Debian GNU/Linux 4.0 (“etch”) リリースノート (AMD64 用)

Josip Rodin, Bob Hilliard, Adam Di Carlo, Anne Bezemer, Rob Bradford, Frans Pop (現在のメンテナ), Andreas Barth (現在のメンテナ), Javier Fernández-Sanguino Peña (現在のメンテナ), Steve Langasek (現在のメンテナ)
<debian-doc@lists.debian.org>

$Id: release-notes.en.sgml,v 1.312 2007-08-16 22:24:38 jseidel Exp $

目次

章 1

はじめに

本リリースノートの主要な目的は、Debian GNU/Linux ディストリビューションのこのリリースにおける主要な変更点、以前のリリースから今回のリリースに安全にアップグレードする方法、そして今回のリリースにアップグレードする際や etch リリースを使用する際にユーザが遭遇する可能性がある既知の問題点について、情報をユーザに提供することです。

既知の問題点をすべてリストアップすることは不可能なので、問題点の予想される広がり具合と影響の大きさの両方に基づいて取捨選択していることに注意してください。

この文書の最新版は常に http://www.debian.org/releases/stable/releasenotes から入手可能です。お読みのバージョンが 1 ヵ月以上前のものでしたら[1]、最新版を取得したほうがよいでしょう。

この文書では、Debian の 1 つ前のリリースからのアップグレード (この場合、sarge からのアップグレード) のみがサポート・記述されていることに注意してください。さらに古いリリースからのアップグレードが必要な場合は、過去のリリースノートを読み、まず sarge へとアップグレードすることをお勧めします。

1.1 この文書に関するバグを報告する

筆者たちは、この文書で説明されているすべての異なるアップグレード手順を試し、また、発生する可能性がありユーザが直面するかもしれないすべての問題も予想しようとしました。

そうはいっても、この文書にバグ (不正確な情報や抜け落ちている情報) を見つけたと思う場合は、release-notes パッケージに対するバグ報告をバグ追跡システム (http://bugs.debian.org/)に提出してください。

[1] PDF 版の場合は最初のページに、HTML 版の場合はフッターに、いつのバージョンかが記載されています。

1.2 アップグレードについての報告をする

sarge から etch へのアップグレードに関連するユーザからの情報はどんなものでも歓迎します。情報を共有するのを厭わない場合は、`upgrade-reports` パッケージに対するバグ報告にアップグレードの結果を含めてバグ追跡システム (http://bugs.debian.org/)に提出してください。報告に添付ファイルを含める場合は、(`gzip` を使用して) 圧縮するようお願いします。

アップグレードについての報告を提出する際には、以下の情報を含めてください。

- アップグレード前後のパッケージデータベースの状態。`/var/lib/dpkg/status` にある `dpkg` の状態データベースと、`/var/lib/aptitude/pkgstates` にある `aptitude` のパッケージ状態情報です。'あらゆるデータや設定情報をバックアップする' on page 13 で説明するように、アップグレードを実行する前にバックアップをとっておくべきです。しかし、これらの情報のバックアップは `/var/backups` にもとられます。
- `script` を使用してとったセッションのログ。'セッションの記録' on page 21 で説明します。
- `aptitude` のログ。`/var/log/aptitude` にあります。

注意: バグ報告に情報を含める前に少し時間をかけてログに機密情報が含まれていないか検査し、機密情報を見つけた場合はそれをログから削除してください。というのも、バグ報告に含まれる情報は公開データベースで公表されるからです。

1.3 この文書のソース

この文書は `debiandoc-sgml` を使用して生成されています。リリースノートのソースは *Debian* ドキュメンテーションプロジェクト *(Debian Documentation Project)* の CVS リポジトリにあります。ウェブからウェブインタフェース (http://cvs.debian.org/ddp/manuals.sgml/release-notes/?root=debian-doc)を使って個々のファイルにアクセスでき、変更を参照できます。CVS へのアクセス方法に関してさらに詳しく知りたい場合は、『Debian ドキュメンテーションプロジェクトの CVS』のページ (http://www.debian.org/doc/cvs)を参照してください。

章 2

Debian GNU/Linux 4.0 の最新情報

このリリースでは、Intel (EM64T) および AMD (AMD64) 双方の 64 ビットプロセッサをサポートした、AMD64 アーキテクチャのサポートが公式に追加されています。前のリリース Debian GNU/Linux 3.1 ('sarge') では、この移植の非公式版が利用可能でした。この非公式版からのアップグレードは、本リリースノートを参照すれば可能なはずですが、Debian による公式のサポートはなされていません。

Debian リリースマネージャが課した条件を満たせなかったので、Motorola 680x0 ('m68k') アーキテクチャは公式にはサポートされなくなりました。最も大きな理由は、パフォーマンスの問題と必須ツールチェインコンポーネントの開発元でのサポートが限られたものだったことです。しかし、公式な安定版リリースの一部ではないとはいえ、m68k 移植版はまだ動作し、引き続きインストールが可能だと予想されます。

Debian GNU/Linux etch で公式にサポートされているアーキテクチャは以下のとおりです。

- Intel x86 ('i386')
- Alpha ('alpha')
- SPARC ('sparc')
- PowerPC ('powerpc')
- ARM ('arm')
- MIPS ('mips' (ビッグエンディアン) と 'mipsel' (リトルエンディアン))
- Intel Itanium ('ia64')
- HP PA-RISC ('hppa')
- S/390 ('s390')
- AMD64 ('amd64')

移植状況の詳細や、お使いの移植版に特有の情報については、Debian の移植版に関するページ (http://www.debian.org/ports/amd64/)で読むことができます。

2.1 ディストリビューションの最新情報

Debian のこの新しいリリースには、一つ前のリリースである sarge に付属していたよりさらに多くのソフトウェアが付属しています。このディストリビューションには、6500 以上の新しいパッケージが含まれており、全体のパッケージ数は 18200 以上になりました。ディストリビューション中のほとんどのソフトウェア、すなわち約 10700 ものソフトウェアパッケージ (これは sarge のパッケージ全体の 68% にあたります) が更新されました。また、かなりの数のパッケージ (sarge のパッケージの 23% にあたる 3500 以上) が、様々な理由でディストリビューションから取り除かれました。これらのパッケージについては更新されておらず、パッケージ管理用のフロントエンドでは 'obsolete' というマークがつけられます。

このリリースで Debian GNU/Linux は、XFree86 から X.Org の 7.1 リリースに移行しました。より広い範囲のハードウェアがサポートされ、自動認識もよりよいものになっています。この移行によって X Window System 用で初めてのコンポジティングウィンドウマネージャの一つである Compiz が使えるようになり、サポートされているデバイスではハードウェア OpenGL アクセラレーションを十分に活用できます。

今回も Debian GNU/Linux にはいくつかのデスクトップアプリケーションやデスクトップ環境が含まれています。特に、GNOME 2.14[1]、KDE 3.5.5a、Xfce 4.4 などがあります。業務用のアプリケーションもアップグレードされました。オフィススイートの OpenOffice.org 2.0.4a や KOffice 1.6、それに GNUcash 2.0.5、GNUmeric 1.6.3、Abiword 2.4.6 などです。

他のデスクトップアプリケーションも更新されました。Evolution 2.6.3 や Gaim 2.0 などです。Mozilla スイートも更新され、主要なプログラムの名前が変わりました。`iceweasel` (バージョン 2.0.0.2) は `Firefox` ウェブブラウザの改名版で、`icedove` (バージョン 1.5) は `Thunderbird` メールクライアントの改名版です。

またこのリリースには、特に挙げるなら、以下のソフトウェアの更新も含まれています。

- GNU C ライブラリのバージョン 2.3.6
- GNU Compiler Collection 4.1 (デフォルトのコンパイラ)
- インタープリタ言語: Python 2.4、PHP 5.2
- サーバソフトウェア:
  - メールサーバ: Exim 4.63 (新規インストール時のデフォルトのメールサーバ)、Postfix 2.3、Courier 0.53、Cyrus 2.2
  - ウェブサーバ: Apache 2.2、fnord 1.10
  - データベースサーバ: MySQL 5.0.32、PostgreSQL 8.1
  - OpenSSH サーバのバージョン 4.3
  - ネームサーバ: Bind 9.3、maradns 1.2
  - ディレクトリサーバ: OpenLDAP 2.3

公式の Debian GNU/Linux ディストリビューションは、現在は 19 ～ 23 枚のバイナリ CD と、ほぼ同数のソース CD の形で提供されています。DVD 版のディストリビューションも利用可能です。

[1] 一部のモジュールは GNOME 2.16 のものです。

2.1.1 パッケージ管理

aptitude は、コンソール上でのパッケージ管理に適したプログラムです。apt-get のコマンドライン操作の大半をサポートしており、apt-get よりも依存関係の解決の面で優れているとわかっています。まだ dselect を使用している場合は、公式のパッケージ管理用フロントエンドとして aptitude に移行すべきでしょう。

etch では、衝突を解決するための先進的な仕組みが aptitude で実装されました。この仕組みは、衝突が検出された原因がパッケージ間の依存関係の変化にある場合に、競合に対する最善の解決方法を見つけようとします。

Secure APT が etch で使えるようになりました。この機能は、強力な暗号化とデジタル署名を簡単にサポートし、ダウンロードしたパッケージを検証できるようにすることで、Debian GNU/Linux システムにさらなるセキュリティを追加します。今回のリリースには、apt のキーリングに新しい鍵を追加するための apt-key ツールも含まれています。debian-archive-keyring パッケージで提供されている apt のキーリングには、デフォルトでは現在の Debian アーカイブを署名するための鍵しか含まれていないからです。

デフォルトの設定では、apt は、検証されていないソースからパッケージをダウンロードする場合に警告を発します。将来のリリースでは、すべてのパッケージについて、ダウンロード前に検証しなければならないようになるでしょう。非公式な apt リポジトリの管理者には、暗号鍵を生成してその鍵で Release ファイルに署名し、さらにその公開鍵を配布する安全な手段を提供することをお勧めします。

さらに詳しい情報は、apt(8)・*Debian* 安全化マニュアル *(Securing Debian Manual)* の Package signing in Debian (http://www.debian.org/doc/manuals/securing-debian-howto/ch7#s-deb-pack-sign) の章・Wiki 文書 (http://wiki.debian.org/SecureApt)を参照してください。

apt に追加されたもう 1 つの機能は、最後に update を実行してから Packages ファイルに加えられた変更のみをダウンロードできる機能です。この機能に関するさらに詳しい情報は 'APT のパッケージインデックスファイルの更新が遅くなります' on page 38 にあります。

2.1.2 debian-volatile が公式サービスに

sarge のリリースとともに非公式のサービスとして導入された *debian-volatile* サービスが、Debian GNU/Linux の公式のサービスとなりました。

つまり、.debian.org のアドレスがサービスに使用されるようになりました[2]。このサービスを既に利用している場合は、忘れずに /etc/apt/sources.list を更新してサービスのアドレスを変更してください。

debian-volatile を利用すると、すぐに古くなってしまう情報を含む安定版 (stable) パッケージをユーザが容易に更新できます。そのような情報としては、例えばウイルススキャナのシグネチャリストや、スパムフィルタのパターンセットがあります。さらに詳しい情報やミラーのリストについては、アーカイブのウェブページ (http://volatile.debian.org/)を参照してください。

[2] これまでの volatile.debian.net というアドレスも当分は引き続き有効です。

2.2 システムの改良

このディストリビューションには etch を新しくインストールする場合に利点となるいくつかの変更が含まれています。しかしそれは sarge からのアップグレードには自動的に適用されないでしょう。このセクションではもっとも重要な変更の概要について述べます。

基本的な開発用パッケージの優先度が低く 優先度が標準 *(standard)* だった多数の開発用パッケージは、優先度が任意 *(optional)* になりました。つまり、これらのパッケージは、もう標準ではインストールされません。このようなパッケージには、標準の C/C++ コンパイラである `gcc` のほか、いくつかのソフトウェア (`dpkg-dev`、`flex`、`make`) や開発用ヘッダ (`libc6-dev`、`linux-kernel-headers`) などがあります。

これらの開発用パッケージをシステムにインストールしたい場合は、`build-essential` パッケージをインストールするのが一番簡単な方法です。これによって上記のパッケージの多くが引きずられてインストールされます。

SELinux は優先度が標準 (standard) に (ただしデフォルトでは無効) SELinux のサポートに必要なパッケージの優先度が標準 *(standard)* に昇格しました。つまり、新規インストールの場合はこれらのパッケージがデフォルトでインストールされます。既存のシステムの場合は次のコマンドを実行すると SELinux をインストールできます。

```
# aptitude install selinux-basics
```

SELinux のサポートはデフォルトでは有効になっていないことに注意してください。SELinux をセットアップして有効にするのに必要な情報は、Debian Wiki (http://wiki.debian.org/SELinux) にあります。

デフォルトの inet スーパーデーモンが変更 etch では、`netkit-inetd` に代わって `openbsd-inetd` がデフォルトの inet スーパーデーモンになりました。サービスが何も設定されていなければ、デーモンは起動しないでしょう。これは標準で正しい動作です。新しいデフォルトの inet スーパーデーモンはアップグレード時に自動的にインストールされます。

デフォルトの `vi` クローンが変更 デフォルトでインストールされる `vi` クローンが、`nvi` から `vim` のコンパクトバージョン (`vim-tiny`) に変わりました。

