Release-Notes for Debian 13 (trixie)

Debian Documentation Team

2025-06-27

目 次

Debian ドキュメンテーションプロジェクト <https://www.debian.org/doc>.

最終更新日: 2025-06-27

第1章　はじめに

この文書は Debian ディストリビューションのユーザーに、バージョン 13 (コードネーム trixie) での大きな変更点を知らせるものです。

このリリースノートでは、リリース 12 (コードネーム bookworm) から今回のリリースへの安全なアップグレード方法や、その際ユーザーが遭遇する可能性がある既知の問題点についての情報をユーザーに提供しています。

この文書の最新版は、https://www.debian.org/releases/trixie/releasenotes から取得できます。

> **注意:** 既知の問題点をすべて列挙するのは不可能なので、問題点の予想される広がり具合と影響の大きさの双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。

Debian の 1 つ前のリリースからのアップグレード (この場合、bookworm 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 bookworm 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

私たちは、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、ユーザーが直面する可能性のある、すべての問題を想定しました。

それにも関わらず、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には、**release-notes** パッケージに対するバグ報告として、バグ追跡システム に提出してください。あなたが発見した問題が既に報告されている場合に備え、まずは 既存のバグ報告 を確認してみると良いでしょう。もしこの文書にさらに内容を付加できるのであれば、どうぞ遠慮なく既存のバグ報告へ情報を追加して下さい。

私たちは、この文書のソースへのパッチを含めた報告を歓迎・推奨します。このドキュメントのソースの取得方法の記述については **この文書のソース** で、より詳細な情報を見つけることができるでしょう。

1.2 アップグレードについての報告をする

bookworm から trixie へのアップグレードに関連するユーザー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として、アップグレードの結果を含めて バグ追跡システム 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/apt/extended_states` にある **apt** 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** 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が、`/var/lib/dpkg/status` のバックアップは `/var/backups` にもあります。
- `script` を使用して作成したセッションのログ。**セッションの記録** で説明します。
- `/var/log/apt/term.log` にある `apt` のログか、`/var/log/aptitude` にある `aptitude` のログ。

注釈: バグ報告に情報を含める前に、慎重に扱うべき情報や機密情報がログに含まれていないかある程度時間をかけて検査し、ログから削除してください。なぜなら、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

この文書のソースは reStructuredText 形式で、sphinx 変換ツールを使っています。HTML 版は、*sphinx-build -b html* を使用して生成しています。PDF 版は、*sphinx-build -b latex* を使用して生成しています。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 Git リポジトリにあります。ウェブから ウェブインターフェース 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。Git 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 VCS 情報ページ を参照してください。

第2章　Debian 13 の最新情報

この章のより詳しい情報は Wiki を参照してください。

2.1 サポートするアーキテクチャ

Debian trixie 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。

- 64-bit PC (`amd64`)
- 64 ビット ARM (`arm64`)
- ARM EABI (`armel`)
- ARMv7 (EABI 浮動小数点ハードウェア ABI, `armhf`)
- 64 ビットリトルエンディアン PowerPC (`ppc64el`)
- 64 ビットリトルエンディアン RISC-V (`riscv64`)
- IBM System z (`s390x`)

Additionally, on 64-bit PC systems, a partial 32-bit userland (`i386`) is available. Please see *i386* **サポートの縮小** for details.

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 の移植版に関するウェブページ で読むことができます。

2.2 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 bookworm に含まれていたよりさらに多くのソフトウェアが含まれています。このディストリビューションには、11294 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 59551 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 42821 ものソフトウェアパッケージ (これは bookworm のパッケージ全体の 72% 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 (bookworm のパッケージの 16% にあたる 9519 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は更新されることはなく、パッケージ管理用のフロントエンドでは 'obsolete' というマークが付けられます。これについては **利用されなくなったパッケージ** を参照してください。

2.2.1 Official support for riscv64

This release for the first time officially supports the riscv64 architecture, allowing users to run Debian on 64-bit RISC-V hardware and benefit from all Debian 13 features.

The Wiki provides more details about riscv64 support in Debian.

2.2.2 PAC/BTI support on arm64

trixie introduces two security features on the arm64 architecture known as Pointer Authentication (PAC) and Branch Target Identification (BTI). They are designed to mitigate Return-Oriented Programming exploits and Jump-Oriented Programming attacks respectively.

The features are enabled automatically if your hardware supports them. The Wiki has information on how to check if your processor supports PAC/BTI and how they work.

2.2.3 デスクトップとよく知られているパッケージ

Debian は今回も複数のデスクトップアプリケーションとデスクトップ環境をサポートしています。中でも GNOME 48, KDE Plasma 6.3, LXDE 13, LXQt 2.1.0, Xfce 4.20 があります。

事務用アプリケーションもオフィススイートを含めてアップグレードされています:

- LibreOffice が 25 へアップグレードされました。
- GNUcash が 5.10 へアップグレードされました。

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:

パッケージ	12 (bookworm) でのバージョン	13 (trixie) でのバージョン
Apache	2.4.62	2.4.63
Bash	5.2.15	5.2.37
BIND DNS サーバ	9.18	9.20
Cryptsetup	2.6	2.7
Emacs	28.2	30.1
Exim 標準の電子メールサーバ	4.96	4.98
GNU Compiler Collection (デフォルトのコンパイラ)	12.2	14.2
GIMP	2.10.34	3.0.2
GnuPG	2.2.40	2.4.7
Inkscape	1.2.2	1.4
GNU C ライブラリ	2.36	2.41
Linux カーネルイメージ	6.1 シリーズ	6.12 シリーズ
LLVM/Clang ツールチェイン	13.0.1, 14.0 (デフォルト) そして 15.0.6	19 (デフォルト), 17 そして 18 が利用可能
MariaDB	10.11	11.8
Nginx	1.22	1.26
OpenJDK	17	21
OpenLDAP	2.5.13	2.6.9
OpenSSH	9.2p1	10.0p1
OpenSSL	3.0	3.4
Perl	5.36	5.40
PHP	8.2	8.4
Postfix MTA	3.7	3.10
PostgreSQL	15	17
Python 3	3.11	3.13
Rustc	1.63	1.85
Samba	4.17	4.22
Systemd	252	257
Vim	9.0	9.1

2.2.4 HTTP Boot Support

The Debian Installer and Debian Live Images can now be booted using "HTTP Boot" on supported UEFI and U-Boot firmware.

On systems using TianoCore firmware, enter the *Device Manager* menu, then choose *Network Device List*, select the network interface, *HTTP Boot Configuration*, and specify the full URL to the Debian ISO to boot.

For other firmware implementations, please see the documentation for your system's hardware and/or the firmware documentation.

第3章　インストール用システム

Debian Installer は公式の Debian インストールシステムです。このインストーラーは、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。

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

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

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

これまでに知られている問題点を列挙した debian-installer の正誤表も https://www.debian.org/releases/trixie/debian-installer#errata で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変更点

Debian Installer は前回の Debian 12 で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善され、ワクワクするような新機能や改善がいくつか追加されました。

bookworm 以降になされた変更の概要に興味がある場合は、Debian Installer の ニュースの履歴 で閲覧可能な、trixie 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。

3.2 Debian Pure Blends のインストール

Debian Junior、Debian Science、Debian FreedomBox といった Debian Pure Blends の選択を直接 Debian インストーラーで行えるようになりました。詳しくは インストールガイド を参照してください。

Debian Pure Blends に関する詳細は、https://www.debian.org/blends/ や wiki を参照してください。

3.3 クラウドへのインストール

クラウドチーム はいくつかの人気のあるクラウドコンピューティングサービス向けの Debian trixie イメージを公開しています。例えば:

- アマゾン ウェブ サービス (AWS)
- Microsoft Azure
- OpenStack
- 仮想マシン (Plain VM)

クラウド用イメージでは `cloud-init` 経由の自動フックを提供し、特別に最適化されたカーネルパッケージと grub 設定を使ってインスタンスを高速に起動することを優先します。様々なアーキテクチャをサポートするイメージを必要に応じて提供し、クラウドサービスが提供するすべての機能をサポートするようクラウドチームは努力しています。

