Vine Linux
開発者ガイド

Project Vine

目次

Vine Linux 開発者ガイド

Project Vine

produced by Project Vine and exciting guys

概　要

Vine Linux の開発にはだれでも自由に参加することができ、必要とする人が必要とするものを作り上げる自由なオペレーティングシステムです。

Vine Linux 開発者になるには特別なスキルが必要なわけではありません。Vine Linux に要望を持つ人、私家版パッケージを作っている人、設定に戸惑った経験を Blog にしている人なども歓迎します。是非、我々と一緒に Vine Linux を作り上げましょう！

このガイドでは、Vine Linux の開発への参加方法と開発において注意すべき事項について説明します。

1　Vine Linux 開発への参加形態

この章では、Vine Linux 開発への参加形態について説明します。とりあえず、興味のない参加形態については、読み飛ばしても差し支えありません。

今後、開発を進めるに従って他の開発者とも関わりを持つことになるでしょう。その際に相手がどのような事をしているのか知っていると話が通じやすくなる場合もあります。読み飛ばした参加形態については、Vine Linux 開発の雰囲気になれて余裕が出てきた頃に読んでみると良いでしょう。

1.1 パッケージのテスト

Vine Linux では、各種アプリケーションを RPM パッケージ（以降、パッケージと呼ぶ）として提供しています。これらパッケージをテストし、バグトラッキングシステム `http://bts.vinelinux.org/`（以降、BTS と呼ぶ）を利用して問題を報告する事で Vine Linux の品質向上に貢献します。

パッケージのテストでは、インストールの可否やアプリケーションの起動の可否はもちろんのこと、一通りアプリケーションの機能を試して問題がないか確認します。一つの機能に対して複数の実行方法 [*1] が用意されている場合は、全ての実行方法を試してください。テストするパッケージがライブラリのように直接実行する事ができないパッケージについては、そのパッケージを必要とするアプリケーションをテストします。

テストの結果、予期しない動作（バグ）を発見した場合は、その動作を再現できるか確かめた上、Vine Linux BTS `http://bts.vinelinux.org/guest.cgi?project=VineLinux&action=top` で新規レポートを作成します。

新規レポートを投稿する場合はユーザ名とパスワードが必要です

スパム防止のため、新規レポートを投稿する場合はゲストユーザであってもユーザ名とパスワードによる認証が必要となっています。

ゲストユーザとして投稿する場合は、ユーザ名に guest、パスワードに vinelinux と入力して認証してください。

後述する項 2.2 でアカウントを登録した場合は、ログイン `http://bts.vinelinux.org/user.cgi` にアクセスし、登録したユーザ名とパスワードで認証してください。 ■

新規レポート作成にあたっては、パッケージ作成者がバグを確実に再現できる様、できるだけ詳細に記述してください。

[*1] メニューから実行する場合とショートカットキーにより実行する場合など

なお、正式なパッケージとして登録する前にパッケージ作成者以外にテストを行ってもらうパッケージを登録するリポジトリが用意されています。リリース済み Vine Linux 用の proposed-updates や開発版 VineSeed 用の test です。これらのリポジトリにパッケージが登録された場合、項 2.1.1 で紹介するメーリングリストに投稿されますのでテストへの協力をお願いします。

パッケージのテストに参加したい場合は、第 2 章と項 3.1 及び項 3.4 を読んでおいてください。

1.2　パッケージの作成・更新

Vine Linux で採用しているアプリケーションの多くは、オープンソースソフトウェアやフリーソフトウェアなどと呼ばれ、コンパイルの必要があるソースファイルのまま配布されています。コンパイル済みのバイナリパッケージが配布されている場合でもメジャーなディストリビューション向けに調整されており、そのまま使えない事が多々あります。

Vine Linux では、エンドユーザが簡単にインストールして使えるようにパッケージャと呼ばれる開発者が依存関係等を解決した上、コンパイルしたものを RPM パッケージと呼ばれる形式にして apt リポジトリに登録します。エンドユーザは、apt-get コマンドや Synaptic などを利用して使いたいパッケージのインストールを要求すれば、そのパッケージに必要とされるパッケージも同時にインストールする事ができます。

RPM パッケージの作成方法については、RPM パッケージの作成方法 `http://vinelinux.org/manuals/making-rpm.html` で細かく説明していますが、この文書では更にパッケージャとして Vine Linux の開発に参加する具体的な方法について記述します。

