスターサーバーから取得する情報は/var/named/example.com.zoneファイルに保存されているものです。

12.2.2. 他のステートメントタイプ

以下に、named.confの中で利用でき、使用頻度に低いステートメントタイプの一覧を示します。

- controls — namedサービス管理用のrndcコマンドを使用するのに必要な各種のセキュリティ要求を設定します。

  一緒に使用する各種オプションを含んだcontrolsステートメントがどのように見えるかを知るには、項12.4.1を参照して下さい。

- key "<*key-name*>" — 名前によって特定の鍵を定義します。鍵は安全な更新やrndc コマンドの使用などのさまざまな行動の認証に使用されるものです。keyでは以下の2つのオプションが使用されます：

  - algorithm <*algorithm-name*> — dsa又はhmac-md5など、使用されるアルゴリズムのタイプ。
  - secret "<*key-value*>" — 暗号化した鍵。

  keyステートメントの書き方については項12.4.2を御覧下さい。

- logging — *channel*と呼ばれる複数タイプのログの使用を許可します。loggingステートメント内でchannelオプションを使用することにより、自己のファイル名(file)、サイズ限定(size)、バージョン指定(version)、及び重要度のレベル(severity)などを持つカスタムタイプのログを構成することができます。1度カスタムチャンネルが定義されると、categoryオプションを使用してチャンネルを分類化でき、namedが再起動した時にログが始まります。

  ディフォルトでは、namedは、syslogデーモンへ標準のメッセージをログします。そしてそれを/var/log/messagesに配置します。これが起こるのは、幾つかの標準チャンネルが数種の重要度レベルでBINDに組み込まれており、その1つとして情報ログのメッセージ(default_syslog)を処理するもの、もう1つは特にデバッグを処理するメッセージ(default_debug)を処理するものなどがあるからです。defaultと呼ばれるデフォルトのカテゴリーは、特殊な設定なしに通常のログを取る組み込みチャンネルを使用します。

  ログプロセスのカスタマイズは、かなり細かなプロセスでこの章の説明範囲から越えてしまうものです。カスタムのBINDログの作成法に関する詳細は項12.7.1の中の*BIND 9* アドミニストレータ参照マニュアルを御覧下さい。

- server — 特に通知とゾーン転送に関して、namedがリモートネームサーバーに対してどう対処するかを左右する特定のオプションを定義します。

  transfer-formatオプションは、1つのリソースレコードがそれぞれのメッセージ(one-answer)と共に送信されるか、又は複数のリソースレコードがそれぞれのメッセージ(many-answers)と一緒に送信されるかを制御します。many-answersはより効率的ですが最新のBINDネームサーバーだけがそれを理解します。

- trusted-keys — この中にはセキュアDNS (DNSSEC)で使用される各種の公開鍵が含まれています。BINDセキュリティに関する詳細は項12.5.3で御覧下さい。

- view "<*view-name*>" — ネームサーバーにコンタクトしているホストに応じて特別なビューを作成します。これにより、幾つかのホストはある特定のゾーンに関して1つの回答を受けることができ、その他のホストは全く別の情報を受け取るようにできます。別の方法として、特定のゾーンだけが特定の信頼できるホストに利用可能となり、信用できないホストは単に他のゾーンに関するクエリをするだけということも出来ます。

  名前が独自であれば、複数のビューも使用できます。match-clients オプションは、ある特定のビューに適用するIP アドレスを指定します。どのようなoptionsステートメントでもビューの中で使用でき、既にnamed用に設定してあるグローバルオプションを上書き出来ます。殆どのviewステートメントはmatch-clientsリストに適用できる複数のzoneステートメントを含んでいます。クライアントのIPアドレスに適合する最初のviewステートメントが採用される為、viewステートメントがリストされている順序が重要です。

  viewステートメントに関する詳細は項12.5.2で御覧下さい。

12.2.3. コメントタグ

以下にnamed.conf内で使用される有効なコメントタグの一覧を示します：

- // — 行の先頭に位置している場合は、その行はnamedによって無視されます。
- # — 行の先頭に位置している場合は、その行はnamedによって無視されます。
- /*及び*/ — テキストがこれらのタグで囲まれている場合、テキストのそのブロックはnamedによって無視されます。

12.3. ゾーンファイル

ゾーンファイルには、特定のネーム空間についての情報が記載されており、デフォルトでnamedの作業ディレクトリ/var/named/にに保存されます。各ゾーンファイル名はzoneステートメントのfileオプションデータに従い、通常example.com.zoneのように該当するドメインに関係し、ゾーンデータが記載されているファイルとして識別できるような名前が付けられます。

各ゾーンファイルには、ディレクティブとリソースレコードが含まれている場合があります。ディレクティブは、ネームサーバーに対して、あることを実行したり、ゾーンに特別の設定を適用したりするよう命令するものです。リソースレコードは、ゾーンのパラメータを定義し、個々のホストに識別を割り当てるものです。ディレクティブはオプションですが、リソースレコードはネームサービスをそのゾーンに提供するため必須です。

すべてのディレクティブとリソースレコードは、定められた個々の行に記載されなくてはなりません。

コメントは、ゾーンファイル内のセミコロン（;）の後に置かれます。

12.3.1. ゾーンファイルディレクティブ

ディレクティブは、ドルサイン文字($)で始まり、その後にディレクティブの名前が続きます。通常はゾーンファイルの先頭に置かれます。

以下のディレクティブが最も一般的に使用されます：

- $INCLUDE — ディレクティブが使用されている場所で、このゾーンファイル内に別のゾーンファイルをインクルードするようnamedを設定します。このディレクティブにより、おもなゾーンファイル以外にも追加ゾーン設定を保存することができます。
- $ORIGIN — ホストだけしか指定していないレコードなど、資格のないレコードに付けるドメイン名を設定します。

  たとえば、ゾーンファイルには以下のような行が含まれていてもかまいません。
  ```
  $ORIGIN example.com
  ```
  後付きのピリオド(.)を持たないリソースレコードに使用されている名前はどれも、その後にexample.comが付加されます。

  

  注意

  ゾーンが/etc/named.conf内に指定されている場合は、$ORIGIN ディレクティブを使用する必要はありません。ゾーンの名前はデフォルトで$ORIGIN ディレクティブ値として使用されます。

- $TTL — デフォルトの*Time to Live*（*TTL*）値をゾーンに設定します。これは、ゾーンのリソースレコードが有効である時間を秒単位で与えられる数です。各リソースレコードはそれぞれ自己のTTL値を含むことが可能で、それがこのディレクティブを上書きします。

この値を増加させると、リモートネームサーバーは、このゾーンの情報をより長時間キャッシュします。こうすると、このゾーンについて行われるクエリの数は減りますが、リソースレコード変更を伝えるのに要する時間は長くなります。

12.3.2. ゾーンファイルリソースレコード

ゾーンファイルの主要コンポーネントはそのリソースレコードです。

ゾーンファイルリソースレコードには、多種のタイプがあります。最もよく使用されるタイプを以下に示します：

- A — アドレスレコード。名前に割り当てるIPアドレスを次の例のように指定します。
  ```
  <host>  IN  A  <IP-address>
  ```
  この<*host*>値が省略された場合、Aレコードはネーム空間の一番上にデフォルトIPアドレスをポイントします。このシステムは、すべての非FQDN要求の対象となります。

  example.comゾーンファイルについて、以下のAレコード例を考えてみましょう：
  ```
       IN  A   10.0.1.3
  server1  IN  A   10.0.1.5
  ```
  example.com用の要求は10.0.1.3をポイントし、server1.example.comの要求は10.0.1.5をポイントします。

- CNAME — Canonical ネームレコードで、1つのネームを別のネームにマップします。このタイプのレコードはエイリアス(別名)レコードとして知られています。

  次の例では、namedに対して、<*alias-name*>に送信された要求はすべてホスト、<*real-name*>をポイントすることを知らせます。CNAMEレコードは、最も一般的にWebサーバーのwwwのように共通名スキームを使用するサービスをポイントするのに使用されます。
  ```
  <alias-name>  IN  CNAME   <real-name>
  ```
  次の例で、1つのAレコードがあるホスト名をあるIPアドレスにバインドし、CNAMEレコードが一般的に使用されるwwwホスト名をそれにポイントしています。
  ```
  server1  IN  A   10.0.1.5
  www    IN  CNAME server1
  ```

- MX — Mail eXchangeレコード。このゾーンによって制御されるネーム空間に送られるメールがどこへ行くのかを知らせます。
  ```
    IN  MX  <preference-value> <email-server-name>
  ```
  上記の例では、<*preference-value*>により、このネーム空間のEメールサーバーを数値的にランク付けすることが出来るため、幾つかのEメールシステムに、他のEメールシステムよりも優先させることが出来ます。最低の<*preference-value*>を持つMXリソースレコードは、他のものよりも優先されますが、同じ値で複数のEメールサーバーを指定してEメールのトラフィックを分散させることができます。

  <*email-server-name*>はホスト名か、FQDNとすることが出来ます。
  ```
    IN  MX  10  mail.example.com.
    IN  MX  20  mail2.example.com.
  ```
  この例では、最初のmail.example.comEメールサーバーはexample.comドメイン宛のEメールを受信する時に、2行目のmail2.example.comEメールサーバーよりも優先されます。

- NS — ネームサーバーレコード。あるゾーンに対して権限のあるネームサーバーを発表する。

  以下にNSレコードの例を示します：
  ```
    IN  NS  <nameserver-name>
  ```
  <*nameserver-name*>はFQDNでなくてはなりません。

次に、2つのネームサーバーがあるドメインについて権限を持っていることが示されます。これらのネームサーバーがどちらもスレーブであるか、それとも1つはマスターであるかは重要ではありません。これらは両方とも権限があると考えられます。

```
  IN  NS  dns1.example.com.
  IN  NS  dns2.example.com.
```

- PTR — PoinTeRレコード。ネーム空間の別の部分を指すよう設計されています。

  PTRレコードは、逆にIPアドレスから名前をポイントするため、主に逆引き名前解決に使用されます。使用中のPTRレコードの例については項12.3.4を参照してください。

- SOA — Start Of Authorityレコード。ネーム空間についての重要な権限ある情報をネームサーバーに示します。

  SOAレコードは、ディレクティブの後に置かれ、ゾーンファイル内の最初のリソースレコードとなります。

  以下の例は基本的なSOAレコードの構成を示しています：

```
@  IN  SOA  <primary-name-server>  <hostmaster-email> (
        <serial-number>
        <time-to-refresh>
        <time-to-retry>
        <time-to-expire>
        <minimum-TTL> )
```

  @記号は、このSOAリソースレコードによって定義されているネーム空間として$ORIGINディレクティブ（$ORIGINディレクティブが設定されていない場合にはゾーンの名前）を置きます。このドメインの権限であるプライマリネームサーバーは<*primary-name-server*>の為に使用され、このネーム空間に関して連絡する人のEメールは<*hostmaster-email*>に置かれています。

  namedが、このゾーンをリロードするべきであることがわかるよう<*serial-number*>は、ゾーンファイルが変更されるたびにインクリメントされます。<*time-to-refresh*>は、ゾーンに変更が行われた場合にマスターネームサーバーに問い合わせるまでどのくらい長く待つべきかをすべてのスレーブサーバーに知らせます。<*serial-number*>値は、スレーブが古いゾーンデータを使用しているかどうか、そしてリフレッシュすべきなのかどうかを判断するためスレーブによって使用されます。

  <*time-to-retry*>は、マスターネームサーバーが回答していない場合に、別のリフレッシュ要求を発行するまでどのくらいの間隔待つべきかをスレーブネームサーバーに知らせます。マスターが、<*time-to-expire*>の経過前にリフレッシュ要求に回答しない場合、スレーブはそのネーム空間についての要求について権限をもつものとして応答するのを停止します。

  <*minimum-TTL*>は、他のネームサーバーが少なくともこの時間(秒単位)の長さだけ、ゾーンの情報をキャッシュすることを要求します。

  BINDでは、すべての時間は秒数で表します。しかし、分（M）、時間（H）、日（D）、週（W）など秒以外の時間単位の短縮形を使用することもできます。表12-1の表は、秒数での時間量と他のフォーマットでの等価時間を示します。

秒	他の時間単位
60	1M
1800	30M
3600	1H
10800	3H
21600	6H
43200	12H

秒	他の時間単位
86400	1D
259200	3D
604800	1W
31536000	365D

表12-1. 他の時間単位と比較した秒数

以下の例は、基本的なSOAリソースレコードが、実際の値で設定されるとどのように表示されるかを示したものです

```
@  IN  SOA  dns1.example.com.  hostmaster.example.com.(
         2001062501;serial
         21600   ;refresh after 6 hours
         3600    ;retry after 1 hour
         604800  ;expire after 1 week
         86400 )  ;minimum TTL of 1 day
```

12.3.3. ゾーンファイルの例

個別に見た場合、ディレクティブとリソースレコードは把握するのが困難です。しかし、共通ファイルとして一緒に置くと理解しやすくなります。

以下に非常に基本的なゾーンファイルの例を示します。

```
$ORIGIN example.com
$TTL 86400
@  IN  SOA  dns1.example.com.  hostmaster.example.com.(
         2001062501;serial
         21600   ;refresh after 6 hours
         3600    ;retry after 1 hour
         604800  ;expire after 1 week
         86400 )  ;minimum TTL of 1 day

   IN  NS  dns1.example.com.
   IN  NS  dns2.example.com.

   IN  MX  10  mail.example.com.
   IN  MX  20  mail2.example.com.

      IN  A   10.0.1.5

server1   IN  A   10.0.1.5
server2   IN  A   10.0.1.7
dns1    IN  A   10.0.1.2
dns2    IN  A   10.0.1.3

ftp     IN  CNAME server1
mail    IN  CNAME server1
mail2    IN  CNAME server2
www     IN  CNAME server2
```

この例では、標準ディレクティブとSOA値が使われています。権限のあるネームサーバーは、dns1.example.comとdns2.example.comに設定され、これらをそれぞれ10.0.1.2と10.0.1.3に結び付けるAレコードがあります。

MXレコードで設定されるEメールサーバーは、CNAMEレコードを介してserver1とserver2をポイントします。server1とserver2の名前は最後がピリオド（.）で終わっていないため、その後

ろに$ORIGINドメインが置かれ、server1.example.comとserver2.example.comに拡張されます。関連Aリソースレコードを通して、そのIPアドレスを決定することができます。

標準名のftp.example.comとwww.example.comで利用できる一般的なFTPとWebのサービスは、CNAMEレコードを使って、これらの名前に合ったサービスにポイントされます。

12.3.4. 逆引き名前解決ゾーンファイル

逆引き名前解決ゾーンファイルは、特定のネーム空間のIPアドレスをFQDNに変換します。これは標準ゾーンファイルにとてもよく似ていますが、PTRリソースレコードがIPアドレスを完全修飾ドメイン名に連結するのに使われるという点で異なっています。

PTRレコードは以下のようになります：

```
<last-IP-digit>  IN  PTR  <FQDN-of-system>
```

The <last-IP-digit>は、特定のシステムのFQDNをポイントすべきIPアドレスの最後の数と一致します。

次の例では、10.0.1.20から10.0.1.25までのIPアドレスが対応するFQDNにポイントされています。

```
$ORIGIN 1.0.10.in-addr.arpa
$TTL 86400
@  IN  SOA  dns1.example.com.  hostmaster.example.com. (
         2001062501 ; serial
         21600   ; refresh after 6 hours
         3600   ; retry after 1 hour
         604800  ; expire after 1 week
         86400 )  ; minimum TTL of 1 day

   IN  NS  dns1.example.com.
   IN  NS  dns2.example.com.

20  IN  PTR  alice.example.com.
21  IN  PTR  betty.example.com.
22  IN  PTR  charlie.example.com.
23  IN  PTR  doug.example.com.
24  IN  PTR  ernest.example.com.
25  IN  PTR  fanny.example.com.
```

このゾーンファイルは、named.confファイルのzoneステートメントでサービスにコールされます。以下のようになります：

```
zone "1.0.10.in-addr.arpa" IN {
 type master;
 file "example.com.rr.zone";
 allow-update { none; };
};
```

ゾーンの命名法を除き、この例と標準zoneステートメントの間にはほとんど違いがありません。逆引き名前解決ゾーンでは、IPアドレスの最初の3つのブロックを逆にし、その後に.in-addr.arpaを添付する必要があることに注意してください。これにより逆引き名前解決ゾーンファイルで使用されるIP番号の1つのブロックがこのゾーンで正しく添付されます。

12.4. rndcの使用法

BINDには、rndcというユーティリティコマンドが含まれています。それを使用することで、ローカルホスト又はリモートホストからのnamed デーモンのコマンドライン管理ができるようになります。

他のシステムの権限のないユーザーによってサーバーのBINDが制御されるのを防ぐため、共有秘密鍵方法を使用して、特定のホストに明示的に特権を与えます。このことは、/etc/named.confとrndcの設定ファイルである/etc/rndc.confの両方で同一の鍵を所有する必要があることを意味します。

12.4.1. /etc/named.confの設定

rndcがnamedサービスに接続される為には、BINDサーバーの/etc/named.confファイルにcontrols ステートメントがなければなりません。

以下の例に示すcontrolsステートメントにより、ローカルホストからrndcが接続できるようになります。

```
controls {
 inet 127.0.0.1 allow { localhost; } keys { <key-name>; };
};
```

このステートメントはnamedにループバックアドレスのデフォルトTCPポート953をリッスンするように指示し、適切な鍵が与えられた場合にローカルホストからのrndcコマンドを許可します。<*key-name*>は、keyステートメントに関連しますが、これも/etc/named.confファイル内にあります。次の例は、keyステートメントのサンプルを示します。

```
key "<key-name>" {
 algorithm hmac-md5;
 secret "<key-value>";
};
```

この場合、<*key-value*>はHMAC-MD5鍵です。以下のコマンドを使用してHMAC-MD5鍵を生成することが出来ます：

```
dnssec-keygen -a hmac-md5 -b <bit-length> -n HOST <key-file-name>
```

鍵は256ビット長以上ある方がいいでしょう。<*key-value*>領域に置くべき実際の鍵は、<*key-file-name*>にあります。

重要

/etc/named.confは、すべてが読み込めるファイルであるため、keyステートメントを別のファイルの中に置いてrootのみの読み取り可能して、次の例のようにincludeステートメントを使用して参照するのが良いでしょう：

```
include "/etc/rndc.key";
```

12.4.2. /etc/rndc.confの設定

keyは/etc/rndc.confの中で最も重要なステートメントです。

```
key "<key-name>" {
 algorithm hmac-md5;
 secret "<key-value>";
```

```
};
```

<*key-name*>と<*key-value*>は/etc/named.conf内での設定とまったく同じでなくてはなりません。

ターゲットサーバーの/etc/named.confに指定された鍵をテストするには次の行を/etc/rndc.confに追加します。

```
options {
 default-server localhost;
 default-key  "<key-name>";
};
```

このコマンドは、グローバルなデフォルト鍵を設定します。しかし、rndcコマンドは、次の例にあるように、異なるサーバーには異なる鍵を使用できます：

```
server localhost {
 key "<key-name>";
};
```

重要

rootユーザー以外は/etc/rndc.confファイルを読み書きできないようにしてください。

12.4.3. コマンド行オプション

rndcコマンドは以下のような形態をとります。

```
rndc <options> <command> <command-options>
```

rndcを適切に設定されたローカルホストで実行する場合、以下のコマンドが利用できます。

- halt — namedサービスをただちに停止します。
- querylog — このネームサーバーに送られたクエリのすべてをログします。
- refresh —ネームサーバーのデータベースをリフレッシュします。
- reload — ゾーンファイルをリロードしますが、以前にキャッシュされた他の回答を全て保存します。このコマンドを使用すると、すべての保存された解決を消失することなくゾーンファイルの変更が出来ます。

  もし、変更が特定のゾーンのみに影響する場合、reloadコマンドの後にそのゾーン名を付加することで1つのゾーンだけをリロードします。
- stats — 現在のnamed統計を/var/named/named.statsファイルにダンプします。
- stop — サーバーを安全に停止し、終了前に動的な更新や*IXFR* (*Incremental Zone ransfers*)データを保存します。

ときには、/etc/rndc.confファイル内のデフォルト設定を上書きしたい場合があるかもしれません。そのような場合には、以下のようなオプションが利用できます：

- -c <*configuration-file*> — rndcにデフォルトの/etc/rndc.conf以外の設定ファイルを使用するよう指示します。
- -p <*port-number*> — デフォルトの953以外のrndc接続用ポート番号を指定します。

- `-s <server>` — 設定ファイルに指定してあるdefault-server以外のサーバーにコマンドを送信するようにrndcに指示します。
- `-y <key-name>` — /etc/rndc.confファイルのdefault-keyオプション以外の鍵を指定します。

これらのオプションについてのさらに詳しい情報は、rndcmanページに記載されています。

12.5. BINDの高度な機能

ほとんどのBIND実装では、namedだけを使用して名前解決サービスを提供したり、特定のドメインかサブドメインの権限として動作したりします。しかしBINDバージョン9には数多くの高度な機能があり、より安全で効率的なDNSサービスを利用することができます。

重要

DNSSEC、TSIG、IXFRなど先進機能のうちいくつかは、この機能に対応したネームサーバーを持つネットワーク環境でのみ使用することができます。ネットワーク環境に非BINDのネームサーバーか、旧式のBINDネームサーバーがある場合には、これらを利用する前に特定の先進機能が利用可能であるかどうかを確認してください。

ここで述べる機能はすべて、*BIND9*管理者リファレンスマニュアルでさらに詳細に説明されています。このマニュアルの説明については、項12.7.1を参照してください。

12.5.1. DNSプロトコル改良

BINDは、*IXFR*（増分ゾーン転送：*Incremental Zone Transfers*）をサポートしています。ここでは、スレーブネームサーバーはマスターネームサーバー上で変更されたゾーンの更新部分をダウンロードするだけです。標準転送プロセスでは、たとえほんのわずかな変更であってもゾーン全体を各スレーブネームサーバーに転送しなくてはなりませんでした。非常に長いゾーンファイルと数多くのスレーブネームサーバーを持つ非常に人気のあるドメインについては、IXFRを利用することにより、通知と更新プロセスのリソース集中を大幅に削減することができます。

IXFRは、動的更新を利用してマスターゾーンレコードの変更を行っている場合にのみ利用可能であることに注意してください。手作業でゾーンファイルを編集して変更を行っている場合は、AXFRが使用されます。動的更新の詳細については、*BIND9*管理者リファレンスマニュアルを参照してください。その詳細は項12.7.1で御覧下さい。

12.5.2. 複数ビュー

named.confのviewステートメントを使用することにより、BINDでは、誰が要求を出しているかに応じて異なる情報を提出することができます。

ローカルネットワーク以外のクライアントには重要なタイプのDNSクエリを拒絶し、内部のクライアントはこれができるようにしたいというような場合、この機能が使用されます。

viewステートメントは、match-clientsオプションを使用してIPアドレスかネットワーク全体を一致させ、特別のオプションとゾーンデータを与えるようにします。

12.5.3. セキュリティ

BINDはマスターネームサーバーとスレーブネームサーバーの両方でゾーンの更新と転送を保護するためのさまざまな方法をサポートしてしています。

- *DNSSEC — DNS SECurity*の短縮形。この機能を利用すると、ゾーン鍵でゾーンを暗号的に署名することができます。

  この方法により、ある特定のゾーンの情報は、受領者がそのネームサーバーの公開鍵を持っている限り、特定の秘密鍵で署名したネームサーバーから来たものとして検証することができます。

  BINDバージョン9はまた、メッセージ認証のSIG(0)公開秘密鍵方法をサポートしています。

- *TSIG — Transaction SIGnatures*の略語です。マスターサーバーとスレーブサーバーに共有秘密鍵が存在することが証明された後でのみ、この機能でマスターからスレーブへの転送が認可されます。

  この機能により標準IPアドレスに基づいた転送許可の方法が強化されます。攻撃者はIPアドレスにアクセスしてゾーンを転送しなくてはならないだけでなく、秘密鍵を知らなくてはならなくなります。

  BINDバージョン9はまた、*TKEY*をサポートしています。これは、ゾーン転送を許可するもう1つの共有秘密鍵方法です。

12.5.4. IPバージョン6

BINDバージョン9は、`A6`ゾーンレコードを使用することによりIPバージョン6(IPv6)環境でのネームサービスを提供することができます。

ネットワーク環境にIPv4ホストとIPv6ホストが両方とも含まれている場合、すべてのネットワーククライアントで`lwresd`軽量リゾルバデーモンを使用しなくてはなりません。このデーモンは、本質的に非常に効率の高いキャッシュのみのネームサーバーであり、IPv6で利用されている新しい`A6`レコードと`DNAME`レコードを理解します。詳細については`lwresd`のmanページを参照してください。

12.6. よくある間違いを避けるために

初心者にとってBIND設定ファイルを編集するときに間違えたりすることはよくあります。以下の問題を避けるように気を付けて下さい:

- ゾーンファイルを編集するときには必ずシリアル番号をインクリメントしてください。

  シリアル番号がインクリメントされなかった場合、マスターネームサーバーは、正しく新しい情報を得ることができるかもしれませんが、スレーブネームサーバーにはその変更は通知されず、そのゾーンのデータをリフレッシュしようとします。

- `/etc/named.conf`ファイルでは、大かっことセミコロンは必ず正しく使ってください。

  セミコロンが省略されていたり、大かっこ部分が閉じていなかったりした場合、`named`は起動を拒否します。

- 忘れずにすべての*FQDN*の後のゾーンファイルにピリオド（`.`）を付け、ホスト名ではピリオドを省略してください。

  ドメイン名の後のピリオドは完全修飾ドメイン名を示します。ピリオドを省略すると、`named`はそのゾーンの名前か、`$ORIGIN`値を名前の後ろに付けてこれを完成させます。

- *named*から他のネームサーバーへのファイアウォールが接続をブロックしている場合、この設定ファイルを編集します。

デフォルトで、BINDバージョン9は1024以上のランダムポートを使用して、他のネームサーバーにクエリを出します。しかし、ファイアフォールの中にはすべてのネームサーバーがポート53を使用して通信することを期待するものもあります。この場合、以下に示す行を/etc/named.confのoptionsステートメントに追加することにより、namedが強制的にポート53を使用するようにできます：

```
query-source address * port 53;
```

12.7. その他のリソース

以下の情報源は、BINDに関する追加情報を提供します。

12.7.1. インストールされているドキュメント

- BINDには、さまざまなトピックを扱うあらゆる範囲のドキュメントがインストールされており、それぞれがその対象ディレクトリに置かれています。
  - /usr/share/doc/bind-<*version-number*>/ — 最新の機能の一覧を含むREADMEファイルを含んでいます。
  - /usr/share/doc/bind-<*version-number*>/arm/ — *BIND9*管理者リファレンスマニュアルのHTMLとSGMLが含まれています。この中には、BINDリソース要件、異なったタイプのネームサーバーを設定する方法、ロードバランシングの実行方法などの高度なトピックについて記載されています。ほとんどの新BINDユーザーにとって、最初に読むのにもっとも適した場所です。
  - /usr/share/doc/bind-<*version-number*>/draft/ — DNSサービスとDNSサービスを扱う方法に関連したさまざまな技術ドキュメントが含まれています。
  - /usr/share/doc/bind-<*version-number*>/misc —特定の高度な問題を扱うよう設計されたドキュメントが含まれています。BINDバージョン8のユーザーは、migrationドキュメントを参照して、BINDバージョン9に移行する際に実行しなくてはならない変更を調べる必要があります。optionsファイルには、/etc/named.confで使用するBIND9に実装されているオプションがすべて一覧表示されています。
  - /usr/share/doc/bind-<*version-number*>/rfc/ — BINDに関連するすべてのRFCドキュメントは、このディレクトリに置かれています。

- man named — BINDネームサーバーデーモンを制御するのに使用されるさまざまな引数を検索出来ます。
- man named.conf — named設定ファイルの中で利用できるオプションの総括的な一覧です。
- man rndc — BINDネームサーバーを制御する為のrndcを使用する時に利用できる異なるオプションを説明しています。
- man rndc.conf — rndc設定ファイル内で利用できるオプションの総括的な一覧です。

12.7.2. 役に立つWebサイト

- http://www.isc.org/products/BIND — BINDプロジェクトのホームページ。現在のリリースについての情報と*BIND9*管理者リファレンスマニュアルのPDFバージョンを含んでいます。
- http://www.redhat.com/mirrors/LDP/HOWTO/DNS-HOWTO.html —解決用キャッシング用ネームサーバーとしてのBINDの使用方法と、ドメインのためのプライマリネームサーバーとして機能させるのに必要なさまざまなゾーンファイルの設定が載っています。

12.7.3. 関連書籍

- *DNS and BIND*（Paul Albitz and Cricket Liu著、O'Reilly & Associates刊）—共通BINDオプションと上級BINDオプションの両方を説明し、DNSサーバーを保証する戦略を提供する人気のあるリファレンス。
- *The Concise Guide to DNS and BIND*（Nicolai Langfeldt著、Que刊）—複数のネットワークサービスとBINDの接続についてタスク指向の技術トピックに重点を置いて述べたもの。

第13章

LDAP（Lightweight Directory Access Protocol）

LDAP(*Lightweight Directory Access Protocol*) とは、ネットワーク上の中枢にある保存情報にアクセスするための一連のオープンプロトコルです。ディレクトリ共有用に*X.500* 基準をベースとしていますが、複雑ではなくリソース集中型です。この為、LDAPは時には"*X.500 Lite*"とも呼ばれます。

X.500の様にLDAPは、ディレクトリを使用して情報を階級式に構成します。これらのディレクトリは各種の情報を保存でき、さらにはネットワーク情報サービス(NIS)と似た方法で使用することができます。LDAPが有効になっているネットワーク上では誰でも、どのマシンからも自分のアカウントにアクセスできます。