クラウドチームは trixie の LTS 期間の終わりまで更新されたイメージを提供します。新しいイメージは、ポイントリリースやとても重要なパッケージのセキュリティ修正後にリリースされます。クラウドチームのサポートポリシーの全文は こちら で確認できます。

詳細は https://cloud.debian.org/ や wiki を参照してください。

3.4 コンテナおよび仮想マシンイメージ

さまざまなアーキテクチャの Debian trixie のコンテナイメージを Docker Hub で利用できます。標準イメージに加えて、ディスク使用量を削減した "slim" バリアントも利用できます。

Hashicorp の Vagrant VM マネージャー向けの仮想マシンイメージは Vagrant Cloud で公開しています。

第4章　Debian 12 (bookworm) からのアップグレード

4.1 アップグレードの準備

アップグレードの前には、*trixie* **で注意すべき点** 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。

バックアップしておくべき主な対象として、/etc、/var/lib/dpkg、/var/lib/apt/extended_states の中身、そして dpkg コマンドの出力などがあります:

```
$ dpkg --get-selections '*' # (the quotes are important)
```

システムの管理に aptitude を使っている場合は、/var/lib/aptitude/pkgstates もバックアップしておくと良いでしょう。

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・KDE といったデスクトップ環境のように) 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、root としてログインするか su や sudo を使って、必要なアクセス権限を得てください。

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。ただ、システムに ssh 接続などでアクセスしてきているユーザが、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、また、作業を続行できるはずです。

万一の対策をしたければ、アップグレードの前に /home パーティションをバックアップするか、アンマウントしておきましょう。

trixie 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、これはアップグレード完了後に実施します。

4.1.3 サービスのダウン期間の準備

システムが提供しているサービスで、アップグレードに含まれるパッケージが関連するサービスがあるかもしれません。この場合、注意して欲しいのですが、アップグレード作業中に関連パッケージが置換・設定される際、これらのサービスが停止します。この間、サービスは利用できなくなります。

これらのサービスに対する実際のダウン期間は、システム中でアップグレードされるパッケージ数に応じて違いますし、このダウン期間には (もしあれば) システム管理者が各パッケージのアップグレードに対する設定の質問への回答に費やす時間も含まれます。アップグレード作業が放置されたままでいて、システムがアップグレード中に入力を必要とした場合、非常に長期間サービスが利用ができなくなる可能性が非常に高いでしょう[1]

アップグレードを行うシステムが、ユーザーやネットワークにとって最も重要なサービスを提供している場合[2]、**システムの最小アップグレード** で記述しているように最小限のシステムアップグレードを行い、次にカーネルのアップグレードと再起動をし、そしてもっとも重要なサービスに関連するパッケージをアップグレードします。これらのパッケージのアップグレードは、**システムのアップグレード** にある完全アップグレードより先に実施します。このようにすれば、これらの最重要サービスが動作しつづけ、そして完全アップグレード作業を行っても利用可能であることを保証し、サービスの停止時間を減らすことができます。

4.1.4 復旧の準備

Debian はシステムがブートできる状態を常に確保するように努めていますが、アップグレード後のシステム再起動で問題に遭遇する可能性は常にあります。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から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、必要な事前準備をしておくことをお勧めします。カーネルをアップグレードして再起動した後、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

[1] debconf の優先度を、とても高いレベルに設定していると設定プロンプトを抑制できますが、デフォルト値があなたのシステムに合わない場合、サービスはそのままでは起動に失敗することでしょう。

[2] 例: DNS や DHCP サービス、特に冗長性やフェイルオーバー機能が無い場合。DHCP の例では、リースタイムがアップグレード作業が完了する時間よりも短い場合、エンドユーザはネットワークから切り離されるでしょう。

グレード中に誤ってシステムが再起動された場合にも、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があります。

緊急時のリカバリ作業について、通常お勧めしているのは trixie 用 Debian インストーラーの **レスキューモード** の利用です。インストーラーを使う利点は、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にあります。より詳しい情報は、インストールガイド (https://www.debian.org/releases/trixie/installmanual にあります) の第 8 章にある"壊れたシステムの復旧"セクションや、Debian インストーラー FAQ を参照してください。

これが失敗するなら、システムを起動してアクセス・修復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 live インストール イメージ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、chroot でその中に入って問題点を調査・解決できるはずです。

initrd を使った起動中のデバッグシェル

initramfs-tools パッケージは生成した initrd にデバッグシェルを収録します[3]。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。このデバッグシェルは、問題の追跡、そしておそらくは修正の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、どのモジュールがロードされているか (cat /proc/modules)、dmesg の出力にドライバのロード失敗のエラーが出ていないか、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期待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。

問題点を何とか解決できたなら、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。

systemd を使った起動中のデバッグシェル

起動が systemd において失敗する場合、カーネルコマンドラインを変更することでデバッグ用の root シェルを追加できます。基本的な起動は成功するがサービスが起動に失敗する場合は、カーネルパラメーターに systemd.unit=rescue.target を追加すると解決の役に立つかもしれません。

それ以外の場合、カーネルパラメーターとして systemd.unit=emergency.target を指定することによって、可能な限り早い段階で root シェルが使えるようになります。ですが、これは root ファイルシステムを読み書き可能な権限でマウントする前に実行されます。以下を手動で実行する必要があるでしょう:

```
# mount -o remount,rw /
```

もう一つのアプローチは debug-shell.service 経由で systemd の "早い段階でのデバッグシェル" を有効にすることです。次の起動時にこのサービスは起動プロセスの初期段階で tty9 にて root のログインシェルを立ち上げます。カーネルの起動パラメーターで systemd.debug-shell=1 と指定して有効にするか、あ

[3] 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。

るいは `systemctl enable debug-shell` で設定を永続的にします (この場合、デバッグが完了した際には再度無効にできます)。

systemd 環境下で起動がおかしいのをデバッグする詳細な情報については、Freedesktop.org の Diagnosing Boot Problems という記事で参照できます。

4.1.5 アップグレード用の安全な環境の準備

重要: (**tinc** のような) VPN サービスを使っている場合、アップグレード作業中に使えなくなる可能性を考慮してください。**サービスのダウン期間の準備** を参照してください。

リモートでのアップグレード時にさらなる安全マージンを得るため、`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作業を行うことを提案します。この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一時的に切断された場合でもアップグレード作業が中断しないようにしてくれます。

micro-evtd パッケージにより提供される watchdog デーモンのユーザは、アップグレードの前にデーモンを止めて watchdog タイマーを無効化し、アップグレード作業の途中で誤ってリブートが起きないようにすべきです:

```
# service micro-evtd stop
# /usr/sbin/microapl -a system_set_watchdog off
```

4.2 "純粋"な Debian からの作業開始

この章で説明しているアップグレードのプロセスは、"純粋"な安定版の Debian システムを想定して書かれています。もし、APT の設定が bookworm 以外で追加のソースを指定している、あるいは他のリリースやサードパーティからパッケージをインストール している場合、確実にアップグレード作業を遂行するため、事態をややこしくするこれらの要因を取り除くことから始めると良いでしょう。

APT がどのソースからパッケージをダウンロードするべきかを判断するのに使っている主要設定ファイルは `/etc/apt/sources.list` ですが、`/etc/apt/sources.list.d/` ディレクトリ内のファイルを使用することもできます。詳細は sources.list(5) を参照してください。もしシステムで複数の source-list ファイルを使用しているのであれば、設定に一貫性があることを確認する必要があるでしょう。

4.2.1 Debian 12 (bookworm) からのアップグレード

12 (bookworm) からのアップグレードのみがサポートされています。Debian のバージョンを表示するには以下を実行します:

```
$ cat /etc/debian_version
```

Debian 12 へのアップグレードが必要な場合、まず https://www.debian.org/releases/bookworm/releasenotes にある Debian 12 のリリースノートの指示に従ってください。

4.2.2 最新のポイントリリースへのアップグレード

またこの手順は、システムが bookworm の最新ポイント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、*bookworm* **システムのアップグレード** 内の指示に従ってください。