パッケージャとして Vine Linux の開発に参加したい場合は、第 2 章、第 3 章、第 4 章、第 5 章を読んでください。

1.3 ドキュメントの作成・更新・査読

Web を検索すると様々な情報を見つける事ができますが、ある程度の知識が要求されたり、古い情報でそのまま利用できなかったり、エンドユーザが取捨選択する事が困難な場合もあります。

Vine Linux の環境に合わせ体系立てたドキュメントを提供する事で、エンドユーザが情報の海に溺れないような基礎知識を得る事ができるようにしたいと考えています。また、英語のままになっているアプリケーションのヘルプを翻訳し、開発元にフィードバックしていくことも行っています。

そのための人材を随時募集中です。まだ、ドキュメントを書ける知識がないと思っても提供されているドキュメントを読んで「ここが分かりにくい」、「ここの通りにしたがドキュメントの通りの実行結果が得られない」というような事をフィードバックして頂けるだけでも結構です。

また、ブログなどを使って自分がつまづいたことなどメモしている方、あなたの日々の成果を公式ドキュメントに反映させませんか。

ドキュメントの作成・更新・査読に参加したい場合は、第 2 章を読んでください。

1.4 パッケージの翻訳

RPM パッケージにはパッケージの簡単な説明（Summary）やある程度詳しい説明（Description）が用意されています。英語の情報は必ず存在しますが、それを翻訳した日本語での情報も用意することができます。しかし、Vine Linux のパッケージの中には、日本語への翻訳がなされてないパッケージも存在します。その結果、Vine Linux のインストーラや Synaptic パッケージマネージャなどで表示される情報が英語のままになってしまいます。

これらのパッケージを捜し出し、日本語訳を行うことで Vine Linux の開発に貢献できます。

翻訳作業は、バグトラッキングシステム `http://bts.vinelinux.org/`

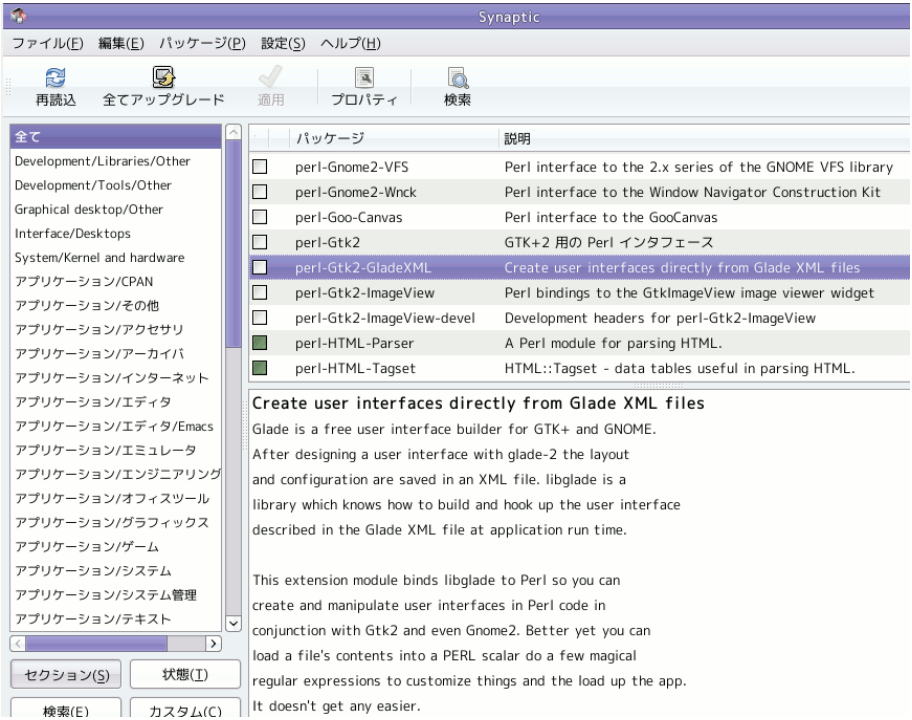

図 1　Summary や Description が英語のままになっている例

の SPEC ファイル翻訳 `http://bts.vinelinux.org/guest.cgi?project=rpmSpecTranslate&action=top` で行います。

パッケージの翻訳に継続して参加してくださる場合は、開発者用メーリングリストへの参加などをご検討ください。（参加方法は、第 2 章を参照してください。）

> **翻訳に対する権利の帰属について**
>
> 投稿して頂いた翻訳に対する権利は、Project Vine に帰属するものとします。

1.5 セキュリティー情報の収集・対応

Vine Linux Security Watch Team http://security.vinelinux.org/（以下、Security Watch Team）は、Vine Linux に関係するセキュリティ情報の収集と対応を行なっています。