しかし多くの場合、LDAPは単に仮想電話帳として使用され、ユーザーは他のユーザーの連絡情報に簡単にアクセスすることが出来ます。ただし、LDAPは普通の電話帳よりももっと柔軟性があり、世界中の他のLDAPサーバーへ参照できることから、随意の世界情報レポジットリを提供します。現在、一般的には大学、政府の各部門、民間企業などの個々の組織内で多く使用されています。

LDAPはクライアント/サーバー型のシステムです。サーバーは、ディレクトリを保存するのに各種のデータベースを使用して、それぞれが迅速で大量の読み込み操作の為に効率化してあります。LDAPのクライアントアプリケーションがLDAPサーバーにアクセスする時は、ディレクトリに問い合わせするか、あるいはそれを変更しようとします。問い合わせの場合は、サーバーはそれに答えるか、又はローカルで回答出来ない場合、サーバーはその問い合わせの回答を持つLDAPサーバーへ案内します。クライアントアプリケーションがLDAPディレクトリの情報を変更しようとしている場合は、サーバーはそのユーザーが変更する権限を持っているかどうかを検証してから情報の追加なり更新なりをします。

本章では、LDAPv2 及びLDAPv3 プロトコルのオープンソース実装であるOpenLDAP 2.0の設定とその使用法を参照します。

13.1. LDAPの使用理由

LDAPを使用することの主要なメリットは、全組織内の情報を中央のレポジットリに統合できることです。例えば、組織内のそれぞれのグループのユーザー一覧を管理するのではなく、LDAPをネットワーク上のどこからでもアクセスできる中央ディレクトリとして使用します。その上、LDAPはSecure Sockets Layer (SSL)とTransport Layer Security (TLS)の両方をサポートしますので、機密データを外部の侵入から保護することが出来ます。

LDAPはまた、ディレクトリを収納するバックエンドデータベースを数多くサポートします。これにより、管理者はサーバーが分配する情報のタイプに最も適したデータベースを起用できる柔軟性を持つことになります。LDAPはまた、適切に定義されたクライアントアプリケーションプログラミングインターフェイス(API)を持つ為、LDAP対応のアプリケーションの数は多く、更にはその質と量も上昇中です。

短所としては、LDAPは設定が難しいことが挙げられます。

13.1.1. OpenLDAP 2.0 の機能強化

OpenLDAP 2.0は数多くの重要な機能を含んでいます。

- *LDAPv3* サポート — OpenLDAP 2.0は、他の改良と共にSimple Authentication and Security Layer (SASL)、Transport Layer Security (TLS), 及びSecure Sockets Layer (SSL)をサポートします。LDAPv2以後、プロトコル内の多くの変更は、LDAPにより高度なセキュリティを与えるように設計されています。

- *IPv6* サポート — OpenLDAP は次世代のインターネットプロトコルのバージョン6 に対応しています。
- *IPC*上での*LDAP* — OpenLDAP はインタープロセスコミュニケーション(IPC)を使用するシステム内で通信できます。これで、ネットワーク上で通信する必要がなくなりセキュリティが向上します。
- *C API*の更新 — これによりプログラマーがLDAPディレクトリサーバーに接続して使用する方法が向上します。
- *LDIFv1* サポート — LDAP Data Interchange Format (LDIF)バージョン1 に対して、完全に準拠しています。
- 機能拡張されたスタンドアロン*LDAP*サーバー — アップデートされたアクセス制御システム、スレッドプーリング、より優れたツール、その他の機能拡張が行われています。

13.2. LDAPの用語

LDAPに関する論議には、LDAP特有の用語群の基本的な理解を必要とします：

- エントリ — エントリとは、LDAP ディレクトリ内でのユニット1つのことです。各エントリはその独特の*Distinguished Name(*区別名*) (DN)*で識別されます。
- 属性 — 属性とは、エントリと直接関連した情報です。例えば、ある組織はLDAPエントリとして表示出来ます。組織と関連した属性は、fax番号、その住所、などがあります。LDAPディレクトリでは、人もエントリとして表示できます。人の通常の属性には、その人の電話番号とeメールアドレスが含まれます。

  属性の幾つかは必須で、その他の属性はオプションになります。それぞれのエントリの為にオブジェクトクラスの定義が、どの属性は必須で、どれがそうでないかを設定しています。オブジェクトクラスの定義は、`/etc/openldap/schema/` ディレクトリ内にある各種のスキーマファイルで確認できます。LDAPスキーマに関する詳細は項13.5で御覧下さい。
- *LDIF — LDAP Data Interchange Format* (LDIF) は、LDAPエントリのASCIIテキスト表示です。LDAPサーバーへインポートするデータ用のファイルはLDIF形式でなければなりません。LDIFエントリは以下の例のようになります：

```
[<id>]
dn: <distinguished name>
<attrtype>: <attrvalue>
<attrtype>: <attrvalue>
<attrtype>: <attrvalue>
```

  各エントリは必要な数の<*attrtype*>: <*attrvalue*>ペアを含んでいます。空白の行はエントリの終了を示します。

用心

  <*attrtype*> と <*attrvalue*> のすべてのペアは、この情報の使用に対応するスキーマで定義されなければなりません。

  "<"と">"に囲まれている全ての値は変数であるため、新規のLDAPエントリが生成されるたびに設定できます。しかしながら、この規則は<*id*> には適用されません。<*id*> は、エントリの編集に使用するアプリケーションによって決定される番号です。

注意

LDIFエントリを手動で編集する必要はないはずです。その代わりにLDAPクライアントアプリケーションを使用します。項13.3でその例の一覧を見ることができます。

13.3. OpenLDAPデーモンとユーティリティ

OpenLDAP ライブラリとツールのセットは以下のパッケージに分配されています：

- `openldap` — OpenLDAPサーバーとクライアントアプリケーションを実行するのに必要なライブラリを含んでいます。
- `openldap-clients` — LDAPサーバー上でディレクトリの表示と変更の為のコマンドラインツールを含んでいます。
- `openldap-servers` — LDAPサーバーの設定と実行に必要なサーバーと他のユーティリティを含んでいます。

2種類のサーバーが`openldap-servers`パッケージの中に含まれています：スタンドアロン*LDAP* デーモン (`/usr/sbin/slapd`)とスタンドアロン*LDAP* 更新複製デーモン(`/usr/sbin/slurpd`)です。

`slapd`デーモンはスタンドアロンLDAPサーバーであり、`slurpd`デーモンはネットワーク上の1つのLDAPサーバーから別のLDAPサーバーへの変更を同期するのに使用されます。`slurpd`デーモンは複数のLDAPサーバーを利用している時にのみに使用されます。

管理の作業をするには、`openldap-servers`パッケージが`/usr/sbin/`ディレクトリへ、以下のユーティリティをインストールする必要があります：

- `slapadd` — LDIFファイルからLDAPディレクトリへエントリを追加します。例えば、`/usr/sbin/slapadd -l` ***`ldif-input`***コマンドは新規のエントリを含むLDIFファイル、***`ldif-input`***で読み込みます。
- `slapcat` — デフォルトフォーマット— Berkeley DB — のLDAPディレクトリからエントリを取り出して、LDIFファイルの中にそれを保存します。例えば、`/usr/sbin/slapcat -l`***`ldif-output`*** コマンドは、LDAPディレクトリからのエントリを含む ***`ldif-output`***というLDIFファイルを出力します。
- `slapindex` — 現在のコンテンツの`slapd`ディレクトリベースの索引を再構成します。
- `slappasswd` — `ldapmodify`で使うユーザーパスワードの値か、あるいは`slapd`の設定ファイル、`/etc/openldap/slapd.conf`の中の`rootpw`の値を生成します。パスワードを作成するには、`/usr/sbin/slappasswd`を実行します。

警告

`slapadd`、`slapcat`、`slapindex`を使用する前に、`/usr/sbin/service slapd stop`を発行して必ず`slapd`を停止させてください。そうしないとLDAPディレクトリの一貫性を失う可能性があります。

これらのユーティリティの使用方法の詳細については、それぞれのmanページを参照してください。

`openldap-clients`パッケージは、LDAPディレクトリのエントリを追加、変更、削除するのに使う`/usr/bin/`の中へツールをインストールします。これらのツールには以下の項目が含まれます：

- `ldapmodify` — ファイル又は標準入力からの入力を受け付け、LDAPディレクトリのエントリを変更します。
- `ldapadd` — ファイル又は標準入力からの入力を受け付け、ユーザーのディレクトリのエントリに追加します。`ldapadd`は実際には`ldapmodify -a`へのハードリンクです。
- `ldapsearch`— shellプロンプトを使用してLDAPディレクトリ内のエントリを検索します。
- `ldapdelete` — ターミナルでのユーザー入力又は、ファイルからの入力を受け付け、LDAPディレクトリからエントリを削除します。

`ldapsearch`を例外として、これらのユーティリティはLDAPディレクトリで変更をしたいそれぞれのエントリ用のコマンドをタイプするよりも、変更したいファイルへの参照をする方がずっと簡単に使用できます。そのようなファイルの形式ついては、それぞれのアプリケーションのman ページに概要があります。

13.3.1. NSS, PAM, 及びLDAP

OpenLDAPパッケージに加えて、Red Hat Linux には、LinuxとUNIX環境の中へ統合できるLDAPの能力を強化する`nss_ldap`と呼ばれるパッケージが含まれています。

`nss_ldap`パッケージは、以下のモジュールを提供します：

- `/lib/libnss_ldap-<glibc-version>.so`
- `/lib/security/pam_ldap.so`

`libnss_ldap-<glibc-version>.so`モジュールによりアプリケーションは、glibcの*Nameservice Switch* (NSS)インターフェイスを経由したLDAPディレクトリを使用してユーザー、グループ、ホスト、そしてその他の情報を検索できます。NSSは、アプリケーションがネットワーク情報サービス(NIS)ネームサービスと単層の認証ファイルと合同で、LDAP使用して認証できるようにします。

`pam_ldap`モジュールの使用で、PAM-認識のアプリケーションはLDAPディレクトリ内に保存してある情報を使用してユーザーの認証ができます。PAM-認識のアプリケーションにはコンソールログイン、POPとIMAPのメールサーバー、及びSambaが含まれます。ネットワーク上でLDAPサーバーを起用することにより、これらのアプリケーションのすべては、同じユーザーIDとパスワードの組合せを使用して認証ができるようになり、管理が非常に簡単になります。

13.3.2. PHP4, Apache HTTP サーバー, 及びLDAP

Red Hat Linux には、PHPサーバーサイドスクリプト言語用のLDAPモジュールを収納したパッケージが含まれています。

`php-ldap`パッケージは`/usr/lib/php4/ldap.so` モジュールを経由してPHP4 HTML埋め込型のスクリプト言語へLDAP サポートを追加します。このモジュールによりPHP4スクリプトは、LDAPディレクトリに保存されている情報にアクセスできます。

重要

Red Hat Linux では、`auth_ldap`パッケージの配布を終了しています。このパッケージはApache HTTP サーバーのバージョン1.3とそれ以前用のLDAPサポートを提供していました。このモジュールの状況の詳細は、Apache Software Foundationの以下のwebサイトで確認して下さい。http://www.apache.org/

13.3.3. LDAP クライアントアプリケーション

ディレクトリの作成と変更をサポートするグラフィカルなLDAPクライアントがありますが、Red Hat Linuxと一緒には配送されていません。そのようなアプリケーションの1つが**LDAP Browser/Editor**です。— Javaベースのツールで以下のサイトでオンラインで入手できます。http://www.iit.edu/~gawojar/ldap.

他のLDAPクライアントのほとんどは読み込み専用でディレクトリにアクセスし、そのまま変更なしにそれを参照して組織全体の情報を得ます。そのようなアプリケーションの例としては、Mozillaベースの Webブラウザ、Sendmail、**Balsa**、**Pine**、**Evolution**、**Gnome Meeting**があります。

13.4. OpenLDAP 設定ファイル

OpenLDAP 設定ファイルは/etc/openldap/ディレクトリの中にインストールされています。以下に最も重要なディレクトリとファイルの簡単な一覧を示します：

- /etc/openldap/ldap.conf — これは、ldapsearch, ldapadd, Sendmail, **Pine**, **Balsa**, **Evolution**, **Gnome Meeting**などのOpenLDAPライブラリを使用する、すべての*client*アプリケーションの為の設定ファイルです。
- /etc/openldap/slapd.conf — これはslapdデーモン用の設定ファイルです。このファイルに関する詳細は項13.6.1で御覧下さい。
- /etc/openldap/schema/ ディレクトリ— このサブディレクトリには、slapdデーモンで使用されるスキーマが含まれます。このディレクトリに関する詳細は項13.5 で御覧下さい。

注意

nss_ldap パッケージがインストールされている場合、それは/etc/ldap.confと言う名前のファイルを作成します。このファイルはnss_ldapパッケージが供給するPAM 及びNSSモジュールによって使用されます。この設定ファイルの詳細は項13.7を御覧下さい。

13.5. /etc/openldap/schema/ディレクトリ

/etc/openldap/schema/ディレクトリは、以前にslapd.at.confファイルとslapd.oc.confファイルに収納されていたLDAP定義を保持しています。すべての属性構文定義とオブジェクトクラス定義は現在、別のスキーマファイルに配置されています。各種スキーマファイルは、includeの行を使用して/etc/openldap/slapd.confを参照することが出来ます。以下のようになります：

```
include /etc/openldap/schema/core.schema
include /etc/openldap/schema/cosine.schema
include /etc/openldap/schema/inetorgperson.schema
include /etc/openldap/schema/nis.schema
include /etc/openldap/schema/rfc822-MailMember.schema
include /etc/openldap/schema/autofs.schema
include /etc/openldap/schema/kerberosobject.schema
```

用心

OpenLDAPによりインストールされたスキーマファイル内で定義されている、スキーマの項目を変更しないでください。

OpenLDAPが使用するスキーマを、デフォルトのスキーマファイルを参考にして追加の属性の種類やオブジェクトクラスをサポートするように拡張することができます。このためには、`/etc/openldap/schema`ディレクトリ内に`local.schema`ファイルを作成します。その後デフォルトのスキーマの`include`行の下に次の行を追加して、この新しいスキーマが`slapd.conf`において参照されるようにします：

```
include     /etc/openldap/schema/local.schema
```

次に、`local.schema`ファイルの内部で属性タイプとオブジェクトクラスを定義します。多くの組織では、デフォルトでインストールされているスキーマファイルからの既存の属性タイプを使用し、新規のオブジェクトクラスを`local.schema` ファイルに追加しています。

ある種の特定した要求に合致するようにスキーマを拡張することは、かなり複雑になりこの章の説明範囲を越える内容です。新規スキーマファイルの書き込みに関する情報はhttp://www.openldap.org/doc/admin/schema.html で御覧下さい。

13.6. OpenLDAP 設定の概要

このセクションは、OpenLDAPディレクトリのインストールと設定についての簡潔な概要を提供します。詳細については以下のURLを参照して下さい：

- http://www.openldap.org/doc/admin/quickstart.html — OpenLDAPについてのwebサイト。*Quick-Start Guide*
- http://www.redhat.com/mirrors/LDP/HOWTO/LDAP-HOWTO.html — Linuxドキュメントプロジェクトの*LDAP Linux HOWTO*、Red Hatのwebサイトにミラーバージョンがあります。

LDAPサーバー構築の基本的なステップは次の通りです：

1. `openldap`, `openldap-servers`,及び`openldap-clients` RPMをインストールします。
2. `/etc/openldap/slapd.conf`ファイルを編集して、ユーザーのLDAPドメインとサーバーを参照します。このファイルの編集法に関する詳細は項13.6.1で御覧下さい。
3. 以下のコマンドを使用して`slapd`をスタートします：
   `/sbin/service/ldap start`

   LDAPを正しく設定した後は、`chkconfig`, `ntsysv` 又は**サービス設定ツール**を使用してLDAPをブート時に開始するように設定できます。サービスの設定については、*Red Hat Linux* カスタマイズガイドの中のサービスに対するアクセスの制御の章を参照して下さい。
4. `ldapadd`かスクリプトにより、LDAPディレクトリにエントリを追加します。
5. `ldapsearch`を使用して`slapd`が情報に正しくアクセスしているか確認します。
6. この時点で、LDAPディレクトリは正常に機能しているはずで、LDAPディレクトリを使うLDAPが有効になったアプリケーションを設定することが出来ます。

13.6.1. /etc/openldap/slapd.confの編集

slapd LDAP サーバーを使用するには、その設定ファイル、/etc/openldap/slapd.confを変更する必要があります。このファイルを編集して正しいドメインとサーバーを指定しなければなりません。

suffixの行はLDAPサーバーが情報を提供する、変更すべきドメイン名を示します：

```
suffix    "dc=your-domain,dc=com"
```

それを完全修飾のドメイン名が表示されるようにします。例えば：

```
suffix    "dc=example,dc=com"
```

rootdnエントリは、LDAPディレクトリ上の操作用に設定されたアクセス制御又は管理制限パラメータで制限されていないユーザー用の*Distinguished Name*(区別名) (*DN*)です。rootdnのユーザーは、LDAPディレクトリのrootユーザーとも考えられます。設定ファイルの中で、rootdnの行をデフォルトの値から、次の例の様に変更します：

```
rootdn    "cn=root,dc=example,dc=com"
```

ネットワーク上でLDAPディレクトリを充填する予定の場合、rootpwの行を変更します。— デフォルトの値を暗号化したパスワード文字列で入れ換えます。暗号化したパスワード文字列を生成するには、次のコマンドを入力します：

```
slappasswd
```

パスワードの入力と確認用の再入力を求められます。そうするとプログラムは生成された暗号化パスワードの結果をターミナルに出力します。

次に、新規に作成された暗号化パスワードを>/etc/openldap/slapd.confのrootpwの行にコピーして、「#」サインを削除します。

終了すると、その行は次と同様な形になります：

```
rootpw {SSHA}vv2y+i6V6esazrIv70xSSnNAJE18bb2u
```

警告

/etc/openldap/slapd.confに指定してあるrootpwディレクティブを含むLDAP パスワードは、TLS暗号化を有効にする場合以外は、ネットワーク上に暗号化なしで送信されます。

TLS暗号化を有効にするには、/etc/openldap/slapd.conf内の説明を再確認して、slapd.conf用のmanページを参照して下さい。

セキュリティ強化の為には、LDAPディレクトリを充填した後でrootpw ディレクティブは、その行の先頭に「#」マークを付けてコメントアウト(無効化)します。

LDAPディレクトリを充填するために、ローカルで/usr/sbin/slapadd コマンドラインツールを使用している時は、rootpw ディレクティブは使う必要がありません。

重要

/usr/sbin/slapaddはrootとして操作する必要があります。しかし、ディレクトリサーバーはldapユーザーとして作動します。その為、ディレクトリサーバーはslapaddによって作成されたファイルは何も変更することが出来ません。この問題を修正するにはslapaddを使用した後で、次のコマンドを入力します：

```
chown -R ldap /var/lib/ldap
```

13.7. システムがOpenLDAPの認証を実行するように設定する

このセクションではOpenLDAPを使って認証するようにRed Hat Linuxシステムを設定する方法について簡単に概要を説明します。OpenLDAPのエキスパートである場合は別ですが、本書の説明以外にも詳しいマニュアルが必要になるでしょう。詳細については項13.9を参照してください。

必要な*LDAP*パッケージのインストール

‘ 最初に、LDAPサーバーとLDAPクライアントの両方のマシンに該当するパッケージがインストールされていることを確認する必要があります。LDAPサーバーには`openldap-servers`パッケージが必要です。

LDAPクライアントマシンでは、`openldap`、`openldap-clients`、`nss_ldap`のパッケージをインストールする必要があります。

設定ファイルの編集

‘

- サーバー上では、LDAPサーバーの`/etc/openldap/slapd.conf`ファイルを編集して、確実に組織の特性に一致するようにします。`slapd.conf`の編集については項13.6.1を参照してください。
- クライアントマシン上では、`/etc/ldap.conf`と `/etc/openldap/ldap.conf`の両方が組織の適切なサーバーと検索ベースの情報を含んでいる必要があります。

  この操作の最も簡単な方法は、**認証設定ツール** (`authconfig-gtk`)を実行して、**ユーザー情報** タブの下部で**LDAPサポートを有効にする**を選択することです。

  また、これらのファイルは手動でも編集できます。
- LDAPを使用する為には、クライアントのマシンで、`/etc/nsswitch.conf`を編集する必要があります。

  この操作の最も簡単な方法は、**認証設定ツール** (`authconfig-gtk`)を実行して、**ユーザー情報** タブの下部で**LDAPサポートを有効にする**を選択することです。

  手動で`/etc/nsswitch.conf`を編集する場合、適当な行に`ldap`を追加します。

  例えば次のようにします：

```
passwd: files ldap
shadow: files ldap
group: files ldap
```

13.7.1. PAMとLDAP

認証の為に標準のPAMが有効になったアプリケーションでLDAPを使用するには**認証設定ツール**(`authconfig-gtk`)を実行して**認証**タブの中の**LDAPサポートを有効にする**を選択します。PAMの設定に関する詳細は、第14章とPAMのmanページを御覧下さい。

13.7.2. 古い認証情報をLDAPフォーマットへ移行

/usr/share/openldap/migrationディレクトリには、認証情報をLDAPフォーマットに移行するための一連のシェルとPerlスクリプトが含まれています

最初にmigrate_common.phファイルを修正し、ドメインを反映させる必要があります。デフォルトのDNSドメインは、そのデフォルトの値から次のように変更する必要があります：

```
$DEFAULT_MAIL_DOMAIN = "your_company";
```

デフォルトのベースも次のように変更する必要があります：

```
$DEFAULT_BASE = "dc=your_company,dc=com";
```

ユーザーデータベースを、1つの読み込み可能なLDAPフォーマットに移行する作業は、同じディレクトリにインストールしてある一連の移行スクリプトの仕事です。表13-1を使用して、ユーザーデータベースを移行する為に実行するスクリプトを決定して下さい。

既存のネームサービス	LDAPは動作しているか	使用するスクリプト
/etc単層ファイル	はい	migrate_all_online.sh
/etc単層ファイル	いいえ	migrate_all_offline.sh
NetInfo	はい	migrate_all_netinfo_online.sh
NetInfo	いいえ	migrate_all_netinfo_offline.sh
NIS (YP)	はい	migrate_all_nis_online.sh
NIS (YP)	いいえ	migrate_all_nis_offline.sh

表13-1. LDAP移行スクリプト

既存のネームサービスに適応するスクリプトを実行します。

注意

これらのスクリプトを使用するには、システム内にPerlをインストールしている必要があります。

/usr/share/openldap/migration/ディレクトリ内のREADME及びmigration-tools.txtファイルは移行の方法に関する詳細情報を提供します。

13.8. OpenLDAP バージョン2.0へのアップグレード

OpenLDAP バージョン2.0では、slapdLDAPサーバーによって使用されるオンディスク保存形式が変更になりました。LDAPをRed Hat Linux 7.0又は以前のバージョンからアップグレードしている場合は、既存のLDAPディレクトリを、次のコマンドを使用してLDIFファイルへ入れ込む必要があります：

```
ldbmcat -n > <ldif_file>
```

上記のコマンドでは、<*ldif_file*>を出力ファイルの名前に変更します。次に、以下のコマンドを入力してこのファイルをOpenLDAP2.0へインポートします：

```
slapadd -l <ldif_file>
```

重要

rootになって/usr/sbin/slapaddを使用します。しかし、ディレクトリサーバーはldapユーザーとして動作します。そのため、ディレクトリサーバーはslapaddで作成されたファイルはどれも変更できません。この問題を修正するにはslapaddを終了した後に次のコマンドを実行します：

```
chown -R ldap /var/lib/ldap
```

13.9. その他のリソース

LDAPに関する情報は沢山あります。システムでLDAPの設定を開始する前に、これらのリソース、特にOpenLDAPのWebサイトとLDAP Linux HOWTOを見直してください。

13.9.1. インストールされているドキュメント

- LDAP man ページ— ldapのmanページは、LDAPについて最初に学習するのに最適な資料です。また、各種のLDAPデーモンやユーティリティのmanページもあります。
- /usr/share/docs/openldap-<*versionnumber*> — 一般的なREADMEと雑多な情報が含まれています。

13.9.2. 役に立つWebサイト

- http://www.openldap.org/ — OpenLDAP プロジェクトの本部です。このwebサイトにはOpenLDAPに設定に関する豊富な情報が含まれています。
- http://www.redhat.com/mirrors/LDP/HOWTO/LDAP-HOWTO.html — 古くてもまだ価値のあるLDAP HOWTOです。
- http://www.padl.com/ — 各種の便利なLDAPツールと共にnss_ldapとpam_ldapに開発をしている企業です。
- http://www.kingsmountain.com/ldapRoadmap.shtml—Jeff Hodges' LDAP Road Mapには、LDAPプロトコルに関するいくつかの便利なFAQや最新ニュースへのリンクがあります。
- http://www.webtechniques.com/archives/2000/05/wilcox—LDAPによるグループ管理に関する便利な情報があります。
- http://www.ldapman.org/articles — ディレクトリツリーの設計やディレクトリ構造のカスタマイズの方法を含む、LDAPの優れた入門書となる記載があります。

13.9.3. 関連書籍

- *Implementing LDAP*（Mark Wilcox著、Wrox Press, Inc.刊）
- *Understanding and Deploying LDAP Directory Services*、Tim Howesほか著、Macmillan Technical Publishing

III. セキュリティへの参照

安全なプロトコルを使用することがシステム管理統合の重要な部分となります。このセクションでは、ユーザー認証、ネットワークアクセス制御、安全なネットワーク通信、侵入感知などに使用される重要な役目のツールについて説明しています。Red Hat Linux システムの安全性に関する詳細情報については、*Red Hat Linux* セキュリティガイドを参照して下さい。

目次

第14章

PAM（Pluggable Authentication Modules）

ユーザーにシステムへのアクセス権を与えるプログラムは、認証というプロセスを通じてユーザーに身元を確認します。歴史的にそのようなプログラムのそれぞれは認証の役割を達成するのに独自の手段を持っていました。Red Hat Linuxでは、多くのそのようなプログラムを*Pluggable Authentication Modules* (*PAM*)という中央化された認証プロセスで使用するように設定されています。

PAMのプラグイン可能なモジュラー構造を用いることによって、システム管理者は担当のシステム用の認証ポリシー設定に多大の柔軟性を持つことが出来ます。

ほとんどの場合、PAM認識アプリケーションのためにPAM設定ファイルを変更する必要はありません。但し、場合によってはPAM設定ファイルを編集する必要がでることがあります。PAMの設定が間違っているとシステムセキュリティへの侵害につながりますので、変更をする前に、これらのファイルの構造を理解することが重要になります。(詳細は項14.3を参照して下さい)。

14.1. PAMの利点

PAMは以下のような利点を提供します：

- 各種アプリケーションで使用できる共通の認証設定を提供します。
- システム管理者とアプリケーション開発者に、認証に対する優れた柔軟性と制御性を与えます。
- アプリケーション開発者は、プログラムを開発する際に独自の認証設定を作成する必要がありません。

14.2. PAM設定ファイル

`/etc/pam.d/`ディレクトリには、PAM認識アプリケーションのための、PAM設定ファイルがあります。PAMの初期のバージョンでは、`/etc/pam.conf`ファイルが使用されていました。しかし、現在このファイルはあまり重要視されません。`/etc/pam.d/`ディレクトリが存在しない場合のみ、`pam.conf` ファイルが読み込まれます。

14.2.1. PAM サービスファイル

各PAM認識プログラム、すなわち「サービス」は`/etc/pam.d/`ディレクトリにファイルを持ちます。これらの各ファイルは、その制御するアクセスに応じてサービスの名前がついています。

サービス名の定義と、PAM設定ファイルを`/etc/pam.d/`ディレクトリにインストールすることはPAM認識プログラムに依存します。例えば、`login`プログラムは、そのサービス名を`/etc/pam.d/login`と定義しています。

14.3. PAM設定ファイルの形式

各PAM設定ファイルには、次のようなフォーマットされたディレクティブのグループが含まれています：

<module interface> *<control flag>* *<module path>* *<module arguments>*

これらのコンポーネントはそれぞれ次のセクションで説明してあります。

14.3.1. モジュールインターフェイス

4つのタイプのPAMモジュールインターフェイスがあり、それぞれ認証プロセスの異なる側面に関連しています：

- `auth` — これらのモジュールは、例えばパスワードの要求とチェックを用いて、ユーザー認証を行います。このインターフェイスのモジュールは、グループメンバーシップや、Kerberosチケットなどの証明書も設定できます。
- `account` — これらのモジュールはアクセスが許可されることをチェックします。例えば、ユーザーのアカウントが期限切れでないか、あるいはユーザーがその時刻のログインを認められているか、をチェックします。
- `password` — これらのモジュールはパスワードの設定と確認に使用されます。
- `session` — これらのモジュールはユーザーセッションを設定して管理します。このインターフェイスのモジュールは、ユーザーのホームディレクトリのマウントやユーザーのメールボックスを使用可能にするなどアクセスの許可に必要な追加のタスクを実行することが出来ます。

注意

個々のモジュールは上記の全てのモジュールインターフェイスあるいはそのいずれかを提供することができます。例えば、`pam_unix.so`は4つのインターフェイスをすべて提供できます。

PAM設定ファイルでは、モジュールインターフェイスは最初に定義されるフィールドです。例えば、設定の標準的な行は次のようになります：

```
auth   required /lib/security/pam_unix.so
```

これはPAMに対して`pam_unix.so`モジュールの`auth`インターフェイスを使用するように指示しています。

14.3.1.1. スタックモジュール

モジュールインターフェイスディレクティブは、スタックされる、つまり次々に置き換えられるので1つの目的の為に複数のモジュールが一緒に使用できます。従って、認証プロセスにおいて、モジュールのスタック順は非常に重要です。