4.2.3 Debian Backports

Debian Backports は Debian 安定版のユーザーがより最新に近いバージョンのパッケージを (テストとセキュリティサポートの不足のトレードオフ込みで) 実行できるようにしてくれます。Debian Backports チームは、次期 Debian リリースからのサブセットパッケージを現在の Debian 安定版リリースで使えるようにするために調整と再コンパイルなどしてメンテナンスしています。

bookworm-backports から取得したパッケージは trixie にあるバージョンよりも小さいバージョン番号なので、ディストリビューションのアップグレードの作業中、"純粋な" bookworm パッケージと同じやり方で trixie へと問題なくアップグレードできるはずです。今の所は潜在的な問題は特定されてないものの backports 経由のアップグレードはテストが少なく、それに応じてよりリスクがあります。

> **注意:** 通常の Debian Backports はサポートされているものの、(bookworm-backports-sloppy を参照している APT source-list の記述を使っている) sloppy backports からのクリーンなアップグレード経路は存在しません。

非公式なソース と同様に、ユーザーはアップグレードの前に APT source-list ファイル群から "bookworm-backports" の記述を削除することが推奨されています。アップグレードの完了後に、"trixie-backports" (https://backports.debian.org/Instructions/ をご覧ください) の追加が検討できるでしょう。

詳細については Backports Wiki ページ を調べてください。

4.2.4 パッケージデータベースの準備

パッケージデータベースの準備が整っているか、アップグレードする前に確認してください。もしパッケージマネージャ **aptitude** や **synaptic** を使っているなら、それらにおいて中断しているアクションがないか確認してください。パッケージマネージャにおいて、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可能性があります。パッケージマネージャにおけるアクションの修正は、APT source-list ファイルに "stable" や "trixie" ではなく、"bookworm" 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。*APT source-list* **ファイルのチェック** も参照してください。

4.2.5 利用されなくなったパッケージ

アップグレードする前に **古いパッケージをシステムから削除する** のも良いでしょう。古いパッケージはアップグレードを難しくさせ、メンテナンスされていなければセキュリティリスクが存在しうるからです。

4.2.6 Debian 由来でないパッケージを削除する

Debian 由来でないパッケージを見つけるには、以下の apt または apt-forktracer を使った 2 つの手法があります。どちらも 100% 正確ではない点を留意して下さい (例: apt の例では、古いカーネルパッケージのように一度は Debian によって提供されていたが今は提供されていないパッケージを表示します)。

```
$ apt list '?narrow(?installed, ?not(?origin(Debian)))'
$ apt-forktracer | sort
```

4.2.7 残っている設定ファイルを取り除く

以前のアップグレードでは使われていない設定ファイルのコピー、パッケージメンテナーによって提供された **古いバージョン** の設定ファイルなどが残されているかもしれません。以前のアップグレードから取り残されたファイルを削除すると混乱を避けることができます。そのような取り残されたファイルを見つけるには:

```
# find /etc -name '*.dpkg-*' -o -name '*.ucf-*' -o -name '*.merge-error'
```

4.2.8 non-free コンポーネントと non-free-firmware コンポーネント

non-free なファームウェアをインストールしていた場合、APT sources-list へ non-free-firmware の追加が推奨されています。

4.2.9 proposed-updates セクション

APT source-list ファイルに proposed-updates セクションを含めている場合は、システムのアップグレードを試みる前に、それらのセクションをファイルから削除してください。これは衝突の可能性を減らすための予防策です。

4.2.10 非公式なソース

システムに Debian 以外のパッケージがインストールされている場合、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してください。当該パッケージが APT source-list ファイルに Debian 以外の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 trixie 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項目と同時にそれも適切に修正してください。

ユーザによっては bookworm システムに **非公式に** バックポートされた "より新しい" バージョンのパッケージが **存在している** こともあるでしょう。そのようなパッケージはファイルが競合する可能性があるので、アップグレードの際にはおそらく問題を起こします[4]。**アップグレード中の注意点** には、もしそのような競合が起きた場合にどうやって対処するのか、という情報があります。

4.2.11 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のディストリビューション (テスト版など) からインストールするように APT を設定している場合、その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` および `/etc/apt/preferences.d/` 内に保存されている) APT の pin 設定を変更しなければならないかもしれません。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。

4.2.12 パッケージの状態をチェックする

アップグレードの方法に関係なく、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のを確認することをお勧めします。次のコマンドは、インストールが未完了のパッケージ (Half-Installed) や設定に失敗したパッケージ (Failed-Config)、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
```

`aptitude` や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
```

または

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

別の方法としては `apt` を使うこともできます。

```
# apt list --installed > ~/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 状態にある場合、アップグレードに失敗します。

[4] Debian のパッケージ管理システムにおいて、別のパッケージを置き換えるように指定されていないパッケージは、通常、別のパッケージの所有ファイルを削除したり置き換えたりすることはできません。

```
$ apt-mark showhold
```

パッケージをローカルで変更・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。

apt 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。

```
# apt-mark hold package_name
```

"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、*APT source-list* **ファイルのチェック** で説明するように APT source-list ファイルが bookworm を指定したままにしておくべきです。

4.3 APT source-list ファイルの準備

アップグレードを始める前に、APT の source-list ファイル (/etc/apt/sources.list および /etc/apt/sources.list.d/ 以下のファイル) に trixie を追加し、bookworm を削除する必要があります。

APT は、あらゆる設定済みアーカイブを通して見つかった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、ファイルで最初に現れたエントリを優先します。したがって、複数のミラーを指定する場合は、最初にローカルのハードディスクを、次に CD-ROM を、最後にリモートミラーを指定すると良いでしょう。

リリースを指定するのに、コードネーム ("bookworm" や "trixie") と状態名 ("oldstable"、"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には、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、リリースが行われた直後に、パッケージが大量に更新可能になったことに気づくでしょう。

Debian は、Debian のリリースに関わる関連情報について最新の状態を保つために役立つ 2 つのアナウンス用メーリングリストを提供しています:

- Debian アナウンスメーリングリストを購読 すれば、Debian が新しいリリースを行う度に通知がきます。例えば、"trixie" の "テスト版 (testing)" から "安定版 (stable)" へ変わった時などです。
- Debian セキュリティアナウンスメーリングリストを購読 すれば、Debian がセキュリティのアナウンスを公開する度に通知を受け取ります。

4.3.1 APT のインターネットソースの追加

新規インストールではデフォルトはネットワークの条件によってあなたに近いサーバから自動的にパッケージをダウンロードできる Debian APT CDN サービスを使うように APT を設定します。これは比較的新しいサービスのため、古いインストールでは以前としてメインの Debian インターネットサーバのひとつまたはミラーのひとつを設定しているかもしれません。もしまだ設定を行っていない場合、APT 設定において CDN サービスを使うように切り替えることを推奨します。

CDN サービスを利用するには、あなたの APT ソース設定にこのような 1 行を追加してください (`main` と `contrib` を使用していると仮定します):

```
deb https://deb.debian.org/debian trixie main contrib
```

新しいソースを追加した後、"`deb`" 行の先頭に、ハッシュ記号 (`#`) を追加して無効にしてください。

しかしながら、もしあなたのネットワークの条件から近い特定のミラーを使用して良い結果が得たいならば、このオプションはまだ利用可能です。

Debian ミラーのアドレスは、https://www.debian.org/mirror/list にあります。

For example, suppose your closest Debian mirror is `https://mirrors.kernel.org`. If you inspect that mirror with a web browser, you will notice that the main directories are organized like this:

```
https://mirrors.kernel.org/debian/dists/trixie/main/...
https://mirrors.kernel.org/debian/dists/trixie/contrib/...
```

与えられたミラーを使うよう APT 設定をするには、このような 1 行を追加してください (再度、`main` と `contrib` を使用していると仮定します):