`ext2/ext3` ファイルシステムでの標準機能の変更点 標準で、新しい ext2 と ext3 ファイルシステムは *dir_index* と *resize_inode* 機能つきで作成されます。*dir_index* 機能は大量のファイルがあるディレクトリでの処理をスピードアップします。*resize_inode* 機能はファイルシステムを使用中に (つまり、マウントされた状態で) サイズの変更を可能にします。

sarge からアップグレードするユーザは、`tune2fs` を使って手動で *dir_index* フラグを追加できます[3]。*resize_inode* フラグはすでに作成済みのファイルシステムには追加できません。`dumpe2fs -h` コマンドで、ファイルシステムにどのフラグが設定されているか調べることができます。

[3] *filetype* フラグは、おそらく sarge より前にインストールしたシステムを除き、ほとんどのファイルシステムですでに設定されているはずです。

etch ではデフォルトエンコーディングが UTF-8 に 新しくインストールした場合の Debian GNU/Linux のデフォルトエンコーディングは UTF-8 です。多くのアプリケーションはデフォルトで UTF-8 を使用するよう設定されています。

etch にアップグレードするユーザで、UTF-8 に変更したい方は、環境とロケール定義の再設定が必要です。システム全体のデフォルトエンコーディングは `dpkg-reconfigure locales` コマンドで変更できます。はじめにあなたの言語と国で UTF-8 ロケールを選択してください。そしてそれをデフォルトにセットしてください。UTF-8 に変更するということは、既存のファイルを以前の (古い) エンコーディングから UTF-8 に変換する必要があることに注意してください。

`utf8-migration-tool` パッケージは UTF-8 への移行を助けるツールです。しかし、このパッケージは etch に含めるのに間に合わなかったので不安定版 (unstable) でしか利用できません。このツールを利用する場合は、事前にデータと設定のバックアップを取っておくことを強くお勧めします。

一部のアプリケーションではまだ UTF-8 環境では正しく動かない可能性があることに注意してください。ほとんどが表示に関する問題です。

Debian Wiki (http://wiki.debian.org/Sarge2EtchUpgrade) には sarge と etch の違いについて、さらに情報があります。

2.3 カーネル関連の主要な変更点

Debian GNU/Linux 4.0 で提供されているカーネルのバージョンは、すべてのアーキテクチャにおいて 2.6.18 です。このリリースはまだ 2.4 系カーネルとある程度は互換性があります。しかし、Debian はもはや 2.4 系カーネルパッケージを提供することはありません。

カーネルそのものとカーネルのパッケージングに大きな変更がありました。変更のうちいくつかはアップグレード手続きを複雑にします。そして etch にアップグレードした後、システムを再起動したとき問題が発生する可能性があります。このセクションでは、最も重要な変更の概要を述べます。起こる可能性のある問題とその対処法については、後の章で述べます。

2.3.1 カーネルパッケージングにおける変更

カーネルパッケージの名前が変更 名前空間を整理するため、Linux カーネルパッケージはすべて `kernel-*` から `linux-*` に名前が変わりました。この変更によって、将来、Debian に Linux 以外のカーネルを含めやすくなるでしょう。

AMD64 用のカーネルが単一の汎用カーネルに sarge では、このアーキテクチャにおいて異なるプロセッサファミリーごとにそれぞれのカーネルフレーバーがありました。システムのプロセッサに合わせて、カーネルを自動的に最適化するというカーネル内の変更により、カーネルのフレーバーを分ける必要がまったくなくなりました。

標準のカーネルで SMP が利用可能に マルチプロセッサシステムには、Linux カーネルの `*-smp` フレーバーはもう必要ありません。AMD64 では、`-smp` という接尾辞のない

linux-image パッケージはユニプロセッサとマルチプロセッサの両方のシステムをサポートしています。

廃止されたパッケージのために、可能な場合は、新しいパッケージに依存する移行用のダミーパッケージが用意されています。

2.3.2 新しい initrd 生成ユーティリティ

AMD64 の Debian カーネルイメージパッケージはシステムを起動するために initrd を必要とします。カーネル内での変更により、sarge での initrd を生成するためのユーティリティであった initrd-toolsはもはや使用されないので廃止されました。その代わりとして 2 つの新しいユーティリティが開発されました。initramfs-tools と yaird です。この 2 つの新しいユーティリティの背後にあるコンセプトはかなり異なっています。Debian Wiki (http://wiki.debian.org/InitrdReplacementOptions) にはその概要があります。両方のユーティリティとも 圧縮された cpio アーカイブである *initramfs* ファイルシステムを使用して initrd を生成します。デフォルトになっているお勧めのユーティリティは、initramfs-tools です。

etch のカーネルにアップグレードすると、デフォルトでは initramfs-tools がインストールされます。

sarge からのアップグレードに必要なため、initrd-tools パッケージは etch にまだ含まれています。次のリリースでは廃止されるでしょう。

2.3.3 動的な /dev 管理とハードウェア検出

etch カーネルは devfs のサポートを提供しません。

devfs が、ユーザ空間での devfs の実装である udev に置き換えられました。

udev は、/dev ディレクトリにマウントされ、カーネルでサポートされているデバイスをそのディレクトリと結びつけます。そしてカーネルモジュールがロード・アンロードされるとき、カーネルにより生成されたイベントに基づき適切にデバイスファイルを動的に作成・削除します。udev は devfs よりさらに万能で、hal (ハードウェア抽象化レイヤ) のような他のパッケージで使用されるサービスを提供します。

udev はカーネルと連動し、ハードウェアを検出したり、検出したデバイスのモジュールをロードする働きをします。そのため、これは hotplug と衝突します。また、sarge においては、システムの起動中にモジュールをロードするのに discover が使用されていました。しかし、etch での discover の新しいバージョンはもうその機能を提供しません。discover はシステムのグラフィックコントローラの検出のために X.Org によってまだ使用されています。

Debian カーネルイメージをインストールすると、デフォルトで udev もインストールされます。initramfs-tools が udev に依存しているためです。

モジュール化されていないカスタムカーネルをコンパイルしたり、yaird のような別の initrd 生成ユーティリティを使用すれば、udev をインストールしなくて済みます。しかし、initramfs-tools がお勧めの initrd 生成ユーティリティです。

章 3

インストールシステム

Debian Installer は公式の Debian インストールシステムです。このインストーラは、様々なインストール方法を提供しています。お使いのシステムにインストールするのにどの方法が利用できるかは、使っているアーキテクチャに依存します。

etch 用のインストーラのイメージは、インストールガイドとともに Debian のウェブサイト (http://www.debian.org/releases/stable/debian-installer/)にあります。

インストールガイドは、Debian 公式 CD/DVD セットの 1 枚目の CD/DVD 内の、次の場所にも含まれています。

```
/doc/install/manual/言語/index.html
```

これまでに知られている問題点を列挙した debian-installer の正誤表 (http://www.debian.org/releases/stable/debian-installer/index#errata)も確認しておくとよいでしょう。

3.1 インストールシステムの最新情報

Debian Installer は、sarge での初めての公式リリース以降も活発に開発されています。その結果、ハードウェアサポートが改良され、ワクワクするような新機能がいくつか追加されました。

このリリースノートでは、インストーラにおける変更点のうち主要なもののみをリストアップします。sarge 以降になされた詳細な変更一覧の概要に興味がある場合は、Debian Installer のニュースの履歴 (http://www.debian.org/devel/debian-installer/News/)で閲覧可能な、etch 用ベータ版やリリース候補版 (RC) のリリースアナウンスを参照してください。

3.1.1 主要な変更点

インストール中の再起動が不要に インストールは以前は 2 つの部分に分かれていました。ベースシステムをセットアップしてそれを起動可能にするステージと、そ

の後で再起動してから、ユーザのセットアップ・パッケージ管理システムのセットアップ・(tasksel を使用した) 追加パッケージのインストールなどの面倒を見る base-config を実行するステージです。

etch では、第 2 ステージが Debian Installer 本体に統合されました。この統合には多数の利点があります。例えば、セキュリティが強化されます。また、インストールの最後の再起動後には新システムが正しいタイムゾーンを既にもった状態となり、デスクトップ環境をインストールしている場合はすぐにグラフィカルユーザインタフェースが起動されます。

新システムでは UTF-8 エンコーディングがデフォルトに インストーラは、古くから使われてきた言語固有のエンコーディング (ISO-8859-1、EUC-JP、KOI-8 など) ではなく、UTF-8 エンコーディングを使用するシステムをセットアップします。

より柔軟なパーティション分割 ガイドつきのパーティション分割を利用して LVM ボリュームにファイルシステムをセットアップ可能になりました。

インストーラは暗号化ファイルシステムもセットアップできます。パーティションの手動分割を使用する場合、dm-crypt と loop-aes のどちらを使用するか、パスフレーズとランダムなキーのどちらを使用するかを選択したり、様々な他のオプションを調整できます。ガイドつきのパーティション分割を利用する場合、インストーラは、他の (/boot 以外の) あらゆるファイルシステムを論理ボリュームとして含む暗号化 LVM パーティションを作成します。

グラフィカルユーザインタフェース グラフィカルユーザインタフェースの方が好きな場合は、installgui をつけてインストーラを起動してみてください。

グラフィカルインストーラは、表現方法が異なるだけで、機能的には通常のインストーラとほぼ同等です。唯一の例外は、グラフィカルフロントエンドが、ランダムなキーを用いた暗号化パーティションのセットアップをサポートしていないということです。

グラフィカルユーザインターフェースの主な利点は、通常のユーザインターフェース (newt) よりも多くの言語をサポートしていることです。グラフィカルインストーラについての詳細と、グラフィカルインストーラと通常のインストーラの重要な相違点は、インストールガイドの付録で記述しています。

注意: グラフィカルユーザインタフェースはすべてのアーキテクチャで利用可能ではありません。

レスキューモード 起動できないなどの問題がシステムに生じた場合、その解決にインストーラを利用できます。最初のステップは通常のインストールと同様ですが、インストーラはパーティション作成プログラムを起動しません。代わりにレスキューオプションのメニューを表示します。

インストーラに rescue を与えて起動するかブートパラメータ rescue/enable=true を追加して、レスキューモードをアクティブにしてください。

root アカウントの代わりに sudo を使用 エキスパートモードでのインストール中に、root アカウントをセットアップしないよう選択できます (root アカウントはロックされま

す)。root アカウントをセットアップしない場合は代わりに sudo をセットアップし、最初のユーザがシステム管理のためにこのコマンドを使用できるようにします。

ダウンロードしたパッケージの暗号署名による検証 インストーラがダウンロードしたパッケージは、暗号署名を通じて、apt を用いて検証されるようになりました。これによって、ネットワーク経由でインストール中のシステムのセキュリティを侵害するのがさらに難しくなりました。

メールの設定が簡単に 「標準システム」のインストール時に、インストーラは、ローカルにメールを配送するだけのメールサーバの基本設定を、システムにセットアップします。このメールサーバは、同一ネットワークに接続している他のシステムとのメールのやりとりはできません。システムに対してローカルでないメールのやりとりを (送信であれ受信であれ) 取り扱えるようにシステムを設定したい場合は、インストール後にメールシステムを再設定しなくてはなりません。

デスクトップの選択 ユーザがデスクトップを選択すると、インストールシステムは、デフォルトのデスクトップとして GNOME デスクトップをインストールします。

しかし、それ以外のデスクトップ環境をインストールしたいユーザには、ブートパラメータに以下を指定して簡単にインストールできます。KDE の場合には tasks="standard, kde-desktop"、Xfce の場合には tasks="standard, xfce-desktop" としてください。ネットワークミラーを追加パッケージ取得元として使わずに、フル CD イメージからインストールしている場合はうまく動作しません。DVD イメージやその他のインストール方法であればきちんと動作します。

KDE デスクトップや Xfce デスクトップをデフォルトでインストールする、独立した CD イメージも用意しています。

新しい言語 翻訳者の多大な努力により、Debian は、テキストユーザインターフェースを使用した場合は 47 の言語でインストールできるようになりました。sarge より 6 言語増えたことになります。このリリースで追加された言語は、ベラルーシ語、エスペラント語、エストニア語、クルド語、マケドニア語、タガログ語、ベトナム語、ウォロフ語です。ペルシャ語とウェールズ語の翻訳は更新されなかったため、このリリースではサポートされなくなりました。

グラフィカルユーザインターフェースを使用する場合、さらに 11 の言語がサポートされています。これらの言語は、非グラフィカル環境では文字を表示できないため、グラフィカルインストーラだけで選択できます。新規にサポートされた言語は、ベンガル語、ゾンカ語、グジャラート語、ヒンディー語、グルジア語、クメール語、マラヤーラム語、ネパール語、パンジャブ語、タミール語、タイ語です。

ロケールを選びたくないユーザは、インストーラの言語選択で C を選択できるようになりました。d-i 言語一覧 (http://d-i.alioth.debian.org/i18n-doc/languages.html)で詳細情報をご覧いただけます。

地域化やタイムゾーンの設定が簡単に 言語・国・タイムゾーンの設定が簡単になり、ユーザが要求される情報の量が減りました。インストーラは、どの言語が選択されたかによってシステムの国とタイムゾーンの情報を推測するようになり、推測できない場合は限られた選択肢だけを表示するようになりました。ユーザは、必要であれば国や言語の一般的でない組み合わせを選択することもできます。