スタックにより、管理者がユーザーに認証を与える前に存在すべき特定の条件を要求することが容易になります。例えば、`rlogin`は、以下のPAM設定ファイルで示してあるように、通常、5つのスタックされた`auth`モジュールを使用します：

```
auth   required  /lib/security/pam_nologin.so
auth   required  /lib/security/pam_securetty.so
auth   required  /lib/security/pam_env.so
auth   sufficient /lib/security/pam_rhosts_auth.so
auth   required  /lib/security/pam_stack.so service=system-auth
```

`rlogin`の使用が認可される前に、PAMは`/etc/nologin`が存在しないこと、暗号化されていないネットワーク接続を通じてrootとしてリモートからログインしようとしていないこと、どんな環境変数でもロードできることを確認します。それがすべて適切なら`rhosts`の認証が実行され、その接続が許可されます。`rhosts`の認証が失敗した場合は、通常のパスワード認証が実行されます。

14.3.2. 制御フラグ

すべてのPAMモジュールは、コールがあると成功又は失敗の結果を生成します。制御フラグは、その結果に対する処理方法をPAMに提供します。モジュールは特定の順序でスタックされるので、制御フラグは、ユーザーにそのサービスへの認証をすることの全体的目的に対して、この特定のモジュールの成功あるいは失敗の重要度を判定します。

4つのタイプの制御フラグが定義されています：

- `required` — 認可が続行するにはモジュールの結果は成功でなければいけません。`required`モジュールが失敗の結果を出すと、インターフェイスを参照している全てのモジュールが完了するまで、ユーザーは結果報告を受け取りません。
- `requisite` — 認可が続行するにはモジュールの結果は成功でなければいけません。但し、`requisite`モジュールの結果が失敗した場合、ユーザーは直ちに最初に失敗した`required`あるいは`requisite`モジュールに関するメッセージを受け取ります。
- `sufficient` — チェックが失敗した場合、無視されるモジュールです。ただし、`sufficient`フラグモジュールのチェックが成功し、そして その上部の`required`フラグモジュールがすべて成功した場合、このタイプのほかのモジュールはチェックされず、ユーザーは認証されます。
- `optional` — チェックが失敗した場合、無視されるモジュールです。チェックに成功しても、結果はこのモジュールインターフェイスの成功や失敗に大した役割はしません。`optional`フラグモジュールが認証の成功に必要になるのは、そのインターフェイスを参照する他のモジュールがない時です。この場合、`optional` モジュールが、そのインターフェイス用の全体のPAM認証を決定します。

重要

`required`モジュールがコールされる順序は重要ではありません。`sufficient`と`requisite`の制御フラグではその順序が重要になります。

より正確な制御の為の新しい制御フラグの構文が現在PAM用に利用可能です。この新しい構文に関する詳細は`/usr/share/doc/pam-<version-number>/` ディレクトリ内にあるPAMドキュメントをお読み下さい。(<*version-number*>はPAMのバージョン番号です)。

14.3.3. モジュールパス

モジュールパスによって、指定されたモジュールインターフェイスとともに使用するプラグ可能なモジュールの場所がPAMに知らされます。通常、`/lib/security/pam_stack.so`のようにモジュールへのフルパスが示されます。ただし、フルパスが提示されない場合、表示されたモジュールは、`/lib/security/` ディレクトリ(PAMモジュールのデフォルトの位置)にあると判断されます。

14.3.4. モジュールの引数

PAMは、幾つかのモジュール用に認証をしている間、プラグ可能なモジュールへ情報を渡すために引数を使用します。

たとえば、`pam_userdb.so`モジュールは、Berkeley DBファイルに保存された秘密鍵を使ってユーザーを認証します。Berkeley DBは、多くのアプリケーションに組み込まれているオープンソースのデータベースシステムです。モジュールが`db`引数を使用することによってBerkeley DBは要求されたサービスに使用するデータベースを判断できます。

PAM設定ファイルの中の標準的な`pam_userdb.so`の行は、以下のようになります：

```
auth   required /lib/security/pam_userdb.so db=<path-to-file>
```

直前の例では、<*path-to-file*>をBerkeley DBデータベースファイルの完全パス名で入れ換えます。

無効な引数は無視され、PAMモジュールの成功あるいは失敗のどちらにも影響しません。但し、殆どのモジュールではエラーが/var/log/messagesファイルに表示されます。

14.4. PAM設定ファイルのサンプル

以下に、PAMアプリケーションの設定ファイルのサンプルを示します。

```
#%PAM-1.0
auth   required /lib/security/pam_securetty.so
auth   required /lib/security/pam_unix.so shadow nullok
auth   required /lib/security/pam_nologin.so
account  required /lib/security/pam_unix.so
password required /lib/security/pam_cracklib.so retry=3
password required /lib/security/pam_unix.so shadow nullok use_authtok
session  required /lib/security/pam_unix.so
```

最初の行は、行頭に「#」マークが付加されているコメントです。

2行目から4行目ではログイン認証のモジュールを3つスタックしています。

```
auth   required /lib/security/pam_securetty.so
```

このモジュールはもしユーザーがrootとしてログインを試行し、さらに、/etc/securettyファイルが存在する場合、ユーザーがログインしようとしているttyがこのファイルに一覧表示されていることを確認します。

```
auth   required /lib/security/pam_unix.so shadow nullok
```

このモジュールはユーザーにパスワードを要求して、/etc/passwdに保存されている情報を使用してそのパスワードをチェックします。パスワードが存在する場合、/etc/shadowをチェックします。pam_unix.soモジュールは自動的にシャドウパスワードを検出して使用し、ユーザーの認証をします。シャドウパスワードに関する詳細は項6.5で御覧下さい。

引数nullokは、pam_unix.soモジュールに対し、空白のパスワードを許可するように指示します。

```
auth   required /lib/security/pam_nologin.so
```

これが最終認証ステップです。/etc/nologinファイルが存在するかどうかを確認します。nologinが存在し、ユーザーがルートでない場合、認証は失敗します。

注意

この例では、最初のauthモジュールが失敗しても、3つのauthモジュールすべてがチェックされます。これはユーザーに、認証のどの段階で拒否されたかを悟られないようにするためです。そのような情報をアタッカーに渡す事は、彼らにシステムをクラックする方法をたやすく類推する事を許します。

```
account  required /lib/security/pam_unix.so
```

このモジュールで、必要なアカウントの確証が実行されます。たとえば、シャドウパスワードが有効な場合、pam_unix.soモジュールのアカウントコンポーネントは、アカウントの期限が切れていないか、ユーザーがパスワード猶予期間内にパスワードを変更していないかをチェックします。

```
password required /lib/security/pam_cracklib.so retry=3
```

パスワードの期限が切れている場合、pam_cracklib.soモジュールのパスワードコンポーネントは新しいパスワードの要求をします。それから、新規に作成されたパスワードに対してテストを実行することにより、それがパスワードに対する辞書型攻撃プログラムによって簡単に判明するものでないことを確認します。最初にこのテストに失敗した場合、retry=3引数に従って、あと２回、強力なパスワードを作る機会があります。

```
password required /lib/security/pam_unix.so shadow nullok use_authtok
```

この行では、プログラムがユーザーのパスワードを変更する場合、pam_unix.soモジュールのpasswordコンポーネントを使ってその変更を行わなければならないことを指定します。これはパスワードを変更しなければならないとpam_unix.soモジュールのauth部が判断した場合のみ行われます。

引数shadowはユーザーのパスワードが更新される時、シャドーパスワードを作るようにモジュールに指示します。

引数nullokは、モジュールにユーザーがパスワードをブランクから変更するのを許可するように指示します。さもなければ、ブランクのパスワードは固定アカウントとして取り扱われます。

この行の最後の引数、use_authtokはPAMモジュールのスタック順序の良い例を示しています。この引数は、モジュールに対しユーザーの新しいパスワードを求めないように伝えます。その代わりに、それ以前のパスワードモジュールで承認されたいかなるパスワードも受け入れます。この方法では、全ての新しいパスワードが、受け入れられる前にセキュアなパスワードのpam.cracklib.soテストをパスしなければいけません。

```
session required /lib/security/pam_unix.so
```

最後の行は、pam_unix.soモジュールのセッションコンポーネントを使用してセッションを管理することを指定しています。このモジュールは、各セッションの始めと終りで、/var/log/messagesにユーザー名とサービスタイプのログを残します。他の機能が必要な場合は、他のセッションモジュールにスタックする事で、補充できます。

次の設定ファイルの例は、rloginプログラム用のauthモジュールスタックを表しています。

```
#%PAM-1.0
auth   required  /lib/security/pam_nologin.so
auth   required  /lib/security/pam_securetty.so
auth   required  /lib/security/pam_env.so
auth   sufficient /lib/security/pam_rhosts_auth.so
auth   required  /lib/security/pam_stack.so service=system-auth
```

最初にpam_nologin.soが/etc/nologinの存在を知るためにチェックをします。存在する場合、root以外は誰もログイン出来ません。

```
auth   required  /lib/security/pam_securetty.so
```

pam_securetty.soは、安全ではないターミナルからrootのログインが行われないようにします。これで、rootによるrlogin試行のすべてが、アプリケーションの限定的なセキュリティガードの理由で、効果的に拒否されます。

ヒント

rootユーザーとしてリモートログインするには、代わりにOpenSSHを使用します。SSHプロトコルの詳細に関しては第18章を御覧下さい。

```
auth   required  /lib/security/pam_env.so
```

この行はpam_env.soモジュールをロードして、それが/etc/security/pam_env.confに指定してある環境変数を設定します。

```
auth   sufficient /lib/security/pam_rhosts_auth.so
```

pam_rhosts_auth.soモジュールは、ユーザーのホームディレクトリにある.rhostsを使用してユーザーを認証します。これが成功すると、PAMは即座に認証の成功を認識します。pam_rhosts_auth.soがユーザーの認証に失敗すると認証の試行は無視されます。

```
auth   required  /lib/security/pam_stack.so service=system-auth
```

pam_rhosts_auth.soモジュールがユーザーの認証に失敗すると、pam_stack.soが通常のパスワード認証を実行します。

引数service=system-authは、ユーザーが/etc/pam.d/system-authにあるシステム認証のPAM設定を経由してパスする必要があることを示します。

ヒント

securettyの結果が失敗になった時、PAMがパスワードを要求するのを防ぐ為に、pam_securetty.soモジュールをrequiredからrequisiteへ変更して下さい。

14.5. PAMモジュールの作成

新しいPAMモジュールは、PAM-認識アプリケーションが使用する為に何時でも追加できます。例えば、開発者が1回制限のパスワード作成法を想像して、そのサポートの為にPAMモジュールを書いた場合、PAM-認識アプリケーションは、リコンパイルや他の修正なしに直ちにその新しいモジュールとパスワードの方法を使用できます。これにより、開発者とシステム管理者は、ミックスアンドマッチするだけでなく、リコンパイルなしに別のプログラム用に認証法をテストすることができます。

モジュールを書くことに関するドキュメントはシステムと一緒に/usr/share/doc/pam-<*version-number*>/ディレクトリに含まれています。(この<*version-number*>は、PAMのバージョン番号です)。

14.6. PAMおよびデバイスの所有権

Red Hat Linuxによって、マシンの物理的コンソール上にログインする最初のユーザーに対し、デバイスの操作や特殊タスクの実行など通常はrootユーザー用に保存してある能力を許可します。これはpam_console.soと呼ばれる、PAM モジュールによって制御されています。

14.6.1. デバイス所有権

Red Hat Linuxのマシンにユーザーがログインすると、pam_console.soモジュールがloginまたはグラフィカルログインプログラム**gdm**と**kdm**にコールされます。もし、このユーザーが物理的なコンソール—*console user*と呼ばれます— へログインする最初のユーザーなら、モジュールは通常ルートに所有される様々なデバイスの所有権をそのユーザーが持つとみなします。コンソールユーザーの最後のローカルセッションが終了するまで、そのユーザーはこれらデバイスの所有権を持ちます。一度ユーザーがログアウトすると、デバイスの所有権はrootユーザーに戻ります。

これに影響され、しかしそれだけに制限されていない状態のデバイスにはサウンドカード、フロッピードライブ、CD-ROMドライブです。

これは、ローカルユーザーにルートへの変更をせずに、これらデバイスの操作を許可します。このように、コンソールユーザーのための一般的なタスクを単純化しています。

/etc/security/console.permsファイルを修正することにより、管理者はpam_console.soによって制御されるデバイスの一覧を編集できます。

14.6.2. アプリケーションアクセス

コンソールユーザーはさらに/etc/security/console.apps/ディレクトリ内のコマンド名の付くファイルと共に特定のプログラムにアクセスを許可されます。

コンソールユーザーがアクセスする重要なアプリケーショングループの一つは、システムを終了または再起動するプログラムです。以下に示します。

- /sbin/halt
- /sbin/reboot
- /sbin/poweroff

これらはPAM-認識アプリケーションなので、使用要求としてpam_console.so モジュールをコールします。

詳細な情報はpam_console、console.perms、console.apps、及びuserhelperのmanページを参照して下さい。

14.7. その他のリソース

以下に、PAMの使用と設定法に関するリソースの一覧を示します。これらのリソースの他に、PAMの設定ファイルがどの様に構成されているかを理解する為にシステム上のPAMの設定ファイルを読んで下さい。

14.7.1. インストールされているドキュメント

- man pam — PAMについての適切な導入知識。PAM設定ファイルの構造と目的が含まれます。
- /usr/share/doc/pam-<*version-number*> — これには*System Administrators'Guide*、*Module Writers' Manual*、*Application Developers' Manual*、及び、PAM標準のコピー、DCE-RFC 86.0が含まれています。

14.7.2. 役に立つWebサイト

- http://www.kernel.org/pub/linux/libs/pam/ — Linux-PAMプロジェクトのための初期配布Webサイト。さまざまなPAMモジュール、FAQ、追加のPAMマニュアルの情報が含まれます。

第15章

TCPラッパーとxinetd

ネットワークサービスへのアクセスの制御は、サーバー管理者にとって最も重要なセキュリティタスクです。幸いRed Hat Linuxには、ぴったりの仕事をするツールが数多くあります。例えば、iptablesベースのファイアウォールは、カーネルのネットワークスタック内で、歓迎できないネットワークパケットをフィルタにかけます。これを運用するネットワークサービスのために、*TCP*ラッパーが、どのホストが"ラップした" ネットワークに接続を許可されるか、されないかを定義することにより、追加の保護層を加えます。ラップしたネットワークサービスの1つは、xinetdスーパーサーバーです。このサービスがスーパーサーバーを呼ばれる理由は、それがネットワークサービスのサブセットへの接続を制御して、アクセス制御をより厳密にするからです。

図15-1は、これらのツールがどのようにしてネットワークサービスを一緒に保護するかの基本的な描写をしています。

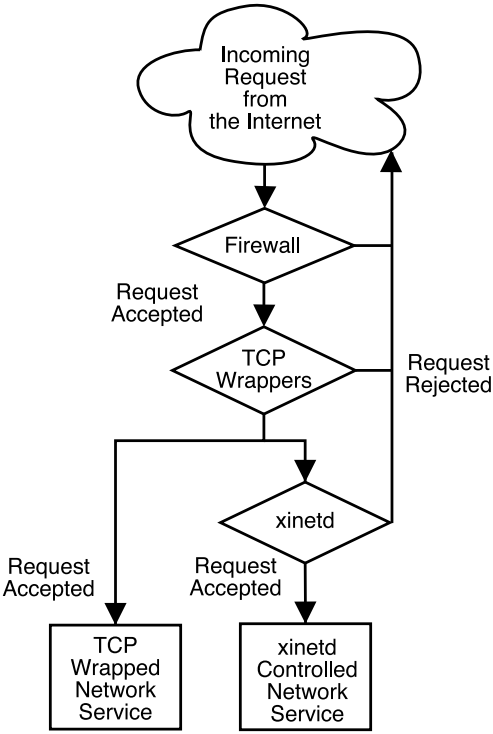

図15-1. ネットワークサービスへのアクセス制御

この章では、ネットワークサービスへのアクセス制御に於けるTCPラッパーの役割とxinetdに焦点を置いています。そして、ロギングと運用管理の両方を強化するために、これらのツールがどの様に使用されるかを説明していきます。iptablesでのファイアウォールについての論議には第16章を参照してください。

15.1. TCPラッパー

TCPラッパーパッケージは(tcp_wrappers)Red Hat Linuxにデフォルトでインストールされており、ネットワークサービスに対しホストベースのアクセス制御を提供します。パッケージ内で最も重要なコンポーネントは/usr/lib/libwrap.a ライブラリです。一般的な表現では、TCPでラップしたサービスは、libwrap.a ライブラリに対してコンパイルされたものを指します。

TCPラップのサービスに接続の試行が有った場合、そのサービスは最初にホストアクセスファイル(/etc/hosts.allow 及び/etc/hosts.deny)へ参照して、クライアントホストが接続を許可されるかどうかを決定します。その後、syslogデーモン(syslogd)を使用して/var/log/secure又は/var/log/messagesへその要求しているホスト名と要求サービスを書き込みます。

もし、クライアントホストが接続を許可された場合、TCPラッパーは要求されたサービスへの接続制御を開放し、それ以上クライアントホストとサーバーとの間の通信を邪魔しません。

アクセスの制御とロギングに加えて、TCPラッパーは先ずクライアントと折衝をするためのコマンドを起動して、それから要求されたネットワークサービスへの接続を拒否するか、制御の開放をします。

TCPラッパーは、すべてのサーバー管理者の強力なセキュリティツールへの価値のある追加となりますので、Red Hat Linux内の殆どのネットワークサービスはlibwrap.a ライブラリに対してリンクされています。そのようなアプリケーションには、/usr/sbin/sshd、/usr/sbin/sendmail、そして/usr/sbin/xinetdが含まれます。

注意

ネットワークサービスバイナリがlibwrap.aに対してリンクされているかどうか判定するには、rootユーザーとして以下のコマンドを入力します：

```
strings -f <binary-name> | grep hosts_access
```

<*binary-name*>は、ネットワークサービスバイナリの名前で入れ換えます。

15.1.1. TCPラッパーの利点

TCPラッパーは、他のネットワークサービス制御技術と比較して以下のような利点を持っています：

- クライアントホストとラップしたネットワークサービス間の透視度 — 接続しようとしているクライアントとラップしたネットワークサービスからはTCPラッパーが使用されているかどうか知ることができません。正式なユーザーは要求したサービスへログインして接続しますが、禁止されたクライアントからの接続は失敗します。
- 複数プロトコルの中央管理 — TCPラッパーは、それが保護するネットワークサービスからは別に稼働するため、多くのサーバーアプリケーションは簡素な管理用設定ファイルの共通セットを共有することが出来ます。

15.2. TCPラッパーの設定ファイル

サービスにクライアントマシンが接続を許可されるかどうか決定するために、TCPラッパーは次の2つのファイルを参照します。これは一般的にホストアクセスファイルと呼ばれるものです：

- /etc/hosts.allow
- /etc/hosts.deny

クライアントの要求がTCPラップしたサービスで受け付けられた時、次のような基本ステップが取られます：

1. サービスが*/etc/hosts.allow*を参照する。 — TCPラップしたサービスは連続的に/etc/hosts.allowファイルを構文解析してそのサービスで指定されている最初の規則を適用します。対応する規則があれば、接続を許可します。そうでなければ、それは第2ステップへと移動します。
2. サービスが*/etc/hosts.deny*を参照する。 — TCPラップしたサービスは連続的に/etc/hosts.denyファイルを構文解析します。対応する規則があれば、接続は拒否されます。そうでなければ、そのサービスへのアクセスは認可されます。

TCPラッパーを使用してネットワークサービスを保護する場合、次の重要な点を考慮する必要があります：

- hosts.allow内のアクセスの規則が最初に適用される為、これらの規則はhosts.deny内に指定してある規則より優先されます。そのため、hosts.allowでサービスへのアクセスが許可された場合、hosts.denyの同じサービスに対するアクセス拒否の規則は無視されます。
- 各ファイル内の規則が上から下に読まれ、該当するサービスに対して最初の適応する規則が1つだけ採用されますので、規則の配置順序は非常に重要です。
- どちらのファイルにもそのサービス用の規則がない場合、あるいはどちらのファイルも存在しない場合、そのサービスへのアクセスは承認されます。
- TCPラップのサービスは、ホストのアクセスファイル規則をキャッシュしません。そのため、hosts.allowまたはhosts.denyへの変更は、ネットワークサービスを再開始しなくても直ちに反映されます。

15.2.1. アクセス規則のフォーマット

/etc/hosts.allowと/etc/hosts.denyのフォーマットは全く同じです。「#」マークで始まる行、又は空白行はすべて無視されます。各規則はその独自の行になければなりません。

それぞれの規則は以下のような基本フォーマットを使用してネットワークサービスへのアクセスを制御します：

```
<daemon list>: <client list> [: <option>: <option>: ...]
```

- <*daemon list*> — プロセス名(サービス名ではなく)のカンマで区切られたリスト、又はALLワイルドカード(項15.2.1.1を参照)。デーモンリストはまた、項15.2.1.3内の演算子を承認し、より高い柔軟性を与えます。
- <*client list*> — ホスト名のカンマで区切られたリスト、ホストIPアドレス、特殊パターン(項15.2.1.2を参照)、あるいは特殊ワイルドカード(項15.2.1.1を参照)など、規則により影響されるホストを識別します。クライアントリストは、項15.2.1.3にある演算子リストも承認して、より高い柔軟性を与えます。
- <*option*> — 規則が発動された時に実施される1つのオプション行動、又はカンマで区切られた行動のリスト。オプションのフィールドは拡張(項15.2.3.4を参照)をサポートし、シェルコマンド、アクセスの許可/拒否、ロギング行動の変更などの開始に使用できます(項15.2.3を参照)。

以下に基本的なホストのアクセス規則のサンプルを示します：

```
vsftpd : .example.com
```

この規則はTCPラッパーに対して、example.comドメインのホスト全てからのFTPデーモン(vsftpd)への接続を監視するように指示します。もし、この規則がhosts.allowにある場合、接続は許可されます。この規則がhosts.denyにある場合は、接続は拒否されます。

次のホストアクセス規則のサンプルはより複雑で、2つのオプションフィールドを使用します：

```
sshd：.example.com \
：spawn /bin/echo ‘/bin/date‘ access denied>>/var/log/sshd.log \
：deny
```

この例の中で、各オプションフィールドが逆スラッシュ(\)の後に始まっていることに注意してください。逆スラッシュの使用は、規則の長さによる問題を防止します。

警告

ホストアクセスファイルの最後の行が、改行マーク([Enter]キーを押して出るマーク)でなければ、そのファイル内の最後の規則は、失敗してエラーが/var/log/messages又は/var/log/secureにログされます。これは、規則の行が逆スラッシュを使用することなく、複数行に跨ぐ場合にも同様の問題となります。以下の例では、以上のどちらかの状況による規則違反のログメッセージの関連する部分を表示しています：

```
warning: /etc/hosts.allow, line 20: missing newline or line too long
```

このサンプルの規則は、SSHデーモンへの接続(sshd)が、example.comドメインのあるホストからの試行である場合、echoコマンドを実行(特別ファイルへの試行をログします)して、接続を拒否するように示しています。オプションのdeny ディレクティブが使用されている為、それがhosts.allowファイル内に存在しても、この行がアクセスを拒否します。利用できるオプションの詳細は項15.2.3で御覧下さい。

15.2.1.1. ワイルドカード

ワイルドカードの使用により、TCPラッパーはデーモンやホストのグループをより簡単に照合できます。これらはアクセス規則のクライアントリストフィールドの中で多く使用されます。

以下のようなワイルドカードが使用できます：

- ALL — すべてを照合します。これはデーモンリストとクライアントリストの両方に使用できます。
- LOCAL — ローカルホストなど、ピリオド(.)のないホストをすべて照合します。
- KNOWN — ホスト名、ホストアドレス、あるいはユーザーが既に判っているホストを照合します。
- UNKNOWN — ホスト名、ホストアドレス、あるいはユーザーが不明なホストをすべて照合します。
- PARANOID — ホスト名とホストアドレスが一致しないホストをすべて照合します。

重要

KNOWN、UNKNOWN、PARANOIDのそれぞれのワイルドカードは、名前解決中で問題となると正式なユーザーがサービスへのアクセスを取得するのを阻止する可能性がありますので、注意して使用して下さい。

15.2.1.2. パターン

パターンはアクセス規則のクライアントリストフィールドに使用して、クライアントホストのグループをより正確に指定できます。

以下にクライアントエントリー用の最も一般的に人気のあるパターンのリストを示します：

- ピリオド*(.)*で始まるホスト名 — ホスト名の先頭にピリオドを付けると、その名前のリストされたコンポーネントを共有する全てのホストを照合します。以下の例では、`example.com`ドメイン内のすべてのホストに適用されます：
  ```
  ALL：.example.com
  ```
- ピリオド*(.)*で終了する*IP*アドレス — ピリオドをIPアドレスの末尾に置くと、あるIPアドレスの最初の数字のグループを共有する全てのホストを照合します。次の例では、`192.168.x.x`ネットワーク内のすべてのホストに適用されます：
  ```
  ALL：192.168.
  ```
- *IP*アドレス/ネットマスクのペア — ネットマスク表現もIPアドレスの特定のグループへのアクセスを制御する為にパターンとして使用されます。次の例では、`192.168.0.0`から`192.168.1.255`までのアドレスを持つ全てのホストに適用されます：
  ```
  ALL：192.168.0.0/255.255.254.0
  ```

- アスタリスク*(*)*— アスタリスクはホスト名のグループやIPアドレスの全体を照合するのに使用されます。これは他のタイプのパターンを含むようなクライアントリストが混合していない場合に有効です。次の例では、`example.com` ドメイン内の全てのホストに適用されます：
  ```
  ALL：*.example.com
  ```
- スラッシュ*(/)* — クライアントリストがスラッシュで始まる場合、それはファイル名として取り扱います。これは、多量のホストを指定する規則が必要な場合に役に立つものです。次の例では、全てのTelnet接続の為の`/etc/telnet.hosts` ファイルへのTCPラッパーを対称にしています：
  ```
  in.telnetd：/etc/telnet.hosts
  ```

他にも使用頻度の低いパターンもTCPラッパーで受け付けられます。詳細はホストアクセスのman 5ページで御覧下さい。

警告

ホスト名やドメイン名など、名前解決を必要とするものに関する規則を作成している場合、侵入者は各種のトリックを使用して正確な名前解決を邪魔します。さらには、DNSサービスへのいかなる妨害は、権限のあるユーザーでさえもネットワークサービスを使用出来ないようにしてしまいます。

可能な限り、IPアドレスを使用するのが最善の策です。

15.2.1.3. 演算子

現在、アクセス制御規則は、1つの演算子、EXCEPTを受け付けます。これはデーモンリストと規則内のクライアントリストで使用できます。

EXCEPT演算子の使用により、同じ規則内でより幅広い照合へと特別な拡張が可能になります。

次の`hosts.allow`からの例では、`cracker.example.com`以外の全ての`example.com`ホストからの、全てのサービスへの接続が許可されます：

```
ALL: .example.com EXCEPT cracker.example.com
```

もう1つの`hosts.allow`ファイルからの例では、192.168.0.*x*のネットワークからのクライアントはFTP以外は、すべてのサービスを使用できます：

```
ALL EXCEPT vsftpd: 192.168.0.
```

注意

組織的な観点からは通常は、EXCEPT演算子は控えめに使用して、他のアクセス制御ファイルで規則の例外を指定する方がよいでしょう。このようにすれば、どの管理者も該当するファイルを調べるだけでサービスへのアクセスを許可するホストか、禁止するホストを知ることができ、さまざまなEXCEPT 演算子を調べる必要がなくなります。

15.2.2. ポートマップとTCPラッパー

portmapのアクセス制御規則を作成する場合、TCPラッパーの実装はホストのルックアップをサポートしないので、ホスト名は使用しないで下さい。この理由で、ホストを指定する時はIPアドレスかキーワードALLのみを、hosts.allow又は、hosts.denyで使用してください。

さらには、portmapアクセスコントロール規則への変更はすぐに反映されない場合があります。

NISやNFSなど幅広く使用されているサービスはportmapに依存して運用されているので、これらの制限に気をつけて下さい。

15.2.3. オプションフィールド

アクセスの許可と拒否についての基本的な規則の他に、Red Hat LinuxのTCPラッパーの実装はオプションフィールドを通じてアクセス制御言語への拡張をサポートしています。ホストアクセス規則内のオプションフィールドを使用することにより、管理者はログの動作の変更、アクセス制御の確定、及びシェルコマンドの始動などの各種タスクを達成することが出来ます。

15.2.3.1. ロギング

オプションフィールドにより管理者は、severityディレクティブを使用してログ設備と規則の優先度を簡単に変更することができます。

以下の例では、example.comドメインのホストからのSSHデーモンへの接続は、emergの優先度を持つデフォルトのauthpriv 設備(設備値が指定されていない為)にログされます：

```
sshd : .example.com : severity emerg
```

severityオプションを使用して設備を指定することもできます。次の例では、alertの優先度を持つlocal0設備へのexample.comドメインのホストによるSSHサービス接続の試行をログします：

```
sshd : .example.com : severity local0.alert
```

注意

実際には、この例はlocal0設備へsyslogデーモン(syslogd)がログするように設定されるまで、機能しません。カスタムログ設備の設定に関する詳細についてはsyslog.confのmanページをお読み下さい。

15.2.3.2. アクセスの制御

オプションフィールドにより、管理者はallow又はdeny ディレクティブを最終オプションとして追加することで、1つの規則内で明確に許可、あるいは拒否ができるようになります。

例えば、次の2つの例は、client-1.example.comからのSSH接続を許可しますが、client-2.example.comからの接続を拒否します：

```
sshd : client-1.example.com : allow
sshd : client-2.example.com : deny
```