Security Watch Team の活動は、専用のメーリングリストや BTS（Bug Tracking System） を通じて行なわれています。 代表的な作業には次のようなものがあります。

- セキュリティ情報の収集
- セキュリティ修正パッケージの作成
- セキュリティ修正パッケージのテスト
- 助言や手助け
- 遅れている作業への催促

協力してくださる方は Vine@vinelinux.org mailto:Vine@vinelinux.org 宛にメールをお願いします。

2 Vine Linux の開発に参加するには

2.1 まずは交流から

まずは、私たち Vine Linux の開発者と語らいましょう。Vine Linux 開発者の交流の場として、開発者用メーリングリストや IRC チャンネルが用意されています。

あなたができること、したいことを私たちにお聞かせください。そして、あなたが作成したパッケージや文書などを私たちにお見せください。

2.1.1 開発者用メーリングリストへの参加方法

Vine Linux 開発者の意見交換や情報共有を目的としたメーリングリストを用意しております。次の手順で参加してください。

1. VineSeed-ctl @ vinelinux.org `mailto:VineSeed-ctl @ vinelinux.org` 宛に

```
subscribe（半角英数で）あなたの名前
```

と書いたメールを送信します。
2.『Subscribe confirmation request (VineSeed ML)』という件名で登録を確認するメールが届きますのでそのまま返信します。
3. 登録に成功すると『Welcome to our (VineSeed ML) You are added automatically』という件名のメールが届きます。
4. 投稿用のメールアドレスは、VineSeed@vinelinux.org `mailto:VineSeed@vinelinux.org` です。まずは、簡単に自己紹介をお願いします。

リリース済みバージョンの開発者も参加してください

開発者用メーリングリストのメールアドレスは、Vine Linux の開発版である VineSeed の名前を冠していますが、VineSeed 専用ではありません。リリース済みバージョン向け追加パッケージ集である VinePlus の開発を行いたい方などすべての開発者が対象です。 ■

登録確認メールに返信する際は返信先をよく確認してください

現在のところ偽装メールの報告はありませんが、返信先が登録用メールアドレスになっている事を確認してください。 ■

Web で過去ログを確認できます

2009 年 2 月 16 日以降 [*2] のメールを Vine Linux ML アーカイブ `http://ml.vinelinux.org/` で公開しています。ご活用ください。 ■

[*2] これより以前のメールは非公開ポリシーであったため公開していません。

2.1.2 開発者 IRC 会議への参加方法

リアルタイムに TODO や問題などについてざっくばらんに議論を行う場として月一回を基準に開発者 IRC 会議を行っています。会議開催予定は、開発者用メーリングリストでアナウンスされます。表 1 を参考に是非参加してください。

表 1 IRC 設定情報

サーバ	irc.linux.or.jp または irc.debian.or.jp
ポート番号	6667
チャンネル	#Vine
文字コード	UTF-8

過去の議事録は定例 IRC 会議 `http://trac.vinelinux.org/wiki/MonthlyIrcMeeting` で参照できます。(こちらでも次回の予定を掲示しています。)

なお、IRC チャンネルは原則として会議時間以外も存在し、雑談や他の開発者との交流を行うことができます。

開発者 IRC 会議の位置づけ

- 開発者用メーリングリストでの議論・決定を妨げるものではありません。
- 会議時間以外に重要な決定を行うことを禁止します。

2.2 開発者アカウントの発行

あなたの開発意欲などを確認できましたら、Project Vine から開発者アカウントの発行を打診します。この際、以下の情報をメールにて送っていただきます。

- アカウント名の候補

半角英数字の 3〜16 文字（先頭は半角英字）で 3 つ以上の候補を用意してください。

- GnuPG 公開鍵
  公開鍵サーバへ登録済みのものをアスキー形式でエキスポートしてください。
  送付いただいた GnuPG 公開鍵は、vine-keyring パッケージを通じて Vine Linux ディストリビューションに組み込まれます。
- GnuPG 公開鍵の Fingerprint（電子指紋）
  メールの署名に含めてください。Web ページや Blog を公開している場合は、プロフィールなどに記載してください。

可能であれば、すでに鍵を登録されている他の開発者と会って、公開鍵に署名してもらってください。

アカウントの登録が終わりましたら、作成したアカウント名とパスワードを返送します。このアカウントで Vine Linux 開発者向け Trac `http://trac.vinelinux.org/` と BTS `http://bts.vinelinux.org/` の編集が可能になります。

必ずパスワードを変更してください

発行されたパスワードをそのまま使い続けないでください。パスワードの変更は、Vine Linux 開発者向け Trac `http://trac.vinelinux.org/` で行えます。

Vine Linux 開発者向け Trac `http://trac.vinelinux.org/` にログイン `http://trac.vinelinux.org/login` 後、ユーザ設定 `http://trac.vinelinux.org/prefs` の Account `http://trac.vinelinux.org/prefs/account` ページでパスワードを変更してください。 ■

Trac 及び BTS とは?