改良されたシステム全体の地域化 以前 localization-config ツールが担当していた多くの国際化・地域化タスクは、現在 Debian インストーラか、個々のパッケージ自身に含まれています。つまり言語を選択すると、標準システムとデスクトップ環境で、自動的にその言語に必要なパッケージ (辞書、ドキュメント、フォント……) をインストールするということです。自動で扱われなくなった設定は、用紙サイズの設定と、言語による X ウィンドウシステムの高度なキーボード設定です。

言語特有のパッケージは、インストール中で利用可能な場合にのみ、自動的にインストールされることに注意してください。

3.1.2 自動インストール

前のセクションで述べたとおりインストーラに多数の変更が加えられたので、インストーラによる、事前設定ファイルを使用した自動インストールのサポートにも変更がありました。つまり、sarge のインストーラで動いた既存の事前設定ファイルがあっても、修正を加えずにそれが新しいインストーラで動くことは期待できません。

幸いなことに、事前設定の使用方法に関する豊富な文書を含んだ付録がインストールガイド (http://www.debian.org/releases/stable/installmanual)につけられました。

etch のインストーラには、インストールをさらに、そしてより簡単に自動化できるようにする、ワクワクするような新機能がいくつか追加されました。このインストーラは RAID や LVM、暗号化 LVM を使用した高度なパーティション分割もサポートしています。詳しくは付属文書を参照してください。

3.2 人気コンテスト

インストールシステムが昔のように popularity-contest パッケージをインストールするよう提案します。このパッケージは sarge にはデフォルトではインストールされませんでしたが、それより前のリリースではインストールされていました。

popularity-contest は、ディストリビューション内のどのパッケージが実際に使われているかについての有益な情報を、Debian プロジェクトに提供してくれます。この情報は、主にインストール CD-ROM に収録されるパッケージの優先順位を決めるために使われますが、Debian 開発者がもはやメンテナのいないパッケージを引き受けるかどうかを決める際にもよく参照されます。

popularity-contest からの情報は匿名で処理されます。この公式の調査に参加して Debian の改良を手伝っていただけるとありがたいです。

章 4

以前のリリースからアップグレードする

4.1 アップグレードの準備

アップグレードの前には、'etch で知っておくべき問題点' on page 37 に書かれている情報も読むことをお勧めします。この章に書かれている問題点は、アップグレードの過程と直接は関係がないかもしれませんが、それでもアップグレードを開始する前に知っておくべき重要事項である可能性があります。

4.1.1 あらゆるデータや設定情報をバックアップする

システムをアップグレードする前に、完全なバックアップを取っておくよう強くお勧めします。少なくとも、失いたくないデータや設定情報だけでもバックアップしておきましょう。アップグレードのツールや処理はきわめて信頼性の高いものですが、アップグレードの最中にハードウェア障害が起こると、システムに大きなダメージを与えることがありえます。

バックアップしたくなるであろう主な対象としては、`/etc`、`/var/lib/dpkg`、`/var/lib/aptitude/pkgstates` の中身、`dpkg --get-selections "*"` (引用符を忘れてはいけません) の出力などでしょう。

アップグレードの過程自体は、`/home` ディレクトリ以下は一切変更しません。とはいえ、(Mozilla スイートの一部や GNOME や KDE のデスクトップ環境などのように) ユーザが初めて新しいバージョンのアプリケーションを起動するときに、既存のユーザ設定を新たなデフォルト値で上書きしてしまうものがあるのも事実です。万一に備えて、ユーザのホームディレクトリにある隠しファイルと隠しディレクトリ (いわゆる「ドットファイル」) をバックアップしておくのがよいでしょう。古い状態に戻したり、再度設定する場合に役立つはずです。ユーザにもこのことについて知らせておいてください。

あらゆるパッケージのインストール処理はスーパーユーザ特権で実行されなければならないため、必要なアクセス権限を得るために root としてログインするか `su` や `sudo` を使ってください。

アップグレードにあたって事前に整えなければならない条件がいくつかあります。実際にアップグレードを実行する前にそれらを確認してください。

4.1.2 事前にユーザに通知する

アップグレードの前には、その予定をすべてのユーザに知らせるとよいでしょう。しかしシステムに ssh 接続などでアクセスしてきているユーザは、アップグレードの最中にそうと気付くことはほとんどないはずですし、作業を続行できるはずです。

万一の用心をしたければ、アップグレードの前にユーザのパーティション (/home) をバックアップして、アンマウントしてしまいましょう。

etch にアップグレードするときはおそらくカーネルをアップグレードしなければならないので、通常は再起動が必要です。通常、再起動はアップグレードが完了した後に行います。

4.1.3 復旧の準備

ドライバやハードウェア検出、デバイスファイルの命名法や順序に関して sarge と etch との間ではカーネルに多くの変更が加えられたため、アップグレードのシステム再起動で問題に直面するリスクが高くなっています。既知の潜在的な問題点の多くは、このリリースノートの本章と次章で述べられています。

上述の理由により、システムが再起動に失敗したり、リモート管理されているシステムならネットワーク接続の確立に失敗した場合に備え、復旧できる手立てを整えておくことが大切です。

ssh 接続経由でリモートアップグレードを行うのなら、リモートのシリアル端末からサーバにアクセスできるよう必要な用心をしておくことを強く勧めます。カーネルをアップグレードして再起動した後、('デバイスの整列順序の変更' on page 30 で述べられているように) いくつかのデバイスの名前が変更され、ローカルコンソール経由でシステム設定を修正しなければならないことがあります。また、アップグレード途中でシステムが予期せぬ再起動を行った場合にも、ローカルコンソールを使って復旧する必要に迫られることもあります。

最初に試すべきもっとも明白なことは、古いカーネルでの再起動です。しかしながら、本文書の別の場所で述べられているいくつかの理由により、これがうまくいくという保証はありません。

古いカーネルでの再起動に失敗するなら、アクセスして修復できるようシステムを起動するための代替手段が必要となるでしょう。1 つのオプションとしては、特別な復旧イメージや Linux ライブ CD を使うことがあります。これらを使って起動した後は、ルートファイルシステムをマウントしてから、問題点を調査解決するために chroot を実行できるはずです。

お勧めしたい別のオプションとしては、etch 用 Debian インストーラのレスキューモードを使うことがあります。同インストーラを使う利点は、多くのインストール手段の中からあなたの状況に最適なものを選べることです。より詳しい情報は、インストールガイド (http://www.debian.org/releases/stable/installmanual)の第 8 章にある "壊れたシステムの復旧" セクションや、Debian Installer FAQ (http://wiki.debian.org/DebianInstaller/FAQ) を参照してください。

initrd を使った起動中のデバッグシェル

initramfs-tools パッケージには、それが生成する initrd の中にデバッグシェル[1]が含まれています。例えば、initrd がルートファイルシステムをマウントできなければ、デバッグシェル内に移るでしょう。同シェルは、問題点を追跡してそれを修正する手助けとなる基本的なコマンドを備えています。

チェックすべき基本的事項としては、次のようなものがあります。/dev 内に適切なデバイスファイルが存在するか、ロードされているモジュール (cat /proc/modules)、dmesg の出力に示されたドライバのロード失敗など。dmesg の出力はまた、どのデバイスファイルがどのディスクに割り当てられているのかも示してくれます。ルートファイルシステムが予期した通りのデバイス上にあるかを確認するために、echo $ROOT の出力もチェックすべきでしょう。

問題点をしっかり解決できたなら、exit とタイプすることでデバッグシェルを終了させ、起動プロセスを失敗した時点から継続できます。もちろん次回の起動時に再び失敗することが無いよう、根本的な問題を修正して initrd を再生成する必要があるでしょう。

4.1.4 アップグレード用の安全な環境の準備

ディストリビューションのアップグレードは、ローカルのテキストモード仮想コンソール (あるいは直接接続されたシリアル端末) から行うか、リモートなら ssh 接続経由で行いましょう。

リモートでのアップグレード時にさらなるセーフティマージンを得るために、screen プログラムが提供する仮想コンソール内でアップグレードプロセスを実行することを提案します。同プログラムは安全な再接続を可能にし、リモート接続プロセスが切断された場合でもアップグレードプロセスが中断しないようにしてくれます。

重要! telnet、rlogin、rsh を用いてアップグレードをしてはいけません。アップグレードマシンの xdm、gdm、kdm などが管理している X セッションからのアップグレードも行うべきではありません。これらのサービスはアップグレードの最中に切断されてしまう可能性が高く、そうなるとアップグレード途中のシステムへの接続が不可能になってしまうからです。

4.1.5 2.2 系カーネルはサポートされなくなりました

2.4.1 より前のカーネルを使用している場合、glibc をアップグレードする前に (最低でも) 2.4 系にアップグレードする必要があります。これは、アップグレードを開始する前に済ませておくべきです。お勧めは、2.4 系のカーネルにアップグレードするのではなく、sarge で提供されているカーネル 2.6.8 に直接アップグレードすることです。

[1] この機能は、ブートパラメータに panic=0 を付加することで無効にできます。

4.2 システムの状態をチェックする

この章で述べられているアップグレード手順は、サードパーティ製のパッケージが無い “純粋” な sarge システムからのアップグレード用です。特に、/usr/X11R6/bin/ 内にプログラムをインストールするサードパーティ製パッケージには、X.Org への移行 (‘XFree86 から X.Org への移行’ on page 39) によってアップグレード時に問題が発生することが知られています。アップグレードプロセスにおいて最大限の信頼性を確保するために、アップグレード開始前にシステムからサードパーティ製パッケージを削除しておいた方が良いでしょう。

またこの手順は、システムが sarge の最新リリースにアップデート済みであるものと想定しています。そうではなかったり、アップグレード済みかどうか不明なら、‘sarge システムのアップグレード’ on page 51内の指示に従ってください。

4.2.1 パッケージマネージャ内の中断中のアクションを確認

パッケージをインストールするのに aptitude の代わりに apt-get を使用すると、時として、aptitude がそのパッケージを “未使用” だとみなし、削除対象とすることがあります。一般的にはアップグレードを行う前に、システムが完全に最新の状態で “クリーン” な状態となっているかを確認するべきです。

このため、パッケージマネージャ aptitude 内で中断しているアクションがあるかどうかを確認すべきでしょう。パッケージマネージャによってあるパッケージが削除あるいは更新の対象となっているなら、アップグレード手順に好ましくない影響を与えるかもしれません。この修正は、sources.list に *stable* や *etch* ではなく、*sarge* が指定されている段階でのみ可能なことに注意してください。‘ソースリストのチェック’ on page 51 も参照してください。

確認するためには、aptitude のユーザインターフェイスを起動して ‘g’ (“Go”) を押してください。何らかのアクションが表示されたなら、その内容を確認して修正するかあるいは提案されたアクションを実行すべきです。いかなるアクションも提案されない場合は、“インストール・削除・更新されるパッケージがありません” というメッセージが表示されるでしょう。

4.2.2 APT の pin 機能を無効にする

特定のパッケージを安定版以外 (テスト版など) のディストリビューションからインストールするように APT を設定しているなら、当該パッケージが新しい安定版リリース内のバージョンにアップグレードできるように、(/etc/apt/preferences 内に保存されている) APT の pin 設定を変更する必要があるかもしれません。APT の pin 機能に関するより詳しい情報は、apt_preferences(5) にあります。

4.2.3 パッケージの状態をチェックする

アップグレードに使う手段に関係なく、まず全パッケージの状態を調べ、全パッケージがアップグレード可能な状態にあることを確認することをお勧めします。次のコマンドは、イ

ンストールが中断していたり設定に失敗したパッケージや、何らかのエラー状態にあるパッケージを表示します:

```
# dpkg --audit
```

dselect や aptitude、あるいは次のようなコマンドを使ってシステムの全パッケージの状態を検査することもできます:

```
# dpkg -l | pager
```

または

```
# dpkg --get-selections "*" > ~/curr-pkgs.txt
```

アップグレード前に、あらゆる hold 状態を解除しておいたほうがよいでしょう。アップグレードに不可欠なパッケージが hold 状態にあるなら、アップグレードに失敗するでしょう。

hold 状態にあるパッケージを記録するのに、aptitude は apt-get や dselect とは異なる手法を用いることに注意してください。aptitude では、以下のように実行して hold 状態にあるパッケージを検出できます:

```
# aptitude search "~ahold" | grep "^.h"
```

apt-get でどのパッケージが hold 状態にあるのかを調べたければ、以下のように実行してください:

```
# dpkg --get-selections | grep hold
```

パッケージをローカルで変更したり再コンパイルしており、パッケージの名前を変えたりバージョン番号に epoch フィールドを追加していないなら、アップグレードしないよう hold 状態にしておかなければなりません。

aptitude でパッケージを “hold” 状態に変更するには、以下のように実行してください。

```
# aptitude hold パッケージ名
```

“hold” 状態を解除するには hold の代わりに unhold を使用してください。

修正が必要なことがあるなら、‘ソースリストのチェック’ on page 51で説明するように sources.list が sarge を指定したままにしておくべきです。

4.2.4 非公式なソースとバックポート

自分のシステムに非 Debian パッケージがあるなら、依存関係の衝突のためアップグレード中に削除されるかもしれないことに注意すべきです。当該パッケージが `/etc/apt/sources.list` に特別なパッケージアーカイブを追加することでインストールされたのなら、そのアーカイブが etch 用にコンパイルされたパッケージも提供しているかをチェックし、Debian パッケージ用のソース行と一緒にそのソース行も適切に修正すべきです。