アクセス制御を規則単位ベースで許可することにより、オプションフィールドは管理者が全てのアクセス規則を1つのシングルファイル、hosts.allow、又はhosts.denyに統合できるようにします。幾らかの人々にとってはこの方法がアクセス規則を簡単に構成させてくれるでしょう。

15.2.3.3. シェルコマンド

オプションフィールドにより、アクセス規則は次の2つのディレクティブを通じてシェルコマンドを始動できます：

- spawn — シェルコマンドを子プロセスとして始動できます。このオプションディレクティブは/usr/sbin/safe_fingerの使用などのタスクを実行し、要求しているクライアントの情報を得たり、echo コマンドを使用して特殊なログファイルを作成したりします。

  次の例では、example.comドメインからTelnetサービスへアクセスを試行するクライアントは、密かに特殊なファイルにログされます：

  ```
  in.telnetd : .example.com \
   : spawn /bin/echo `/bin/date` from %h>>/var/log/telnet.log \
   : allow
  ```

- twist — 要求されたサービスを指定されたコマンドで入れ換えます。このディレクティブは侵入者に対する仕掛け(蜜の壷といいます)をするのによく使用されます。また接続しているクライアントにメッセージを送信するためにも使用されます。ツイストコマンドは、規則行の末尾に置く必要があります。

  以下の例では、example.comドメインからFTPサービスへアクセスを試行しているクライアントがechoコマンドによりメッセージを受け取ります。

  ```
  vsftpd : .example.com \
  : twist /bin/echo "421 Bad hacker, go away!"
  ```

シェルコマンドのオプションに関する情報はhosts_optionsのmanページで御覧下さい。

15.2.3.4. 拡張

拡張は、spawnやtwistディレクティブと一緒に使用して、クライアント、サーバー、プロセスなどの関連の情報を提供します。

以下にサポートされている拡張のリストを示します：

- %a — クライアントのIPアドレス。
- %A — サーバーのIPアドレス。
- %c — ユーザー名とホスト名、又はユーザー名とIPアドレスなどのような各種のクライアントの情報を提供します。
- %d — デーモンのプロセス名。
- %h — クライアントのホスト名(又は、それがない場合、IPアドレス)。
- %H — サーバーのホスト名(又は、それがない場合、IPアドレス)。

- %n — クライアントのホスト名。ない場合は、unknownが出力されます。もしクライアントのホスト名とホストアドレスが一致しない場合は、paranoidが出力されます。
- %N — サーバーのホスト名。ない場合は、unknownが出力されます。サーバーのホスト名とホストアドレスが一致しない場合、paranoidが出力されます。
- %p — デーモンのプロセスID。
- %s — デーモンのプロセスやサーバーのホスト又はIPのアドレスなどのさまざまなタイプのサーバー情報。
- %u — クライアントのユーザー名。ない場合はunknownが出力されます。

以下の規則のサンプルは、spawnコマンドと一緒に拡張を使用してカスタマイズログファイルの中のクライアントホストを識別しています。

これはTCPラッパーに対して、SSHデーモン(sshd)への接続がexample.comドメインのホストから試行された場合は、echoコマンドを実行して、クライアントのホスト名(%h拡張を使用して)を含むその試行を特殊ファイルへログするように指示しています：

```
sshd : .example.com \
: spawn /bin/echo `/bin/date` access denied to %h>>/var/log/sshd.log \
: deny
```

同様に、拡張はクライアントに送り返すメッセージを個人化するのに使用されます。次の例では、example.comドメインからFTPサービスへアクセスを試行しているクライアントは、彼らがサーバーに立ち入り禁止となったことを通知されています：

```
vsftpd : .example.com \
: twist /bin/echo "421 %h has been banned from this server!"
```

利用できる拡張、及び追加のアクセス制御オプションに関する総合的説明には、hosts_accessの為のmanページのセクション5 (man 5 hosts_access)とhosts_optionsのmanページを御覧下さい。

TCPラッパーに関する追加のリソースについては、項15.5を参照して下さい。

15.3. xinetd

xinetdデーモンは、FTP、IMAP、Telnetを含む人気のあるネットワークサービスのサブセットへのアクセスを制御するTCPラップしたスーパーサービスです。これは、アクセス制御、強化ロギング、結合、方向転換、リソース活用制御などの為にサービス特有の設定オプションを提供します。

クライアントホストが、xinetdで制御されたネットワークサービスへ接続を試みた場合、スーパーサーバーは要求を受けて、いずれかのTCPラッパーアクセス制御規則をチェックします。アクセスが許可される場合、xinetdはそのサービス用の自身のアクセス規則の元で接続が許可されることを確認します。そしてサービスがその割り当て以上のリソースを消費していないことや、定義された規則を違反していないことも確認します。その後、要求されたサービスのインスタンスを開始し、その接続の制御を通過させます。1度接続が確立されると、xinetdはもう、クライアントホストとサーバー間の通信を邪魔することはありません。

15.4. xinetdの設定ファイル

xinetdの設定ファイルは以下のようになります：

- /etc/xinetd.conf — グローバルなxinetd設定ファイル。

- /etc/xinetd.d/ディレクトリ— 全てのサービス特有のファイルを含んだディレクトリ。

15.4.1. /etc/xinetd.confファイル

/etc/xinetd.confには、xinetdの制御の元で全てのサービスが影響を受ける一般的な構成の設定が含まれています。xinetd サービスが開始される時に1回だけ読み込まれるため、設定の変更が反映されるようにするためには管理者はまた、xinetdサービスを再起動する必要があります。以下にサンプルの/etc/xinetd.confファイルを示します：

```
defaults
{
   instances       = 60
   log_type        = SYSLOG authpriv
   log_on_success     = HOST PID
   log_on_failure     = HOST
   cps          = 25 30
}
includedir /etc/xinetd.d
```

これらの行は、xinetdのさまざまな側面を制御します：

- instances — xinetdが1度に対処できる最大の要求数を設定します。
- log_type — xinetdを設定してauthprivログ設備を使用します。これはログエントリーを/var/log/secureファイルに書き込みます。FILE /var/log/xinetdlogのようなディレクティブをここに追加すると、/var/log/ディレクトリの中にxinetdlogと言うカスタムログファイルを作成します。
- log_on_success — xinetdを設定して、接続が成功したかどうかをログします。デフォルトでは、リモートホストのIPアドレスと要求をプロセスしているサーバーのプロセスIDが記録されます。
- log_on_failure — xinetdを設定して、接続失敗があるかどうか又は、接続が許可されていないかどうかをログします。
- cps — xinetdを設定してどのサービスにも毎秒25接続以上は許可しないようにします。この限度に到達した場合は、サービスは30秒だけ休憩します。
- includedir /etc/xinetd.d/ —/etc/xinetd.d/ディレクトリにあるサービス特有の設定ファイルで宣言してあるオプションを含めます。このディレクトリの詳細については項15.4.2を参照して下さい。

注意

多くの場合、/etc/xinetd.conf内のlog_on_successとlog_on_failureの両方の設定は、さらにサービス特有のログファイルで編集されます。この理由で、このファイルが示す以上に該当するサービスログにはより多くの情報が表示される可能性があります。ロギングオプションの詳細については項15.4.3.1を御覧下さい。

15.4.2. /etc/xinetd.d/ディレクトリ

/etc/xinetd.d/ディレクトリ内のファイルには、xinetdで管理されている各サービスの設定ファイルと、そのサービスに関連したファイルの名前が含まれています。xinetd.confの場合と同様にこのファイルはxinetd サービスが開始する時にだけ読み込まれます。変更を反映させるためには、管理者はxinetd サービスを再起動する必要があります。

/etc/xinetd.d/ディレクトリ内のファイルのフォーマットは/etc/xinetd.confと同じ慣例を使用します。各サービスの設定が別々のファイルに格納されている主な理由は、カスタマイズを簡単にすることと、他のサービスに影響を与える可能性が少なくなるからです。

これらのファイルがどのように構成されるかを知るには、/etc/xinetd.d/telnetファイルを考慮して下さい：

```
service telnet
{
    flags     = REUSE
    socket_type   = stream
    wait      = no
    user      = root
    server     = /usr/sbin/in.telnetd
    log_on_failure += USERID
    disable    = yes
}
```

これらの行は、telnetサービスのさまざまな側面を制御します：

- service — サービス名を定義します。通常、/etc/servicesファイルにリストしてあるサービスに合わせます。
- flags — 接続の為の必要な数の属性をセットします。REUSEはxinetdに対してTelnet接続のソケットを再使用するように指示します。
- socket_type — ネットワークソケットのタイプをstreamに設定します。
- wait — サービスがシングルスレッド(yes)か又はマルチスレッド(no)かを定義します。
- user — プロセスが実行に使用するユーザーIDを定義します。
- server — 始動する予定のバイナリ実行ファイルを定義します。
- log_on_failure — 既にxinetd.confに定義している物に追加して、log_on_failure用のロギングパラメータを定義します。
- disable — サービスが有効かどうか定義します。

15.4.3. xinetd設定ファイルの変更

xinetdで保護されているサービス用には利用できる多くの種類のディレクティブがあります。このセクションはより一般的に使用されているオプションの一部を強調して説明します。

15.4.3.1. ロギングオプション

以下のロギングオプションは、/etc/xinetd.confと/etc/xinetd.d/ディレクトリ内のサービス特有の設定ファイルの両方で利用できます。

以下により一般的に使用されているロギングオプションの一覧を表示します：

- ATTEMPT — 失敗の試行があった事実をログします(log_on_failure)。
- DURATION — リモートシステムによって使用されたサービスの時間の長さをログします(log_on_success)。
- EXIT — サービスの終了ステータス、又は終結シグナルをログします(log_on_success)。
- HOST — リモートホストのIPアドレス(log_on_failureとlog_on_success)をログします。
- PID — 要求を受けているサーバーのプロセスIDをログします(log_on_success)。

- RECORD — サービスがスタートできない場合に、リモートシステムに関する情報を記録します。loginやfingerなどの特定のサービスのみがこのオプション(log_on_failure)を使用できます。
- USERID — 全てのマルチスレッドシステムの為にRFC 1413で定義された方法を使用しているリモートユーザーをログします(log_on_failureとlog_on_success)。

ロギングオプションの総合的リストについては、xinetd.confのmanページを御覧下さい。

15.4.3.2. アクセス制御のオプション

xinetdのユーザーは、TCPラッパーホストアクセス規則の使用か、xinetd設定ファイル経由でのアクセス制御の用意か、又はその両方を選択できます。TCPラッパーホストアクセス制御ファイルの使用に関する情報は項15.2で見ることができます。このセクションではサービスへのアクセスを制御するためのxinetdの使用を説明します。

注意

TCPラッパーとは異なり、xinetdの管理者がxinetdサービスを再起動した場合にのみ、アクセス制御への変更が反映されます。

xinetdホストアクセス制御は、TCPラッパーで使用されている方法と異なります。TCPラッパーが全てのアクセス設定を2つのファイル、/etc/hosts.allowと/etc/hosts.denyに配置するのに対して、/etc/xinetd.d内の各サービスファイルはそれ自身のアクセス制御規則を含むことができます。

以下のホストアクセスオプションは、xinetdによってサポートされています：

- only_from — 特定のホストのみにサービスの使用を許可します。
- no_access — リストにあるホストのサービス利用を拒否する。
- access_times — 特定のサービスが利用できる時間幅を指定する。この時間幅は24時間形式でHH:MM-HH:MMの表示をする必要があります。

only_fromとno_accessオプションはIPアドレス又はホスト名の一覧、あるいはネットワーク全体の指定ができます。TCPラッパーの様に、強化ロギング設定と共にxinetdアクセス制御を結合することにより、それぞれの接続試行の記録を詳細に取りながら、禁止されているホストからの要求をブロックしてセキュリティを強化できます。

例えば、以下の/etc/xinetd.d/telnetファイルは、特定のネットワークグループからのTelnetアクセスをブロックして、許可されたユーザーにさえも、ログイン可能な時間帯を制限する為に使用できます：

```
service telnet
{
    disable    = no
    flags      = REUSE
    socket_type  = stream
    wait       = no
    user       = root
    server     = /usr/sbin/in.telnetd
    log_on_failure += USERID
    no_access    = 10.0.1.0/24
    log_on_success += PID HOST EXIT
    access_times  = 09:45-16:15
}
```

この例では、10.0.1.2などの10.0.1.0/24ネットワークのクライアントシステムが、Telnetサービスにアクセスを試みた場合、以下のように表示したメッセージを受け取ることになります：

```
Connection closed by foreign host.
```

さらに、彼らのログイン試行は、次のように/var/log/secureの中に記録されます：

```
May 15 17:38:49 boo xinetd[16252]: START: telnet pid=16256 from=10.0.1.2
May 15 17:38:49 boo xinetd[16256]: FAIL: telnet address from=10.0.1.2
May 15 17:38:49 boo xinetd[16252]: EXIT: telnet status=0 pid=16256
```

TCPラッパーをxinetdアクセス制御と併用する場合、2つのアクセス制御メカニズム間の関係を理解することが重要です。

クライアントが接続の要求をした時、xinetdによって続けられる動きの順序を以下に示します：

1. xinetdデーモンはlibwrap.aライブラリコールを経由して、TCPラッパーホストアクセス規則にアクセスします。もし拒否の規則がクライアントホストに適合するならば、その接続は切断されます。許可の規則がクライアントホストと適合する場合、接続がxinetdに渡されます。
2. xinetdデーモンは、xinetdサービスと要求されたサービスの両方の為に、自身のアクセス制御規則をチェックします。もし、拒否の規則がクライアントホストに適合する場合、接続は切断されます。その他の場合は、xinetdは要求されたインスタンスを開始して接続の制御を渡します。

重要

TCPラッパーアクセス制御をxinetdアクセス制御と併用する場合、注意が必要です。設定ミスは良からぬ結果を招くことになります。

15.4.3.3. バインドとリダイレクトオプション

xinetd用のサービス設定は、サービスをあるIPアドレスにバインドし、そのサービス用の要求を別のIPアドレス、ホスト名、あるいはポートへリダイレクトします。

バインディングは、サービス特有の設定ファイルの中でbindオプションと共に制御されており、サービスをシステム上のIPアドレスと連結します。設定されると、bindオプションは、正式なIPアドレスの為の要求のみをそのサービスへアクセスする許可をします。この方法で異なるサービスは異なるネットワークインターフェイスに需要ベースでバインドできます。

これは、特に複数のネットワークアダプターか、複数のIPアドレスを設定したシステムには便利なものです。このようなシステムでは、Telnetなどの安全でないサービスはプライベートなネットワークに接続されているインターフェイス上でのみリッスンするようにして、インターネット接続のインターフェイスではリッスンしないように設定できます。

redirectオプションは、IPアドレス又はホスト名とそれに続くポート番号を受け付けます。サービスを設定して、このサービス用の要求を全て指定したホストか、又はポート番号にリダイレクトできるようにします。この機能は同じシステム上のポートから別のポートへポイントする、あるいは要求を同じマシン上の別のIPアドレスへリダイレクトする、あるいは要求を全く別のシステムとポート番号へ移動する、あるいはこれらのオプションの組合せをする場合に使用できます。この様にして、システム上の特定のサービスに接続しているユーザーは、問題なく他のシステムへ回送されるようにできます。

xinetdデーモンは、リクエストを送信したクライアントマシンと実際にサービスを提供するホストが接続されている間だけ有効なプロセスを生成し、2つのシステム間でデータを転送することによって、このリダイレクトを実現します。

bindとredirectオプションの利点は、それらが一緒に使用される時にはっきりと判別できます。サービスをシステム上の特定のIPアドレスにバインドして、その後そのサービス用の要求を1番目のマシンだけが見ることができる2番目のマシンにリダイレクトすることにより、内部のシステムが全く別のネットワークの為にサービスを提供することに使用できます。別の方法として複数ホームのマシン上の特定のサービスが、既知のIPアドレスへの露呈されることを制限したり、またそのサービスの要求をその目的用に特定の設定をしたマシンへリダイレクトするのに使用できます。

例えば、Telnetサービス用にこの設定でファイアウォールとして使用されているシステムを考えて見ましょう：

```
service telnet
{
    socket_type = stream
    wait = no
    server = /usr/sbin/in.telnetd
    log_on_success += DURATION USERID
    log_on_failure += USERID
    bind         = 123.123.123.123
    redirect       = 10.0.1.13 21 23
}
```

このファイル内のbindとredirectオプションは、マシン上のTelnetサービスがインターネットに向けた外部IPアドレス(123.123.123.123)にバインドされていることを確定します。さらには、123.123.123.123に送信されたTelnetサービスへの要求は2番目のネットワークアダプターを通じて、内部IPアドレスの10.0.1.13にリダイレクトされ、これはファイアウォールと内部のシステムしかアクセスできないようになっています。ファイアウォールはその後、その2つのシステム間で通信をしますので、接続しているシステムは実際には別のマシンに接続されている状態でも123.123.123.123に接続されているように見えます。

この機能は、ブロードバンド接続で1つの固定IPアドレスをしているユーザーに特に役に立ちます。NAT(Network Address Translation)を使用するとき、内部専用のIPアドレスを使用する、ゲートウェイマシンの背後にあるシステムはゲートウェイマシン外部のマシンからは利用できません。しかし、xinetdで制御されている特定のサービスがbindとredirect オプションで設定されている場合、そのゲートウェイマシンは外部システムとサービスを提供するように設定されている特定の内部マシン間でのプロキシの一種として動作することができます。さらには、各種のxinetdアクセス制御とロギングオプションが、リダイレクトされたサービスの同時接続の数量を制限するなどの、追加の保護が提供されることになります。

15.4.3.4. リソース管理のオプション

xinetdデーモンは、DoS(Denial of Serviceサービスの拒否)攻撃からに対する基本的レベルの保護を追加することができます。以下のリストでは、そのような終りのない攻撃を制限できるディレクティブを表示します：

- per_source — ソースIPアドレス毎のサービス用のインスタンスの最大数を定義します。これは整数のみを引数として受け付け、xinetd.conf内とxinetd.d/のサービス特有の設定ファイル内で使用できます。
- cps — 毎秒の接続最大数を定義します。このディレクティブは中間にスペースを入れた2つの整数引数を使います。1番目は1秒間に接続が許可される最大数です。2番目はサービスを再開始する時にxinetdが待機する必要のある時間の秒数です。整数のみを引数として受け付け、xinetd.conf内とxinetd.d/ディレクトリのサービス特有の設定ファイル内の両方で使用できます。
- max_load — サービスのCPU使用限界を定義します。これは引数に小数点を受け付けます。

他にも利用できるxinetd用のリソース管理オプションはあります。詳細は*Red Hat Linux* セキュリティガイド内のサーバーセキュリティという章を御覧下さい。また、xinetd.confのmanページも参照して下さい。

15.5. その他のリソース

TCPラッパーとxinetdに関する追加の情報が、システムドキュメントとWebにあります。

15.5.1. インストールされるドキュメント

システムに付随するドキュメントとWebに、TCPラッパー、xinetdの追加情報が記載されています。

- /usr/share/doc/tcp_wrappers-<*version*>/ — ここに含まれているREADMEファイルで、TCPラッパーの動作説明と、存在する各種ホスト名とホストアドレスの偽装について説明しています。
- /usr/share/doc/xinetd-<*version*>/ — ここにあるREADMEファイルでは、アクセス制御とsample.conf ファイルの側面と共に、/etc/xinetd.d/ディレクトリ内のサービス特有の設定ファイルを編集する各種アイデアを説明しています。
- man 5 hosts_access — TCPラッパーのホスト制御アクセスファイルの為のmanページ。
- man hosts_options — TCPラッパーのオプションフィールドのmanページ。
- man xinetd.conf — xinetdの設定オプションをリストしたmanページ。
- man xinetd — xinetdスーパーサービスデーモンのmanページ。

15.5.2. 役に立つWebサイト

- http://www.xinetd.org—xinetdの発祥地で、設定ファイルのサンプル、すべての機能の一覧、有益なFAQが含まれています。
- http://www.macsecurity.org/resources/xinetd/tutorial.shtml—特定のセキュリティ条件に合わせてデフォルトのxinetd設定ファイルを修正するさまざまな方法を詳細に説明した講習です。

15.5.3. 関連書籍

- *Red Hat Linux* セキュリティガイド ; Red Hat, Inc. — TCPラッパーとxinetdに関する特定の提案を持つ、ワークステーション、サーバー、及びネットワークセキュリティの概要を提供します。
- *Hacking Linux Exposed* by Brian Hatch, James Lee, and George Kurtz; Osbourne/McGraw-Hill — TCPラッパーとxinetdに関する情報と特徴とした優秀なセキュリティ関連のリソースです。

第16章

iptables

Red Hat Linuxにはネットワーク用の高度なツールパケットフィルタリング (カーネル内でネットワークスタックへネットワークパケットが進入、通過、退出するのを制御するプロセス)がインストールされています。バージョン2.4以前のカーネルはパケットフィルタリングの為にipchainsに依存しており、フィルタリングプロセスの各ステップでパケットに適用される規則の一覧を使用していました。バージョン2.4の到来により、iptables (ネットフィルタとも言います)が導入され、ipchainsと似ていますが、ネットワークパケットのフィルタに利用出来る活動範囲及び制御を大幅に拡張します。

この章では、パケットフィルタリングの基礎に焦点を当て、ipchainsとiptablesの違いを明確にし、iptablesコマンドで使用できるさまざまなオプションを説明し、システムを次にリブートしてもフィルタリング規則を保持できる方法を示します。

iptablesの規則の作成とこれらの規則に基づくファイアウォールの設定については、項16.5を参照してください。

警告

2.4カーネル下でデフォルトのファイアーウォール機能はiptablesです。しかし、ipchainsが既に起動しているなら、iptablesは使う事ができません。ipchainsがシステム起動時に存在すれば、カーネルはエラーを表示しiptablesの起動に失敗します。

これらのエラーメッセージはipchainsの機能に影響を与えるものではありません。

16.1. パケットフィルタリング

トラフィックは、パケットとしてネットワーク内を移動します。ネットワークパケットは特定のサイズと形式のデータの集合体です。ファイルをネットワーク越しに転送する為に送信側のコンピュータは利用するネットワークプロトコルの規則を使用し、ファイルをパケットに分割してネットワークで送信します。各パケットにファイルデータの小片が格納されます。パケットを受信すると、受信側のコンピュータはパケットからファイルを再構築します。

各パケットには、ネットワーク内を移動して送信先に到達するための情報が含まれています。パケットが移動する途中にあるコンピュータと送信先のマシンは、パケットの送信元、送信先、タイプなどの情報を知ることができます。ほとんどのパケットはデータを伝送するように設計されていますが、特別な方法でパケットを使用するプロトコルもあります。たとえば、「TPC（伝送制御プロトコル）」で使用されるSYNパケットは2つのシステム間の通信を開始するためのもので、データは含まれていません。

Linuxカーネルにはパケットをフィルタリングするための機能が組み込まれているので、一部のパケットだけがシステムに入ってくるようにすることができます。2.4カーネルのネットフィルタには3つの組込み型テーブル、すなわち規則一覧が含まれています。それは以下のようなものです：

- filter — ネットワークパケットを処理するデフォルトのテーブル。
- nat — 新規接続を作成するパケットの変更に使用。
- mangle — パケット変更の特定のタイプに使用。

これらの各テーブルは順番に組込み型のチェーンのグループを持ち、これはネットフィルタによってパケット上で実行されるアクションに相当するものです。

フィルタテーブル用の組込み型チェーンは以下のようになります：

- *INPUT* — ホスト用のターゲットとされているネットワークパケットに適用します。
- *OUTPUT* — ローカル生成のネットワークパケットに適用します。
- *FORWARD* — ホストを通ってルーティングしたネットワークパケットに適用します。

natテーブル用の組込み型チェーンは以下のようになります：

- *PREROUTING* — ネットワークパケットが到着するとそれを変更します。
- *OUTPUT* — ローカル生成のネットワークパケットが送信される前にそれを変更します。
- *POSTROUTING* — ネットワークパケットが送信される前にそれを変更します。

mangleテーブルの組込み型チェーンは以下の様になります：

- *INPUT* — ホスト用にターゲットされているネットワークパケットを変更します。
- *OUTPUT* — ローカル生成のネットワークパケットを送信される前に変更します。
- *FORWARD* — ホストを通してルーティングしたネットワークパケットを変更します。
- *PREROUTING* — 着信のネットワークパケットをルーティングされる前に変更します。
- *POSTROUTING* — ネットワークパケットを送信される前に変更します。

Linuxシステムから受信したり、Linuxシステムへ送信したりするネットワークパケットの全ては、少なくとも1つのテーブルによって左右されます。

パケットは、チェーンの末尾に出てくる前に各テーブルの中で複数の規則に対してチェックされます。これらの規則の構成と目的は変化しますが、特定のプロトコルとネットワークサービスを使用する時には通常、特定のIPアドレス、又はアドレスのセットから届く、あるいはそこへ向かうパケットを識別しようとします。

その目的地に関係なく、パケットがテーブルのある特定の規則に適合すると、あるターゲット、すなわち、アクションがパケットに適用されます。規則が適合するパケットの為にACCEPTターゲットを指定している場合、パケットは規則チェックの残りの部分をスキップして、その目的地に進むことを許可されます。規則がDROPターゲットを指定している場合、パケットはシステムへのアクセスを拒否されてパケットを発信したホストには何も返信されません。規則がQUEUEターゲットを指定している場合、パケットはユーザースペースへパスされることになります。規則がオプションのREJECTターゲットを指定している場合、パケットはドロップされます。しかしエラーパケットがパケットの発信元へ送られます。

それぞれのチェーンはACCEPT、DROP、REJECT、QUEUEのいずれかのデフォルトポリシーを持っています。このチェーン内の規則のどれもパケットに適応しない場合、パケットはデフォルトポリシーに従って扱われます。

iptablesコマンドはこれらのテーブルを設定するだけでなく、必要であれば、新しいテーブルのセットアップもします。

16.2. iptablesとipchainsの違い

一見したところでは、ipchainsとiptablesは非常に似ているように思われます。どちらの方法でパケットフィルタリングを行っても、Linuxカーネル内で有効な規則のチェーンを使用して、システムに出入りできるパケットを決定するだけでなく、特定の規則を満たすパケットについての動作を決定します。しかし、iptablesを使用すると非常に柔軟な方法でパケットをフィルタリングできるので、管理者はシステムを複雑にせずにきめ細かい制御を行うことができます。

特に、ipchainsを使い慣れているユーザーは、次のようなipchainsとiptablesの著しい違いを認識してから、iptablesを使用してください。

- *iptables*の元では、フィルタリングする各パケットは複数のチェーンではなく*1*つのチェーンのみの規則を使用して処理されます。つまり、ipchainsを使用するシステムに入ってき

たFORWARDパケットは、INPUT、FORWARD、OUTPUTの各チェーンを通らないと送信先に進めませんが、iptablesでは送信先がローカルシステムの場合はINPUTチェーン、ローカルシステムがパケットを生成した場合はOUTPUTチェーンのみにパケットが送信されます。この理由で、パケットを実際に調べる規則の中で特定のパケットを取り込む様に設計された規則を用意します。

- *DENY*ターゲットは*DROP*に変更されました。 ipchainsでは、チェーン内の規則に一致したパケットはDENYターゲットに送られます。このターゲットはiptablesの元ではDROPに変更されなければいけません。
- 規則の中でオプションを設置する時、その順番が大切です。 以前は、ipchainsで規則のオプションの順序はあまり関係ありませんでした。iptablesコマンドでは、もっと厳密な構文を使用します。例えば、iptablesコマンドの中のプロトコル(ICMP、TCP、UDP)は、送信元又は送信先のポートの前で指定する必要があります。
- 規則で使用するネットワークインターフェイスを指定する場合、*INPUT*チェーンか*FORWARD*チェーンでは着信インターフェイス（*-i*オプション）、*FORWARD*チェーンか*OUTPUT*チェーンでは発信インターフェイス（*-o*オプション）のみを使用する必要があります。これは、着信インターフェイスではOUTPUTチェーンが使用されず、INPUTチェーンが発信インターフェイスを通過するパケットによって見えないためです。

iptablesは根本的に作り直されたネットワークフィルターなので、変更点はこれがすべてではありません。詳細については、項16.5の中の*Linux 2.4 Packet Filtering HOWTO*を参照して下さい。

16.3. iptablesコマンドで使用するオプション

カーネルがパケットをフィルタできるようにする規則は、iptablesコマンドを実行することで有効になります。iptablesコマンドを実行する時には以下のオプションを指定します：

- パケットタイプ — コマンドがフィルタするパケットのタイプを指示します。
- パケットの送信元/送信先 — パケットの送信元、又は送信先に応じてコマンドがフィルタするパケットのタイプを指示します。
- ターゲット — 上記の基準に適合するパケットに対して実行されるアクションを指示します。

現行のiptables規則と共に使用されるオプションは、規則を有効にする為に規則全体の目的と条件を元にして論理的にグループ化する必要があります。

16.3.1. テーブル

iptablesの強力な特徴として、複数のテーブルを使用して、特定のパケットの行方を決定できます。iptablesの柔軟性のお蔭で、特定の目的に合わせてテーブルを作成して、それを/lib/modules/<*kernel-version*>/kernel/net/ipv4/netfilter/ ディレクトリの中に保存できます。この<*kernel-version*>とはカーネルのバージョン番号のことです。

デフォルトのテーブルはfilterという名前で、標準的なINPUT、OUTPUT、FORWARDの各チェーンが組み込まれています。これは、ipchainsで使用する標準的なチェーンと少し似ています。しかし、iptablesには、特別なパケットフィルタリング作業を行う2つのテーブルがデフォルトで追加されています。natテーブルを使用するとパケットに記録されている送信元と送信先のアドレスを変更することができ、mangleテーブルを使用すると特別な方法でパケットを変更することができます。