Vine Linux 開発者向け Trac `http://trac.vinelinux.org/` は、Vine Linux の開発に関する情報を整理するために使用されています。

また、BTS `http://bts.vinelinux.org/` は、Vine Linux のバグや要望などを管理するためのシステムです。 ■

オフラインミーティングに参加しませんか?

不定期ですが、Vine Linux の開発者が主催するオフラインミーティングを行っています。これは、他の開発者と会う絶好の機会です。開発者用メーリングリストでアナウンスしますので是非参加してみてください。

実は、GnuPG 公開鍵があるだけでは「信頼」の担保はできません。それが本当に本人によって作成されたものであるか、証明する必要があります。GnuPG の認証局はどこにもありませんので、開発者同士が実際に会ってお互いの身分を確認し、お互いの公開鍵に署名しあう事で公開鍵の信頼性を高めます。

もし、他の開発者から公開鍵に署名してもらいたい場合は、運転免許証やパスポートなどの写真付き公的身分証明書と自分の GnuPG の鍵指紋を印刷またはメモしたものを持参してください。 ■

GnuPG とは

GnuPG とは、The GNU Privacy Guard の略で IETF が発行した RFC4880 `http://www.ietf.org/rfc/rfc4880.txt` で定義されている OpenPGP 規格の GNU プロジェクトによる実装です。データの暗号化と復号、データへの署名などを行う事ができます。

GnuPG の作成方法などの詳細は、付録付録 A や付録付録 B を参照してください。 ■

3　パッケージ作成のための基礎知識

3.1　更新パッケージの提供期間

Vine Linux 5.1 のように既に正式にリリースされたバーションでパッケージにセキュリティホールやバグが発見された場合に更新パッケージを提供するのは、次のメジャーバージョンリリースから 1 年後までとします。

Vine Linux 5.0 から提供期間を変更しました

Vine Linux 5.0 から、上記の定義になりました。Vine Linux 4.x の更新パッケージ提供終了は、2010 年 8 月 24 日までです。 ■

Vine Linux のメジャーバージョンとマイナーバージョンについて

Vine Linux のバージョン番号は、メジャーバージョンとマイナーバージョンをピリオド（.）で繋げたものとなっています。

Vine Linux 5.1 であれば、メジャーバージョンは 5、マイナーバージョンは 1 です。

なお、Vine Linux が提供する個々のパッケージにおいては、更にマイクロバージョンやビルド番号を付加しているものも存在します。 ■

3.2　VineSeed について

VineSeed とは、次期メジャーバージョンの開発過程を α 版として公開しているものです。基本的に開発のための環境として提供され、インストール媒体は提供されません。

VineSeed 環境の構築要領については、付録付録 C を参照してください。

3.3 収録可能なパッケージのライセンスについて

Vine Linux に収録可能なパッケージは、ライセンス的に明らかに再配布可能なものでなければなりません。例えば、Open Source Initiative `http://opensource.org/` がオープンソースと認めているライセンス `http://opensource.org/licenses/alphabetical` であれば、収録可能です。

再配布に許可が必要なパッケージは、必ず著作権者に許可を取ってください。また、商業的な配布を禁止しているものは基本的に収録できないと考えて下さい。問題があるパッケージを発見した場合は、事前連絡なく削除します。

また、Vine Linux では Vine Linux 5 以降、ライセンス以外での制約があるソフトウェア（例えば特許）については、ソースコードの再配布およびバイナリの配布を実施していません。そのようなソフトウェアのために self-build パッケージを導入しています。

もし、判断に迷うライセンスがあれば、開発者用メーリングリスト等でご相談ください。

self-build パッケージとは?