自分の sarge システムに、Debian に存在するパッケージの 非公式にバックポートされた“新” バージョンをインストールしているユーザもいるでしょう。そのようなパッケージはファイルの衝突を引き起こすことにより、アップグレード中に問題を引き起こす場合がほとんどでしょう[2]。ファイル衝突が発生したときの対処方法については、‘アップグレード中の注意点’ on page 27にいくつかの情報があります。

4.3 パッケージのマークを手作業で外す

依存関係に引きずられてインストールされたいくつかのパッケージが `aptitude` によって削除されるのを防ぐために、それらが *auto* パッケージであるとの指定を手作業で外す必要があるでしょう。この中には、デスクトップインストール時の OpenOffice や Vim が含まれます。

```
# aptitude unmarkauto openoffice.org vim
```

さらにカーネルメタパッケージを使ってインストールした場合、2.6 系のカーネルイメージも対象となります。

```
# aptitude unmarkauto $(dpkg-query -W 'kernel-image-2.6.*' | cut -f1)
```

注意: 以下のように実行することで、aptitude 内でどのパッケージが *auto* と指定されているのかを確認できます:

```
# aptitude search 'i~M <パッケージ名>'
```

4.4 APT の取得先 (ソース) の準備

アップグレードを始める前に、`apt` の設定ファイル `/etc/apt/sources.list` を編集して、パッケージの取得先を決める必要があります。

`apt` は、“`deb`” 行にあるすべてのパッケージを見比べ、最も大きなバージョン番号のパッケージをインストールします。同じパッケージが取得可能な場合は、先に現れた行を優先し

[2] Debian のパッケージ管理システムは、対象パッケージを置き換えるように指定されていない限り、通常はあるパッケージが別のパッケージが所有しているファイルを削除したり置き換えることを許可しません。

ます (つまり、複数のミラーを指定している場合は、最初にローカルのハードディスクを、次に CD-ROM を、最後に HTTP/FTP ミラーを指定するといいでしょう)。

リリースを指定するのに、コードネーム (sarge や etch) と状態名 (old-stable、stable、testing、unstable) のどちらもよく使用されます。コードネームによる指定は、新しいリリースが出たときに驚かずに済むという利点があるため、ここではコードネームを使用しています。当然ですが、コードネームを使用している場合は自分でリリースアナウンスに注意を払わなければいけません。代わりに状態名を使用している場合は、リリースの直後にパッケージの更新が大量に利用可能になったことに気づくでしょう。

4.4.1 APT のインターネットソースの追加

デフォルトの設定では、メインの Debian インターネットサーバを使ってインストールするようになっています。ですがここでは、`/etc/apt/sources.list` を編集して、他のミラー (できればネットワーク的に最も近いミラー) を使うようにするほうがよいでしょう。

Debian の HTTP/FTP ミラーのアドレスは、`http://www.debian.org/distrib/ftplist` を参照してください。一般には HTTP ミラーのほうが FTP ミラーよりも高速です。

例えば、一番近くにある Debian ミラーが `http://mirrors.kernel.org/debian/` だったとしましょう。このミラーをウェブブラウザや FTP プログラムで見てみると、main などのディレクトリが以下のように構成されていることがわかります。

```
http://mirrors.kernel.org/debian/dists/etch/main/binary-amd64/...
http://mirrors.kernel.org/debian/dists/etch/contrib/binary-amd64/...
```

このミラーを `apt` で使うには、次の行を `sources.list` ファイルに追加します。

```
deb http://mirrors.kernel.org/debian etch main contrib
```

'`dists`' は書かなくても暗黙のうちに追加されます。そしてリリース名の後の引数がそれぞれ用いられ、複数のディレクトリの各々のパス名に展開されます。

新しいソースを追加した後、`sources.list` 内の既存の "`deb`" 行の先頭にシャープ記号 (#) を追加して、それらを無効にしてください。

4.4.2 APT のローカルミラーソースの追加

HTTP や FTP のパッケージミラーを使うのではなく、ローカルディスク (多分 NFS マウントされたもの) にあるミラーを使うよう、`/etc/apt/sources.list` を変更したいこともあるかもしれません。

例えばパッケージのミラーが `/var/ftp/debian/` にあり、main などのディレクトリが次のように配置されているとします。

```
/var/ftp/debian/dists/etch/main/binary-amd64/...
/var/ftp/debian/dists/etch/contrib/binary-amd64/...
```

これを apt から使うには、次の行を sources.list ファイルに追加します。

```
deb file:/var/ftp/debian etch main contrib
```

'dists' は書かなくても暗黙のうちに追加されます。そしてリリース名の後の引数がそれぞれ用いられ、複数のディレクトリの各々のパス名に展開されます。

新しいソースを追加した後、sources.list 内の既存の "deb" 行の先頭にシャープ記号 (#) を追加して、それらを無効にしてください。

4.4.3 APT の CD-ROM/DVD ソースの追加

CD だけでインストールをしたい場合は、/etc/apt/sources.list 中の "deb" 行の先頭にシャープ記号 (#) を置き、それらを無効にしてください。

CD-ROM ドライブをマウントポイント /cdrom にマウントすることを許可している行が /etc/fstab にあるかどうかを確認してください (apt-cdrom を使う場合は、マウントポイントを /cdrom 以外にはできません)。例えば /dev/hdc が CD-ROM ドライブなら、/etc/fstab には次のような行が必要です。

```
/dev/hdc /cdrom auto defaults,noauto,ro 0 0
```

第 4 フィールドの defaults,noauto,ro の間にはスペースがあってはいけません。

これが正しく機能しているか調べるには、CD を挿入して以下を実行してみてください。

```
# mount /cdrom    # マウントポイントに CD をマウントします
# ls -alF /cdrom  # CD のルートディレクトリを表示します
# umount /cdrom   # CD をアンマウントします
```

問題がなければ

```
# apt-cdrom add
```

を、Debian Binary CD-ROM それぞれに対して実行してください。各 CD に関するデータが APT のデータベースに追加されます。

4.5 パッケージのアップグレード

以前の Debian GNU/Linux からのアップグレード方法のお勧めは、パッケージ管理ツール aptitude を用いる方法です。このプログラムはパッケージに関する判断を apt-get よりも安全に行います。

まず、必要なパーティションが read-write モードでマウントされていることを忘れずに確認しましょう (特にルートパーティションと /usr パーティション)。以下のようなコマンドラインが使えます。

```
# mount -o remount,rw /マウントポイント
```

次に、(/etc/apt/sources.list 内の) APT ソースのエントリが “etch” と “stable” のいずれか一方を指定していることを念入りにチェックしてください。sarge を指し示すソースエントリが含まれないようにすべきです。注意: CD-ROM のソース行は “unstable” を指定していることがよくあります。これは混乱の元かもしれませんが、変更すべきではありません。

4.5.1 セッションの記録

ここで強くお勧めしたいのですが、/usr/bin/script プログラムを使って、このアップグレード作業の記録を取るようにしましょう。こうすれば何らかの問題が生じたときに、何が起こったかを記録しておくことができ、バグ報告の必要が生じた場合に、その正確な情報を提供できます。記録を開始するには次のように入力します。

```
# script -t 2>~/upgrade-etch.time -a ~/upgrade-etch.script
```

typescript ファイルは /tmp や /var/tmp のような一時ディレクトリには置かないでください (これらのディレクトリのファイルはアップグレードや再起動の際に削除されることがありますから)。

typescript はまた、スクロールしてスクリーンから消えた情報を見ることができるようにもしてくれるでしょう。(Alt-F2 を使って) 2 番の仮想コンソールに切り替えて、ログインしてから less -R ~root/upgrade-etch.script と実行すれば当該ファイルを見ることができます。

アップグレード完了後に script を停止するには、プロンプトから exit と入力してください。

script に *-t* スイッチをつけておいた場合は、以下のようなコマンドを使用して scriptreplay プログラムでセッション全体をリプレイできます。

```
# scriptreplay ~/upgrade-etch.time ~/upgrade-etch.script
```

4.5.2 パッケージリストの更新

まず、新しいリリースで利用可能なパッケージの一覧を取得する必要があります。そのためには以下のコマンドを実行してください。

```
# aptitude update
```

このコマンドを初めて実行して新しいソースを更新する際、ソースの取得性に関する警告がいくつか表示されます。これらの警告は無害なもので、コマンドを再び実行したときには表示されません。

4.5.3 アップグレードするのに十分な領域があることを確認する

システムアップグレードの前には、'残りのシステムのアップグレード' on page 26 で説明するシステム全体のアップグレードを開始するときに十分なハードディスク領域があるかどうかを確認しなければいけません。まず、ネットワーク経由で取得してインストールする必要があるどのようなパッケージも、/var/cache/apt/archives (およびダウンロード中には partial/ サブディレクトリ) に保存されます。したがって、システムにインストールされるパッケージをダウンロードして一時的に保存できるよう、/var/ を保持しているファイルシステムパーティションに十分な空き領域があることを確認しなければなりません。ダウンロード後にはおそらく、アップグレードされるパッケージ (これらには、より大きなバイナリやより多くのデータが含まれている可能性があります) と、アップグレードに伴って依存関係に引きずられて新たに導入されるパッケージの両方のインストールのために、他のファイルシステムパーティションにさらに領域が必要になるでしょう。システムに十分な空き領域がない場合、アップグレードが不完全な状態で終わり、復旧が困難になる可能性があります。

aptitude と apt のどちらを使っても、インストールに必要なディスク領域の詳細な情報が表示されます。アップグレードを実行する前に、次のように実行して必要な領域の推定値を見ることができます。

```
# aptitude -y -s -f --with-recommends dist-upgrade
[ ... ]
更新: XXX 個、新規インストール: XXX 個、削除: XXX 個、保留: XXX 個。
yyyMB 中 xx.xMB のアーカイブを取得する必要があります。展開後に追加で AAAMB
のディスク容量が消費されます。
パッケージのインストールまたは削除。
```

[3]

アップグレードをするのに十分な領域がない場合、事前に領域を解放するのを忘れないようにしてください。以下のことを実行するとよいでしょう。

[3] アップグレード手順の初めにこのコマンドを実行すると、以降のセクションで説明するような理由でエラーが発生する可能性があります。その場合は、このコマンドを実行してディスク領域の推定値を見る前に、まず'システムの最小アップグレード' on page 23で説明するとおりシステムの最小アップグレードを行い、さらに'カーネルのアップグレード' on page 25のようにカーネルをアップグレードする必要があります。

- インストールのために以前 (/var/cache/apt/archive に) ダウンロードしたパッケージを削除する。apt-get clean または aptitude clean を実行してパッケージキャッシュを一掃すると、以前ダウンロードしたパッケージファイルをすべて削除できます。
- もう使用しない古いパッケージを削除する。popularity-contest をインストールしていれば、popcon-largest-unused を使って、システムで使用していないパッケージのうち最も大きな領域を占めているものをリストアップできます。deborphan や debfoster を使って時代遅れのパッケージを見つけることも可能です ('時代遅れ (Obsolete) なパッケージ' on page 33 を参照してください)。それらのツールを使う代わりに aptitude を「ビジュアルモード」で起動すれば、古いパッケージは、「廃止された、またはローカルで作成されたパッケージ」の下に見つかります。
- あまりにも大きな領域を占めており現在は必要ないパッケージを削除する (アップグレード後にいつでも再インストール可能なのですから)。dpigs (debian-goodies パッケージに含まれています) や wajig (wajig size を実行してください) を用いると、ディスク領域の大部分を占めているパッケージをリストアップできます。
- /var/log/ の下にあるシステムログを一時的に他のシステムに移動するか、永久に削除する。

パッケージを安全に削除するための注意として、'ソースリストのチェック' on page 51で説明するように、sources.list が sarge を指し示すよう設定を戻しておくことが望ましいです。

4.5.4 システムの最小アップグレード

sarge で必要なパッケージの一部と etch で必要なパッケージの一部が衝突するため、直接 aptitude dist-upgrade を実行すると、多くの場合、一時的に固定しておきたいパッケージが多数削除される結果となります。そのため、まずはこれらの競合状態を打開するための最小アップグレードを行い、その上で完全な dist-upgrade を行う、という 2 段階のアップグレード過程を踏むことをお勧めします。

まず、次のコマンドを実行してください。

```
# aptitude upgrade
```

このコマンドには、アップグレードしても他のパッケージをインストール・削除する必要がないパッケージだけをアップグレードする、という効果があります。

最小アップグレードが完了したら、以下のコマンドを実行してください。

```
# aptitude install initrd-tools
```

このステップによって、libc6 と locales が自動的にアップグレードされ、SELinux サポート用のライブラリ群 (libselinux1) が引きずられてインストールされます。この時点

で、xdm や gdm、kdm などといった実行中のいくつかのサービスが再起動されます。その結果、ローカルの X11 セッションは切断されます。

次のステップは、どのようなパッケージ群がシステムにインストールされているかによって変化します。このリリースノートでは、どのような方法をとるべきかに関する一般的なアドバイスをします。しかし、確信がもてない場合は、それぞれの方法でアップグレードを先に進める前に、どのパッケージを削除するよう提案されているのかきちんと調べることをお勧めします。