各テーブルにはテーブルの目的に基づいて必要な作業を行うデフォルトのチェーンが含まれていますが、どのテーブルにも新しいチェーンを追加することが出来ます。

16.3.2. 構造

多くのiptablesコマンドの構造は、次のようになります。

```
iptables [-t <table-name>] <command> <chain-name> <parameter-1> \
    <option-1> <parameter-n> <option-n>
```

この例では、<*table-name*>オプションによってデフォルトのfilterテーブル以外のテーブルを使用できます。<*command*>オプションは、 <*chain-name*>オプションで指定される規則の追加あるいは削除など、実行する特定の作業を指示します。<*chain-name*>以降にあるのはパラメータとオプションのペアで、パケットが規則に適合した場合に起こることを定義します。

iptablesコマンドの構造を見てみると、他の殆どのコマンドとは異なり、iptablesコマンドの長さと複雑性はその目的に基づいて変更出来ることを認識しておくことが大切です。チェーンから規則の1つを削除する簡単なコマンドは、とても短くできますが、特定のサブネットから送信され特定のパラメータとオプションを使用するパケットをフィルタするように設計したコマンドは、かなり長くなります。iptables コマンドを作成する時、幾つかのパラメータとオプションは、さらに以前のオプションの要求を指定するために、他のパラメータとオプションを必要とする可能性があることを認識しておくとよいでしょう。有効な規則を構成するには、他のオプションセットを要求するパラメータとオプションがすべて満足されるまで継続する必要があります。

iptables -hと入力すると、iptablesコマンドの構造の総合的な一覧が表示されます。

16.3.3. コマンド

コマンドでは、iptablesが行う特定の動作を指定します。1つのiptablesコマンド文字列について指定できるのは、1つのコマンドだけです。ヘルプコマンドを除くすべてのコマンドは、大文字で入力します。

iptablesのコマンドには、次のようなものがあります：

- -A—指定したチェーンの終わりにiptables規則を追加します。これは、チェーン内の規則の順序が問題でない場合に単純に規則を追加するためのコマンドです。
- -C—指定したチェーンに追加する前に特定の規則をチェックします。このコマンドは、パラメータとオプションを追加するときにプロンプトが表示されるので、複雑なiptables規則を作成する場合に便利です。
- -D—番号で指定した規則を特定のチェーンから削除します。たとえば、チェーン内の5番目の規則を削除する場合は5を指定します。規則全体を入力すると、それに一致する規則がチェーンから削除されます。
- -E — ユーザーが定義したチェーンの名前を変更します。テーブルの構造にはまったく影響を与えません。
- -F—選択したチェーンからすべての規則を削除します。チェーンを指定しない場合は、すべてのチェーンのすべての規則が削除されます。
- -h — コマンドの構造の一覧とコマンドパラメータ、オプションの簡単な説明などが表示されます。
- -I — ユーザー定義の数値で指定された位置にチェーン内の規則を挿入します。数値が指定されていない場合、iptablesはそのコマンドをチェーンの最上部に置きます。


警告

規則を追加するときは、使用するオプション（-Aか-I）に注意してください。チェーン内にある規則の順番はどの規則をどのパケットに適用するかを決定するのに重要です。

- -L—コマンドの後で指定するチェーン内にあるすべての規則を一覧表示します。チェーンとテーブルを指定しない場合は、デフォルトのfilterテーブル内にあるすべてのチェーンのすべての規則が一覧表示されます。それ以外の場合は、次の構文を使用して、規則を一覧するチェーンとテーブルを指定します。
  iptables -L <*chain-name*> -t <*table-name*>

規則の番号や説明を可能にする-Lコマンドの強力なオプションについては、項16.3.7を参照してください。

- -N—指定した名前で新しいチェーンを作成します。
- -P—特定のチェーンについてデフォルトのポリシーを設定します。これによって、パケットがチェーン内にあるすべての規則を満たさない場合に、ACCEPTやDROPなど特定のターゲットに送ることができます。
- -R — 特定のチェーンの規則を置き換えます。規則の番号はチェーン名の後で指定する必要があります。チェーン内の最初の規則が、規則番号「1」になります。
- -X—指定したチェーンを削除します。どのテーブルについても、あらかじめ組み込まれているチェーンは削除できません。
- -Z—特定のテーブルについてすべてのチェーンのバイトとパケットカウンタを0にします。

16.3.4. パラメータ

特定のチェーンにおける規則の追加、削除、挿入、交換などの規則を含む一定のiptablesコマンドを指定すると、パケットフィルタリング規則を構築するためのパラメータが必要になります。

- -c—特定の規則のカウンタをリセットします。このパラメータでは、PKTSオプションか、BYTESオプションを使用してリセットするカウンタを指定できます。
- -d — 規則を満たすパケットの送信先ホスト名、IPアドレス、ネットワークのどれかを設定します。ネットワークと一致する場合、以下のようなIPアドレス/ネットマスクがサポートされます：
  - *N.N.N.N/M.M.M.M* — ここで*N.N.N.N*はIPアドレスの範囲であり、*M.M.M.M*はネットマスクです。
  - *N.N.N.N/M* — ここで*N.N.N.N*はIPアドレスの範囲であり、*M*はネットマスクです。
- -f—断片化されたパケットのみに規則を適用します。

  このパラメータの後で!オプションを使用すると、断片化されていないパケットのみに規則が適用されます。
- -i — eth0やppp0などの着信ネットワークインターフェイスを設定します。iptablesでは、このオプションパラメータを使用できるのは、filterテーブルの場合はINPUTチェーンとFORWARDチェーンと共に、またnatテーブルとmangleテーブルの場合はPREROUTINGチェーンと共に、使用する時だけです。

  このパラメータはまた、以下のような特殊オプションもサポートします：
  - !—このパラメータで指定したインターフェイスを規則から除外します。
  - + — 特定の文字列に一致するすべてのインターフェイスを一致の対象とするワイルドカード文字です。たとえば、-i eth+というパラメータを指定すると、システム上にあるすべてのイーサネットインターフェイスに規則が適用され、ppp0など他のインターフェイスには適用されません。

  -i—パラメータを使用する場合にインターフェイスを指定しないと、すべてのインターフェイスが対象となります。
- -j — パケットが特定の規則を満たした場合に特定のターゲットにジャンプするよう指定します。-jオプションで使用できるターゲットには、ACCEPT、DROP、QUEUE、RETURNという標準のオプションと、LOG、MARK、REJECTなど、Red Hat Linux iptables RPM パッケージにデフォルトでロードされているモジュールを経由して利用可能な拡張オプションがあります。この詳細と他のターゲットに関する情報はiptablesのmanページを御覧下さい。

  現在のチェーン外にあるユーザー定義のチェーンに、規則を満たすパケットを送ることができます。そうすることで他の規則もそのパケットに適用できます。

ターゲットを指定しない場合、いかなる動作も行わずにパケットが通過します。しかし、パケットは特定の規則を満たしたので、その規則のカウンタには1が加えられます。

- -o — 1つの規則の為に発信ネットワークを設定します。filterテーブルの場合はOUTPUTチェーンとFORWARDチェーン、natテーブルとmangleテーブルの場合はPOSTROUTINGチェーンのみで使用できます。このパラメータのオプションは、着信ネットワークインターフェイスパラメータ（-i）の場合と同じです。
- -p — 規則についてIPプロトコルを設定します。icmp、tcp、udp、allのどれか、サポートしているプロトコルを指定できます。さらには、/etc/protocolsに一覧表示してあるプロトコルも使用できます。規則を作成している時点にこのオプションが省略されていると、allオプションがデフォルトになります。
- -s — 送信先パラメータ（-d）と同じ構文を使用して、特定のパケットの送信元を設定します。

16.3.5. 比較オプション

異なるネットワークプロトコルは、特別な比較オプションを用意して、そのプロトコルを使用して特定のパケットと一致するように特殊な設定をします。もちろん、このプロトコルは最初に、-p tcp <*protocol-name*>を使用してiptablesコマンドの中に指定しておき(ここで<*protocol-name*>はターゲットプロトコルです。)、そのプロトコル用のオプションを利用できるようにする必要があります。

16.3.5.1. TCPプロトコル

TCPプロトコル（-p tcp）では、以下の比較オプションを使用できます。

- --dport — パケットの送信先ポートを設定します。ネットワークサービス名（wwwやsmtpなど）、ポート番号、ポート番号の範囲のいずれかを使用できます。ネットワークサービスの名前や、エイリアスとそのネットワークサービスが使用するポート番号を閲覧するには、/etc/servicesファイルを参照してください。--destination-portの比較オプションは、--dportと同義となります。

  ポート番号の範囲を指定するには、-p tcp --dport 3000:3200のように2つの番号をコロン（:）で区切ります。最大の有効範囲は、0:65535です。

  また、--dportオプションの後で感嘆符（!）をフラグとして使用すると、iptablesに対してそのネットワークサービスか、あるいはポートを使用しないすべてのパケットを比較するように指定できます。
- --sport — --dportと同じオプションを使用して、パケットの送信元ポートを設定します。--source-portの比較オプションは--sportと同義です。
- --syn—一般に*SYN*パケットと呼ばれる、通信を開始するよう設計されたすべてのTCPパケットを規則の対象にします。データを伝送するパケットは影響を受けません。--syn オプションの後で感嘆符（!）をフラグとして使用すると、SYNパケット以外のすべてのパケットが対象になります。
- --tcp-flags — 特定のビット（フラグ）を持つTCPパケットを規則と比較される様にします。--tcp-flagsの比較オプションは2つのパラメータを受け付けます。1番目のパラメータはマスクで、パケット内でフラグが検査できるようにします。2番目のパラメータでは、一致するように設定する必要のあるフラグを指定します。

  使用できるフラグは以下のようになります：

  - ACK
  - FIN
  - PSH
  - RST

- SYN
- URG
- ALL
- NONE

たとえば、-p tcp --tcp-flags ACK,FIN,SYN SYNと指定すると、SYNフラグが設定されていてACKフラグとFINフラグは設定されていないTCPパケットのみが規則を満たします。

感嘆符(!)を--tcp-flagsの後で使用すると、比較オプションの対応が逆転されます。

- --tcp-option—特定のパケットで設定できるTCP特有のオプションを比較しようとします。感嘆符（!）を使用すると、意味を反対にすることができます。

16.3.5.2. UDPプロトコル

UDPプロトコル（-p udp）では、以下のオプションを使用できます。

- --dport — サービス名、ポート番号、ポート番号の範囲のどれかを使用して、UDPパケットの送信先ポートを指定します。--destination-portの比較オプションは--dportと同義となります。このオプションのさまざまな使用法については、項16.3.5.1の--dport比較オプションを参照してください。
- --sport — サービス名、ポート番号、ポート番号の範囲のどれかを使用して、UDPパケットの送信元ポートを指定します。--source-portの比較オプションと同義です。このオプションのさまざまな使用法については、項16.3.5.1の--sport比較オプションを参照してください。

16.3.5.3. ICMPプロトコル

ICMP（Internet Control Message Protocol）を使用するパケットの場合(-p icmp)、次のオプションを使用して比較を行うことができます：

- --icmp-type—規則を満たすICMPタイプの番号か名前を設定します。有効なICMP名の一覧は、iptables -p icmp -hというコマンドを実行すると表示されます。

16.3.5.4. その他の比較オプションがあるモジュール

その他の比較オプションもiptablesコマンドによってロードされるモジュールで利用できます。比較オプションモジュールを使用するには、-m <*module-name*> などの-mオプションを使用して、名前の指定でモジュールをロードする必要があります(<*module-name*>はモジュールの名前で入れ換えます)。

デフォルトで多数のモジュールを使用することができます。ユーザー独自のモジュールを作成して、新しい比較オプションを使用することもできます。

多くのモジュールがありますが、ここではよく使用されるモジュールのみを説明します。

- limitモジュール— この使用により特定の規則を満たすパケットの数を制限できます。大量の一致パケットが同じメッセージでログを一杯にしたりシステムのリソースを無駄に使用することがないようにして、規則の一致をログする時に特に便利です。

  limitモジュールは以下のようなオプションを有効にします：

  - --limit—特定の時間帯に比較する回数を設定します。<*number*>/<*time*>という形式で回数と時間を指定します。たとえば、--limit 5/hourと指定すると、1時間に5回だけ規則が比較されます。

回数と時間を指定しない場合は、デフォルト値の3/hourが使用されます。

- --limit-burst—同時に比較できるパケットの数を制限します。このオプションは、--limitオプションとともに使用してください。このオプションでは、同時に比較できるパケットの最大数を指定します。

  値を指定しない場合、5つのパケットだけが規則を満たすことができます。

- stateモジュール— 接続状態について比較を有効にします。

  stateモジュールは以下のようなオプションを有効にします：

  - --state — 以下の接続状態についてパケットを比較します：
    - ESTABLISHED—確立された接続内にある他のパケットに関係があるパケットが規則を満たします。
    - INVALID—既知の接続に結び付けられないパケットが規則を満たします。
    - NEW—新しい接続を作成しているパケットか、あるいはそれまでになかった双方向接続の一部となっているパケットが規則を満たします。
    - RELATED—既存の接続と何らかの関係がある新しい接続を開始するパケットが規則を満たします。

    これらの接続状態を複数組み合わせて使用するには、-m state --state INVALID,NEWのようにカンマで区切ります。

- macモジュール— ハードウェアMACアドレスの比較を有効にします。

  macモジュールは以下のようなモジュールを有効にします：

  - --mac-source — パケットの送信元であるネットワークインターフェイスカードのMACアドレスを比較します。この規則からMACアドレスを除外するには、--mac-source比較オプションの後に感嘆符(!)を付けます。

他のモジュールで使用できる比較オプションを確認するには、iptablesのmanページを参照して下さい。

16.3.6. ターゲットオプション

パケットが特定の規則を満たすと、規則はそのパケットのさまざまな行方を決定し場合によっては追加動作をさせることも可能です。各チェーンにはデフォルトのターゲットがあり、そのチェーンの規則を満たすパケットがない場合か、あるいはパケットが満たした規則のいずれもがターゲットを指定していない場合に使用されます。

標準(デフォルト)のターゲットには以下のようなものがあります：

- <*user-defined-chain*> — <*user-defined-chain*>はテーブル内のユーザー定義のチェーンの名前で入れ換えます。このターゲットはパケットをターゲットチェーンに渡します。
- ACCEPT—パケットが送信先または別のチェーンに移動することを許可します。
- DROP — パケットを送信したシステムには何も通知せずにパケットをドロップします。パケットを送信したシステムは不具合の報告も受けません。
- QUEUE — ユーザースペースのアプリケーションで処理されるようにパケットをキューに登録します。

- RETURN — 現在のチェーン内の規則に対するパケットのチェックを停止します。RETURNターゲットのパケットが別のチェーンから呼び出されたチェーンの規則を満たす場合、そのパケットは最初のチェーンに戻され、そこで規則チェックが再開されます。組み込み型のチェーンでRETURN規則を使用していてパケットが前のチェーンに戻れない場合は、現在のチェーンのデフォルトターゲットによって処理が決定されます。

これらの標準ターゲットのほかに、「ターゲットモジュール」と呼ばれる拡張機能で各種のターゲットを使用できます。比較オプションモジュールの詳細については、項16.3.5.4を参照してください。

多くの拡張ターゲットモジュールがありますが、ほとんどは特定のテーブルか状況のみに適用されます。デフォルトでRed Hat Linuxに含まれているターゲットモジュールでよく使用されるものには、次のようなものがあります：

- LOG — 規則を満たすすべてのパケットを記録します。パケットを記録するのはカーネルなので、出力先は/etc/syslog.confファイルによって決定されます。デフォルトの出力先は、/var/log/messagesファイルです。

  LOGターゲットでは、さまざまなオプションを使用して記録を行う方法を指定できます：

  - --log-level — イベントを記録する優先順位を設定します。優先順位の一覧については、syslog.confのmanページを参照してください。
  - --log-ip-options—IPパケットのヘッダーで設定されているオプションを記録します。
  - --log-prefix — ログを記録するときに、行の先頭に29文字までの文字列を設置します。これは、パケットの記録とともに使用するsyslogフィルタを作成する場合にも便利です。
  - --log-tcp-options — TCPパケットのヘッダーで設定されているオプションを記録します。
  - --log-tcp-sequence—パケットのTCPシーケンス番号を記録します。

- REJECT — パケットを送信したシステムにエラーパケットを送り返して、パケットをドロップします。

  REJECTターゲットでは、--reject-with <*type*>(<*type*>は拒絶のタイプ)オプションを使用して、エラーパケットと共に返送される詳細情報を指定できます。他のオプションが使用されていない場合、メッセージport-unreachableがデフォルトで与えられる<*type*>のエラーです。使用可能な<*type*>オプションの総合一覧はiptablesのmanページで御覧下さい。

natテーブルを使用したIPマスカレード、又はmangleテーブルを使用したパケット変更で役にたつものなど、その他のターゲット拡張の幾つかは、iptablesのmanページを参照してください。

16.3.7. リストオプション

デフォルトのリストコマンドiptables -Lは、デフォルトのフィルタテーブルの現在のチェーンについて非常に基本的な概要情報を提供します。追加のオプションは更に詳細情報を提供します：

- -v—各チェーンがチェックしたパケット数とバイト数、各規則を満たしたパケット数とバイト数、特定の規則に適用されるインターフェイスなど、冗長な情報を出力します。
- -x—数値の正確な値を出力します。負荷が大きいシステムでは、特定のチェーンか、あるいは規則がチェックしたパケット数とバイト数の終わりにK（キロ）、M（メガ）、G（ギガ）を付けて表現を省略する場合があります。このオプションを指定すると、正確な値が出力されます。
- -n—IPアドレスとポート番号を、デフォルトのホスト名とネットワークサービスの形式ではなく数値形式で出力します。
- --line-numbers—各チェーンの規則の横にチェーン内の順序番号を出力します。このオプションは、特定の規則を削除する場合や規則を挿入する場所を探す場合に便利です。
- -t — テーブル名を指定します。

16.4. iptablesの情報の格納

iptablesコマンドで作成した規則は、メモリー(RAM)に格納されます。したがって、iptablesの規則を設定した後でシステムを再起動すると、その規則は失われます。ネットフィルタ規則がシステムの再起動毎に残るようにするには、その内容が保存される必要があります。これを実行するには、rootでログインして次のように入力します：

```
/sbin/service iptables save
```

これによって、iptablesの初期化スクリプトは/sbin/iptables-saveプログラムを実行し、現在のiptablesの設定を/etc/sysconfig/iptablesファイルに書き込みます。このファイルは、ルートユーザー以外は読めないようにしてください。

次にシステムをブートすると、iptablesの初期化スクリプトは/sbin/iptables-restoreコマンドを実行して、/etc/sysconfig/iptablesファイルに保存されていた規則を再び適用します。

/etc/sysconfig/iptablesファイルに保存する前に新しいiptablesの規則をテストするのは良い考えですが、別のシステムにあるファイルからiptablesの規則をコピーすることもできます。この方法を使用すると、iptablesの規則を複数のマシンに同時にすばやく配布することができます。

重要

他のマシンに/etc/sysconfig/iptablesファイルを分配する場合は、その後で/sbin/service iptables restartと入力して、新しい規則を有効にする必要があります。

16.5. その他のリソース

iptablesを用いたパケットフィルタリングに関する追加情報は以下の情報源を参照して下さい。

16.5.1. インストールされるドキュメント

- man iptables — 各種のコマンド、パラメータ、オプションについての広汎な説明が含まれています。

16.5.2. 役に立つWebサイト

- http://netfilter.samba.org — LinuxのIPファイアウォールの保守を行っているRusty Russellによる各種の有益なガイドや遭遇する可能性がある問題に対処するためのFAQなど、iptablesに関する多彩な情報が含まれています。このサイトのHOWTOドキュメントでは、基本的なネットワークの概念、2.4カーネルのパケットフィルタリングとNATの設定などの課題を扱っています。
- http://www.linuxnewbie.org/nhf/Security/IPtables_Basics.html — パケットがLinuxカーネルを通過する様子に関する非常に基本的で一般的な説明と、簡単なiptablesコマンドを作成する方法が含まれています。
- http://www.redhat.com/support/resources/networking/firewall.html — このページは、様々なパケットフィルター情報の最新リンクがあります。

第17章
Kerberos

kerberosは、ネットワークへのユーザー認証をする為、対称鍵暗号法を用いた、MITで開発されたネットワーク認証プロトコルです。—ネットワーク上でパスワードを送る必要がなくなります。Kerberosを使用してネットワークサービスにユーザーを認証すると、認証のないユーザーが、ネットワークのトラフィックを監視してパスワードを取り込もうとしても適切に阻止されます。

17.1. Kerberosの利点

従来のネットワークシステムのほとんどはパスワードベースの認証体系を使用しています。この様な認証体系は、特定のネットワークサーバにユーザーがユーザー名とパスワードを供給して認証を得る手続きを要求します。残念ながら多くのサービス内で認証情報の送信は暗号化されていません。この様な認証体系を安全にする為には、ネットワークが外部からアクセス不可能にし、そのネットワーク上の全てのコンピュータとユーザーは信頼できて、信頼する価値を持つ必要があります。

この様にしたケースでさえも、インターネットに接続されてしまうと、そのネットワークはもう安全でない可能性があります。アクセスを取得するアタッカーは、簡単なパケット解析(パケットスニッファとも呼ぶ)を使用し、上述の方法で転送されたユーザー名とパスワードを盗み取り、ユーザーアカウントと全体のセキュリティ基盤が被害を受けます。

Kerberosのおもな設計目標は、ネットワーク経由で送信されるプレーンテキストのパスワードを無くすと言う事です。Kerberosを適切に使用することで、パケットスニッファが与えるネットワーク上の脅威を効率的に抹消します。

17.1.1. Kerberosの欠点

Kerberosにより、一般的で、極度のセキュリティ脅威は除去できますが、さまざまな理由により、Kerberosを実装することは難しいことがあります:

- `/etc/passwd`や`/etc/shadow`といった、標準的なUNIXパスワードデータベースからKerberosパスワードデータベースへとユーザーのパスワードを移行するのは、単調な作業となる可能性があります。というのは、この移行を自動的に実行するメカニズムが存在しません。これに関する詳細は、以下のURLでオンライン「Kerberos FAQ」の中の質問番号2.23を参照して下さい:

  http://www.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html.

- Kerberosは、Red Hat Linuxを実行するほとんどのサーバーで使用されているPAM（Pluggable Authentication Modules）システムとの互換性を部分的にしか持っていません。この問題の詳細については、項17.4を参照してください。

- アプリケーションにおいてKerberosを使用するためには、Kerberosのライブラリを正しくコールするために、アプリケーションのソースを修正しなければいけません。アプリケーションによっては、アプリケーションサイズ、又はその設計の為に深刻な問題になることがあります。その他の互換性のないアプリケーションには、サーバとクライアント側が通信出来るように変更する必要があります。ここでも又、広範囲に及ぶプログラミングが必要になります。デフォルトではkerberosをサポートしていないクローズドソースアプリケーションと併用することもしばしば問題となります。

- Kerberosでは、信用できるユーザーが信用できないネットワーク上で信用できないホストを使用していることを想定します。Kerberosの第一目標は、プレーンテキストのパスワードがそのネットワーク経由で送信されないようにすることです。ただし、不正なユーザーがいずれかのホスト、—特に*key distribution center*(*KDC*)— と呼ばれる認証のためのチケットを発行するホストにアクセスしていると、Kerberosの認証システム全体が被害を受ける危険性があります。

- Kerberosは「ALL or Nothing」のソリューションです。ネットワークでKerberosを使用することに決めた場合には、Kerberosによる認証を使用しないサービスにパスワードを送付すると、パケットスニッファーに奪われる危険を犯す事を忘れないで下さい。そうなると、ネットワークでKerberosを使用するメリットはまったくありません。Kerberosによってネットワークを保護するためには、プレーンテキストのパスワードを送信するクライアント/サーバアプリケーションをすべて*kerberos*化するか、ネットワークにおいてこのように不安定なアプリケーションの使用をやめるか、どちらかを行わなければいけません。

17.2. Kerberosの用語

Kerberosにも様々なサービスを定義するための独自の用語があります。Kerberosの機能を理解する前に知っておく必要のある用語を以下に示します。

暗号文

‘ 暗号化されたデータ。

クライアント

‘ Kerberosからチケットを得る事ができるネットワーク上の実体名目(ユーザー、ホスト、アプリケーションなど)。

証明書キャッシュまたはチケットファイル

‘ ユーザーと各種ネットワークサービスの間の通信を暗号化するための鍵を含むファイル。Kerberos 5は、その他のキャッシュタイプ（たとえば共有メモリ）を使用するための枠組みを提供しますが、ファイルの方が徹底してサポートされています。

暗号ハッシュ

‘ ユーザの認証に使う一方向ハッシュ。プレインテキストよりセキュアですが、経験豊富なクラッカーにとっては、容易に解読されるものです。

GSS-API

‘ 汎用セキュリティサービスアプリケーションプログラムインターフェイス(GSS-API) [RFC-2743]は一連の機能セットで、裏で動作している機構の特別な知識がなくても、クライアントがサーバへの認証に使用し、サーバはクライアントを認証するのに使用できます。ネットワークサービス(例；IMAP)がGSS-APIを使用していると、Kerberosを使って認証が出来ます。

鍵

‘ データを暗号化/複号化する際に使用されるデータ。暗号化されたデータの復号化は、正しい鍵(又は超越した想像力)なしでは不可能です。

KDC (Key Distribution Center)

‘ Kerberosのチケットを発行するサービス。通常はTicket Granting Serverと同一のホスト上で動作します。

key table 又はkeytab

‘ 暗号化されていないプリンシパルとその鍵の一覧を含むファイル。サーバーは、`kinit`を使用せずに、keytabファイルから必要な鍵を取り出します。デフォルトのkeytabファイルは`/etc/krb5.keytab`です。KDC 管理サーバ、`/usr/kerberos/sbin/kadmind`のみが、その他のファイルを使用するサービスです。（それは`/var/kerberos/krb5kdc/kadm5.keytab`を使用します）。

kinit

‘ ログインしているプリンシパルはkinitコマンドにより、初期のTGT(Ticket Granting Ticket)を取得し、キャッシュ保存できます。kinit コマンドの使用についての詳細はそのman ページを御覧下さい。

プリンシパル

‘ プリンシパルはKerberosを使用して認証できる、独特のユーザー名、あるいはサービス名。プリンシパル名の形式は、root[/instance]@REALMです。一般的なユーザーの場合、rootは、ユーザーのログインIDと等しくなります。instanceは、オプションです。プリンシパルが1つのインスタンスを持つ場合、インスタンスとrootをスラッシュ（"/"）で区切ります。空の文字列（""）も実際には有効なインスタンスとなります（デフォルトのNULL インスタンスとは異なります）が、使用すると混乱のもとになります。1つのrealmに属するすべてのプリンシパルは、独自の鍵を持ちます。鍵はパスワードから導き出されるか、サービス用にランダムに設定されます。

realm

‘ Kerberosを使用したネットワーク。KDCと呼ばれる一台または少数台のサーバーと非常に多数になる可能性のあるクライアントから構成されます。

サービス

‘ ネットワーク経由でアクセスされるプログラム。

チケット

‘ 特定のサービスに関してクライアントの身元を識別するための、一時的な電子証明書のセット。

TGS (Ticket Granting Service)

‘ ユーザーが実際にアクセスするために使用する目的のサービスに対し、チケットを発行するサーバー。TGSは通常、KDCと同一のホスト上で動作します。

TGT (Ticket Granting Ticket)

‘ あらためてKDCに対して要求しなくても、クライアントが追加のチケットを取得できるようにする特殊なチケット。

暗号化のないパスワード

‘ プレインテキストの、人間に読み取れるパスワード。

17.3. Kerberosの機能

Kerberosは、他の認証方法とは異なります。個別のユーザーを個別のネットワークサービスに認証するのではなく、Kerberosは対称暗号法と信用できるサードパーティ—Key Distribution Center (KDC)として知られています。—を使用して一連のネットワークサービスへユーザーを認証します。ひとたびKDCにユーザーが認証されると、その通信固有のチケットをユーザのマシンに送信し、パスワードを使う認証をユーザに求める代わりに全てのKerberos化されたサービスはユーザーマシン上のこのチケットを探します。

kerberos化されたネットワーク上で、ユーザーが自分のワークステーションにログインすると、ユーザーのプリンシパルがKey Distribution Centerに送信され、Ticket Granting Service (TGS)からのTicket Granting Ticket(TGT）を要求します。この要求は、ログインプログラムによって送信するので、ユーザーには透過的です。またログインした後でユーザーがkinitプログラムを使って送信したりすることができます。

KDCは、データベース内にプリンシパルが存在するかどうかをチェックします。プリンシパルが見つかった場合、KDCはTGSに指示してTGTを作成し、ユーザーの鍵を使用してそのTGTを暗号化してからユーザーへと返信します。

クライアントマシン上のログインか、kinitプログラムのどちらかが、ユーザーの鍵（ユーザーのパスワードから計算されます）を使用してTGTを復号化します。ユーザーの鍵は、クライアントマシン上でのみ使用され、ネットワークには送信されません。

TGTは一定の時間(通常10時間)が経過すると有効期限が切れるように設定されていますが、クライアントマシンの証明書キャッシュの中に保存されます。有効期間が設定されているのは、TGTが盗まれたとしても、侵入者が使用できるのを一定の時間に限定するためです。一度TGTが発行されると、TGTの有効期限が切れるか、ログアウトして再度ログインするまで、ユーザーはKDCへ再度パスワードを入力する必要はありません。

ユーザーがネットワークサービスにアクセスする必要が生じた場合は、クライアントソフトウェアがTGTを使って、TGS（Ticket Granting Service）にそのサービス用の新規のチケットを要求します。サービス用のチケットはそのサービスに対し、透過的にユーザーを認証するのに使用されます。