```
deb https://mirrors.kernel.org/debian trixie main contrib
```

"`dists`" を書かなくても、暗黙のうちに追加します。リリース名の後の各引数は、パスの末尾につけて、複数のディレクトリに展開するのに用います。

再度、あなたの新しいソースを追加した後、既存のアーカイブエントリを無効にしてください。

4.3.2 APT のローカルミラーソースの追加

HTTP パッケージミラーを使うのではなく、ローカルディスク (おそらくは NFS マウントされたもの) にあるミラーを使うよう、APT source-list ファイルを変更したいことがあるかもしれません。

例えばパッケージのミラーが `/var/local/debian/` にあり、主なディレクトリの配置が次のようになっているとします。

```
/var/local/debian/dists/trixie/main/...
/var/local/debian/dists/trixie/contrib/...
```

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

```
deb file:/var/local/debian trixie main contrib
```

"`dists`" を書かなくても、暗黙のうちに追加します。リリース名の後の各引数は、パスの末尾につけて、複数のディレクトリに展開するのに用います。

新しいソースを追加した後、APT source-list ファイル内の既存のアーカイブエントリの先頭にハッシュ記号 (#) を追加して無効にしてください。

4.3.3 APT の光学メディアソースの追加

DVD (や CD、Blu-ray ディスク) **だけ** を使いたい場合は、すべての APT source-list ファイル内の既存エントリの先頭にハッシュ記号 (#) 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 /media/cdrom にマウントできるようにしている行が /etc/fstab にあるかどうかを確認してください。例えば /dev/sr0 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/sr0 /media/cdrom auto noauto,ro 0 0
```

第 4 フィールドの noauto,ro の単語の間には、スペースを入れてはいけません。

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 /media/cdrom    # this will mount the CD to the mount point
# ls -alF /media/cdrom  # this should show the CD's root directory
# umount /media/cdrom   # this will unmount the CD
```

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。

4.4 パッケージのアップグレード

推奨する方法はパッケージ管理ツール apt を使って前の Debian リリースからアップグレードすることです。

注釈: apt は対話式な用途を目的としており、スクリプトの中で使うべきではありません。スクリプトの中では字句解析に適していて安定した出力をもつ apt-get を使うべきです。

まず、必要なすべてのパーティション (特にルートパーティションと /usr パーティション) を read-write モードでマウントするのを忘れずに行いましょう。それには以下のようなコマンドを使います。

```
# mount -o remount,rw /mountpoint
```

次に、(/etc/apt/sources.list や /etc/apt/sources.list.d/ 以下のファイル内の) APT ソースのエントリが "trixie" 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。bookworm を指し示すソースエントリが含まれてはいけません。

注釈: 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**。

4.4.1 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セッション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、必要に応じてバグ報告に正確な情報を含めることができます。記録を開始するには次のように入力します。

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

typescript を再度実行する必要がある場合 (例: システムを再起動する必要がある場合) は、どのアップグレード手順のログを取っているのかを示すため、別の **手順** 番号を使ってください。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内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)。

また、typescript ファイルに記録することで、スクロールしてスクリーンから消えた情報をもう一度見ることができるようにもなります。システムのコンソールの前に居る場合は、(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-trixie.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。

apt は /var/log/apt/history.log に変更されたパッケージの状態を、/var/log/apt/term.log に端末の出力を記録します。

script に *-t* スイッチをつけておいた場合は、以下のように 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-trixie-step.time ~/upgrade-trixie-step.script
```

4.4.2 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。

```
# apt update
```

注釈: apt-secure のユーザは aptitude や apt-get を使うと問題を見つけることができるかもしれません。apt-get の場合、apt-get update --allow-releaseinfo-change を使うことができます。

4.4.3 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、**システムのアップグレード** 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てください。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パッケージは、すべて /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インストール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。

apt で、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できます。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。

```
# apt -o APT::Get::Trivial-Only=true full-upgrade
[ ... ]
XXX upgraded, XXX newly installed, XXX to remove and XXX not upgraded.
Need to get xx.xMB of archives.
After this operation, AAAMB of additional disk space will be used.
```

注釈: 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、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** で説明しているシステムの最小アップグレードを行う必要があります。

アップグレードをするのに十分な領域がない場合は、apt が以下のような警告メッセージを出します。

```
E: You don't have enough free space in /var/cache/apt/archives/.
```

この場合、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。

- インストールのために、以前 (/var/cache/apt/archives に) ダウンロードしたパッケージを削除する。apt 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。
- 忘れ去られたパッケージを削除する。さらに、bookworm で手作業でパッケージをインストールするのに aptitude や apt 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います。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対して、依存元パッケージが削除されたためにもう不要となった場合に、余分だというマークをつけることができるでしょう。手作業でインストールしたパッケージには削除されるマークをつけません。自動的にインストールされたがもはや使われていないパッケージを削除するには、以下を実行してください:

  ```
  # apt autoremove
  ```

You can also use `debfoster` to find redundant packages. Do not blindly remove the packages this tool presents, especially if you are using aggressive non-default options that are prone to false positives. It is highly recommended that you manually review the packages suggested for removal (i.e. their contents, sizes, and descriptions) before you remove them.

- 多くの容量を占めていて現在必要の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インストールし直せます)。**popularity-contest** をインストールしていれば、`popcon-largest-unused` を使うことにより、容量の多くを占めていて使うことのないパッケージの一覧が得られます。`dpigs` (**debian-goodies** パッケージに収録) や `wajig` (`wajig size` を実行) により、ディスク容量を消費するだけのパッケージを検索することができます。こういった情報は **aptitude** を使って検索することもできます。`aptitude` を full-terminal モードで起動し、表示 > フラットなパッケージ一覧を新規に作成 を実行し、`l` を押して、`~i` と入力します。`S` を押して `~installsize` と入力します。すると、作業しやすい一覧が得られます。
- 翻訳や地域化用ファイルが不要なら、それらをシステムから削除する。**localepurge** パッケージをインストールして設定すれば、選んだ少数のロケールのみがシステムに残るようにすることが可能です。これによって、`/usr/share/locale` の消費するディスク領域を減らせるでしょう。
- `/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。
- 仮設の `/var/cache/apt/archives` を使用する。すなわち、別のファイルシステム (USB ストレージデバイス、一時的なハードディスク、既に使用されているファイルシステムなど) を仮設のキャッシュディレクトリとして拝借することができます。

注釈: アップグレード中にネットワーク接続が途切れる可能性があるので、NFS マウントは使用しないでください。

以下は、`/media/usbkey` にマウントされた USB ドライブがある場合を例とします。

1. 今までに、インストールのためにダウンロードされたパッケージを削除します。

```
# apt clean
```

2. ディレクトリ `/var/cache/apt/archives` を、USB ドライブにコピーします。

```
# cp -ax /var/cache/apt/archives /media/usbkey/
```

3. 現在のキャッシュディレクトリに、仮のキャッシュディレクトリをマウントします。

```
# mount --bind /media/usbkey/archives /var/cache/apt/archives
```

4. アップグレード後に、元々の `/var/cache/apt/archives` ディレクトリを復活させます。

```
# umount /var/cache/apt/archives
```

5. 残っている `/media/usbkey/archives` を削除します。

仮設のキャッシュディレクトリは、システムにマウントされているファイルシステムであれば何にでも作成できます。

- システムの最小アップグレードを行う (**システムの最小アップグレード** 参照)、あるいは完全アップグレードにしたがって、システムの部分的なアップグレードを行う。これによって、システムを部分的にアップグレードが可能になり、完全アップグレード前にパッケージキャッシュの削除ができます。

パッケージを安全に削除するための注意として、*APT source-list* **ファイルのチェック** で説明するように、APT source-list ファイルが bookworm を指し示すよう設定を戻しておくことが望ましいです。

4.4.4 監視システムの停止

apt があなたのコンピューターで動作しているサービスを一時的に停止する必要があるかもしれないため、アップグレードの最中に終了された他のサービスを再起動できるような監視用サービスを予め停止しておくのが良い考えでしょう。Debian では、**monit** がそのようなサービスの例です。