どの場合でも削除されるだろうと予想されるパッケージには、base-config、hotplug、xlibs、netkit-inetd、python2.3、xfree86-common、xserver-c があります。etch で時代遅れとなるパッケージのさらに完全な一覧については、'時代遅れ (Obsolete) なパッケージ' on page 33を参照してください。

デスクトップシステムのアップグレード

このアップグレード手順は、sarge の「デスクトップ」タスクがインストールされているシステムで正しく機能することが確認されています。「デスクトップ」タスクがインストールされているか gnome パッケージまたは kde パッケージがインストールされているシステムでは、この手順に沿ってアップグレードすれば、おそらく最もよい結果になるでしょう。

次のようにして libfam0c102 パッケージや xlibmesa-glu パッケージがシステムにインストールされているかを調べたときにまだインストールされていないという結果になった場合は、おそらく、この方法は適用する方法としてふさわしくありません。

```
# dpkg -l libfam0c102 | grep ^ii
# dpkg -l xlibmesa-glu | grep ^ii
```

完全なデスクトップシステムがインストールされている場合、以下のコマンドを実行してください。

```
# aptitude install libfam0 xlibmesa-glu
```

X のパッケージがいくつかインストールされているシステムのアップグレード

X のパッケージがいくつかインストールされていても完全な「デスクトップ」タスクがインストールされているわけではないシステムには、異なる方法を使う必要があります。この方法は、一般に、xfree86-common パッケージがインストールされているシステムに当てはまります。そのようなシステムとしては、tasksel のサーバタスクがインストールされている一部のサーバシステムなどがあります。というのも、これらのタスクのうち一部には、グラフィカルな管理ツールを含むものがあるからです。X は実行できても完全な「デスクトップ」タスクはインストールされていないようなシステムを使用するのがおそらく正しい方法でしょう。

```
# dpkg -l xfree86-common | grep ^ii
```

まず、libfam0c102 パッケージと xlibmesa-glu パッケージがインストールされているか、次のようなコマンドで確かめてください。

```
# dpkg -l libfam0c102 | grep ^ii
# dpkg -l xlibmesa-glu | grep ^ii
```

libfam0c102 パッケージがシステムにインストールされていない場合は、以下のコマンドラインに libfam0 を含めないでください。また、xlibmesa-glu パッケージがインストールされていない場合は、以下のコマンドラインに xlibmesa-glu を含めないでください[4]。

```
# aptitude install x11-common libfam0 xlibmesa-glu
```

注意しなければならないのは、libfam0 をインストールすると、File Alteration Monitor (fam) や RPC ポートマッパー (portmap) がまだシステムで利用可能になっていない場合はそれらもインストールされる、ということです。どちらのパッケージも、(内部である) ループバックネットワークデバイスにバインドするように設定できはしますが、システムで新たなネットワークサービスを有効にすることになります。

X のサポートがインストールされていないシステムでのアップグレード

X のないシステムでは aptitude install コマンドを追加実行する必要はないはずなので、次のステップへとそのまま進めます。

4.5.5 カーネルのアップグレード

etch に含まれているバージョンの udev は、バージョン 2.6.15 より前のカーネル (これには sarge のカーネル 2.6.8 も含まれます) をサポートしておらず、逆に、sarge に含まれるバージョンの udev は最新のカーネルでは正しく動作しません。さらに、etch に含まれているバージョンの udev をインストールすると、2.4 系 Linux カーネルで使用されていた hotplug が強制的に削除されます。

その結果、このアップグレードを行うと、以前のカーネルパッケージはおそらく正しく起動しなくなります。また同様に、アップグレードの最中には、udev がアップグレードされている一方で最新のカーネルがまだインストールされていない状況が存在します。もし、アップグレードがまだ完了していないこのような状況でシステムを再起動することになったら、ドライバの検出やロードを正しく行えないためにシステムは起動できないでしょう (リモートからアップグレードしている場合には、発生する可能性があるこのような事態に対する心

[4]次のようなコマンドを用いれば、libfam0 と xlibmesa-glu をインストールする必要があるか判断した上で、必要に応じてそれらをインストール対象として自動的に選択します。

```
# aptitude install x11-common \ $(dpkg-query -showformat '${Package} ${Status}\n' -W
libfam0c102 xlibmesa-glu \ | grep 'ok installed$' | sed -e's/ .*//; s/c102//')
```

構えについて、‘アップグレード用の安全な環境の準備’ on page 15の忠告を参照してください)。

したがって、「デスクトップ」タスク、あるいはそれ以外でも受け入れ難い数のパッケージ削除を伴うパッケージがシステムにインストールされているのでなければ、この段階でカーネルだけをアップグレードすることをお勧めします。

このカーネルアップグレードを実行するには、次のコマンドを実行してください。

```
# aptitude install linux-image-2.6-フレーバー
```

カーネルパッケージのどのフレーバーをインストールすべきか判断するための手助けが欲しい場合は、‘カーネルメタパッケージのインストール’ on page 29を参照してください。

デスクトップの場合、残念ながら、新しいカーネルパッケージのインストールが新しい udev のインストールの直後に確実に行われるようにするのは不可能です。したがって、hotplug を完全にサポートしているカーネルがシステムにインストールされていない状況が、どのくらい長くかはわかりませんが存在します。システムの起動が hotplug に依存しないような設定に関する情報については、‘カーネルと関連パッケージのアップグレード’ on page 29を参照してください。

4.5.6 残りのシステムのアップグレード

さて、アップグレードの主要部分を続行する準備が整いました。以下のコマンドを実行してください:

```
# aptitude dist-upgrade
```

これによってシステムの完全なアップグレードを行います。すなわち、すべてのパッケージの最新版を入手し、リリース間でのパッケージの依存関係の変化すべてを解決します。必要に応じて、新しいパッケージ (通常は更新版のライブラリや、名前の変わったパッケージ) をインストールしたり、衝突している古いパッケージを削除したりもします。

CD-ROM のセットからアップグレードする場合には、アップグレード作業の最中に CD を交換するよう、数回指定されることになります。同じ CD を複数回入れなければならないかもしれません。これは、相互に依存しているパッケージが別々の CD に分散していることもあるからです。

現在インストールされているパッケージの更新版が、他のパッケージのインストール状態を変更しなければならないような場合には、そのパッケージは現在のバージョンのままになります (「固定されている」と表示されます)。この状態は、aptitude でこれらのパッケージをインストール対象として選択するか、aptitude -f install パッケージ名 を試すかのどちらかで解決できます。

4.5.7 パッケージの署名の取得

アップグレードを終えたら、新しいバージョンの apt を使用してシステムのパッケージ情報を更新できます。新しいバージョンの apt には、パッケージの署名を確認するための仕組みが新たに含まれています。

```
# aptitude update
```

アップグレードによって、Debian パッケージアーカイブの署名用の鍵を取得して有効にする作業は終わっているでしょう。パッケージソースとして他の (非公式の) ものを追加すると、apt は、そのパッケージソースからダウンロードされるパッケージが正規のものかどうかや改竄されていないかどうかを確認できないことについて、警告を表示します。さらに詳しく知りたい場合は、'パッケージ管理' on page 5を参照してください。

新しいバージョンの apt を使用し始めたユーザは、apt が、パッケージインデックス一覧全体をダウンロードするのではなくパッケージ差分ファイル (pdiff) をダウンロードするようになったことに気付くでしょう。この機能についてさらに詳しく知りたい場合は、'APT のパッケージインデックスファイルの更新が遅くなります' on page 38を参照してください。

4.5.8 アップグレード中の注意点

aptitude や apt-get、dpkg を使用した操作が次のようなエラーで失敗に終わるかもしれません。

```
E: Dynamic MMap ran out of room
```

この場合、デフォルトのキャッシュ容量では不十分だということになります。これを解決するには、/etc/apt/sources.list から不要な行を削除もしくはコメントアウトするか、キャッシュサイズを増やします。キャッシュサイズを増やすには、/etc/apt/apt.conf に APT::Cache-Limit を設定します。以下のコマンドを実行すれば、アップグレードするには十分な値が設定されます:

```
# echo 'APT::Cache-Limit "12500000";' >> /etc/apt/apt.conf
```

ここでは、/etc/apt/apt.conf ファイル内にまだこの値を設定していない場合を想定しています。

場合によっては衝突や事前依存のループのために、APT の APT::Force-LoopBreak オプションを有効にして、必須パッケージを一時的に削除しなければならないかもしれません。その場合 aptitude はこのことを警告してアップグレードを中断します。aptitude のコマンドラインに -o APT::Force-LoopBreak=1 を指定すれば、この状態を回避できます。

システムの依存関係の構造が非常に混乱していて、手動での介入が必要となることもあります。通常これは aptitude を用いるか、あるいは

```
# dpkg --remove パッケージ名
```

として、目ざわりなパッケージを消す作業になります。または次の作業でもよいかもしれません。

```
# aptitude -f install
# dpkg --configure --pending
```

極端な場合には、コマンドラインから次のように入力して、再インストールしなければならないかもしれません。

```
# dpkg --install /path/to/パッケージ名.deb
```

「純粋な」sarge システムからのアップグレードでは、ファイルの衝突は起こらないはずですが、非公式なバックポートパッケージをインストールしているなら起こるかもしれません。ファイルの競合が起こると、次のようなエラーになります:

```
(<package-foo-file> から) <package-foo> を展開しています...
dpkg: <package-foo> の処理中にエラーが発生しました (--install):
 '<some-file-name>' を上書きしようとしています。これはパッケージ <package-bar>
 にも含まれています
dpkg-deb: サブプロセス paste がシグナル (Broken pipe) によって強制終了しました
 以下のパッケージの処理中にエラーが発生しました:
 <package-foo>
```

ファイルの衝突を解消するには、エラーメッセージの最後の行に表示されたパッケージを強制的に削除します:

```
# dpkg -r --force-depends パッケージ名
```

問題が修正できたら、先に示したように aptitude コマンドを繰り返し入力すれば、アップグレードを再開できます。

アップグレードの最中に、いくつかのパッケージの設定・再設定に関する質問が表示されます。/etc/init.d と /etc/terminfo ディレクトリに置かれるファイルと /etc/manpath.config に関しては、パッケージメンテナのバージョンに置き換えるようにしてください。システムの整合性を保つためには 'yes' と答えることが必要になります。古いバージョンも .dpkg-old という拡張子で保存されていますので、戻すのはいつでもできます。

どうすればよいかわからなくなったら、そのパッケージやファイルの名前を書き留めておいて、その問題解決は後回しにしましょう。typescript ファイルを検索すれば、アップグレードの最中に画面に表示された情報を見直すこともできます。

4.6 カーネルと関連パッケージのアップグレード

このセクションでは、カーネルのアップグレード方法を説明し、このアップグレードに際して生じる可能性がある問題点を確認します。Debian で提供されている linux-image-* パッケージのどれか一つをインストールしても、カスタマイズしたカーネルをソースからコンパイルしてもかまいません。

このセクションに書かれている多くの情報は、ユーザが Debian のモジュール化されたカーネルのうち一つを initramfs-tools や udev とともに使用しているのを前提にしている、ということに注意してください。initrd を必要としないカスタムカーネルを使用するのを選択している場合や、initrd 生成ユーティリティとして異なるものを使用している場合は、このセクションの情報の一部は適切ではないかもしれません。

また、udev がシステムにインストールされていない場合には、ハードウェアの検出に hotplug を使い続けられることにも注意してください。

4.6.1 カーネルメタパッケージのインストール

sarge から etch への dist-upgrade を実行する際には、新しい linux-image-2.6-* メタパッケージをインストールすることを強くお勧めします。このパッケージは、dist-upgrade の過程で自動的にインストールされるかもしれません。次のように実行すると、このパッケージがインストールされたか確認できます。

```
# dpkg -l "linux-image*" | grep ^ii
```

何も出力されない場合は、新しい linux-image パッケージを手作業でインストールする必要があります。利用可能な linux-image-2.6 メタパッケージの一覧を見るには次のように実行してください。

```
# apt-cache search linux-image-2.6- | grep -v transition
```

どのパッケージを選択すればよいのかわからない場合は、uname -r を実行し、似た名前をもつパッケージを探してください。例えば、コマンドの結果が '2.4.27-3-686' の場合は linux-image-2.6-686 をインストールすることをお勧めします。利用可能なパッケージのうち最良のものを選ぶ手助けとして、次のように apt-cache を用いて各パッケージのパッケージ説明・詳細版を見てもよいでしょう。

```
# apt-cache show linux-image-2.6-686
```

インストールするカーネルイメージが決まったら、aptitude install でインストールします。新しいカーネルがインストールされたら、再起動できる機会に再起動し、新しいバージョンのカーネルを有効にしてください。

もうちょっと冒険したい人には、自分のカスタムカーネルをコンパイルする方法も Debian GNU/Linux は提供しています。kernel-package をインストールして、/usr/share/doc/kernel-package の文書を読んでみてください。

4.6.2 2.6 系カーネルからのアップグレード

現在 sarge で 2.6 系カーネルを使用している場合、システムパッケージを完全にアップグレードした後で、このアップグレードを実行します (‘パッケージのアップグレード’ on page 21を参照してください)。

可能なら、カーネルパッケージのアップグレードをメインの `dist-upgrade` と分けることで、一時的に起動しないシステムにしてしまうことを極力避けられます。この手順の説明については‘カーネルのアップグレード’ on page 25をご覧ください。カーネルパッケージのアップグレードは、‘システムの最小アップグレード’ on page 23で説明した最小アップグレードの手順の後以外では行うべきでないことに注意してください。