警告

Kerberosシステムではいつでもネットワーク上のどのユーザにも、Kerberos化されていないサービスに対して、プレインテキストでパスワード認証を送ると、被害を受けることになります。したがって、Kerberos化されていないサービスの使用は推薦できません。このようなサービスの例として、telnetやftpがあります。理想的ではありませんが、他のセキュアなプロトコル、OpenSSHやSSL等で安全なサービスを使う事をお勧めします。

これは、どのようにKerberos認証がネットワーク上で動作するかを概観したに過ぎません。Kerberos認証に関する探求には、項17.7を参照してください。

注意

Kerberosは、正しく動作するために特定のネットワークサービスに依存しています。まず、Kerberosはネットワーク上のマシン群の間で、ほぼ正確に時計の同期が取られていることを必要とします。ネットワークに対し、ntpdなどの時計の同期化プログラムをセットアップする必要があります。ntpdの設定に付いての詳細は/usr/share/doc/ntp-<*version-number*>/index.htmでネットワークタイムプロトコルサーバの設定法を御覧下さい。

また、Kerberosの特定の部分がDNS（Domain Name Service）に依存していますで、ネットワーク上のDNSエントリとホストがすべて正しく設定されていることを確認してください。この詳細については/usr/share/doc/krb5-server-<*version-number*>の中で、PostScript形式とHTML形式で提供されている*Kerberos V5 System Administrator's Guide*を御覧下さい。

17.4. Kerberos とPAM

現在のところ、kerberos化されたサービスは、まったくPAM（Pluggable Authentication Modules）を利用していません。— kerberos化されたサービスは完全にPAMをバイパスします。ただし、PAMを使用したアプリケーションでは、pam_krb5モジュール（pam_krb5パッケージで提供されます）がインストールされているならば、認証用にKerberosを利用できます。pam_krb5パッケージには、loginやgdmなどのサービスが、ユーザーを認証したり、ユーザーのパスワードを使って初期証明書を取得したりすることを可能にする、サンプルの設定ファイルが含まれています。ネットワークサービスに対するアクセスが、常にkerberos化されたサービスまたはIMAPなどのGSS-APIを使用したサービスを使用して行われるならば、そのネットワークはかなり安全だと考えることができます。

管理者は、無差別にネットワークサービスでユーザーがKerberosパスワードを使用した認証を得る許可をしないように注意する必要があります。これらのサービスで使用するプロトコルのほとんどは、ネットワーク経由で送信する前にパスワードを暗号化しませんのでKerberosシステムの価値を無駄に

してしまいます。例としては、ユーザーにTelnet上でKerberosパスワードを使用して認証できる許可を与えるべきではありません。

次の章では、基本的なKerberosサーバーの構築方法を説明します。

17.5. Keberos 5サーバーの設定

Kerberosを構築するには、最初にサーバをインストールします。スレーブサーバーを構築するには、マスターとスレーブサーバー関係を構築する詳細が*Keberos 5 Installation Guide* (`/usr/share/doc/krb5-server-`<バージョン番号> ディレクトリの中)にありますので参照してください。

基本的なKerberosサーバを設定するには、以下のステップに従います：

1. Kerberos 5をインストールする前に、時計同期とDNSがサーバーで正常に動作している事を確認してください。Kerberosサーバーと各クライアント間の時計同期は特に注意してください。もし、サーバーとクライアントの時計が5分以上異なっていたら、(これはデフォルトのKeberos 5設定時間です。)Kerberosクライアントはサーバーに認証されません。この時計同期は、正規のユーザーと偽って、古いKerberos チケットを用いるアタッカーを防止するために必要です。

   Keberosを用いていない場合でも、ネットワークでクライアント/サーバー互換のNTP(Network Time Protocol)の設定をすべきです。Red Hat Linuxには、簡単にインストールできる`ntp`パッケージが含まれています。Network Time Protocolサーバの設定に関する詳細には`/usr/share/doc/ntp-`<*version-number*>`/index.htm`を参照して、NTPに関するその他の情報については http://www.eecis.udel.edu/~ntpを御覧下さい。

2. KDCが実行するように決定している専用マシンに、`krb5-libs`, `krb5-server`, `krb5-workstation`をインストールします。このマシンは特にセキュアであることが必要です。—可能なら、KDC以外の他のサービスは実行しないことが望まれます。

   Kerberosを管理するのに、GUI(Graphical User Interface)を使いたい場合は、`gnome-kerberos`パッケージもインストールしてください。このパッケージには、`krb5`というチケットを管理するGUIツールが含まれています。

3. realm名とドメイン-realm間マッピングを反映するためには`/etc/krb5.conf`と`/var/keberos/krb5kdc/kdc.conf`設定ファイルを編集してください。簡単なreamlは*`EXAMPLE.COM`*と*`example.com`*の例をドメイン名—大文字か小文字か、正しいフォーマットを確かめて下さい—で置き換え、そしてKDCを*`kerberos.example.com`*からKerberosサーバー名に変更することで、構築できます。慣習的に、realm名は大文字で、DNSホスト名とドメイン名は小文字です。これらファイル形式の詳細については、該当するマニュアルページを参照ください。

4. シェルプロンプトから`krb5_util`ユーティリティを使ってデータベースを作成します：
   ```
   /usr/kerberos/sbin/kdb5_util create -s
   ```
   `create`コマンドはKerberos realmの鍵を格納するために使用するデータベースを作成します。`-s`スイッチは、マスターサーバー鍵を格納する*stash*ファイルを作成します。鍵を読むためのstashファイルが無い場合は、Kerberosサーバー(`krb5kdc`)は起動する度に、ユーザーにマスターサーバーパスワード(鍵を再生成するのに使われる)の入力を促します。

5. `/var/kerberos/krb5kdc/kadm5.acl`ファイルを編集します。このファイルはどのプリンシパルがKerberosデータベースにどのレベルでアクセスするかを決める`kadmind`で使われます。多くの場合、以下の様に一行で編集できます：
   ```
   */admin@EXAMPLE.COM  *
   ```
   ほとんどのユーザーは、データベース上に単一のプリンシパル(例えば*joe@EXAMPLE.COM*の例ように*NULL*あるいは空で）で表示されます。この設定を用いて、第２のプリンシパルを持っているユーザーは(例えば*joe/admin@EXAMPLE.COM*のように)*admin*の例を使ってrealmのKerberosデータベース上で全権限を使う事ができます。

一旦、kadmindがサーバー上で起動すると、realm内のクライアントやサーバーからkadminを起動する事で、どのユーザーもそのサービスにアクセスできます。しかし、kadm5.aclファイルに記載されているユーザだけが、自身のパスワード変更以外なら、どのような変更もデーターベースに対して行えます。

注意

kadminユーティリティはネットワーク越しにkadmindサーバーと通信しており、認証を扱うためにKerberosを使います。当然、ネットワーク越しにサーバーに接続する前に、ネットワークを管理する第一プリンシパルを作成する必要があります。第一プリンシパルを作成するにはkadmin.localコマンドを使用します。これは特にKDCと同じホストで使用するように設計してあり、認証用にKerberosを使用しません。

第一プリンシパルを作成するには、KDCターミナルで次のkadmin.localコマンドを入力します：

```
/usr/kerberos/sbin/kadmin.local -q "addprinc username/admin"
```

6. 以下のコマンドでKerberosを起動します：
```
/sbin/service krb5kdc start
/sbin/service kadmin start
/sbin/service krb524 start
```
7. addprincコマンドとkadmin使用してユーザーのためのプリンシパルを追加します。kadminとkadmin.localはKDCのコマンドラインインターフェイスです。この中では、kadminプログラムを起動した後に多くのコマンドが利用できます。詳細はkadminのmanページを御覧下さい。
8. システムがチケットを発行できるか確かめます。最初に、kinitを実行してチケットを生成し、証明書キャッシュファイルに格納します。それからklistを使用してキャッシュ内の証明書一覧を表示して、その後、kdestroyを用いて、キャッシュとその中身の証明書を破棄します。

注意

デフォルトでは、kinitは、最初にシステム(Kerberosサーバーではない）にログインした時に使ったアカウントのログインユーザー名を用いて、認証しようとします。そのシステムのユーザー名がKerberosデータベースのプリンシパルと合致していない場合は、エラーメッセージが表示されます。この場合は、コマンドラインの引数としてプリンシパルの名前をkinitに与えます。(kinit*principal*)。

以上のステップを完了すると、Kerberosサーバーは起動し作動してるはずです。次は、Kerberosクライアントの設定をします。

17.6. Kerberos 5クライアントの設定

Kerberos 5クライアントの設定は、サーバーの設定に比べて少なくて済みます。最小限、クライアントパッケージをインストールして、正しいkrb5.conf設定ファイルを各クライアントに供給してください。Kerberos化したrshとrlogin にも幾らかの設定変更が必要です。

1. KerberosクライアントとKDCで時間同期が達成されていることを確認してください。詳細は項17.5を参照してください。さらに、Kerberosクライアントプログラムを設定する前に、Kerberosクライアント上でDNSが適切に動作している事を確証して下さい。
2. 全てのクライアントマシンにkrb5-libsとkrb5-workstation パッケージをインストールしてください。それぞれのクライアントには、1つのバージョンの /etc/krb5.confを供給する必要があります。通常これは、KDCで使用されるのと同じkrb5.confファイルです。

3. realm内のそれぞれのワークステーションに、Kerberos化したrshやrloginを用いて接続するユーザーを許可する前に、ワークステーションにxinetdパッケージをインストールしてKerberosデータベース内に自らのホストプリンシパルを作成する事が必要です。kshdとklogind サーバープログラムも、サービスプリンシパル用の鍵にアクセスする必要があります。

   kadminを使って、KDC上のワークステーション用のホストプリンシパルを追加します。この場合の例としては、ワークステーションのホスト名があります。-randkey オプションをkadminのaddprincコマンドに付けて使用すると、プリンシパルが作成され、ランダム鍵が割り当てられます：
   ```
   addprinc -randkey host/blah.example.com
   ```

   これで、プリンシパルを作成できました。ワークステーション上で、kadminを起動し、そしてkadminの中のktaddコマンドを使って、ワークステーション用の鍵を引き出せます：
   ```
   ktadd -k /etc/krb5.keytab host/blah.example.com
   ```

4. 他のkerberos化したネットワークサービスを使用したい場合は、それを起動しなければなりません。以下に一般的なkerberos対応のサービスの一覧とそれらを有効にする方法を示します：

   - rshとrlogin — kerberos化したrshとrloginを使うためには、klogin、eklogin、kshellが有効でなければなりません。
   - Telnet — kerberos化したTelnetを使用するには、krb5-telnetを有効にする必要があります。.
   - FTP — FTP アクセスを用意するには、ftpのrootでプリンシパル用の鍵を作成し、引き出す必要があります。FTPサーバの完全修飾形のホスト名への事例を設定して、それからgssftpを有効にします。
   - IMAP — imapパッケージに含まれているIMAPサーバは、正しい鍵を/etc/krb5.keytabで見付けることが出来るなら、Kerberos 5を利用して、GSS-API認証を使います。プリンシパルのrootはimapである必要があります。
   - CVS — CVSのkerberos化したgserverは、cvsのrootでプリンシパルを使用し、その他の面ではCVSpserverと全く同じです。

   サービスを可能にする詳細は*Red Hat Linux* カスタマイズガイドのサービスアクセス管理章を参照してください。

17.7. その他のリソース

Kerberosに関する詳細な情報は、以下を参照してください。

17.7.1. インストールされているドキュメント

- /usr/share/doc/krb5-server-<*version-number*> — *Kerberos V5 Installation Guide*及び*Kerberos V5 System Administrator's Guide*をPostScript とHTML 形式で参照できます。krb5-server パッケージをインストールしておく必要があります。
- /usr/share/doc/krb5-workstation-<*version-number*> — *Kerberos V5 UNIX User's Guide*をPostScript とHTML 形式で参照できます。krb5-workstationパッケージをインストールしておく必要があります。

17.7.2. 役に立つWebサイト

- http://web.mit.edu/kerberos/www — MITの*Kerberos: The Network Authentication Protocol*についてのホームページ。

- http://www.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html — Kerberosに関するFAQ(よくある質問とその回答)。
- ftp://athena-dist.mit.edu/pub/kerberos/doc/usenix.PS — Jennifer G. Steiner、Clifford Neuman、Jeffrey I. Schiller共著のPostScript 版*Kerberos: An Authentication Service for Open Network Systems*。Kerberosについて記述したオリジナルの論文です。
- http://web.mit.edu/kerberos/www/dialogue.html — *Designing an Authentication System: a Dialogue in Four Scenes*。オリジナルはBill Bryant著の1988年版で、改訂版はTheodore Ts'o著の1997年版です。この文献は、Kerberos型の認証システムの作成を通じて思索した2人の開発者の対話を記述したものです。会話形式の議論を行っているため、本書はKerberosの初心者にとって優れた手引となっています。
- http://www.ornl.gov/~jar/HowToKerb.html — *How to Kerberize your site* は、ネットワークをKerberos化する為の良い参考書となります。
- http://www.networkcomputing.com/netdesign/kerb1.html — *Kerberos Network Design Manual* は、Kerberosシステムの充実した概要が提供されています。

第18章

SSHプロトコル

SSH™の使用でユーザーは、リモートからホストシステムへログインが可能となります。FTPやTelnetとは異なり、SSHはログインセッションを暗号化しますので、侵入者がクリアテキストのパスワードを取り出すことは有り得ません。

SSHには、**telnet**や**rsh**などの古くて低レベルセキュリティのリモートホスト用ログインアプリケーションを置き換える設計がなされています。これに関連した**scp**と呼ばれるプログラムは、**rcp**などホスト間でファイルをコピーする設計の古いプログラムを置き換えます。これらの古い方のアプリケーションはクライアントとサーバー間で送信されるパスワードを暗号化しない為、可能な限りこれらを避けて下さい。リモートシステムにログインする時に安全な手段を使用することは、クライアントシステムとリモートホスト両方のリスクを減少させます。

18.1. SSHの特徴

SSH (これはSecure *SH*ellの略)は、クライアント/サーバのアーキテクチャを使用して2つのシステム間で安全な接続を確立するためのプロトコルです。

SSHプロトコルでは、次の保護手段が取られます：

- 初期接続の後、クライアントは以前に接続したのと同じサーバに接続していることを確証できます。
- クライアントは強力な128ビット暗号を使用して、サーバに認証情報を送付します。
- セッションの間に送受信されたデータの全ては、128ビット暗号を使用して送信されますので、それを横取りしても復号と読み取りが非常に難しくなります。
- クライアントはサーバからX11アプリケーション[1]を伝送できます。この*X11*フォワーディングと呼ばれる技術はネットワーク上でグラフィカルアプリケーションを使用する為の安全な手段を提供します。

SSHプロトコルは、それが送受信するすべてを暗号化するため、そのままでは不安全なプロトコルを安全にするために使用されます。ポートフォワーディングと呼ばれる技術を使用すれば、SSHサーバがPOPなどの普段は安全でないプロトコルを安全にする経路になり、全体的にシステムとデータのセキュリティを向上します。

Red Hat Linuxには、一般的なOpenSSHパッケージ（`openssh`）、OpenSSHサーバーパッケー(`openssh-server`)及びクライアントパッケージ(`openssh-clients`)も含まれています。OpenSSHのインストールと活用に付いての案内は*Red Hat Linux* カスタマイズガイドの中にある*OpenSSH*というタイトルの章を御覧下さい。また、OpenSSHパッケージはOpenSSLパッケージ(`openssl`)を必要とすることに注意して下さい。OpenSSLは、OpenSSHが暗号化された通信を用意できるようにするための幾つかの重要な暗号化法ライブラリをインストールします。

SSHプロトコルを使用できるクライアントプログラムとサーバープログラムはたくさんあります。現在使用されている主要なほとんどすべてのオペレーティングシステムに合わせて、各種のSSH クライアントバージョンが用意されています。

---

1. X11 とはX11R6ウィンドウディスプレイシステムのことで、伝統的にX と呼ばれます。Red Hat Linuxには、X11R6を基本にした、広範囲で使用されるオープンソースのX ウィンドウシステムである**XFree86**が含まれています。

18.1.1. SSHを使用する理由

悪質なコンピュータユーザーは、あるシステムへのアクセスを得る為にネットワークトラフィックに対して接続破壊、干渉、経路変更などを可能にするさまざまなツールを所持しています。一般的な表現ではこれらの脅威は以下のカテゴリーに分類できます:

- *2*つのシステム間の通信の干渉 — この手口では、侵入者は、ネットワーク上の通信機構の間のどこかに介在し、その間を通過する情報をコピーします。侵入者は、干渉し情報を横取りするか、又はその情報を変更してその本来の受信者に送り付ける可能性があります。

  この攻撃は、パケットスニッファ(一般的なネットワークユーティリティ)を使用してマウントできるものです。

- 特定ホストの偽装 — この策略の場合は、侵入者のシステムが通信の本来の受信者として振舞うように設定されています。この策略が成功すれば、ユーザーのシステムは間違った相手と通信していることに気が付きません。

  この攻撃は、DNSポイゾニング(汚染)、[2] 又はIPスプーフィングとして知られる技術でマウントできるものです。[3]

どちらの手口でも重要な機密情報を干渉することができ、その干渉が悪意のある物である場合、悲惨な結果となりえます。

SSHがリモートのシェルログインとファイルコピーの為に使用されるのであれば、これらのセキュリティ脅威は大幅に減少できます。これはSSHのクライアントとサーバーが自身の身元を証明するのにデジタル署名を使用する為です。さらには、クライアントとサーバー間のすべての通信が暗号化されています。ローカルとリモートのシステムのみが持つ鍵を使用して暗号化してあるため、この通信のどちらかの身元をスプーフ(偽装) しようとしても成功しません。

18.2. SSH プロトコルのバージョン

SSHプロトコルを使用することで、このプロトコルの仕様に合わせて作成されたクライアントプログラムやサーバープログラムは安全に通信し合い、相互に使用し合うことができます。

現在SSHには2つのバージョンが存在します。SSHバージョン1は、特許を持つ数種の暗号化アルゴリズムを使用(但し、特許の幾つかは期限が切れています)しますが、通信ストリームに侵入者のデータ挿入を認める可能性のあるセキュリティホールへの弱味を持ちます。Red Hat Linuxに収納されているOpenSSHセットはデフォルトでSSHバージョン2を使用しますが、バージョン1もサポートします。

重要

可能な限りはSSHのバージョン2互換のサーバー及びクライアントを使用されることが推奨されます。

18.3. SSH接続のイベントシーケンス

以下の一連のイベントにより2つのホスト間のSSH通信の一貫性を保護する援助をします。

2. DNSポイゾニングは、侵入者がDNSサーバーをクラックしてクライアントシステムを悪意を持って複写偽装したホストに向けることで発生します。
3. IPスプーフィングは侵入者が、ネットワーク上で信頼されたホストから送信されたように偽装したネットワークパッケージを送信することにより発生します。

- 暗号化したハンドシェークがなされてクライアントは正しいサーバーと通信していることを証明できます。
- クライアントとリモートホスト間の接続上のトランスポート層は対称暗号を使って暗号化します。
- クライアントが自分自身をサーバーに対して認証します。
- リモートクライアントは、ここで暗号化した接続を介してリモートホストと安全に交流が出来ます。

18.3.1. トランスポート層

トランスポート層のおもな役割は、認証時と認証後に2つのホスト間で安全な通信を確立することです。トランスポート層は、データの暗号化と復号を処理して、データパケットを送受信する際に一貫性を確保することによってこれを実現します。さらに、トランスポート層は、圧縮も提供できるため、情報の転送をスピードアップします。

SSHクライアントがサーバーと通信すると、2つのシステムがトランスポート層を正しく構築できるように、鍵となる情報が交換されます。この交換の間、以下のステップが実行されます：

- 鍵の交換する
- 公開鍵暗号化アルゴリズムを決定する
- 対称暗号化アルゴリズムを決定する
- メッセージ認証アルゴリズムを決定する
- 使用するハッシュアルゴリズムを決定する

鍵を交換する際、サーバーは独自のホスト鍵を使ってクライアントに自分自身を証明します。このクライアントが以前にこの特定のサーバーと1度も通信したことがないと、クライアントはサーバーの鍵を知りませんので、接続はできません。OpenSSHは、ユーザーが通知を受け新しいホスト鍵の受理を確認した後に、サーバーのホスト鍵を承認するすることでこの問題を処理しています。これで、次回からの接続で、サーバーのホスト鍵と、クライアントに保存されているバージョンを突き合わせることができるため、クライアントが本当に目的のサーバーと通信していることを証明できます。それ以降、ホスト鍵が適合しなくなった場合、接続を達成するにはユーザーはクライアントに保存されているバージョンを削除する必要があります。

用心

ローカルシステムは目的のサーバーと侵入者がセットアップした偽りのサーバーとの違いがわからないので、最初に接続した時点で、侵入者がサーバーを偽装することは可能です。これを防ぐには、最初に接続する前に、又はホスト鍵の不適合の時点でサーバー管理者と連絡を取り、新しいSSHサーバーの一貫性を確かめるべきです。

SSHは、ほとんどすべての種類の公開鍵アルゴリズムやエンコーディング形式を使用できるように作られています。最初の鍵交換で交換したり秘密値を共有するハッシュ値を生成し、2つのシステムは、認証と、この接続を介して送信されるデータの保護を行うために、新しい鍵とアルゴリズムの計算をただちに開始します。

特定の鍵とアルゴリズムをつかって、ある程度のデータを送信した後(正確な量は、SSHの実装に依存します。)、別のハッシュ値と共有秘密値のセットを生成する別の鍵への交換が起こります。その為、仮に侵入者がハッシュ値と共有秘密値を解読できたとしても、その情報は、限定された時間だけしか役に立ちません。

18.3.2. 認証

トランスポート層が、2つのシステム間で情報の受け渡しを行うための安全なトンネルを作成し終わると、サーバーは、秘密鍵でエンコードした署名の使用や、パスワードの入力などのサポートされている各種の認証方法をクライアントに伝えます。クライアントは、サポートされている方法の1つを使って、サーバーに対し自分自身を認証します。

SSHサーバーとクライアントは、さまざまなタイプの認証をサポートするように構成でき、その認証方法は、各側で最適に制御することができます。サーバーは、セキュリティモデルに基づいて、どの暗号化方法をサポートするかを指定することができ、クライアントは、利用できるオプションの中から、認証方法を試行する順序を選ぶことができます。SSHプロトコル層の安全性という性質のおかげで、ホストとパスワードベースの認証など、安全性に欠けるような認証方法でも安心して使用することができます。

18.3.3. チャンネル

SSHトランスポート層で認証が正常終了すると、マルチプレキシングと呼ばれる技術により複数のチャンネルが開きます。[4]これらのチャンネルで、各種の端末セッション用と送信されたX11セッション用の通信が処理されます。

クライアントとサーバーは両方とも新しいチャンネルを作成できます。各チャンネルには、それぞれの側で別々の番号が割り当てられます。クライアントが新しいチャンネルを開こうとする場合、クライアントはそのチャンネル番号を要求と一緒に送ります。この番号情報は、サーバーに格納されており、そのチャンネルの通信方向決定に使用されます。この目的は、さまざまなタイプのセッションが互いに影響しないようにすることであり、1つのセッションが終了した時点で、そのチャンネルは、基本SSH接続を切断することなく閉じることができます。

チャンネルは、データを順序正しく送受信できる、フロー制御もサポートしています。このように、チャンネル上でのデータ送信は、チャンネルが開いたというメッセージをホストが受け取るまで開始されません。

クライアントとサーバーは、クライアントが要求するサービスやユーザーがネットワークに接続した方法に依存する、各チャンネルの特徴を自動的にネゴシエートします。このことは、プロトコルの基本的な構造を変更すること無しに、異なるタイプのリモート接続を扱う大きな自由度をもたらします。

18.4. OpenSSHの設定ファイル

OpenSSHには、2つの異なる設定ファイルのセットがあります。1つはクライアントプログラム用（`ssh`、`scp`、`sftp`）で、もう1つは、サーバーデーモン用（`sshd`）です。

システム全体のSSH設定情報は、`/etc/ssh/`ディレクトリに格納されます：

- `moduli`—セキュアなトランスポート層を構築するために重要な、Diffie-Hellman鍵交換に使用する、Diffie-Hellmanグループが格納されます。SSHセッションを開始するとき、鍵が交換されると、片方単独では決定できない共有秘密値を生成します。この値は、その後、ホスト認証を行う際に使われます。
- `ssh_config`—デフォルトのシステム全体のSSHクライアント設定ファイル。ユーザーが、ホームディレクトリ内に利用可能な独自の設定ファイルを持っている場合、その設定ファイルの値が優先します。（`~/.ssh/config`）
- `sshd_config` — `sshd`デーモン用の設定ファイル。

4. マルチプレクス接続は、共有の共通媒体を介して送信される複数の信号から成り立ちます。SSHでは、共通の安全な接続を介して各種チャンネルが送信されます。

- ssh_host_dsa_key — sshdデーモンで使用するDSA秘密鍵。
- ssh_host_dsa_key.pub — sshdデーモンで使用するDSA公開鍵。
- ssh_host_key — SSHプロトコルのバージョン1のsshdデーモンで使用するRSA秘密鍵。
- ssh_host_key.pub — SSHプロトコルのバージョン1のsshdデーモンで使用するRSA公開鍵。
- ssh_host_rsa_key — SSHプロトコルのバージョン2のsshdデーモンで使用するRSA秘密鍵。
- ssh_host_rsa_key.pub — SSHプロトコルのバージョン2のsshdで使用するRSA公開鍵。

ユーザー固有のSSH設定情報は、ユーザーのホームディレクトリ内の~/.ssh/ディレクトリに格納されます：

- authorized_keys — このファイルはサーバー用に認可された公開鍵の一覧を保有しています。クライアントからサーバーに接続されたとき、サーバーはこのファイルに格納してある署名付きの公開鍵を確認することによりクライアントを認証します。
- id_dsa — ユーザーのDSA秘密鍵が格納されます。
- id_dsa.pub—ユーザーのDSA公開鍵が格納されます。
- id_rsa — SSHプロトコルのバージョン2のsshで使用するRSA秘密鍵。
- id_rsa.pub — SSHプロトコルのバージョン2のsshで使用するRSA公開鍵。
- identity — SSHプロトコルのバージョン1のsshで使用するRSA秘密鍵。
- identity.pub — SSHプロトコルのバージョン1のsshで使用するRSA公開鍵。
- known_hosts — ユーザーがアクセスするSSHサーバーのDSAホスト鍵が格納されています。SSHクライアントが正しいSSHサーバーに接続しているか確かめるために、このファイルは非常に重要です。

重要

SSHサーバーのホスト鍵が変更された場合、クライアントはユーザーに対し、テキストエディタを使用してknown_hostsファイルからサーバーのホスト鍵を削除するまで接続は進行できないことを知らせます。但し、これを実行するまえに、システム管理者に連絡してサーバーが侵害されていないがどうか確認すべきです。

SSH設定ファイルで利用できる各種ディレクティブ関係の情報については、sshとsshdのmanページを参照してください。

18.5. SSHの詳細

安全なコマンドラインインターフェイスは、SSHを使用できる数多くの方法のまさに開始点です。正しい量の帯域幅が割り当てられていれば、X11セッションをSSHチャンネル上で送信することができます。また、TCP/IPフォワーディングを使用することで、以前は安全性に欠けていたシステム間のポート接続を特定のSSHチャンネルにマップすることができます。

18.5.1. X11フォワーディング

確立されているSSH接続上でX11セッションを開くことは、ローカルマシンでXプログラムを実行するのと同じくらい簡単なことです。セキュアシェルプロンプトからXプログラムを実行すると、SSHクライアントとSSHサーバーが現在のSSH接続内で新しい安全なチャンネルを作成し、Xプログラムのデータがそのチャンネルを介して透過的にクライアントマシンに送られます。

X11フォワーディングは非常に便利です。たとえば、X11フォワーディングを使用して、up2dateとの安全でインタラクティブなセッションを構成できます。これを実行するには、**ssh**を使用してサーバーに接続して以下を入力します：

```
up2date &
```

サーバー用のrootパスワードを入力すると、その後、**Red Hat 更新エージェント**が表示されリモートユーザーは安全にリモートシステムの更新をすることが出来ます。

18.5.2. ポートフォワーディング

SSHを用いると、そのままでは不安全なTCP/IPプロトコルをポートフォワーディング経由で安全にすることが出来ます。この技術を使用する時、SSHサーバーはSSHクライアントに対して暗号化されたコンジット(経路)になります。

ポートフォワーディングは、クライアント上のローカルポートをサーバー上のリモートポートにマップすることで、動作します。SSHで、サーバーのどのポートも、クライアント上のどのポートにもマッピングできます。SSHを動作させるために、ポート番号を適合させる必要はありません。

ローカルホスト上で接続を受信待機するTCP/IPポートフォワーディングチャンネルを作成するには、次のコマンドを実行します：

```
ssh -L local-port:remote-hostname:remote-port username@hostname
```

注意

ポートフォワーディングを設定して1,024以下のポートで受信待機する場合はrootアクセス権が必要です。

従って、暗号化された接続でPOPを使用しているmail.example.comというサーバー上の電子メールをチェックするには、次のコマンドを使用します：

```
ssh -L 1100:mail.example.com:110 mail.example.com
```

ポートフォワーディングのチャンネルがクライアントマシンとメールサーバーの間で設定されると、ローカルホスト上のポート1100を使用して新しいメールをチェックするように、POPメールクライアントに指示できます。クライアントシステム上のポート1100へ送られた要求はどれもmail.example.comサーバーに安全に転送されます。

mail.example.comでSSHサーバーが実行されていなくても、同じネットワーク上の別のマシンで実行している場合は、この接続の一部を安全にする為にSSHを使用出来ます。但し、以下のように少々異なるコマンドを必要とします：

```
ssh -L 1100:mail.example.com:110 other.example.com
```