4.4.5 システムの最小アップグレード

完全アップグレード (以下に記述しています) を直接行った場合、残しておきたいパッケージが大量に削除されてしまうことが時折あります。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 **システムのアップグレード** にあるような完全なアップグレードを行う、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。

これをまず行うには、以下のコマンドを実行してください。

```
# apt upgrade --without-new-pkgs
```

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。

システムの容量が少なく、容量による制約のため完全アップグレードが実行できない場合にも、システムの最小アップグレードは有用です。

apt-listchanges パッケージがインストールされていれば、(デフォルト設定で) パッケージのダウンロード後にアップグレードされるパッケージについての重要な情報をページャで表示します。読み終わったら q を押してページャを終了し、アップグレードを続けてください。

4.4.6 システムのアップグレード

これまでの手順を実行し終わったら、アップグレードの主要な部分を続ける準備ができています。以下のコマンドを実行してください。

```
# apt full-upgrade
```

これによってシステムの完全なアップグレードが行われ、すべてのパッケージの最新版がインストールされ、リリース間で発生しうるパッケージの依存関係の変化すべてが解決されます。必要に応じて、新しいパッケージ (通常は、新しいバージョンのライブラリや、名前の変わったパッケージ) がインストールされたり、衝突した古いパッケージが削除されたりもします。

CD/DVD/BD のセットからアップグレードする場合には、アップグレードの最中に、おそらく特定のディスクを入れるよう何回か指示されることになるでしょう。同じディスク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ディスクに分散しているためです。

現在インストールされているパッケージを新しいバージョンへとアップグレードする際に、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、または apt install パッケージ名 を実行してみると、解決できます。

4.5 アップグレード中の注意点

以下の章では、trixie へのアップグレードの最中に現れるかもしれない既知の問題を記述しています

4.5.1 「即時設定は動作しません」で full-upgrade が失敗する

apt full-upgrade の途中でパッケージをダウンロードした後に失敗となり、

```
E: Could not perform immediate configuration on 'package'.  Please see man 5 apt.conf␣
↪under APT::Immediate-Configure for details.
```

と表示することがあります。これが起きた場合は、代わりに apt full-upgrade -o APT::Immediate-Configure=0 を実行することでアップグレードを進められるはずです。

この問題の暫定的な別の対処の可能性として、bookworm と trixie の両方のソースを一時的に sources.list に追加して apt update を実行する方法があります。

4.5.2 予期されるパッケージの削除

trixie へのアップグレード作業では、システム中のパッケージ削除を尋ねてくるかもしれません。実際のパッケージ一覧は、インストールしてあるパッケージの構成によって異なってくるでしょう。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。trixie で時代遅れ (obsoleted) となったパッケージの詳細については、**利用されなくなったパッケージ** を参照してください。

4.5.3 衝突 (Conflicts) あるいは事前依存 (Pre-Depends) のループ

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 apt はこのことを警告してアップグレードを中断します。apt のコマンドラインにオプション -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あまりに問題だらけで、手動での介入が必要となることもあります。通常、手動での介入とは、apt を用いるか、あるいは

```
# dpkg --remove package_name
```

で問題の原因となるパッケージを消す作業になります。または次の方法を用いてもよいかもしれません。

```
# apt -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/package_name.deb
```

4.5.4 ファイルの衝突

"純粋" な bookworm 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、非公式の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:

```
Unpacking <package-foo> (from <package-foo-file>) ...
dpkg: error processing <package-foo> (--install):
trying to overwrite `<some-file-name>',
which is also in package <package-bar>
dpkg-deb: subprocess paste killed by signal (Broken pipe)
Errors were encountered while processing:
<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 package_name
```

問題が修正できたら、先程説明した apt コマンドを再度入力すれば、アップグレードを再開できます。

4.5.5 設定の変更

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 ディレクトリと /etc/manpath.config ファイルに関しては、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 という拡張子をつけられて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もう一度見ることもできます。

4.5.6 コンソール接続へセッションの変更

システムのローカルコンソールを使ってアップグレードを実行している場合、アップグレードの最中に何回かコンソールが別の画面へ移動してしまい、アップグレード作業が見えなくなることに気づくかもしれません。例えば、グラフィカルインターフェイスがあるシステムではディスプレイマネージャが再起動した際に起こります。

仮想ターミナル 1 に戻るには (グラフィカルの起動画面の場合は) Ctrl+Alt+F1、あるいは (ローカルのテキストモードコンソールの場合には) Alt+F1 を使う必要があります。F1 は、アップグレードが実行されている仮想ターミナルの番号と同じ番号のファンクションキーと置き換えてください。異なったテキストモードのターミナル間で切り替えを行うには、Alt+左矢印 か Alt+右矢印 も使えます。

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明確にします。Debian で提供されている **linux-image-*** パッケージのいずれか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 のモジュラーカーネルのいずれかを **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た場合や、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。

4.6.1 カーネルメタパッケージのインストール

bookworm から trixie への full-upgrade を実行する際、新しい linux-image-* メタパッケージを、過去にインストールしていない場合にはインストールすることを強くお勧めします。これらのメタパッケージは、アップグレードの最中に自動的に新しいバージョンのパッケージを取得します。次のように実行すると、どのパッケージがインストールされているのかを確認できます:

```
$ dpkg -l 'linux-image*' | grep ^ii | grep -i meta
```

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か、linux-image メタパッケージをインストールする必要があります。利用可能な linux-image 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- | grep -i meta | 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、コマンドの出力が '4.9.0-8-amd64' の場合は **linux-image-amd64** 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 を用いて各パッケージのパッケージ説明の詳細版を参照してもよいでしょう。以下は例です:

```
$ apt show linux-image-amd64
```

インストールするカーネルイメージが決まったら、apt install でインストールします。次回、再起動可能になった際に新しいカーネルバージョンが提供するメリットを得るために再起動を実行する必要があります。ですが、アップグレード後の最初の再起動を実行する前に **アップグレード後、再起動前にすること** を参照してください。

少し勇気のある人には、Debian 上で簡単に自分のカスタムカーネルをコンパイルするやり方があります。**linux-source** パッケージで提供されるカーネルソースをインストールしてください。バイナリパッケージの構築には、ソース中の Makefile 中の deb-pkg ターゲットが使えます。さらなる情報は、Debian Linux カーネルハンドブック にあります。**debian-kernel-handbook** パッケージでも利用できます。

可能であればカーネルパッケージのアップグレードをメインの full-upgrade と分けることで、一時的にでも起動不能なシステムにしてしまうことを極力避けられます。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** 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。

4.7 次のリリースへの準備

アップグレードの後で、次のリリースに向けてできるいくつかの準備があります。

- **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** や **利用されなくなったパッケージ** で説明するように、余分、あるいは時代遅れ (obsolete) のパッケージを削除してください。それらのパッケージが使用する設定ファイルを確認し、パッケージの完全削除 (purge) によって、設定ファイルも含めて削除することを検討してください。**削除したパッケージを完全削除する** についても参照をお願いします。

4.7.1 削除したパッケージを完全削除する

一般的に、削除したパッケージを完全に削除 (purge) するのは賢明なことです。以前のリリースアップグレード (つまりは bookworm へのアップグレード) の際に削除されているパッケージである、あるいはパッケージがサードパーティベンダーから提供されたものである場合、尚のこととなります。特に、古い init.d スクリプトは問題を起こすことが知られています。

> **注意:** パッケージの完全削除 (purge) は通常ログファイルについても完全に削除を行うので、まずはこのバックアップを行ったほうが良いでしょう。

以下のコマンドは、設定ファイルをシステムに残して削除されたパッケージの一覧を (もしあれば) 表示します:

```
$ apt list '~c'
```

apt purge を実行すればパッケージを削除できます。一度でこれらのパッケージを削除したい場合は、以下のコマンドで実施できます:

```
# apt purge '~c'
```

4.8 利用されなくなったパッケージ

trixie では大量の新規パッケージが導入された一方で、bookworm に存在していた非常の少量の古いパッケージの破棄や削除が行われています。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 プロジェクトでは通常、trixie がリリースされてから 1 年後にセキュリティサポートを終了します[5]。そして、その時点から他のサポートも提供しません。利用可能な代替手段で置き換えられるのであれば、そうすることをお勧めします。