自分のカスタムカーネルを使用していて、etch で提供されているカーネルを使いたい場合も、この手順で行えます。カーネルのバージョンが `udev` でサポートされていない場合、最小アップグレードの後でアップグレードするのをお勧めします。`udev` でサポートされている場合は、安全にシステム全体のアップグレードを待っていれば OK です。

4.6.3 デバイスの整列順序の変更

etch は、以前のリリースよりも強固なハードウェア検出機構を特徴としています。しかし、それによってシステム上のデバイス検出順が変わり、それがデバイス名の割り当て順に影響するかもしれません。例えば、2 つの異なるドライバと結び付いた 2 つのネットワークアダプタがある場合、eth0 と eth1 が参照するデバイスは入れ替わるかもしれません。この新しい機構によって、例えば実行中の etch システムでイーサネットアダプタを交換するなどした場合、新しいアダプタにも新しいインタフェース名が割り当てられるようになる、ということに注意してください。

`udev` のルールを使用すると、ネットワークデバイスが並び替えられないようできます。もっと正確に言うと、`/etc/udev/rules.d/z25_persistent-net.rules` で定義できます[5]。その他には、起動時に物理デバイスに特定の名前を割り当てる、`ifrename` ユーティリティを使用できます。詳細は `ifrename(8)` と `iftab(5)` をご覧ください。この 2 つ (`udev` と `ifrename`) は同時に使用できません。

ストレージデバイスについては、`initramfs-tools` を用いて、現在と同じ順序でストレージデバイスドライバモジュールをロードするように設定することで、この順序の変更を防げます。このためには、`lsmod` の出力に目を通し、システム上のストレージモジュールがロードされた順序を特定してください。`lsmod` は、ロードされた順序とは逆の順序でモジュールをリストアップします。つまり、リストの最初のモジュールは最後にロードされていたものです。以上は、カーネルが安定した順番で読み込むデバイス (PCI デバイスなど) にしか、効果がないことに注意してください。

しかし、最初に起動した後にモジュールを削除したりロードしなおしたりすると、この順序にも影響が出ます。また、カーネルには静的にリンクされたドライバが含まれている可能性があり、そのようなドライバの名前は `lsmod` の出力に現れません。`/var/log/kern.log`

[5] このルールは、ネットワークインターフェースに固有名をつけるため、`/etc/udev/rules.d/z45_persistent-net-generator.rules` スクリプトで自動生成しています。このシンボリックリンクを削除して、`udev` による NIC の固有名を無効にできます。

や dmesg の出力に目を通すと、これらのドライバの名前やロード順を解読できるかもしれません。

これらのモジュール名を、起動時にロードされるべき順序で /etc/initramfs-tools/modules に追加してください。モジュール名の一部は sarge と etch では異なるかもしれません。例えば、sym53c8xx_2 は sym53c8xx になりました。

その上で update-initramfs -u -k all を実行し、initramfs イメージを再生成する必要があります。

一旦 etch のカーネルと udev を使用し始めたら、ドライバのロード順に依存しないエイリアスでディスクにアクセスするよう、システムの設定を変更してもよいでしょう。これらのエイリアスは /dev/disk/ 階層にあります。

4.6.4 起動タイミングの問題

initramfs-tools で生成した initrd を使用してシステムを起動する場合、時として、udev によるデバイスファイルの作成が、起動スクリプトの動作に間に合わないことがあります。

通常の症状としては、ルートファイルシステムがマウントできず起動に失敗し、デバッグシェルに落ちます。しかしその後チェックしても、必要なデバイスはすべて /dev により提供されているのです。このケースは、ルートファイルシステムが USB ディスクや RAID にある場合、特に lilo を使用している場合に観察されています。

この問題に対処するには、ブートパラメータに rootdelay=*9* を指定してください。タイムアウトの値 (秒) は、調整に必要な時間を指定してください。

4.7 再起動の前にすべきこと

aptitude dist-upgrade が終了したら、「公式」にはアップグレードは終了したことになります。しかし次に再起動する前に、面倒を見てやらなければならないことがいくつかあります。

4.7.1 devfs からのコンバート

Debian のカーネルは、もう devfs をサポートしません。そのため devfs のユーザは、etch のカーネルで起動する前に手作業でシステムを切り替える必要があります。

/proc/mounts に 'devfs' という文字列がある場合、大抵 devfs を使用しています。devfs スタイルの名前を参照している設定ファイルは、すべて udev スタイルの名前を使うように調整する必要があります。devfs スタイルのデバイス名を参照する可能性があるファイルには、/etc/fstab、/etc/lilo.conf、/boot/grub/menu.lst、/etc/inittab などがあります。

生じる可能性がある問題に関するさらに詳しい情報が、バグ報告 #341152 (http://bugs.debian.org/341152) で入手可能です。

4.7.2 lilo の再実行

(sarge をインストールしたときに場合によってはデフォルトのブートローダとなる) lilo をブートローダとして使用している場合は、アップグレード後に以下のように lilo をもう一度実行しておくことを強くお勧めします。

```
# /sbin/lilo
```

注意しなくてはならないのは、システムのカーネルをアップグレードしていない場合でもこの操作が必要になるということです。それは、パッケージのアップグレードによって lilo の第 2 ステージが変更されているからです。

また、/etc/kernel-img.conf の内容を調べ、*do_bootloader = Yes* と書かれていることを確認してください。この設定のとおり、カーネルをアップグレードした後には必ずブートローダが再実行されます。

lilo を再び実行しているときに何らかの問題が発生した場合は、vmlinuz と initrd へのシンボリックリンクが / 内に存在するか、また /etc/lilo.conf の内容に食い違いがないか、確認してください。

再起動する前に lilo を再実行し忘れたり、手動で再実行する前にシステムが偶発的に再起動してしまった場合、システムは起動できなくなるでしょう。その場合、システム起動時に lilo プロンプト全体は表示されず、最初の *LI* だけが表示されます[6]。この状態からシステムを復旧させる方法に関する情報については、'復旧の準備' on page 14 を参照してください。

4.7.3 mdadm のアップグレード

mdadm は、MD アレイ (RAID) を initial ramdisk から再構築したりシステム初期化シーケンス中に再構築するのに設定ファイルを必要とするようになりました。パッケージのアップグレードを終えた後、**再起動する前に**必ず /usr/share/doc/mdadm/README.upgrading-2.5.3.gz に書かれている説明を読み、それに従ってください。このファイルの最新版は http://svn.debian.org/wsvn/pkg-mdadm/mdadm/trunk/debian/README.upgrading-2.5.3?op=file から入手可能です。問題が生じた場合は参考にしてください。

4.8 次期リリースへの準備

アップグレードの後で、次期リリースに向けてできるいくつかの準備があります。

- update-grub プログラムの位置が /sbin/update-grub から /usr/sbin/update-grub に変更されたため、grub を使用している場合は、/etc/kernel-img.conf を編集して update-grub プログラムの位置を変更してください。

[6] lilo の起動エラーコードに関するさらに詳しい情報は、The Linux Bootdisk HOWTO (http://tldp.org/HOWTO/Bootdisk-HOWTO/a1483.html) (日本語訳 (http://www.linux.or.jp/JF/JFdocs/Bootdisk-HOWTO-12.html)) を参照してください。

- 新しいカーネルイメージのメタパッケージが、古いものの依存関係で引きずられてインストールされた場合、自動的にインストールされたというマークがついています。これは以下のようにして修正してください。

```
# aptitude unmarkauto $(dpkg-query -W 'linux-image-2.6-*' | cut -f1)
```

- 以下を実行して、sarge のカーネルメタパッケージを削除してください。

```
# aptitude purge kernel-image-2.6-<フレーバー>
```

- /etc/network/options にある設定オプションをすべて、/etc/sysctl.conf に移動してください。詳しくは /usr/share/doc/netbase/README.Debian を参照してください。
- '時代遅れ (Obsolete) なパッケージ' on page 33 で説明しているように、時代遅れ (obsolete) のパッケージや、使用していないパッケージを削除してください。そういったファイルが使用する設定ファイルを確認し、設定ファイルを削除するのにパッケージを完全削除 (purge) するかどうかを検討してください。

4.9 廃止予定のパッケージ

Lenny のリリースからは、さらに多くのサーバパッケージが廃止されます。したがって、これらのパッケージを今のうちに新しいバージョンに更新しておくと、システムを Lenny へと更新する際の手間が省けます。

廃止予定のパッケージには以下のものがあります。

- apache 1.x (後継となるパッケージは apache2)
- bind8 (後継となるパッケージは bind9)
- php4 (後継となるパッケージは php5)
- postgresql-7.4 (後継となるパッケージは postgresql-8.1)
- exim 3 (後継となるパッケージは exim4)

4.10 時代遅れ (Obsolete) なパッケージ

数千個の新規パッケージが導入された一方で、etch では sarge にはあった 2,000 個以上の古いパッケージが破棄されたり削除されてもいます。これら時代遅れのパッケージをアップグレードする手段は提供されていません。時代遅れのパッケージを使い続けても構いませんが、Debian プロジェクトは通常 etch がリリースされてから 1 年後に[7]そのようなパッケージ

[7] あるいはその期間中に別のリリースが出ない限り。ある時点では、通常 2 個のみの安定版リリースがサポートされています。

へのセキュリティサポートを打ち切り、その後は他のサポートも提供しないのが常です。もし存在するのなら、利用可能な代替品に置き換えることをお勧めします。

あるパッケージが本ディストリビューションから削除された理由は、数多くあります——上流ではもはや保守されていないため、そのパッケージを保守することに興味を抱く Debian 開発者がもはやいないため、提供していた機能が別のソフトウェア (あるいは新バージョン) に取って代わられたため、バグのために etch にはもはや適さないとみなされたため、などです。最後の場合では、当該パッケージが “不安定版” ディストリビューション内には存在していることがあります。

更新されたシステム内のどのパッケージが “時代遅れ” なのかを検出するのは、パッケージ管理用フロントエンドが当該パッケージにその旨のマークをつけてくれるので簡単です。aptitude を使っているのなら、当該パッケージが「廃止された、またはローカルで作成されたパッケージ」欄に列記されているのに気づくでしょう。dselect も同じようなセクションを提供しますが、表示される一覧はわずかに異なっています。さらに、sarge で手作業でパッケージをインストールするのに aptitude を使っていたのなら、手作業でインストールされたパッケージの記録が取られており、依存元パッケージが削除されればもはや不要となる依存関係のみによって引きずられてインストールされたパッケージに時代遅れのマークをつけることができるでしょう。また aptitude は、deborphan とは異なり、手作業でインストールしたパッケージには時代遅れのマークをつけません (依存関係によって自動でインストールされたものにはマークをつけます)。

時代遅れのパッケージを見つけるのに使える追加ツールとしては、以下のものがあります——deborphan や debfoster、cruft。deborphan を強くお勧めしますが、同プログラムは (デフォルトモードでは) 時代遅れのライブラリ—— “libs” や “oldlibs” セクション内にあり、他のパッケージに使われていないパッケージ——しか報告しません。これらのプログラムが表示したパッケージをやみくもに削除しないでください。特に、誤報しやすい非デフォルトのオプションを積極的に使っている場合はなおさらです。実際に削除する前に、削除を提案されたパッケージを手作業で調査 (その中身やサイズ、説明文など) することを強くお勧めします。

Debian バグ追跡システム (http://bugs.debian.org/)は、パッケージが削除された理由についての情報を提供してくれることがよくあります。あるパッケージ自体についてのアーカイブ化されたバグ報告と、ftp.debian.org 擬似パッケージ (http://bugs.debian.org/cgi-bin/pkgreport.cgi?pkg=ftp.debian.org\&archive=yes)のアーカイブ化されたバグ報告の両方を調査すべきです。

4.10.1 ダミーパッケージ

sarge のいくつかのパッケージは etch では複数のパッケージに分割されていますが、これは大半がシステムの保守性を改善するためです。この場合におけるアップグレードを容易にするために、etch はしばしば “ダミーの” パッケージ—— sarge での古いパッケージと同じ名前で、新規パッケージを導入するための依存関係を備えた空のパッケージ——を提供しています。これらの “ダミー” パッケージはアップグレード後は Obsolete 扱いとされ、安全に削除することができます。

大半の (すべてではない) ダミーパッケージの説明文には、その目的が記されています。しかしながらダミーパッケージの説明文は統一されていないため、自分のシステム内のダミー

パッケージを検出するために deborphan を --guess オプションつきで使うこともできます。いくつかのダミーパッケージは、アップグレード後に削除されることを意図しておらず、代わりに時間とともに変化するプログラムの利用可能な最新バージョンの記録用として使われることに注意してください。

章 5

etch で知っておくべき問題点

5.1 生じうる問題

変化には当然のように副作用がつきもので、どこか他の場所でバグを出してしまうこともあります。ここには現時点で私たちが知っている問題点を記載しています。正誤表・関連パッケージの付属文書・バグ報告や、'もっと読みたい' on page 47で触れられているその他の情報も読んでください。