この例では、クライアントマシンのポート1100からポート22上のSSH接続を介してSSHサーバー`other.example.com`にPOP要求を転送しています。これで、`other.example.com`は、`mail.example.com`上のポート110に接続して、新しいメールがあるどうかをチェックします。この手法で安全なのは、クライアントシステムと`other.example.com`SSHサーバー間の接続だけであることに注意して下さい。

ポートフォワーディングは、ネットワークファイアウォールを利用して情報を安全に取得するのにも使用できます。標準ポート(22)経由のSSHトラフィックは許可するけれども、他のポートへのアクセスは阻止するようにファイアウォールを設定した場合、阻止されたポートを使用する2つのホスト間の接続は、確立されたSSH接続を介して通信をリダイレクトすることで可能になります。

注意

ポートフォワーディングを使用してこのように接続を転送すると、クライアントシステム上のユーザーは、誰でも、そのサービスに接続できるようになります。もしクライアントシステムに侵入があった場合、侵入者も転送されたサービスにアクセスできます。

ポートフォワーディングを担当しているシステム管理者は、`/etc/ssh/sshd_config`内のAllowTcpForwarding行で`No`パラメータを指定して、`sshd`サービスを再起動することで、サーバーのこの機能を無効にすることができます。

18.6. リモート接続における SSHの必要条件

SSHを本当に効果的にするには、Telnet やFTPなどの安全性に欠ける接続プロトコルの使用は禁止すべきです。そうしないとSSHを使用して1つのセッションでユーザーのパスワードが保護されても、Telnetを使用してログインすればそのパスワードは盗聴される可能性があります。

使用不可にするサービスの幾つかを下に示します：

- `telnet`
- `rsh`
- `rlogin`
- `vsftpd`

システムへの安全性に欠ける接続手段を無効にする為には、コマンドラインプログラムchkconfig、ncursesベースのプログラム**ntsysv**、又はグラフィカルアプリケーションである**サービス設定ツール**(`redhat-config-services`)のいずれかを使用します。これらのツール全てにはrootでのアクセスが必要です。

ランレベルや`chkconfig`、`ntsysv`や**サービス設定ツール**を使ったサービスの設定についての詳細は、*Red Hat Linux* カスタマイズガイドのサービスに対するアクセスの制御の章を参照してください。

第19章

Tripwire

Tripwireのデータ保全性保証ソフトウェアを使用すると、重要なシステムファイルとディレクトリに対する変更をすべて検出することで、それらの信頼性を確認することができます。Tripwireは、自動化された定期的な検証手段を通してこれを達成します。Tripwireは検査したファイルが変更されたことを感知した場合、電子メールを介してシステム管理者に通達します。Tripwireは、追加、変更、又は削除されたファイルを積極的に識別出来ますので、復元の必要があるファイルの数を最低限に保ち、侵略からの回復時間を短縮します。これらの機能により、サーバへの侵入検知と損害評価を求めるシステム管理者にとっては、Tripwireは素晴らしいツールとなります。

Tripwireは、ファイル位置のデータベース、変更日時、その他のデータに照らしてファイルやディレクトリを比較することにより機能します。データベースは、特定の時点に指定されたファイルとディレクトリを取ったスナップショット— 基準(ベースライン)を含んでいます。基準データベースの内容は、システムが侵入されうる状態になる前(ネットワークに接続する前)に生成する必要があります。基準データベースが作成されると、Tripwireは現在のシステムをこれと比較し、変更、追加、削除のいずれかがあれば報告します。

Tripwireは、Red Hat Linuxの安全な状態を監査する為の貴重なツールではありますが、TripwireはRed Hat, Inc.でサポートされません。Tripwireに関しての詳細情報が必要な場合は、このプロジェクトのサイトhttp://www.tripwire.orgを参照して下さい。

19.1. Tripwireの使用方法

次のフローチャートは、Tripwireがどのように動作するかを示しています：

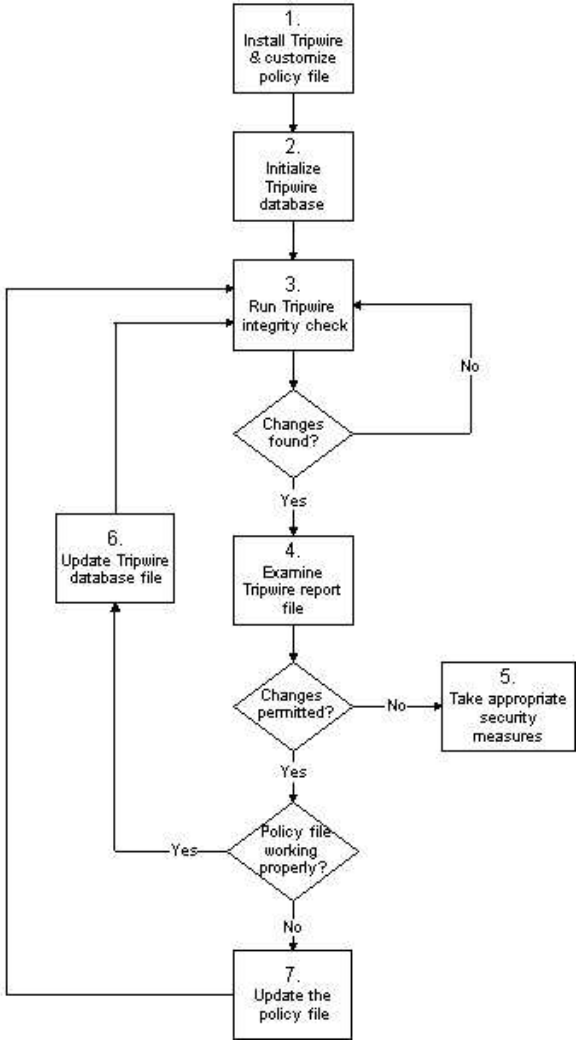

図19-1. Tripwireの使用

図19-1で示してある番号付のブロックをもう少し詳しく説明します。

*1. Tripwire*をインストールしてポリシーファイルをカスタマイズする。

‘ Tripwire RPMをインストールします。(項19.2を参照)。次に、サンプルの設定ファイル（`/etc/tripwire/twcfg.txt`）とポリシーファイル（`/etc/tripwire/twpol.txt`）をカスタマイズし、設定スクリプト（`/etc/tripwire/twinstall.sh`）を実行します。詳細は項19.3を参照してください。

*2. Tripwire*データベースを初期化する。

‘ 新規の、署名済みTripwireポリシーファイル（`/etc/tripwire/tw.pol`）に基づき、監視する重要なシステムファイルのデータベースを構築します。詳細は項19.4を参照してください。

*3. Tripwire*保全性チェックを実行する。

‘ 新しく作成されたTripwireデータベースと実際のシステムファイルを比較し、不足しているファイルや変更されたファイルを検索します。詳細は項19.5を参照してください。

*4. Tripwire*レポートファイルを検証する。

‘ `/usr/sbin/twprint`でTripwireレポートを表示して保全性違反を調べます。詳細は項19.6.1を参照して下さい。

5. 無許可の保全性違反が発生した場合、適切なセキュリティ対策を実施する。

‘ 監視中のファイルが不正に変更されている場合は、バックアップからオリジナルと差し替えるか、プログラムを再インストール、又はオペレーティングシステムを完全に再インストールします。

6. ファイルの変更が有効である場合、*Tripwire*データベースファイルを検査して更新する。

‘ 監視するファイルへの変更が意図的である場合、Tripwireデータベースを編集して結果レポート内のこれらの変更を無視します。詳細は項19.7を御覧下さい。

7. ポリシーファイルが検証に失敗する場合、*Tripwire*ポリシーファイルを更新する。

‘ Tripwireで監視するファイルの一覧や保全性違反の取り扱い方法を変更するには、用意してあるポリシーファイル(`/etc/tripwire/twpol.txt`)を更新して、署名済みのコピー(`/etc/tripwire/tw.pol`)、を再生成し、Tripwireデータベースを更新します。詳細は項19.8を参照してください。

それぞれの手順の詳しい説明は、この章の該当するセクションを参照してください。

19.2. Tripwire RPMのインストール

Tripwireをインストールするもっとも簡単な方法は、Red Hat Linux のインストール時にTripwire RPM を選択することです。ただしRed Hat Linux をすでにインストールしている場合は、`rpm`コマンド、又は**パッケージ管理ツール (パッケージ管理ツール)**を使用してRed Hat Linux 9 CD-ROMからTripwire RPMをインストールします。

Tripwireがインストールしてあるかどうか不明な場合は、シェルプロンプトで次のコマンドを入力します：

```
rpm -q tripwire
```

Tripwireがインストールしてある場合には、このコマンドは次の出力を表示します：

```
tripwire-<version-number>
```

上記出力の内、<*version-number*>とは、パッケージのバージョン番号です。

Tripwireがインストールされていない場合は、シェルプロンプトに戻ります。

以下に、CD-ROMからRPMコマンドラインアプリケーションを使ってTripwireを見付けて、インストールする方法を大まかに説明します：

1. Red Hat Linux9インストールCD-ROMの中の*CD 2*を挿入します。
2. CD-ROMが自動マウントされない場合は、次のコマンドを入力します：

```
mount /mnt/cdrom
```

3. 以下のように入力してTripwire RPMがCD-ROM上にあることを確認します：
   ```
   ls /mnt/cdrom/RedHat/RPMS/ | grep tripwire
   ```
   このRPMがCD-ROM上のある場合は、パッケージの名前が表示されます。

   RPMがCD-ROM上にない場合は、シェルプロンプトに戻ります。この場合、最初にそのCD-ROMをアンマウントして、1番から3番までのステップを繰り返し、他のRed Hat Linux 9インストールCD-ROMをチェックする必要があります。

   アンマウントするには、CD-ROMアイコンを右クリックして**Eject**を選択するか、あるいはシェルプロンプトで次のように入力します：
   ```
   umount /mnt/cdrom
   ```
4. Tripwire RPMを見付けた後では、rootユーザーで次のようにコマンドを入力して、それをインストールします：
   ```
   rpm -Uvh /mnt/cdrom/RedHat/RPMS/tripwire*.rpm
   ```

`/usr/share/doc/tripwire-<version-number>/` ディレクトリ内にあるTripwire用のリリースノートとREADMEファイルを見付けることが出来るでしょう。(ここで<*version-number*>とは、ソフトウェアのバージョン番号です) これらのドキュメントには、デフォルトのポリシーファイルとその他のトピックに関する重要な情報が含まれています。

19.3. Tripwireのカスタマイズ

Tripwire RPMのインストールが終了すると、ソフトウェアを初期化するために次のステップを全て実行する必要があります：

19.3.1. `/etc/tripwire/twcfg.txt`の編集

このサンプルTripwire設定ファイルの編集は必須ではありませんが、ユーザーの立場によっては編集することが必要になるかも知れません。例えば、Tripwire ファイルの場所を変更する、電子メール設定をカスタマイズする、又はリポート用の詳細レベルをカスタマイズするなどの場合に必要です。

以下に`/etc/tripwire/twcfg.txt`ファイル内で必須となるユーザー設定可能な変数を示します：

- `POLFILE` — ポリシーファイルの場所を指定します；`/etc/tripwire/tw.pol`がデフォルトの値です。
- `DBFILE` — データベースファイルの場所を指定します；`/var/lib/tripwire/$(HOSTNAME).twd`がデフォルトの値です。
- `REPORTFILE` — レポートファイルの場所を指定します；デフォルトでこの値は`/var/lib/tripwire/report/$(HOSTNAME)-$(DATE).twr`にセットされています。
- `SITEKEYFILE` — サイトキーファイルの場所を指定します；`/etc/tripwire/site.key`がデフォルトの値です。
- `LOCALKEYFILE` — ローカルキーファイルの場所を指定します；`/etc/tripwire/$(HOSTNAME)-local.key`がデフォルトの値です。

重要

設定ファイルを編集しても、上記のいずれかの変数を未定義のままにすると、設定ファイルは無効になります。この状態になると、`tripwire` コマンドを実行した時に、エラーを報告して終了してしまいます。

サンプルの/etc/tripwire/twcfg.txtファイル内の残りの設定可能な変数はオプションとなります。これには以下が含まれます：

- EDITOR —Tripwireで呼び込まれるテキストエディタを指定します。デフォルトの値は/bin/viです。
- LATEPROMPTING — trueにセットされている場合、この変数は、Tripwireがユーザーにパスワードを要求するまで出来るでけ長く待つようにして、それによりパスワードがメモリ内に存在する時間を最小限にするよう設定します。デフォルトの値はfalseです。
- LOOSEDIRECTORYCHECKING — trueにセットされている場合、この変数は、Tripwireを監視中のディレクトリ内のファイルが変更された場合は報告して、ディレクトリ自身の変更は報告しないように設定します。これでTripwireレポートの余剰を制限します。デフォルトの値はfalseです。
- SYSLOGREPORTING — trueにセットされている場合、この変数は、Tripwireがユーザー設備を経由してsyslog デーモンへ情報を報告するように設定します。ログレベルはnoticeにセットしてあります。詳細はsyslogdのmanを御覧下さい。デフォルトの値はfalseです。
- MAILNOVIOLATIONS — trueにセットされている場合、この変数は、Tripwireが違反の発生にかかわらず、一定の期間で電子メールの報告を出すように設定します。デフォルトの値はtrue です。
- EMAILREPORTLEVEL — 電子メール報告の詳細レベルを指定します。この変数の有効な値は0から4です。デフォルトの値は3となっています。
- REPORTLEVEL — twprintコマンドにより生成されたレポート用の詳細レベルを指定します。この値はコマンドライン上で書き換え出来ますがデフォルトでは3にセットされています。
- MAILMETHOD — Tripwireが使用すべきメールプロトコルを指定します。有効な値は、SMTPとSENDMAILです。デフォルト値はSENDMAILです。
- MAILPROGRAM — Tripwireが使用すべきメールプログラムを指定します。デフォルト値は/usr/sbin/sendmail -oi -tです。

サンプル設定ファイルを編集した後は、サンプルポリシーファイルを設定する必要があります。

警告

セキュリティの目的で、インストールスクリプトの実行、又は署名済みの設定ファイルを再生成した後には、プレインテキストである/etc/tripwire/twcfg.txtのすべてのコピーを削除するか、又は安全な場所に保存する必要があります。他の方法としては権限を変更して、他からは読み取れないようにします。

19.3.2. /etc/tripwire/twpol.txtの編集

必須ではないのですが、システム上の特定のアプリケーション、ファイル、ディレクトリ等を考慮して、大幅にコメントされているこのサンプルTripwireポリシーファイルを編集する必要があります。RPMの無変更のサンプル設定に頼ることはシステムを適切に保護できない可能性があります。

ポリシーファイルを変更することは、ファイルや使用していないプログラムへの誤報を低減すること、及び電子メール通知などの機能の追加によりTripwireの使用価値を向上します。

注意

電子メールによる通知はデフォルトでは設定されていません。この機能の設定の詳細については項19.8.1を御覧下さい。

設定スクリプトを実行した後でサンプルポリシーファイルを変更する場合は、署名済みポリシーファイルの再生成方法を項19.8で御覧下さい。


警告

セキュリティの目的で、インストールスクリプトの実行、又は署名済み付の設定ファイルを再生成した後には、プレインテキストである`/etc/tripwire/twpol.txt`の全てのコピーを削除するか、又は安全な場所に保存する必要があります。他の方法としては権限を変更して、他からは読み取れないようにします。

19.3.3. `twinstall.sh`スクリプトの実行

rootユーザーとして、シェルプロンプトで`/etc/tripwire/twinstall.sh`と入力して設定スクリプトを実行します。`twinstall.sh`スクリプトがサイトとローカルのパスワードを尋ねてきます。これらのパスワードはTripwireファイルを保護する為の暗号化キーを生成するのに使用されます。

サイトとローカルのパスワードを選択するとき、次のガイドラインを考慮する必要があります：

- 独特のパスワードを、英数文字及び記号で最低8文字から最大1023文字までにして使用する。
- パスワード内には引用符号は使用しない。
- Tripwireのパスワードは、完全にシステム用のrootやその他のパスワードとは別のものとして設定する。
- サイトキーとローカルキーはそれぞれ独特のパスワードとする。

サイトキーパスワードはTripwire設定ファイルとポリシーファイルの両方を保護します。ローカルキーパスワードはTripwireデータベースとレポートファイルを保護します。


警告

パスワードを忘れた場合は、署名済みファイルを解読する方法はありません。パスワードを忘れた場合、ファイルは使用できず、設定スクリプトを再度実行する必要があります。

設定、ポリシー、データベース、レポートファイル等を暗号化することで、Tripwireは、サイトとローカルの両パスワードを持っていない人物がそれらを読み込むことを防ぎます。これは侵入者がシステムのrootアクセスを取得したとしても、その形跡を消すためのTripwireファイル変更はできないという意味です。

一度暗号化されて署名されると、`twinstall.sh`スクリプトを実行して生成される設定とポリシーのファイルは、名前変更や移動はすべきではありません。

19.4. Tripwireデータベースの初期化

データベースを初期化する時に、Tripwireはポリシーファイルの規則に基づいて一連のファイルシステムオブジェクトを構成します。このデータベースは保全性チェックの為の基礎としての役割をします。

Tripwireデータベースを初期化するには、次のコマンドを使用します：

```
/usr/sbin/tripwire --init
```

このコマンドは実行に数分かかることがあります。

これらのステップを正しく終了すると、Tripwireは、重要なファイル内の変更を調べるのに必要なファイルシステムの基礎スナップショットを持つようになります。Tripwireデータベースの初期化が終了すると、初期保全性チェックを実行する必要があります。このチェックはコンピュータをネットワークにつないで作成を始める前にすべきものです。詳細の説明は項19.5を御覧下さい。

満足できる状態にTripwireが設定できたら、これで自由にシステムを作業に使用することが出来ます。

19.5. 保全性チェックの実行

デフォルトでは、Tripwire RPMは/etc/cron.daily/ディレクトリにtripwire-checkというシェルスクリプトを追加します。このスクリプトは、1日1回自動的に保全性チェックを実行します。

しかし、以下のコマンドを入力すれば、いつでもTripwireの保全性チェックを実行することが出来ます：

```
/usr/sbin/tripwire --check
```

保全性チェックで、Tripwireは現在実際に使用されているファイルシステムオブジェクトを、データベースに記録されているそれらのファイルのプロパティと比較します。違反内容は画面に出力され、レポートの暗号化したコピーが/var/lib/tripwire/report/に中に生成されます。このレポートの内容は項19.6.1に概要が示されている通り、twprintコマンドを使用して表示することが出来ます。

特定の保全性違反が発生した際に電子メールの通知を受けるようにしたい場合は、ポリシーファイルをそのように設定します。その設定の仕方と機能のテストに関しての説明は項19.8.1を御覧下さい。

19.6. Tripwire レポートの検査

暗号化されたTripwireレポートとデータベースを表示するには/usr/sbin/twprintを使用します。

19.6.1. Tripwire レポートの表示

twprint -m rコマンドは、Tripwireレポートの内容を読みやすいテキストで表示します。どのレポートファイルを表示するかを、twprintに指定する必要があります。

twprintコマンドでTripwireレポートを表示するは、次のように（すべて1行で）入力します：

```
/usr/sbin/twprint -m r --twrfile /var/lib/tripwire/report/<name>.twr
```

コマンドの-m rオプションがtwprintに、Tripwireレポートを復号するように指示します。--twrfileオプションはtwprintに使用するTripwireレポートファイルを指定します。

Tripwireレポートの名前は、Tripwireがレポート作成にチェックしたホストの名前と作成日時を含んでいます。既に保存されているレポートは何時でも表示することができます。Tripwireレポートのの一覧を見るにはls /var/lib/tripwire/reportと入力します。

見つかった違反や生成されたエラーの数によっては、レポートはかなり長くなります。レポートの最初の部分は、次のようになります：

```
Tripwire(R) 2.3.0 Integrity Check Report

Report generated by:     root
Report created on:     Fri Jan 12 04:04:42 2001
Database last updated on:   Tue Jan  9 16:19:34 2001
```

```
===============================================================================
Report Summary:
===============================================================================
Host name:          some.host.com
Host IP address:       10.0.0.1
Host ID:           None
Policy file used:       /etc/tripwire/tw.pol
Configuration file used:   /etc/tripwire/tw.cfg
Database file used:     /var/lib/tripwire/some.host.com.twd
Command line used:      /usr/sbin/tripwire --check

===============================================================================
Rule Summary:
===============================================================================
-------------------------------------------------------------------------------
Section: Unix File System
-------------------------------------------------------------------------------
 Rule Name        Severity Level  Added  Removed Modified
 ---------        --------------  -----  ------- --------
 Invariant Directories  69       0    0    0
 Temporary directories  33       0    0    0
* Tripwire Data Files   100       1    0    0
 Critical devices     100       0    0    0
 User binaries      69       0    0    0
 Tripwire Binaries    100       0    0    0
```

19.6.2. Tripwireデータベースの表示

twprintを使用して、Tripwireデータベース全体やデータベース内の選択したファイルに関する情報を表示することもできます。これにより、システム上のどれくらいの情報をTripwireが追跡しているかを確認できます。

Tripwireデータベース全体を表示するには、次のコマンドを入力します：

```
/usr/sbin/twprint -m d --print-dbfile | less
```

このコマンドを実行すると膨大な量のデータが出力されます。以下の出力例は、そのうちのごく最初の部分です：

```
Tripwire(R) 2.3.0 Database

Database generated by:    root
Database generated on:    Tue Jan 9 13:56:42 2001
Database last updated on:   Tue Jan 9 16:19:34 2001

===============================================================
Database Summary:
===============================================================
Host name:          some.host.com
Host IP address:       10.0.0.1
Host ID:           None
Policy file used:       /etc/tripwire/tw.pol
Configuration file used:   /etc/tripwire/tw.cfg
Database file used:     /var/lib/tripwire/some.host.com.twd
Command line used:      /usr/sbin/tripwire --init

===============================================================
Object Summary:
===============================================================
```

```
-------------------------------------------------------------------
# Section: Unix File System
-------------------------------------------------------------------
  Mode    UID     Size    Modify Time
  ------   ----------- ---------- ----------
/
  drwxr-xr-x root (0)  XXX    XXXXXXXXXXXXXXXX
/bin
  drwxr-xr-x root (0)  4096    Mon Jan 8 08:20:45 2001
/bin/arch
  -rwxr-xr-x root (0)  2844    Tue Dec 12 05:51:35 2000
/bin/ash
  -rwxr-xr-x root (0)  64860   Thu Dec 7 22:35:05 2000
/bin/ash.static
  -rwxr-xr-x root (0)  405576  Thu Dec 7 22:35:05 2000
```

Tripwireが追跡している特定のファイル（例：/etc/hosts）に関する情報を表示するには、次のように入力します：

```
/usr/sbin/twprint -m d --print-dbfile /etc/hosts
```

このコマンドを実行すると、次のような出力が表示されます：

```
Object name: /etc/hosts

Property:       Value:
-------------     -----------
Object Type      Regular File
Device Number     773
Inode Number     216991
Mode         -rw-r--r--
Num Links       1
UID          root (0)
GID          root (0)
```

その他のオプションについては、twprintのmanページを参照してください。

19.7. Tripwire データベース更新

保全性チェックを実行して違反が見つかった場合、まず最初に、発見された違反が現実のセキュリティ侵害なのか、正当な変更によるものかを判断する必要があります。最近、アプリケーションのインストールや重要なシステムファイルの編集をした場合、Tripwireは保全性チェック違反を正しく報告します。この場合、Tripwireデータベースを更新して、以後それらの変更が違反として報告されないようにします。一方、システムファイルに対して不正な変更がされていて保全性チェック違反が報告された場合は、バックアップからオリジナルのファイルを復元するか、違反が極端な場合はオペレーティングシステムを再インストールします。

Tripwireデータベースを更新してレポート内の違反を承認するには、Tripwireは先ずデータベースに対してレポートファイルを相互参照して、その後、レポートファイルから承認できる違反を統合します。データベースを更新する時には、最新のレポートを使用するように気を付けて下さい。

Tripwireデータベースを更新するには、次のコマンドを（すべて1行で）入力します。ここで*name*とは最新のレポートファイルの事です：

```
/usr/sbin/tripwire --update --twrfile /var/lib/tripwire/report/<name>.twr
```

Tripwireは、その設定ファイル内のEDITOR行で指定したデフォルトのテキストエディタを使用してレポートファイルを表示します。ここで、Tripwireデータベース内に更新したくないファイルがあれば、選択を解除できます。


重要

データベースの中では、承認された保全性違反のみを変更することが重要です。

Tripwireデータベースの更新予定のすべては、ファイル名の前が[x]で始まります。以下の例のようになります：

```
Added:
[x] "/usr/sbin/longrun"

Modified:
[x] "/usr/sbin"
[x] "/usr/sbin/cpqarrayd"
```

承認された保全性違反を特定して、Tripwireデータベースへの追加から外すにはxを取り除きます。

デフォルトのテキストエディタviの中でファイルを編集するには、iを入力して[Enter]キーを押して挿入モードに入り、それから必要な変更をします。終了すると[Esc]キーを押して:wqと入力してから[Enter]キーを押します。

エディタが閉じてから、ローカルパスワードを入力するとデータベースが再構成され、署名されます。

新規のTripwireデータベースが書き込まれた後は、新しく承認された保全性違反には警告表示が出なくなります。

19.8. Tripwire ポリシーファイルの更新

Tripwireデータベースに記録されているファイルを変更したり、電子メールに設定を変更したり、報告される違反の程度を変更するには、Tripwireポリシーファイルを編集する必要があります。

まず、サンプルポリシーファイル（/etc/tripwire/twpol.txt）に必要な変更を加えます。このファイルを削除している場合(Tripwireの設定終了後は実行すべき操作)、次のコマンドを発行してそれを再生成することが出来ます：

```
twadmin --print-polfile > /etc/tripwire/twpol.txt
```

このポリシーファイルへの一般的な変更は、システム上に存在しないファイルをコメントアウトしてそれらがTripwireレポートでファイルが見付かりませんのエラーを生成しないようにすることです。例えば、システムに/etc/smb.confファイルがない場合、次の例のように#印をtwpol.txt行の先頭に付けて、コメントアウトすることで、Tripwireがそれを捜索しないようにします：

```
#  /etc/smb.conf          -> $(SEC_CONFIG);
```

次に、新規の署名済み/etc/tripwire/tw.polファイルを作成して、このポリシー情報に基づいてデータベースファイルを更新します。編集したポリシーファイルが/etc/tripwire/twpol.txtとすると、次のようにコマンドを入力します（すべて1行で入力します）：

```
/usr/sbin/twadmin --create-polfile -S site.key /etc/tripwire/twpol.txt
```

サイトパスワードの入力を求められます。入力すると、twpol.txtファイルが暗号化されて署名されます。

新しい/etc/tripwire/tw.polファイルを作成したら、必ずTripwireデータベースを更新します。もっとも確実な方法は、現在のデータベースを削除して、新しいポリシーファイルを使用して新たにデータベースを作成するやり方です。

使用中のTripwireデータベースの名前がbob.domain.com.twdとすると、このデータベースを削除するには次のコマンドを入力します：

```
rm /var/lib/tripwire/bob.domain.com.twd
```

続けて、次のコマンドを入力して更新したポリシーファイルを使用した新しいデータベースを作成します：

```
/usr/sbin/tripwire --init
```

データベースが正しく変更されているかどうかを確認するには、手動で最初の保全性チェックを実行してレポートの内容を確認します。これらの作業の手順は、項19.5と項19.6.1を参照してください。

19.8.1. Tripwireと電子メール

Tripwireでは、ポリシーファイル内のある特定のタイプのルールに対する違反が発生したときに、任意の宛先に電子メールを送信できます。Tripwireにこの設定をするには、まず特定の保全性違反が発生した場合に連絡する相手の電子メールアドレスと、監視するルールを設定する必要があります。管理者が何人もいる大きなシステムでは、異なる違反に対してそれぞれ異なるグループに通知することが出来ます。

誰に何について通知するか及びどの違反ルールで彼らに報告するかが決定すると、/etc/tripwire/twpol.txtの編集で、それぞれのルールのルールディレクティブセクションに**emailto=**行を追加します。具体的には、severity=行の後ろにコンマを追加し、次の行にemailto=を付け、その次に送信先のE-mailアドレスを入力します。複数のアドレスをセミコロンで区切って指定すると、複数のE-mailを送信できます。

たとえば、ネットワークプログラムが変更された際に、johnrayとbobの二人の管理者に通知するには、ポリシーファイル内のNetworking Programsルールディレクティブを次のように変更します：

```
(
 rulename = "Networking Programs",
 severity = $(SIG_HI),
 emailto = johnray@domain.com;bob@domain.com
)
```

ポリシーファイルを変更してから、項19.8の指示に従って更新と暗号化した署名済みのTripwireポリシーファイルのコピーを生成します。

19.8.1.1. テストメッセージの送信

Tripwireの電子メール通知設定による実際のメッセージ送信をテストするには、次のコマンドを実行します：

```
/usr/sbin/tripwire --test --email your@email.address
```

指定した電子メールアドレスにただちにtripwireプログラムによって、テストメッセージが送信されます。

19.9. Tripwire 設定ファイルの更新

Tripwireの設定ファイルを変更したい場合は、先ずサンプル設定ファイル/etc/tripwire/twcfg.txtを編集する必要があります。このファイルを削除している場合(Tripwireの設定を終了後すべき操作)、次のコマンドを使用してそれを再生成することが出来ます：

```
twadmin --print-cfgfile > /etc/tripwire/twcfg.txt
```

Tripwireは、twadminコマンドにより、設定テキストファイルが正しく署名されて、/etc/tripwire/tw.polに変換されるまでどんな設定変更も認識しません。

/etc/tripwire/twcfg.txtテキストファイルから設定ファイルを再生成するには次のコマンドを使用します：

```
/usr/sbin/twadmin --create-cfgfile -S site.key /etc/tripwire/twcfg.txt
```