パッケージが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——もう上流で保守されていない、そのパッケージの保守作業に興味を抱く Debian 開発者がもういない、提供していた機能が別のソフトウェア (または新しいバージョン) に取って代わられた、バグのために trixie にはもう適さないと見なされた、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まだ存在していることがあります。

"廃止、あるいはローカルで作成されたパッケージ" は以下のコマンドラインでまとめて表示・削除できます:

```
$ apt list '~o'
# apt purge '~o'
```

Debian バグ追跡システム は、パッケージが削除された理由についての追加情報を提供してくれることがよくあります。そのパッケージ自体と ftp.debian.org 擬似パッケージ の両方の、アーカイブ化されたバグ報告を調べてください。

trixie での廃止パッケージ一覧については、**特記すべき廃止されたパッケージたち** を参照して下さい。

4.8.1 移行用ダミーパッケージ

bookworm からのいくつかのパッケージは trixie においてアップグレードを簡単にできるよう設計された空の代用品である移行用ダミーパッケージによって置き換えられるかもしれません。以前は 1 つのパッケージであったアプリケーションがいくつかのパッケージに分割された場合、移行用パッケージは古いパッケージと同じ名前で、新しいパッケージをインストールするための適切な依存関係をもって提供されるかもしれません。これをインストールした後は冗長なダミーパッケージを安全に削除できます。

The package descriptions for transitional dummy packages usually indicate their purpose. However, they are not uniform; in particular, some "dummy" packages are designed to be kept installed, in order to pull in a full software suite, or track the current latest version of some program.

[5] あるいは、1 年以内でも別のリリースが出るときに。一般に、どの時点でも、サポートされる安定版リリースは 2 つだけです。

第5章　trixie で注意すべき点

新しいリリースで導入された変更点には副作用が避けられず、どこか他の場所でバグを出してしまうことがあります。この章では、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** で触れられているその他の情報も読んでください。

5.1 trixie へアップグレードする際に注意すべきこと

この項では bookworm から trixie へのアップグレードに関連した項目を取り扱います。

5.1.1 i386 サポートの縮小

trixie 以降、i386 は標準でサポートされるアーキテクチャではなくなりました: 公式にカーネルや Debian インストーラーは i386 向けに提供されません。多くのプロジェクトで i386 をサポートしないため、i386 向けのパッケージが利用可能なのはごく一部に限られます。このアーキテクチャが唯一残されているのは、multiarch を利用した 64-bit (amd64) システム上の chroot でレガシーコードをサポートするためです。

i386 アーキテクチャは 64-bit (amd64) CPU 上でのみ利用することを意図しています。命令セットが SSE2 サポートを必要とするため、Debian 12 でサポートされていた 32-bit CPU の多くで動作しません。

i386 なシステムを動かしているユーザーは、trixie にアップグレードすべきではありません。そのかわりに、Debian では amd64 にシステムを再インストールして延命するか、可能なら該当ハードウェアの使用をやめることを推奨しています。技術的には Cross-grading により再インストールをともなわないアップグレードも可能ですが、リスクがある代替手段です。

5.1.2 openssh-server no longer reads ~/.pam_environment

The Secure Shell (SSH) daemon provided in the **openssh-server** package, which allows logins from remote systems, no longer reads the user's `~/.pam_environment` file by default; this feature has a history of security problems and has been deprecated in current versions of the Pluggable Authentication Modules (PAM) library. If you used this feature, you should switch from setting variables in `~/.pam_environment` to setting them in your shell initialization files (e.g. `~/.bash_profile` or `~/.bashrc`) or some other similar mechanism instead.

Existing SSH connections will not be affected, but new connections may behave differently after the upgrade. If you are upgrading remotely, it is normally a good idea to ensure that you have some other way to log into the system before starting the upgrade; see **復旧の準備**.

5.1.3 OpenSSH no longer supports DSA keys

Digital Signature Algorithm (DSA) keys, as specified in the Secure Shell (SSH) protocol, are inherently weak: they are limited to 160-bit private keys and the SHA-1 digest. The SSH implementation provided by the **openssh-client** and **openssh-server** packages has disabled support for DSA keys by default since OpenSSH 7.0p1 in 2015, released with Debian 9 ("stretch"), although it could still be enabled using the `HostKeyAlgorithms` and `PubkeyAcceptedAlgorithms` configuration options for host and user keys respectively.

The only remaining uses of DSA at this point should be connecting to some very old devices. For all other purposes, the other key types supported by OpenSSH (RSA, ECDSA, and Ed25519) are superior.

As of OpenSSH 9.8p1 in trixie, DSA keys are no longer supported even with the above configuration options. If you have a device that you can only connect to using DSA, then you can use the `ssh1` command provided by the **openssh-client-ssh1** package to do so.

In the unlikely event that you are still using DSA keys to connect to a Debian server (if you are unsure, you can check by adding the `-v` option to the `ssh` command line you use to connect to that server and looking for the "Server accepts key:" line), then you must generate replacement keys before upgrading. For example, to generate a new Ed25519 key and enable logins to a server using it, run this on the client, replacing `username@server` with the appropriate user and host names:

```
$ ssh-keygen -t ed25519
$ ssh-copy-id username@server
```

5.1.4 The last, lastb and lastlog commands have been replaced

The **util-linux** package no longer provides the `last` or `lastb` commands, and the **login** package no longer provides `lastlog`. These commands provided information about previous login attempts using `/var/log/wtmp`, `/var/log/btmp`, `/var/run/utmp` and `/var/log/lastlog`, but these files will not be usable after 2038 because they do not allocate enough space to store the login time (the Year 2038 Problem), and the upstream developers do not want to change the file formats. Most users will not need to replace these commands with anything, but the **util-linux** package provides a `lslogins` command which can tell you when accounts were last used.

There are two direct replacements available: `last` can be replaced by `wtmpdb` from the **wtmpdb** package (the **libpam-wtmpdb** package also needs to be installed) and `lastlog` can be replaced by `lastlog2` from the **lastlog2** package (**libpam-lastlog2** also needs to be installed). If you want to use these, you will need to install the new packages after the upgrade, see the util-linux NEWS.Debian for further information. The command `lslogins --failed` provides similar information to `lastb`.

If you do not install **wtmpdb** then we recommend you remove old log files `/var/log/wtmp*`. If you do install **wtmpdb** it will upgrade `/var/log/wtmp` and you can read older wtmp files with `wtmpdb import -f <dest>`. There is no tool to read `/var/log/lastlog*` or `/var/log/btmp*` files: they can be deleted after the upgrade.

5.1.5 RabbitMQ no longer supports HA queues

High-availability (HA) queues are no longer supported by **rabbitmq-server** starting in trixie. To continue with an HA setup, these queues need to be switched to "quorum queues".

If you have an OpenStack deployment, please switch the queues to quorum before upgrading. Please also note that beginning with OpenStack's "Caracal" release in trixie, OpenStack supports only quorum queues.

5.1.6 RabbitMQ cannot be directly upgraded from bookworm

There is no direct, easy upgrade path for RabbitMQ from bookworm to trixie. Details about this issue can be found in bug 1100165.

The recommended upgrade path is to completely wipe the rabbitmq database and restart the service (after the trixie upgrade). This may be done by deleting `/var/lib/rabbitmq/mnesia` and all of its contents.

5.1.7 MariaDB major version upgrades only work reliably after a clean shutdown

MariaDB does not support error recovery across major versions. For example if a MariaDB 10.11 server experienced an abrupt shutdown due to power loss or software defect, the database needs to be restarted with the same MariaDB 10.11 binaries so it can do successful error recovery and reconcile the data files and log files to roll-forward or revert transactions that got interrupted.

If you attempt to do crash recovery with MariaDB 11.8 using the data directory from a crashed MariaDB 10.11 instance, the newer MariaDB server will refuse to start.

To ensure a MariaDB Server is shut down cleanly before going into major version upgrade, stop the service with

```
# service mariadb stop
```

followed by checking server logs for `Shutdown complete` to confirm that flushing all data and buffers to disk completed successfully.