5.1.1 udev に関連したデバイスでの問題

udev は広範囲にわたってテストされていますが、いくつかのデバイスでは若干の修正が必要になるという問題があるかもしれません。大抵の問題は、デバイスファイルの権限や所有者が変更されるというものです。デフォルトではデバイスファイルが作成されない場合もあります (/dev/video や /dev/radio など)。

udev は、これらの問題に対処する設定の仕組みを提供しています。詳しくは、udev(8) や /etc/udev を見てください。

5.1.2 ネットワーク上の特定の場所に TCP が届かなくなりました

2.6.17 以降の Linux は、RFC 1323 で指定された TCP ウィンドウのスケーリングを積極的に使用しています。サーバによってはこれに対しておかしな挙動を示し、ウィンドウサイズを誤って認識することがあります。さらに詳しく知りたい場合は、バグ報告 #381262 (http://bugs.debian.org/381262) や #395066 (http://bugs.debian.org/395066)、#401435 (http://bugs.debian.org/401435) を参照してください。

これらの問題を回避する方法は、通常 2 つあります。許可される TCP ウィンドウサイズの最大値を小さな値に戻す (好ましい) 方法と、TCP ウィンドウスケーリングオプションを完全に無効にする (非推奨の) 方法です。debian-installer の正誤表ページ (http://www.debian.org/devel/debian-installer/errata)にあるコマンドの例を参照してください。

5.1.3 APT のパッケージインデックスファイルの更新が遅くなります

デフォルトでは、etch バージョンの apt は、(aptitude update を実行するときに) APT パッケージインデックスのファイルをアップデートするのに、(パッケージインデックスのファイル全体ではなく) 差分ファイルをダウンロードする pdiff という新しい方式を使います。この新しい機能は使用するバンド幅を抑え、多くのシステムで速くなります。残念ながら、高速なネットワーク接続 (もしくはミラーまでが非常に近距離) で、頻繁にはアップデートしないシステムでは、アップデート時に遅くなってしまうという正反対の影響もあります。パッケージインデックス全体をダウンロードするよりも、差分ファイルをマージする方が時間がかかってしまうためです。/etc/apt/apt.conf 設定ファイルに Acquire::Pdiffs "false"; を追加すれば、この機能を無効にできます。

この変更は、主に Debian GNU/Linux の *unstable* や *testing* ブランチを使っているユーザに影響します。これらのアーカイブは、変更が生じやすいためです。etch のユーザは、主にセキュリティアーカイブのパッケージ状況をアップデートするときに、この機能に気づくでしょう。

5.1.4 ネットワークの初期化を同期させないと予測不能な挙動の原因となります

ネットワークインターフェイスのドライバをロードするのに udev を使うシステムでは、udev の非同期な性質ゆえ、システム起動時に /etc/init.d/networking が実行されるまでにネットワークドライバがロードされないことがあります。/etc/network/interfaces に (auto に加えて) allow-hotplug を含めると、ネットワークインターフェイスが利用可能になるとすぐに有効になるよう設定できますが、これが、ブートシーケンスがネットワークサービスを開始する前に完了するという保証はありません。ネットワークインターフェイスがない場合に、正しく動かないサービスもあります。

5.1.5 WPA による安全なワイヤレスネットワークを使う場合の問題

sarge では、wpasupplicant パッケージはシステムのサービスとして、/etc/default/wpasupplicant および ユーザが提供する /etc/wpasupplicant.conf により設定されていました。

etch では、/etc/init.d/wpasupplicant はなくなり、Debian パッケージでは wireless-tools など他のパッケージと同様に /etc/network/interfaces に統一されました。これはつまり、wpasupplicant は直接システムサービスを提供しないということを意味しています。

wpasupplicant の設定についての情報は、/etc/network/interfaces ファイルの例が書いてある /usr/share/doc/wpasupplicant/README.modes.gz を参照してください。Debian で wpasupplicant パッケージを使用する際の最新情報は、Debian Wiki (http://wiki.debian.org/WPA) にあります。

5.1.6 非 ASCII 文字を含むファイル名での問題

ファイル名に非 ASCII 文字を含むファイルがある vfat、ntfs、iso9660 ファイルシステム

を、utf8 オプションをつけずにマウントした場合、ファイル名を使おうとすると失敗します。次のようなエラーが表示されます: 'Invalid or incomplete multibyte or wide character'。可能な解決方法は、ファイル名に非 ASCII 文字を含むファイルがある vfat、ntfs、iso9660 ファイルシステムをマウントする際には、`defaults,utf8` をマウントオプションとしてつけることです。

`utf8` オプションをつけると、vfat で大文字小文字を区別せずにファイル名を扱うという機能を、Linux カーネルがサポートしていないことに注意してください。

5.1.7 Nvidia チップセットでのハードウェア IOMMU によるデータ破壊

Nvidia のチップセットを使っていて、かつ 3GB を越す RAM を積んでいる AMD64 システムでは、ハードウェア IOMMU が使われるとデータ破壊が時々起こるという問題が確認されています。この問題はまだ Linux カーネルの開発者とハードウェアメーカーが調査中で、上流でも公式の修正版はリリースされていません。データの完全性を保護するために、これらのシステムのユーザには、適切な解決方法が見つかるまでは、`iommu=soft` をカーネルの起動オプションに手動で追加して、起動時からハードウェア IOMMU の使用を無効にするよう忠告します。

この問題についてのさらなる情報は、Debian のバグ報告 #404148 (http://bugs.debian.org/404148) と Linux カーネルのバグ報告 #7768 (http://bugzilla.kernel.org/show_bug.cgi?id=7768) にあります。

5.1.8 サウンドが機能しなくなった

まれに、アップグレード後にサウンドが機能しなくなる可能性があります。この問題が生じた場合は、alsa のチェックリストを一通り実行してください。つまり、root ユーザとして `alsaconf` を実行する、あなたのユーザを `audio` グループに追加する、alsamixer を用いて音量レベルを上げてミュートでない状態にする、arts あるいは esound を停止する、OSS モジュールがロードされていないのを確認する、スピーカーがオンになっているのを確認する、`cat /dev/urandom > /dev/dsp` というコマンドが root できちんと機能するかを確認する、という手順を踏んでください。

5.2 XFree86 から X.Org への移行

X.Org への移行は多少の構造的変化を伴います。インストール済みのパッケージがすべて Debian 製のパッケージで etch にも含まれている場合、アップグレードは問題なく行われるはずです。しかし、知っておくべき変更点がいくつかあることが経験的に明らかになっています。というのも、これらによってアップグレードの過程で問題が生じる可能性があるからです。

最も重要な変更点は、`/usr/X11R6/bin` が削除され、`/usr/bin` へのシンボリックリンクとして残されるだけになるということです。つまり、新しいパッケージがインストールされる時点で `/usr/X11R6/bin` が空になっていなければいけません。新しいパッケージは、

/usr/X11R6/bin を使用していたほとんどのパッケージと衝突するようになっていますが、場合によっては手作業での介入が必要になります。絶対に、X セッション内からはディストリビューションのアップグレードを実行しないようにしてください。

X.Org のインストール中にアップグレードが中断した場合、/usr/X11R6/bin にまだ残っているファイルがあるか調べなければなりません。そうしたら (そこにファイルがあれば) どの Debian パッケージがそのファイルをインストールしたかを dpkg -S で調べ、該当するパッケージを dpkg --remove で削除できます。後で代わりのパッケージをインストールできるよう、削除したパッケージを書き留めておいてください。アップグレードを続ける前に /usr/X11R6/bin にあるファイルはすべて削除する必要があります。

さらに詳しい情報やその他の問題点については http://wiki.debian.org/Xorg69To7 を読んでください。

再起動後に X.Org に問題が見られる場合は、/etc/init.d/xfs restart を実行してフォントサーバを再起動する必要もあるでしょう。この問題は、/etc/X11/fs/xfs.options が no-restart-on-upgrade という行を含んでいるのにフォントパスが変更されたために生じます。

5.3 多くのアプリケーションが 8 ビットでの表示をサポートしていません

X.Org および最新のライブラリにアップグレードした後、8 ビットのカラー深度しか表示できない X ターミナルは動かなくなります。これは、Cairo 2D ベクトルグラフィックライブラリ (libcairo2) が 8 ビット疑似カラーをサポートしていないからです。このライブラリは、GNOME や Xfce デスクトップをはじめ、abiword のような Gtk2+ ツールキットを用いてコンパイルされた多くのデスクトップアプリケーションで使われています。

いくつかの Sun のマシンと、Tektronix・NCD・IBM・SGI の X ターミナルの他、X ウィンドウをリモートに転送するいくつかのシステムが、この影響を受けることが分かっています。可能なら、これらのターミナルで 16 ビットカラーを使うよう設定すべきです。

さらなる情報は、Freedesktop のバグ報告 bug #4945 (https://bugs.freedesktop.org/show_bug.cgi?id=4945) にあります。

5.4 exim から exim4 へのアップグレード

etch リリースで時代遅れとして扱われるようになったパッケージの 1 つに、メール転送エージェント (MTA) の exim があります。このパッケージは、完全に新しいパッケージである exim4 によって置き換えられました。

それだけでなく、exim (バージョン 3.xx) はもう何年もの間、上流でメンテナンスされていないので、Debian ではこのバージョンのサポートを打ち切りました。まだ exim 3.xx を使用している場合は、インストールしている exim を exim4 に手作業でアップグレードしてください。exim4 は既に sarge に含まれているので、etch へのアップグレードの前に sarge システム上でこのアップグレード作業を行うか、etch へのアップグレードの後で都合のよいとき

に行うかは選択可能です。ただ、古い exim パッケージが自動的にアップグレードすることはなく、sarge のサポートが打ち切られた後はそのパッケージへのセキュリティサポートは行われないということだけは覚えておいてください。

debconf の設定によっては、exim4 のインストール中に、設定に関する質問を全くされない可能性があることに注意してください。質問をされない場合、システムはデフォルトの「ローカル配信のみ」のセットアップになります。dpkg-reconfigure exim4-config コマンドを使用すると設定しなおすことが可能です。

Debian の exim4 パッケージ群は大規模に文書化されています。パッケージのホームページは Debian Wiki の http://wiki.debian.org/PkgExim4 で、README ファイルはパッケージ内部だけでなく http://pkg-exim4.alioth.debian.org/README/README.Debian.html にもあります。

README ファイルにはパッケージ化 (Packaging) に関する章があります。この章では、Debian で提供されている複数の異なるパッケージの差異について説明しています。また README ファイルには Exim 3 からのアップグレード (Updating from Exim 3) に関する章があります。この章は、実際に移行を行う際に役に立つでしょう。

5.5 apache2 のアップグレード

Apache は新しいバージョン 2.2 にアップグレードしました。普通のユーザはこのアップグレードによる影響を受けませんが、知っておくべき、生じる可能性がある問題点がいくつかあります。

http://httpd.apache.org/docs/2.2/upgrading.html に、上流での変更が記載されています (訳注: 本リリースノート執筆時点では、リンク先の日本語のページに記載されている情報は 1.3 から 2.0 で入った変更なので、2.0 から 2.2 で入った変更については英語のページを参照してください)。このページを読み、特に以下の内容を記憶に留めておいてください。

- すべてのモジュールを再コンパイルする必要があります
- 認証モジュールについて、名前の変更や再分類が行われました
- 設定オプションの一部は名前が変わりました

Debian 特有の変更としては、デフォルトパッケージが ssl をサポートするようになったため、SSL という文字列にもう意味がなくなった、などが挙げられます。

(apache2-mpm-itk パッケージで利用可能な) 実験的な ITK MPM を使用している場合、cgi モジュールはデフォルトでは適切に有効になりません。適切に有効にするには、次のように手作業で mod_cgid を無効にして mod_cgi を有効にする必要があります。

```
# cd /etc/apache2/mods-enabled
# rm cgid.conf cgid.load
# ln -s ../mods-available/cgi.load .
# /etc/init.d/apache2 force-reload
```

5.6 Zope と Plone のアップグレード

Zope 関連のすべてのソフトウェアが更新されています。(時代遅れになった、もしくは、新しい Zope、CMF、Plone とは適合しなくなった) 多くのソフトウェアが、このディストリビューションからは削除されています。

残念ながら、複雑な zope や plone のサーバをアップグレードする、簡単で確実な方法はありません。Plone には移行ツールがありますが、経験として、自動移行は簡単に失敗してしまいます。

このため、etch バージョンへの移行テストと並行して、sarge でインストールした Zope/Plone を実行しつづけられるようにシステムを設定するようお勧めします。

これを達成する簡単で安全な方法は、sarge システムを別のハードディスクやパーティションにコピーして、2 つのうちの 1 つだけをアップグレードすることです。chroot を使えば、etch バージョンと並行して sarge バージョンを実行できます。

etch システムで Zope/Plone の新旧両バージョンを同時にインストールするのは不可能です。一つには、古いパッケージは python2.3 に依存していて、python2.4 と同時にはインストールできないからです。

5.7 GNU tar のワイルドカード展開 (glob)

以前のバージョンの GNU tar は、アーカイブからファイルを展開したり、ファイルのリストを作ったりするときに、シェル形式の glob をしていたようです。例として、

```
tar xf foo.tar '*.c'
```

は、'.c' で終わるファイルをすべて展開していました。この挙動は文書化されておらず、従来の tar の実装とは互換性がありません。そこで、バージョン 1.15.91 からの GNU tar はデフォルトでは glob しなくなりました。例えば、上のコマンドは、アーカイブから '*.c' という名前のファイルを展開せよ、と解釈されます。