設定ファイルは、Tripwireポリシーやアプリケーションで追跡するファイルを変更しない為、Tripwireデータベースを再生成する必要はありません。

19.10. Tripwireファイルの場所の参照

Tripwireで作業する前に、アプリケーション用の重要なファイルがある場所を知っておく必要があります。Tripwireはそのファイルを、それぞれのファイルの役目に従ってさまざまな場所に保存します。

- /usr/sbin/ディレクトリの中には、以下のようなプログラムがあります：
  - tripwire
  - twadmin
  - twprint

- /etc/tripwire/ディレクトリ内には、次のようなファイルがあります：
  - twinstall.sh — Tripwireの初期化スクリプト。
  - twcfg.txt — Tripwire RPMによって供給される設定ファイル。
  - tw.cfg — twinstall.shスクリプトによって作成された署名済みの設定ファイル。
  - twpol.txt — Tripwire RPMによって供給されるサンプルポリシーファイル。
  - tw.pol — twinstall.shスクリプトによって作成される署名済のポリシーファイル。
  - キーファイル— twinstall.shスクリプトによって作成されるローカルキーとサイトキーで、拡張子.key が最後に付く。

- twinstall.shインストールスクリプトを実行した後、/var/lib/tripwire/ディレクトリには以下のようなファイルがあります：
  - Tripwire データベース— システムファイルのデータベースで、.twdの拡張子を持つ。
  - Tripwire レポート— report/ディレクトリはTripwire レポートが保存されている場所です。

次のセクションでは、Tripwireシステムの中で、これらのファイルが持つ役割について説明します。

19.10.1. Tripwireのコンポーネント

前述のセクションに出てくるTripwireシステム内の役割の詳細を以下に示します。

/etc/tripwire/tw.cfg

‘ これは、Tripwireデータファイルの場所などシステム固有の情報が保存される暗号化されたTripwire設定ファイルです。twinstall.shインストーラースクリプトとtwadminコマンドは、設定ファイルのテキストバージョン、/etc/tripwire/twcfg.txtを使用してこのファイルを生成します。

インストールスクリプトを実行した後、システム管理者はtwadminを使用して/etc/tripwire/twcfg.txtの編集とtw.cfgの署名済みコピーを再生成することにより、パラメータを変更することが出来ます。その方法の詳細については、項19.9を御覧下さい。

/etc/tripwire/tw.pol

‘ アクティブなTripwireポリシーは、コメント、ルール、ディレクティブ、変数などを含む暗号化されたファイルです。このファイルはTripwireがシステムをチェックする方法を指示します。ポリシーファイル内のそれぞれのルールは監視されるシステムオブジェクトを指定します。ルールはまた、オブジェクトへの変更の報告するものと無視するものを記述します。

システムオブジェクトとは、監視したいファイルやディレクトリです。各オブジェクトはオブジェクト名で識別されます。プロパティはTripwire ソフトウェアが監視できるオブジェクトの1つの特徴を示します。ディレクティブはポリシーファイル内の一連のルールの条件付の処理を制御します。インストール中にサンプルテキストポリシーファイル、/etc/tripwire/twpol.txtがアクティブなTripwireポリシーファイルを生成するのに使用されます。

インストールスクリプトを実行した後、システム管理者は、twadmin コマンドを使って、/etc/tripwire/twpol.txtの編集とtw.pol ファイルの署名済みコピーを再生成することによりTripwireポリシーファイルを更新できます。その方法に関する情報は項19.8で御覧下さい。

/var/lib/tripwire/host_name.twd

‘ 最初の初期化では、Tripwireは署名済みポリシーファイルのルールを使用してデータベースファイルを作成します。このTripwireデータベースファイルが、既知の安全な状態における、システムの基準スナップショットです。Tripwireは、この基準と現在のシステムを比較してどのような変更が発生したか判断します。この比較は保全性チェックと呼ばれます。

/var/lib/tripwire/report/host_name-date_of_report-time_of_report.twr

‘ 保全性チェックを実行すると、/var/lib/tripwire/reportディレクトリにレポートファイルが作成されます。レポートファイルには、保全性チェックでポリシーファイルルールに違反した、ファイルの変更がすべて記録されます。Tripwireのレポートは次の慣例を使用して名前が付けられます：*host_name-date_of_report-time_of_report*.twr. これらはTripwireデータベースと実際のシステムファイルとの相違の詳細を報告します。

19.11. その他のリソース

Tripwireは、この章で紹介している以上のことが出来ます。Tripwireに関する詳細情報を得るには以下の資料を御覧下さい。

19.11.1. インストールされているドキュメント

- `/usr/share/doc/tripwire-<version-number>` — `/etc/tripwire/`ディレクトリにある設定ファイルとポリシーファイルのカスタマイズ方法に関する最適な入門文書です。
- `tripwire`、`twadmin`、`twprint`の各ユーティリティについてのヘルプは、それぞれのmanページを参照してください。

19.11.2. 役に立つWebサイト

- http://www.tripwire.org — Tripwire Open Source Projectのホームページです。Tripwireに関する最新ニュースややFAQ一覧が掲載されています。
- http://sourceforge.net/project/showfiles.php?group_id=3130 — Tripwire プロジェクトによる最新のオフィシャルドキュメントへリンクします。

IV. 付録

目次

付録A.
一般的なパラメータとモジュール

この付録は、一般的なハードウェアデバイスのドライバー[1] に利用できる可能性のあるパラメータの一部を示すために記載されています。これはRed Hat Linuxではカーネルモジュールと呼ばれます。ほとんどの場合、デフォルトのパラメータが機能します。しかし、デバイスを正常に機能させる為に追加のモジュールパラメータが必要であるとか、そのデバイスの為にモジュールのデフォルトパラメータを上書きする必要がある時があります。

インストールする時点では、Red Hat Linuxは安定したインストール環境を構成するために、限られたデバイスドライバーのサブセットを使用します。インストールプログラムは多種多様なハードウェア上でのインストールをサポートしますが、幾つかのドライバー(SCSIアダプター用、ネットワークアダプター用、多くのCD-ROMドライブ用など含む)はインストールカーネルに収納されていません。これらはユーザーによって起動時にモジュールとしてロードされる必要があります。インストールプロセス中にどこで追加のカーネルモジュールを見付けることが出来るかに関しては、*Red Hat Linux* インストールガイドの準備の為のステップの章で代わりの起動方法のセクションを参照してください。

インストールが完了すると、カーネルモジュールを通じてかなりの数のデバイス用のサポートが存在します。

A.1. モジュールパラメータの指定

場合によっては、正常に機能するようにモジュールのロード時にパラメータを指定することが必要なこともあります。これは以下の2つの方法のいずれかで実行します：

- 1行に完全なパラメータセットを指定します。たとえば、`cdu31=0x340,0`パラメータを使用すると、Sony CDU 31か33を割り込み（IRQ）なしでポート340に設定できます。
- 個々のパラメータを指定します。最初のパラメータセットのうち1つ、又はそれ以上が必要ない場合には、この方法で指定します。たとえば、`cdu31_port=0x340 cdu31a_irq=0`を上記の例と同じCD-ROMのパラメータとして使用することができます。*OR*は、最初のパラメータ方法が終了し、2番目の方法が開始する場所を示すために、CD-ROM、SCSI、この付録に掲載されたイーサネットテーブルで使用されています。

注意

特定のパラメータを持つモジュールをロードしているときは、両方ではなくどちらか一方の方法だけを使用します。

用心

パラメータにカンマが含まれる場合は、カンマの後にスペースを入れないでください。

1. ドライバーとは、システムが特定のハードウェアデバイスをLinuxで使用出来るようにするソフトウェアです。ドライバーがないと、カーネルがデバイスの正しい使用方法を認識できないことがあります。

A.2. CD-ROMモジュールパラメータ

注意

一覧表示してあるCD-ROMドライブのすべてがサポートされている訳ではありません。以下のRed Hatのwebサイトにあるハードウェア互換一覧(Hardware Compatibility List)で使用するデバイスがサポートされているかどうか確認して下さい。http://hardware.redhat.com

通常は、ドライバディスクをロードしてデバイスを指定してから、パラメータを指定しますが、もっとも一般的に使用されているパラメータの1つ（hd*X*=cdrom）は、インストール時にブートプロンプト（boot:）で入力することができます。このような例外は、既にカーネルの一部であるIDE/ATAPI CD-ROMサポートを処理するために許されています。

次の表で、パラメータが一覧表示されていないモジュールの大半は、自動検索でハードウェアを見つけるか、モジュールソースコードを手作業で変更して再コンパイルをしなければならないかのいずれかです。

ハードウェア	モジュール	パラメータ
ATAPI/IDE CD-ROM ドライブ		hd*X*=cdrom
Aztech CD268-01A, Orchid CD-3110, Okano/Wearnes CDD110, Conrad TXC, CyCDROM CR520, CyCDROM CR540 (IDE以外)	aztcd.o	aztcd=*io_port*
Sony CDU-31A CD-ROM	cdu31a.o	cdu31a=*io_port*,*IRQ OR* cdu31a_port=*base_addr* cdu31a_irq=*irq*
Philips/LMS CDROMドライブ206 cm260ホストアダプタカード付き	cm206.o	cm206=*io_port,IRQ*
Goldstar R420 CD-ROM	gscd.o	gscd=*io_port*
ISP16, MAD16, or Mozartサウンドカード CD-ROMインターフェイス(OPTi 82C928 and OPTi 82C929) Sanyo/Panasonic, Sony, 又はMitsumiドライブと併用	isp16.o	isp16=*io_port*,*IRQ*,*dma*, *drive_type OR* isp16_cdrom_base=*io_port* isp16_cdrom_irq=*IRQ* isp16_cdrom_dma=*dma* isp16_cdrom_type=*drive_type*
Mitsumi CD-ROM, 標準	mcd.o	mcd=*io_port*,*IRQ*
Mitsumi CD-ROM, テスト用	mcdx.o	mcdx=*io_port_1*,*IRQ_1*, *io_port_n*,*IRQ_n*
Optics storage 8000 AT "Dolphin"ドライブ、Lasermate CR328A	optcd.o	
パラレルポートIDE CD-ROM	pcd.o	
SB Pro 16 互換	sbpcd.o	sbpcd=*io_port*

ハードウェア	モジュール	パラメータ
Sanyo CDR-H94A	`sjcd.o`	sjcd=*`io_port`* *OR* sjcd_base=*`io_port`*
Sony CDU-535 & 531 (一部のProcommドライブ)	`sonycd535.o`	sonycd535=*`io_port`*

表A-1. ハードウェアパラメータ

これらのモジュールの使用例をいくつか示します：

設定	例
セカンダリIDEチャンネルのマスターに設定されたATAPI CD-ROM	hdc=cdrom
非IDE Mitsumi CD-ROM ポート340、IRQ 11	mcd=0x340,11
テスト用ドライバを使用する非IDEのMitsumi CD-ROMドライブ3台、IOポート300、304、320、IRQ 5、10、11	mcdx=0x300,5,0x304,10,0x320,11
Sony CDU 31 か33 ポート340、IRQなし	cdu31=0x340,0 *OR* cdu31_port=0x340 cdu31a_irq=0
Aztech CD-ROM ポート220	aztcd=0x220
PanasonicタイプCD-ROM SoundBlaster インターフェイス、ポート230	sbpcd=0x230,1
Phillips/LMS cm206 and cm260 at IO 340 and IRQ 11	cm206=0x340,11
Goldstar R420 at IO 300	gscd=0x300
Mitsumi ドライブ、MAD16サウンドカード、IO Addr 330 IRQ 1, DMA使用	isp16=0x330,11,0,Mitsumi
Sony CDU 531、IOアドレス320	sonycd535=0x320

表A-2. ハードウェアパラメータの設定例

注意

多くのSound Blasterカードは IDEインターフェイスを搭載しています。そのようなカードでは、`sbpcd`パラメータを使用する必要がありません。hd*X*パラメータのみを使用してください。(*X*は適切なドライブ割り当て文字です)。

A.3. SCSIパラメータ

ハードウェア	モジュール	パラメータ
Adaptec 28xx, R9xx, 39xx	`aic7xxx.o`	

ハードウェア	モジュール	パラメータ
3wareストレージコントローラ	`3w-xxxx.o`	
NCR53c810/820/720, NCR53c700/710/700-66	`53c7,8xx.o`	
AM53/79C974 (PC-SCSI)ドライバー	`AM53C974.o`	
ほとんどのBuslogic (現Mylex)製カードは部品番号に「BT」が付いています	`BusLogic.o`	
Mylex DAC960 RAIDコントローラ	`DAC960.o`	
MCR53c406aベースのSCSI	`NCR53c406a.o`	
Initio INI-A100U2W	`a100u2w.o`	a100u2w=*io*,*IRQ*,*scsi_id*
Adaptec AACRAID	`aacraid.o`	
Advansys SCSIカード	`advansys.o`	
Adaptec AHA-152x	`aha152x.o`	aha152x=*io*,*IRQ*,*scsi_id*
Adaptec AHA 154x amd 631xベース	`aha1542.o`	
Adaptec AHA 1740	`aha1740.o`	
Adaptec AHA-274x, AHA-284x, AHA-29xx, AHA-394x, AHA-398x, AHA-274x, AHA-274xT, AHA-2842, AHA-2910B, AHA-2920C, AHA-2930/U/U2, AHA-2940/W/U/UW/AU/, U2W/U2/U2B/, U2BOEM, AHA-2944D/WD/UD/UWD, AHA-2950U2/W/B, AHA-3940/U/W/UW/, AUW/U2W/U2B, AHA-3950U2D, AHA-3985/U/W/UW, AIC-777x, AIC-785x, AIC-786x, AIC-787x, AIC-788x , AIC-789x, AIC-3860	`aic7xxx.o`	
ACARD ATP870U PCI SCSIコントローラ	`atp870u.o`	
Compaq Smart Array 5300コントローラ	`cciss.o`	
Compaq Smart/2 RAIDコントローラ	`cpqarray.o`	
Compaq FibreChannelコントローラ	`cpqfc.o`	
Domex DMX3191D	`dmx3191d.o`	

ハードウェア	モジュール	パラメータ
Data Technology Corp DTC3180/3280	dtc.o	
DTP SCSIホストアダプタ(EATA/DMA) PM2011B/9X ISA, PM2021A/9X ISA, PM2012A, PM2012B, PM2022A/9X EISA, PM2122A/9X, PM2322A/9X, SmartRAID PM3021, PM3222, PM3224	eata.o	
DTP SCSI Adapters PM2011, PM2021, PM2041, PM3021, PM2012B, PM2022, PM2122, PM2322, PM2042, PM3122, PM3222, PM3332, PM2024, PM2124, PM2044, PM2144, PM3224, PM3334	eata_dma.o	
Sun Enterpriseネットワークアレイ(FC-AL)	fcal.o	
Future Domain TMC-16*xx* SCSI	fdomain.o	
NCR5380（汎用ドライバー）	g_NCR5380.o	
ICP RAIDコントローラ	gdth.o	
I2Oブロックドライバー	i2o_block.o	
IOMEGA MatchMakerパラレルポートSCSIアダプタ	imm.o	
Always IN2000 ISA SCSIカード	in2000.o	in2000=*setup_string*:*value* *OR* in2000 *setup_string*=*value*
Initio INI-9X00U/UW SCSIホストアダプタ	initio.o	
IBM ServeRAID	ips.o	
AMI MegaRAID 418, 428, 438, 466, 762	megaraid.o	
NCR SCSIコントローラ、810/810A/815/825/825A/860/875/876/895チップセット	ncr53c8xx.o	ncr53c8xx=*option1*:*value1*, *option2*:*value2*,... *OR* ncr53c8xx="*option1*:*value1* *option2*:*value2*..."
Pro Audio Spectrum/Studio 16	pas16.o	
PCI-2000 IntelliCache	pci2000.o	
PCI-2220I EIDE RAID	pci2220i.o	
IOMEGA PPA3パラレルポートSCSIホストアダプタ	ppa.o	

ハードウェア	モジュール	パラメータ
Perceptive Solutions PSI-240I EIDE	psi240i.o	
Qlogic 1280	qla1280.o	
Qlogic 2*x*00	qla2x00.o	
QLogic Fast SCSI FASXXX ISA/VLB/PCMCIA	qlogicfas.o	
QLogic ISP2100 SCSI-FCP	qlogicfc.o	
QLogic ISP1020インテリジェントSCSIカードIQ-PCI, IQ-PCI-10, IQ-PCI-D	qlogicisp.o	
Qlogic ISP1020 SCSI SBUS	qlogicpti.o	
Future Domain TMC-885, TMC-950 Seagate ST-01/02, Future Domain TMC-8*xx*	seagate.o	controller_type=2 base_address=*base_addr* irq=*IRQ*
sym53c416チップセット搭載カード	sym53c416.o	sym53c416=*PORTBASE*,*[IRQ]* *OR* sym53c416 io=*PORTBASE* irq=*IRQ*
Trantor T128/T128F/T228 SCSIホストアダプタ	t128.o	
Tekram DC-390(T) PCI	tmscsim.o	
UltraStor 14F/34F (非24F)	u14-34f.o	
UltraStor 14F、24F、34F	ultrastor.o	
WD7000シリーズ	wd7000.o	

表A-3. SCSIパラメータ

これらのモジュールの使用例をいくつか示します：

設定	例
Adaptec AHA1522、ポート330、IRQ 11、SCSI ID 7	aha152x=0x330,11,7
Adaptec AHA1542、ポート330	bases=0x330
Future Domain TMC-800 at CA000, IRQ 10	controller_type=2 base_address=0xca000 irq=10

表A-4. SCSIパラメータ設定例

A.4. イーサネットパラメータ

重要

殆どの最近のイーサネットベースのネットワークインターフェイスカード(NIC)は、設定を変更するのにモ

ジュールパラメータを必要としません。その代わりにそれらはethtool又はmii-toolを使用して設定できます。これらのツールが機能しない場合にのみ、モジュールパラメータを修正すべきです。

これらのツールの使用に関する情報は、ethtoolとmii-toolのそれぞれのmanページを参考にしてください。

ハードウェア	モジュール	パラメータ
3Com 3c501	3c501.o	3c501=*io_port,IRQ*
3Com 3c503 and 3c503/16	3c503.o	3c503=*io_port*,*IRQ OR* 3c503 io=*io_port_1*,*io_port_n* irq=*IRQ_1,IRQ_n*
3Com EtherLink Plus (3c505)	3c505.o	3c505=*io_port,IRQ OR* 3c505 io=*io_port_1*,*io_port_n* irq=*IRQ_1*,*IRQ_2*
3Com EtherLink 16	3c507.o	3c507=*io_port,IRQ OR* 3c507 io=*io_port* irq=*IRQ*
3Com EtherLink III	3c509.o	3c509=*io_port,IRQ*
3Com ISA EtherLink XL "Corkscrew"	3c515.o	
3Com EtherLink PCI III/XL Vortex (3c590, 3c592, 3c595, 3c597) Boomerang (3c900, 3c905, 3c595)	3c59x.o	full_duplex= *0* is off *1* is on
RTL8139, SMC EZカードFastイーサネット	8139too.o	
RTL8129か、RTL8139 Fastイーサネットチップセットを使用するRealTekカード	8139too.o	
Apricot 82596	82596.o	
Ansel Communications Model 3200	ac3200.o	ac3200=*io_port*,*IRQ OR* ac3200 io=*io_port_1*,*io_port_n* irq=*IRQ_1*,*IRQ_n*
Alteon AceNIC Gigabit	acenic.o	
Aironet Arlan 655	arlan.o	
Allied Telesis AT1700	at1700.o	at1700=*io_port*,*IRQ OR* at1700 io=*io_port* irq=*IRQ*
Broadcom BCM5700 10/100/1000イーサネットアダプタ	bcm5700.o	
Crystal SemiconductorCS89[02]0	cs89x0.o	

ハードウェア	モジュール	パラメータ
EtherWORKS DE425 TP/COAX EISA, DE434 TP PCI, DE435/450 TP/COAX/AUI PCI DE500 10/100 PCI Kingston, LinkSys, SMC8432, SMC9332, Znyx31[45], Znyx31[45]、DC21040 (SROMなし), DC21041[A], DC21140[A], DC21142, DC21143チップセット搭載のZnyx346 10/100カード	`de4x5.o`	de4x5=*`io_port`* *OR* de4x5 io=*`io_port`* de4x5 args='eth*`X[fdx]`* autosense=*`MEDIA_STRING`*'
D-Link DE-600イーサネットPocketアダプター	`de600.o`	
D-Link DE-620イーサネットPocketアダプター	`de620.o`	
DIGITAL DEPCA & EtherWORKS DEPCA, DE100, DE101, DE200 Turbo, DE201Turbo DE202 Turbo TP/BNC, DE210, DE422 EISA	`depca.o`	depca=*`io_port`*,*`IRQ`* *OR* depca io=*`io_port`* irq=*`IRQ`*
Digi Intl. RightSwitch SE-X EISA、PCI	`dgrs.o`	
Davicom DM9102(A)/DM9132/ DM9801 Fastイーサネット	`dmfe.o`	
Intel Ether Express/100 ドライバー	`e100.o`	e100_speed_duplex=*`X`* *If `X` =* 0 = autodetect speed and duplex 1 = 10Mbps, half duplex 2 = 10Mbps, full duplex 3 = 100Mbps, half duplex 4 = 100Mbps, full duplex
Intel EtherExpress/1000 Gigabit	`e1000.o`	
Cabletron E2100	`e2100.o`	e2100=*`io_port`*,*`IRQ`*,*`mem`* *OR* e2100 io=*`io_port`* irq=*`IRQ`* mem=*`mem`*
Intel EtherExpress Pro10	`eepro.o`	eepro=*`io_port`*,*`IRQ`* *OR* eepro io=*`io_port`* irq=*`IRQ`*
Intel i82557/i82558 PCI EtherExpressPro ドライバー	`eepro100.o`	

ハードウェア	モジュール	パラメータ
Intel EtherExpress 16 (i82586)	eexpress.o	eexpress=*io_port*,*IRQ* *OR* eexpress io=*io_port* irq=*IRQ* options= 0x10 10base T half duplex 0x20 10base T full duplex 0x100 100base T half duplex 0x200 100baseT full duplex
SMC EtherPower II 9432 PCI (83c170/175 EPICシリーズ)	epic100.o	
Racal-Interlan ES3210 EISA	es3210.o	
ICL EtherTeam 16i/32 EISA	eth16i.o	eth16i=*io_port*,*IRQ* *OR* eth16i ioaddr=*io_port* IRQ=*IRQ*
EtherWORKS 3 (DE203, DE204 and DE205)	ewrk3.o	ewrk=*io_port*,*IRQ* *OR* ewrk io=*io_port* irq=*IRQ*
A Packet Engines GNIC-II Gigabit	hamachi.o	
HP PCLAN/plus	hp-plus.o	hp-plus=*io_port*,*IRQ* *OR* hp-plus io=*io_port* irq=*IRQ*
HP LAN イーサネット	hp.o	hp=*io_port*,*IRQ* *OR* hp io=*io_port* irq=*IRQ*
100VG-AnyLanネットワークアダプタHP J2585B, J2585A, J2970, J2973, J2573 Compex ReadyLink ENET100-VG4, FreedomLine 100/VG	hp100.o	hp100=*io_port*,*name* *OR* hp100 hp100_port=*io_port* hp100_name=*name*
IBM Token Ring 16/4, Shared-Memory IBM Token Ring 16/4	ibmtr.o	ibmtr=*io_port* *OR* io=*io_port*
AT1500, HP J2405A, 殆どのNE2100/クローン	lance.o	
Mylex LNE390 EISA	lne390.o	
NatSemi DP83815 Fastイーサネット	natsemi.o	
NE1000 / NE2000 (非-pci)	ne.o	ne=*io_port*,*IRQ* *OR* ne io=*io_port* irq=*IRQ*
PCI NE2000カードRealTEk RTL-8029, Winbond 89C940, Compex RL2000, PCI NE2000クローン、NetVin, NV5000SC, Via 82C926, SureCom NE34	ne2k-pci.o	
Novell NE3210 EISA	ne3210.o	

ハードウェア	モジュール	パラメータ
MiCom-Interlan NI5010	`ni5010.o`	
NI5210 card (i82586イーサネットチップ)	`ni52.o`	ni52=*`io_port`*,*`IRQ`* *OR* ni52 io=*`io_port`* irq=*`IRQ`*
NI6510イーサネット	`ni65.o`	
IBM Olympic-based PCI token ring	`olympic.o`	
AMD PCnet32 and AMD PCnetPCI	`pcnet32.o`	
SIS 900/701G PCI Fastイーサネット	`sis900.o`	
SysKonnect SK-98*XX* Gigabit	`sk98lin.o`	
SMC Ultra and SMC EtherEZ ISAイーサカード(8K, 83c790)	`smc-ultra.o`	smc-ultra=*`io_port`*,*`IRQ`* *OR* smc-ultra io=*`io_port`* irq=*`IRQ`*
SMC Ultra32 EISAイーサネットカード(32K)	`smc-ultra32.o`	
Sun BigMacイーサネット	`sunbmac.o`	
Sundance ST201 Alta	`sundance.o`	
Sun Happy Mealイーサネット	`sunhme.o`	
Sun Quadイーサネット	`sunqe.o`	
ThunderLAN	`tlan.o`	
Digital 21x4x Tulip PCIイーサネットカード、SMC EtherPower 10 PCI(8432T/8432BT) SMC EtherPower 10/100 PCI(9332DST) DEC EtherWorks 100/10 PCI(DE500-XA) DEC EtherWorks 10 PCI(DE450) DEC QSILVER's, Znyx 312 etherarray Allied Telesis LA100PCI-T Danpex EN-9400, Cogent EM110	`tulip.o`	io=*`io_port`*
VIA VT86c100A Rhine-II PCIか、3043 Rhine-I D-Link DFE-930-TX PCI 10/100 のいずれかを搭載するVIA Rhine PCI Fast イーサネットカード	`via-rhine.o`	
AT&T GIS (前NCR) WaveLan ISAカード	`wavelan.o`	wavelan=*`[IRQ,0],io_port, NWID`*

ハードウェア	モジュール	パラメータ
WD8003 and WD8013-「互換」イーサネットカード	`wd.o`	wd=*`io_port`*,*`IRQ`*,*`mem`*, *`mem_end`* *OR* wd io=*`io_port`* irq=*`IRQ`* mem=*`mem`* mem_end=*`end`*
Compex RL100ATX-PCI	`winbond.o`	
Packet Engines Yellowfin	`yellowfin.o`	

表A-5. イーサネットモジュールパラメータ

これらのモジュールの使用例をいくつか示します：

設定	例
NE2000 ISAカード、IOアドレス300、IRQ 11	ne=0x300,11 ether=0x300,11,eth0
Wavelan card、IO 390、IRQ自動検索、NWIDを0x4321で使用	wavelan=0,0x390,0x4321 ether=0,0x390,0x4321,eth0

表A-6. イーサネットパラメータ設定例

A.4.1. 複数のイーサネットカードの使用

1台のマシンで複数のイーサネットカードを使用することができます。それぞれのカードが別々のドライバーを使用する場合は（たとえば3c509とDE425の場合）、`/etc/modules.conf`へ各カードに対する`alias`（場合によっては`options`も）行を追加するだけです。詳細については、*Red Hat Linux* カスタマイズガイドのカーネルモジュールを参照してください。

2枚のイーサネットカードが同じドライバー（たとえば2枚の3c509や、3c595と3c905）を使用する場合は、ドライバーのオプション行で2枚のカードのアドレスを指定する（ISAカードの場合）か、単にカードごとに`alias`行を1行追加する（PCIカードの場合）かのいずれかが必要です。

複数のイーサネットカードの使用に関する詳細は、以下のサイトで*Linux Ethernet-HOWTO*を参照してください。http://www.redhat.com/mirrors/LDP/HOWTO/Ethernet-HOWTO.html.

索引

Symbols

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

X

あとがき

Red Hat Linux マニュアルはDocBook SGML v4.1形式で書かれています。HTML版とPDF版はカスタムDSSSLスタイルシートとカスタムjade wrapperスクリプトを使用して作成されています。DocBook SGMLファイルは、PSGMLモードの支持を使用して、**Emacs**で書かれています。

Garrett LeSageがアドモーショングラフィクスを製作しました(注意、ヒント、重要、用心、警告など)。これらは自由にRed Hatのドキュメントと一緒に使用することができます。

Red Hat Linux製品ドキュメントチームは以下のメンバーから構成されています。:

Sandra A. Moore — *Red Hat Linux x86* インストールガイドの主任ライター/管理人; *Red Hat Linux* 入門ガイドの支援ライター。

Tammy Fox — *Red Hat Linux* カスタマイズガイドの主任ライター/管理人; *Red Hat Linux* 入門ガイドの支援ライター; カスタムDocBook スタイルシートとスクリプトのライター/管理人

Edward C. Bailey — *Red Hat Linux* システムアドミニストレーションプレミアの主任ライター/管理人; *Red Hat Linux x86* インストールガイドの支援ライター

Johnray Fuller — *Red Hat Linux* 参照ガイドの主任ライター/管理人; *Red Hat Linux* セキュリティガイドの共同ライター/共同管理人; *Red Hat Linux* システムアドミニストレーションプレミアの支援ライター

John Ha — *Red Hat Linux* 入門ガイドの主任ライター/管理人; *Red Hat Linux* セキュリティガイドの共同ライター/共同管理人; *Red Hat Linux* システムアドミニストレーションプレミアの支援ライター

James Kiyoko Hashida — *Red Hat Linux* カスタマイズガイド及び*Red Hat Linux* 参照ガイドの翻訳者：橋田喜代人 ; Noriko Mizumoto — *Red Hat Linux x86* インストールガイド及び*Red Hat Linux* 入門ガイドの翻訳者：水本紀子