If it didn't shut down cleanly, restart it to trigger crash recovery, wait, stop again and verify that second stop was clean.

For additional information about how to make backups and other relevant information for system administrators, please see /usr/share/doc/mariadb-server/README.Debian.gz.

5.1.8 Ping no longer runs with elevated privileges

The default version of ping (provided by **iputils-ping**) is no longer installed with access to the *CAP_NET_RAW* linux capability, but instead uses `ICMP_PROTO` datagram sockets for network communication. Access to these sockets is controlled based on the user's Unix group membership using the `net.ipv4.ping_group_range` sysctl. In normal installations, the **linux-sysctl-defaults** package will set this value to a broadly permissive value, allowing unprivileged users to use ping as expected, but some upgrade scenarios may not automatically install this package. See `/usr/lib/sysctl.d/50-default.conf` and the kernel documentation for more information on the semantics of this variable.

5.1.9 Dovecot configuration changes

The *dovecot* email server suite in trixie uses a configuration format that is incompatible with previous versions. Details about the configuration changes are available at docs.dovecot.org.

In order to avoid potentially extended downtime, you are strongly encouraged to port your configuration in a staging environment before beginning the upgrade of a production mail system.

5.1.10 Significant changes to libvirt packaging

The **libvirt-daemon** package, which provides an API and toolkit for managing virtualization platforms, has been overhauled in trixie. Each driver and storage backend now comes in a separate binary package, which enables much greater flexibility.

Care is taken during upgrades from bookworm to retain the existing set of components, but in some cases functionality might end up being temporarily lost. We recommend that you carefully review the list of installed binary packages after upgrading to ensure that all the expected ones are present; this is also a great time to consider uninstalling unwanted components.

In addition, some conffiles might end up marked as "obsolete" after the upgrade. The `/usr/share/doc/libvirt-common/NEWS.Debian.gz` file contains additional information on how to verify whether your system is affected by this issue and how to address it.

5.1.11 アップグレード後、再起動前にすること

`apt full-upgrade` が完了した時点で、"正規"のアップグレードは完了しています。trixie へのアップグレードについては、再起動の実行前に必要となる特別な作業はありません。

5.2 アップグレード後も影響がある項目

5.2.1 The directories /tmp and /var/tmp are now regularly cleaned

On new installations, *systemd-tmpfiles* will now regularly delete old files in `/tmp` and `/var/tmp` while the system is running. This change makes Debian consistent with other distributions. Because there is a small risk of data loss, it has been made "opt-in": the upgrade to trixie will create a file /etc/tmpfiles.d/tmp.conf which reinstates the old behavior. This file can be deleted to adopt the new default, or edited to define custom rules. The rest of this section explains the new default and how to customize it.

The new default behavior is for files in `/tmp` to be automatically deleted after 10 days from the time they were last used (as well as after a reboot). Files in `/var/tmp` are deleted after 30 days (but not deleted after a reboot).

Before adopting the new default, you should either adapt any local programs that store data in `/tmp` or `/var/tmp` for long periods to use alternative locations, such as `~/tmp/`, or tell *systemd-tmpfiles* to exempt the data file from deletion by creating a file `local-tmp-files.conf` in `/etc/tmpfiles.d` containing lines such as:

```
x /var/tmp/my-precious-file.pdf
x /tmp/foo
```

Please see systemd-tmpfiles(8) and tmpfiles.d(5) for more information.

5.2.2 セキュリティサポートにおける制限事項

Debian がセキュリティ問題に対する最小限のバックポートを約束できないパッケージがいくつか存在しています。これらについては以下の章で触れられています。

注釈: **debian-security-support** パッケージが、インストールされたパッケージのセキュリティサポート状況を確認するのに役立ちます。

ウェブブラウザとそのレンダリングエンジンにおけるセキュリティ更新の状態

Debian 13 は複数のブラウザエンジンを含んでおり、これらは一定の割合でセキュリティ脆弱性の影響を受けます。高い脆弱性率と長期ブランチ形式での upstream でのサポートが限定的なことによって、セキュリティ修正をバックポートしてこれらのブラウザならびにブラウザエンジンをサポートする事が難しくなっています。さらに、ライブラリとの相互依存性のため、開発元での新しいリリースへの更新を極めて難しくしています。**webkit2gtk** ソースパッケージを使ったアプリケーション (例: **epiphany**) はセキュリティサポートの対象ですが、qtwebkit (**qtwebkit-opensource-src** ソースパッケージ) を使っているアプリケーションはセキュリティサポートの対象外です。

一般的なウェブブラウザ利用として我々は Firefox または Chromium を推奨しています。安定版向けに現行の ESR リリースをリビルドすることで最新を維持します。同じ手法が Thunderbird にも適用されます。

一旦リリースが oldstable となると、公式サポート対象のブラウザは標準的な保証期間の更新を受け続けられないかもしれません。例えば、Chromium は oldstable では通常の 12 ヶ月 ではなく 6 ヶ月のセキュリティサポートのみを受けます。

Go および Rust 言語ベースのパッケージ

現在、Debian のインフラは静的リンクを行うパッケージをリビルドすることに問題を抱えています。Go および Rust のエコシステムの成長に伴い、インフラが強化されメンテナンスが行き届くようになるまでは、限定的なセキュリティサポートとなります。

多くの場合ですが Go あるいは Rust の開発用ライブラリへの更新は、定期的なポイントリリースでのみ提供されます。

5.3 廃止および非推奨となった事柄について

5.3.1 特記すべき廃止されたパッケージたち

以下は、よく知られていて特に廃止されたパッケージの一覧です (説明については **利用されなくなったパッケージ** 参照)。

廃止パッケージの一覧には以下が含まれます:

- NSS サービス gw_name の開発は 2015 年に停止しました。関連パッケージ **libnss-gw-name** は将来の Debian リリースにて削除される可能性があります。upstream の開発者は代わりに **libnss-myhostname** の利用を推奨しています。
- The **pcregrep** package has been removed from trixie. It can be replaced with `grep -P` (`--perl-regexp`) or `pcre2grep` (from **pcre2-utils**).

5.3.2 trixie で非推奨となったコンポーネント

次のリリースである Debian 14 (コードネーム forky) では、いくつかの機能が非推奨となります。14 へ更新する際にトラブルを防ぐためには、ユーザーは他の選択肢へ移行する必要があります。

これには以下の機能が含まれます:

- The **sudo-ldap** package will be removed in forky. The Debian sudo team has decided to discontinue it due to maintenance difficulties and limited use. New and existing systems should use **libsss-sudo** instead.

  Upgrading Debian trixie to forky without completing this migration may result in the loss of intended privilege escalation.

  For further details, please refer to bug 1033728 and to the NEWS file in the **sudo** package.

- The **sudo_logsrvd** feature, used for sudo input/output logging, may be removed in Debian forky unless a maintainer steps forward. This component is of limited use within the Debian context, and maintaining it adds unnecessary complexity to the basic sudo package.

For ongoing discussions, see bug 1101451 and the NEWS file in the **sudo** package.

- The **libnss-docker** package is no longer developed upstream and requires version 1.21 of the Docker API. That deprecated API version is still supported by Docker Engine v26 (shipped by Debian trixie) but will be removed in Docker Engine v27+ (shipped by Debian forky). Unless upstream development resumes, the package will be removed in Debian forky.

- The **openssh-client** and **openssh-server** packages currently support GSS-API authentication and key exchange, which is usually used to authenticate to Kerberos services. This has caused some problems, especially on the server side where it adds new pre-authentication attack surface, and Debian's main OpenSSH packages will therefore stop supporting it starting with forky.

  If you are using GSS-API authentication or key exchange (look for options starting with `GSSAPI` in your OpenSSH configuration files) then you should install the **openssh-client-gssapi** (on clients) or **openssh-server-gssapi** (on servers) package now. On trixie, these are empty packages depending on **openssh-client** and **openssh-server** respectively; on forky, they will be built separately.