さらなる情報は、/usr/share/doc/tar/NEWS.gz を見てください。

5.8 NIS と Network Manager

etch の nis に入っている ypbind のバージョンでは、Network Manager のサポートが含まれています。このサポートは、コンピュータがネットワークから切断されたと Network Manager が報告するときに、ypbind が NIS クライアントの機能を無効にする原因となります。Network Manager は、通常、コンピュータが使われていない場合に切断されたと報告するので、NIS クライアントシステムを使っている NIS ユーザは、これらのシステムに載っている Network Manager のサポートが無効になっているのを確認すべきです。

これは、network-manager パッケージをアンインストールするか、/etc/default/nis を編集して、YPBINDARGS に -no-dbus を追加すれば済みます。

新規にインストールした Debian ではデフォルトで -no-dbus が使われますが、これまでのリリースではデフォルトではありませんでした。

5.9 安全でない php の設定の非推奨化

長年、PHP で register_globals の設定をオンにすることは安全でなく危険であるとわかっており、このオプションはこれまでかなりの期間、デフォルトではオフになっていました。この設定が、あまりにも危険であるとして、ついに Debian システムでは非推奨となりました。同様のことが safe_mode と open_basedir の欠陥にも当てはまります。これらの欠陥はやはり、かなりの期間メンテナンスされていません。

本リリース以降、Debian セキュリティチームは、安全でないとわかっている PHP の多数の設定についてはセキュリティサポートを提供しません。最も重大なのは、register_globals がオンになっているために生じる問題への対処は、もはやなされないということです。

register_globals を必要とする旧式のアプリケーションを実行する場合は、例えば Apache の設定ファイルを用いて、該当する各パスのみに対してこの設定を有効にしてください。さらに詳しい情報は、PHP の付属文書のディレクトリ (/usr/share/doc/php4 や /usr/share/doc/php5) に含まれている README.Debian.security ファイルで入手可能です。

5.10 Mozilla 製品のセキュリティの状態

Mozilla のプログラムである firefox と thunderbird (Debian ではそれぞれ iceweasel と icedove に名前が変更されています) は多数のユーザにとって重要なツールです。しかし残念なことに、上流のセキュリティポリシーは上流の新しいバージョンに更新するようユーザに強いることで、これは、セキュリティアップデートには大きな機能変更を含めないという Debian のポリシーと矛盾します。いつになるかは今のところわかりませんが、etch のサポート期間の間に、Mozilla 製品のサポートがもはや Debian セキュリティチームにとって実現不可能になり、Debian セキュリティチームが Mozilla 製品のセキュリティサポートの終了を発表するときが来るかもしれません。Mozilla 製品をインストールするときはこのことを考慮に入れ、セキュリティサポートの終了が問題になると考えられる場合は Debian で提供されている代替プログラムの使用を検討してください。

5.11 KDE デスクトップ

etch に含まれているバージョンの KDE では、メディアの取り扱い方法が、device: / を用いたアドレスから media: / を用いたアドレスへと変化しました。一部のユーザ設定ファイルには device: / を用いたリンクが含まれている可能性があるので、それらは新

しいアドレスに合うよう修正すべきです。特に、`~/.kde/share/apps/konqsidebartng/virtual_folders/services` には `device: /` を用いた参照があります。このファイルは、新規ユーザをセットアップしたときには作成されないので、安全に削除できます。

KDE デスクトップ環境は、sarge に含まれていたバージョンから etch に含まれているバージョンまでに多くの変更が加えられました。さらに詳しい情報は KDE 3.5 のリリースノート (http://www.kde.org/announcements/announce-3.5.php)にあります。

5.12 GNOME デスクトップに関する変更とサポート

sarge で GNOME デスクトップを使用していた場合、etch になって Debian でのデフォルトの設定に導入された変更のうち一部は役立たないでしょう。極端な場合、GNOME デスクトップは過去の設定内容を適切に扱えず、正しい振舞いをしない可能性があります。

GNOME デスクトップの設定を大幅に変更していないのであれば、ユーザのホームディレクトリ内の `.gconf` ディレクトリを別の名前 (`.gconf.old` など) に変更するとよいでしょう。そうすれば、新しいセッションを開始したときに `.gconf` ディレクトリは作成しなおされ、etch のデフォルトの設定を含むようになります。

etch のリリースから、Debian には、もはやサポートされていない GNOME バージョン 1 リリースのパッケージの大半が含まれなくなりました。とはいえ、GNOME 2 を使うよう更新されていないいくつかの Debian パッケージをサポートするため、一部のパッケージは残されています。GTK1.2 のパッケージはまだ完全に保守され続けています。

GNOME デスクトップ環境は、sarge に含まれていたバージョンから etch に含まれているバージョンまでに多くの変更が加えられました。さらに詳しい情報は GNOME 2.14 のリリースノート (http://www.gnome.org/start/2.14/notes/en/)にあります。

5.13 デフォルトのエディタ

`vim` をデフォルトのエディタとして使用していた場合、アップグレードの過程でデフォルトのエディタが `nano` に変更される可能性があります。

管理者は、すべてのユーザ向けのデフォルトのエディタを変更したい場合、以下のようにして alternatives システムを更新してください。

```
# update-alternatives --config editor
```

ユーザは、デフォルトのエディタを変更したい場合、自分のプロファイルに以下のような行を加えて環境変数 *EDITOR* を設定してください。

```
EDITOR=vi
export EDITOR
alias editor=$EDITOR
```

5.14 message of the day

/etc/motd は、システムの再起動のたびに /etc/init.d/bootmisc.sh によってテンプレート /etc/motd.tail から生成しなおされる /var/run/motd へのシンボリックリンクになりました。したがって、/etc/motd に加えられた変更は失われます。/etc/motd.tail に加えられた変更は、システムの再起動時以外には、自動的に /etc/motd に適用されることはありません。

また、/etc/default/rcS の変数 EDITMOTD にはもう何の効果もありません。motd の更新を無効にしたい場合や、message of the day の内容を独自に管理したい場合は、単に、/etc/motd のシンボリックリンクが /etc/motd.static などの異なるファイルを指すようにし、加えたい変更をそちらのファイルに加えてください。

5.15 emacs21* 上での Unicode サポートは非デフォルト

emacs21 と emacs21-nox は、デフォルトでは Unicode を使用するよう設定されていません。さらに詳しい情報や回避方法については、Bug #419490 (http://bugs.debian.org/419490) を参照してください。

章 6

Debian GNU/Linux に関するさらなる情報

6.1 もっと読みたい

このリリースノートやインストールガイドを越えた、Debian GNU/Linux に関するより進んだ文書は、Debian Documentation Project (DDP) から公開されています。DDP は Debian のユーザや開発者向けに、品質の高い文書を作成することを目的としています。Debian リファレンス、Debian メンテナ入門、Debian FAQ などなど、たくさんの文書があります。現在利用可能なリソースの詳細すべては DDP のウェブサイト (http://www.debian.org/doc/ddp)から得られます。

それぞれのパッケージの文書は /usr/share/doc/パッケージに インストールされています。ここには、著作権情報、Debian 固有の詳細、開発元の文書すべて、などが置かれています。

6.2 助けを求めるには

Debian ユーザ向けのヘルプ・アドバイス・サポートなどは、いろいろな場所から得られます。しかしこれらを頼りにするのは、その問題について徹底的に文書を調査してからにしましょう。このセクションでは新しく Debian ユーザになった人向けに、これらを簡単に紹介します。

6.2.1 メーリングリスト

Debian ユーザが最も興味を引かれるであろうメーリングリストは debian-user (英語) リストおよび debian-user-言語 (各国語) リストでしょう。これらのリストの詳細や講読のしかたについては、http://lists.debian.org/ を見てください。利用にあたっては、あなたの疑問に対する答えが以前の投稿ですでに答えられていないかどうか、アーカイブをチェックしてください。また標準的なメーリングリストのエチケットに従うようにしてください。

6.2.2 インターネットリレーチャット (IRC)

Debian には、Debian ユーザのサポートや援助のために専用の IRC チャンネルが OFTC IRC ネットワークにあります。このチャンネルにアクセスするには、お好みの IRC クライアントを irc.debian.org に接続し、#debian に join してください。

チャンネルのガイドラインに従い、他のユーザをきちんと尊重してください。ガイドラインは Debian Wiki (http://wiki.debian.org/DebianIRC) で閲覧できます。

OFTC についてさらに詳しく知りたい場合は、ウェブサイト (http://www.oftc.net/)を訪ねてみてください。

6.3 バグを報告する

私たちは Debian GNU/Linux を高品質な OS にするよう努めていますが、だからといって私たちの提供するパッケージにバグが皆無というわけではありません。Debian の「オープンな開発体制」という考え方に合致し、また、ユーザに対するサービスとして、私たちは報告されたバグに関するすべての情報を bugs.debian.org (http://bugs.debian.org/) にあるバグ追跡システム (Bug Tracking System: BTS) で提供しています。

もしディストリビューションや、その一部であるパッケージされたソフトウェアにバグを見つけたら、将来のリリースで修正できるよう、その問題点の報告をお願いします。バグを報告するには正しい電子メールアドレスが必要です。これをお願いしているのは、バグを追跡できるようにするため、そして追加情報が必要になった場合に開発者が報告者に連絡できるようにするためです。

バグ報告は、reportbug プログラムを使って送信することもできますし、電子メールを使って手で送ることもできます。バグ追跡システムに関する詳細やその使い方については、リファレンスカード (doc-debian パッケージをインストールしていれば /usr/share/doc/debian にあります) をお読み頂くか、またはバグ追跡システム (http://bugs.debian.org/)のウェブサイトからオンラインで入手することもできます。

6.4 Debian に貢献する

Debian への貢献は専門家でなくてもできます。問題を抱えたユーザを、いろいろなサポートメーリングリスト (http://lists.debian.org/)で助けてあげることも、立派なコミュニティへの貢献です。開発メーリングリスト (http://lists.debian.org/)に参加して、ディストリビューション開発に関する問題を見つける (そして解決する) ことも、もちろん非常に助けになります。Debian を高品質なディストリビューションに保つため、バグを報告して (http://bugs.debian.org/)その原因の特定や解決に際して開発者を助けてください。執筆が得意なら、文書 (http://www.debian.org/doc/ddp)作成や既存文書のご自分の言語への翻訳 (http://www.debian.org/international/)に積極的に参加し、そこで貢献するのもよいでしょう。

もっと時間が自由になるなら、Debian に属するフリーソフトウェア集の一部を管理してみるのはどうでしょうか。皆が Debian に入れてほしいと思っているソフトウェアを引き受

けて管理するのは、特に価値の高い貢献です。これに関する詳細は、作業が望まれるパッケージ (http://www.debian.org/devel/wnpp/)をご覧になってください。Debian にはいくつかサブプロジェクトが存在しており、特定のアーキテクチャへの移植、Debian Jr. (http://www.debian.org/devel/debian-jr/)、Debian Med (http://www.debian.org/devel/debian-med/) などが進められています。これらのうち、あなたが興味を持っているグループに参加するのもよいでしょう。

いずれにしても、あなたが何らかの形でフリーソフトウェアコミュニティに関わっているのなら、それがユーザとしてであれ、プログラマ、ライター、翻訳者のいずれとしてであれ、すでにあなたはフリーソフトウェア運動を助けてくださっているのです。貢献することは報いのあることですし、楽しいことです。新しい人々に出会う機会も増えます。きっと暖かな、楽しい気持ちになれるはずです。

付 録 A

sarge システムの管理

この付録には、etch へアップグレードする前に sarge パッケージを確実にインストールしたりアップグレードする方法についての情報が述べられています。特定の状況でのみ必要となるでしょう。

A.1 sarge システムのアップグレード

基本的には、これまで行ってきた sarge のあらゆるアップグレードと違いはありません。唯一異なるのは、‘ソースリストのチェック’ on page 51で説明するようにパッケージリスト内に sarge パッケージがまだ含まれているのを確認する必要があることです。

Debian ミラーを使用してシステムをアップグレードする場合、システムは自動的に最新の sarge ポイントリリースへとアップグレードされます。

A.2 ソースリストのチェック

`/etc/apt/sources.list` 内で ’stable’ を指定している行があるなら、効率よく etch を “使う” 用意ができています。すでに `apt-get update` を実行済みでも、以下の手順に従えば問題なく元に戻すことができます。

etch からパッケージのインストールもしてしまっているなら、おそらくこれ以上 sarge からパッケージをインストールしても無意味でしょう。この場合、続けるかどうかを自分で決断しなければなりません。パッケージをダウングレードすることはできますが、その方法はここでは扱いません。

(root になってから) お気に入りのエディタで `/etc/apt/sources.list` を開き、`deb http:` や `deb ftp:` で始まるすべて行の中に “`stable`” が指定されているかどうかを調べてください。もしあるなら、`stable` を `sarge` に変更してください。

`deb file:` で始まっている行があるなら、その行が指定している場所が sarge か etch のどちらのアーカイブなのかを独力で調べなければならないでしょう。

重要!: `deb cdrom:` で始まっている行は、絶対に変更しないでください。変更するとその行は無効になって、もう一度 `apt-cdrom` を実行しなければならなくなるでしょう。'cdrom' ソースが "`unstable`" を指定していても心配しないでください。混乱するかもしれませんが、これで正常なのです。

変更が済んだら、ファイルを保存してから

```
# apt-get update
```

と実行して、パッケージリストを更新してください。