- sbuild-debian-developer-setup has been deprecated in favor of sbuild+unshare

  sbuild, the tool to build Debian packages in a minimal environment, has had a major upgrade and should work out of the box now. As a result the package **sbuild-debian-developer-setup** is no longer needed and has been deprecated. You can try the new version with:

```
$ sbuild --chroot-mode=unshare --dist=unstable hello
```

- The **fcitx** packages have been deprecated in favor of **fcitx5**

  The **fcitx** input method framework, also known as **fcitx4** or **fcitx 4.x**, is no longer maintained upstream. As a result, all related input method packages are now deprecated. The package **fcitx** and packages with names beginning with **fcitx-** will be removed in Debian forky.

  Existing **fcitx** users are encouraged to switch to **fcitx5** following the fcitx upstream migration guide and Debian Wiki page.

5.4 既知の重大なバグ

「Debian は準備が出来たらリリースされる (Debian releases when it's ready) 」とはいうものの、それは残念ながら既知のバグが存在しないという意味ではありません。リリース作業の一環として、深刻度 (severity) が「重大 (serious)」以上のすべてのバグはリリースチームが精力的に追いかけていますが、 trixie リリース作業の最終工程において「無視をする (ignored)」とタグ付けがされた これらのバグ一覧 は Debian バグ追跡システム で確認ができます。以下のバグはリリース時に trixie へ影響があったものであり、このドキュメント中で触れる価値があるでしょう:

バグ番号	パッケージ名 (ソースあるいはバイナリ)	説明
1032240	**akonadi-backend-mysql**	akonadi server fails to start since it cannot connect to mysql database
1032177	**faketime**	faketime が (i386 では) 動作しない
918984	**src:fuse3**	fuse -> fuse3 のアップグレードパスを bookworm に向けて提供指定ください
1016903	**g++-12**	tree-vectorize: O2 レベルで間違ったコードが生成される (-fno-tree-vectorize は動作している)
1034752	**src:gluegen2**	non-free なヘッダを含む

第6章　Debian 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 (https://www.debian.org/releases/trixie/installmanual にあります) を越えた、Debian に関するより詳細な文書が、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、Debian リファレンス・Debian 新メンテナガイド・Debian FAQ などなどの様に、品質の高い文書を作成することを目的としています。現在利用可能なリソースのすべてについて、詳細は DDP のウェブサイト および Debian Wiki を参照して下さい。

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 手助けを求めるには

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、入手できるドキュメントでその問題について調査してからにしましょう。この章では新しく Debian ユーザになった人にとって役立つであろう、これらのリソース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、https://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 で参照できます。

For more information on OFTC please visit the website.

6.3 バグを報告する

私たちは Debian 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バグ追跡システム (Bug Tracking System: BTS) で提供しています。この BTS は https://bugs.debian.org/ で閲覧できます

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有効な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、リファレンス文書 (**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 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ーを、いろいろなサポート メーリングリスト 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 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 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。`how-can-i-help` というツールが作業するのに適した報告済みのバグを探すのに役立つでしょう。執筆が得意なら、文書 作成や既存文書の自分の言語への 翻訳 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のデータベース 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や、特定のユーザー層向けの Debian Pure Blends などがあります。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、プログラマー、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。きっと暖かで楽しい気持ちになれるはずです。

第7章　アップグレードの前に bookworm システムを調整する

この付録には、trixie へアップグレードする前に bookworm 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。

7.1 bookworm 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 bookworm 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、*APT source-list* **ファイルのチェック** で説明するように、パッケージリスト内に bookworm への参照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 bookworm ポイントリリースへとアップグレードされます。

7.2 APT source-list ファイルのチェック

APT source-list ファイル (sources.list(5) 参照) 内の行で "stable" を指定している行があるなら、trixie への準備が事実上できています。もしアップグレードへの準備がまだできていない場合には、これはお望みの設定ではないかもしれません。すでに `apt 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。

trixie 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 bookworm 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 ユーザーとして、お気に入りのエディタで関連の APT source-list ファイル (`/etc/apt/sources.list` など) を開き、

- `deb http:`
- `deb https:`
- `deb tor+http:`
- `deb tor+https:`
- `URIs: http:`
- `URIs: https:`
- `URIs: tor+http:`

- `URIs: tor+https:`

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 "bookworm" に変更してください。

`deb file:` または `URIs: 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 bookworm か trixie のどちらのアーカイブなのかを自分で調べなければならないでしょう。

重要: `deb cdrom:` または `URIs: 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 update
```

と実行して、パッケージリストを更新してください。

7.3 Performing the upgrade to latest bookworm release

To upgrade all packages to the state of the latest point release for bookworm, do

```
# apt full-upgrade
```

7.4 古く不要になった設定ファイルを削除する

システムを trixie へアップグレードする前に、古い設定ファイル (`/etc` 以下にある `*.dpkg-{new,old}` ファイルなど) をシステムから削除することを推奨します。

第8章　リリースノートの貢献者たち

たくさんの人々がリリースノートを手伝ってくれました。以下の方々もそうですが、他にもいらっしゃいます。

- Adam D. Barrat (2013 年での様々な修正),
- Adam Di Carlo (以前のリリース),
- Andreas Barth aba (2005 - 2007 間のリリース),
- Andrei Popescu (さまざまな貢献),
- Anne Bezemer (以前のリリース),
- Bob Hilliard (以前のリリース),
- Charles Plessy (GM965 問題の解説),
- Christian Perrier bubulle (Lenny インストールについて),
- Christoph Berg (PostgreSQL 固有の問題について),
- Daniel Baumann (Debian Live),
- David Prévot taffit (Wheezy リリースについて),
- Eddy Petrișor (さまざまな貢献),
- Emmanuel Kasper (バックポート),
- Esko Arajärvi (X11 アップグレードの書き直し),
- Frans Pop fjp (以前のリリース (Etch) について),
- Giovanni Rapagnani (数え切れない貢献),
- Gordon Farquharson (ARM 移植版関連),
- Hideki Yamane henrich (2006 年から貢献),
- Holger Wansing holgerw (2009 年から貢献),
- Javier Fernández-Sanguino Peña jfs (Etch および Squeeze　のリリースについて),
- Jens Seidel (ドイツ語翻訳、数え切れない貢献),
- Jonas Meurer (syslog 関連),
- Jonathan Nieder (Squeeze および Wheezy リリースについて),
- Joost van Baal-Ilić joostvb (Wheezy および Jessie リリースについて),

- Josip Rodin (以前のリリース),
- Julien Cristau jcristau (Squeeze および Wheezy リリースについて),
- Justin B Rye (英語の修正),
- LaMont Jones (NFS 問題の解説),
- Luk Claes (編集者のモチベーション管理),
- Martin Michlmayr (ARM 移植版関連),
- Michael Biebl (syslog 関連),
- Moritz Mühlenhoff (さまざまな貢献),
- Niels Thykier nthykier (Jessie リリースについて),
- Noah Meyerhans (数え切れない貢献),
- Noritada Kobayashi (日本語翻訳 (コーディネート)、数え切れない貢献),
- Osamu Aoki (さまざまな貢献),
- Paul Gevers elbrus (buster リリースについて),
- Peter Green (カーネルバージョンメモ),
- Rob Bradford (Etch リリース),
- Samuel Thibault (d-i でのブライユ点字サポートの解説),
- Simon Bienlein (d-i でのブライユ点字サポートの解説),
- Simon Paillard spaillar-guest (数え切れない貢献),
- Stefan Fritsch (Apache 関連の解説),
- Steve Langasek (Etch リリース),
- Steve McIntyre (Debian CD),
- Tobias Scherer ("proposed-update" の解説),
- victory victory-guest (マークアップの修正, 2006 年より貢献),
- Vincent McIntyre ("proposed-update" の解説),
- W. Martin Borgert (Lenny リリースノートの編集、DocBook XML への変換).

この文書はたくさんの言語に翻訳されています。翻訳者に大きな感謝を捧げます! 日本語への翻訳は以下の方が行いました。やまね ひでき (日本語訳 (全般))