ライセンス以外での制約があるソフトウェアのソースコードを配布元からダウンロードし、コンパイルした上、インストールするという一連の手順を実行します。

通常のパッケージと同様に apt-get コマンドや Synaptic パッケージマネージャを使ってインストールを行うことが可能です。ただし、コンパイルが必要な分、通常のパッケージに比べてインストール完了までに時間がかかります。

3.4　パッケージが属するカテゴリについて

パッケージは、その性質によりいくつかのカテゴリに分類されます。カテゴリとは、apt リポジトリにおけるパッケージの分類単位です。

カテゴリへの分類は、Project Vine によって実施され、テスト目的のカテゴリ（testing および propose-updates）を除き、パッケージのアップデート時には意識する必要はありません。

main のサブカテゴリ

カテゴリ main は内部的に 3 つのサブカテゴリに分類されています。

3.5　リリース済み Vine Linux のメンテナンスポリシー

セキュリティホールやバグの修正、機能強化のためにリリース済み Vine Linux 用のパッケージを更新する必要がある場合、パッケージの種類によって以下のポリシーに従うものとします。

3.5.1　main

core

- 原則としてパッチによる対応とし、バージョンアップはしない。
- 著しくパッチの管理が困難、または upstream のメンテナンスに問題がある場合、API/ABI 完全互換である場合に限ってマイナーバージョンアップ可能。ただし、承認プロセスをへる必要がある。

lib 系

- 原則としてパッチによる対応とし、バージョンアップはしない。

表 2　カテゴリの種類

カテゴリ	対象パッケージ
main	Vine Linux を構成する上で必須のパッケージ。原則としてリリース時のバージョンに固定されます。
updates	セキュリティフィックスまたはバグフィックスにより更新された main カテゴリのパッケージ
plus	main には含まれないものの、主要なパッケージや一般的に多く使われているパッケージ。このカテゴリは定期的にメンテナンスされているパッケージが対象です。新規パッケージでメンテナンス宣言されたものはこのカテゴリが分類されます。
extras	特定のメンテナが存在しなくなったパッケージや、継続して更新されないパッケージ。新規パッケージのうち継続してメンテナンスする予定が決まっていないものも含みます。新規パッケージで特に宣言されない場合はこのカテゴリが分類されます。
nonfree	ライセンスや特許等の理由により、利用や再配布に制限のあるソフトウェアを含みます。
orphaned	メンテナンスが止まったものや長期間更新されないもの、不要になったパッケージ。古いパッケージの履歴を残すために存在するもので、そのまま利用すべきではありません。このカテゴリは apt repository では管理されないので、apt-get コマンドではインストールできません。
testing	セキュリティホールが発見されたパッケージを修正したものを updates へ収録する前にテスト目的でパッケージを収録。Security Watch Team `http://security.vinelinux.org/` のメンバー以外には非公開とする。
proposed-updates	バグの修正や機能強化を目的に更新したパッケージを updates へ収録する前にテスト目的でパッケージを収録

表 3　カテゴリ main のサブカテゴリ

サブカテゴリ	対象パッケージ
core	システムとして機能するために必須のパッケージ。インストールオプションで最小構成を選んだ場合でもこのサブカテゴリのパッケージはインストールされます。
main-cd	インストール CD に収録されるパッケージ
main	容量の関係でインストール CD に収録できなかったパッケージ

- ABI/API 完全互換である場合は、bugfix/enhancement も可能。ただし、承認プロセスをへる必要がある。

■server 系

- 原則としては、API, ABI 設定ファイルなどを含め完全な互換がない限りはバージョンアップはしない。
- 変更点を把握した上でユーザに迷惑がかからないことが確認できた場合は、BUGFIX/ENHANCEMENT としてバージョンアップも可能。ただし、承認プロセスをへる必要がある。

■user application 系　ユーザのデータや使用状態に問題がおこらない範囲であれば、バグの修正や機能強化のためのバージョンアップを行えます。

3.5.2　plus

- API/ABI 互換である場合には バージョンアップ可能
- 上記以外の場合は、
  - 依存されていないか、依存されているものが全て自分がメンテナである場合は バージョンアップ 可能。
  - 依存されているものを全て対応できるが、自分がメンテナじゃないものが含まれている場合は要相談。（メンテナ以外が update したい場合の方

法参照）

3.5.3 extras

依存関係を壊さない限りバージョンアップ可能です。継続してメンテナンスできる場合は、plus への移動を開発用メーリングリストで依頼してください。

3.5.4 nonfree

plus に従う。self-build 系は、原則として、so name の変更があるバージョンアップはしない。

3.6 SPEC ファイルのバージョン管理について

現在は VineSeed 向けパッケージのみですが、Subversion を利用して SPEC ファイルのバージョン管理を行っています。

管理されている SPEC ファイルは、http://trac.vinelinux.org/browser/projects/specs `http://trac.vinelinux.org/browser/projects/specs` で確認することができます。（この URI は、閲覧専用です。）

標準的な Subversion リポジトリに従って、branches,tags,trunk の 3 つのディレクトリを作成していますが、branches,tags の運用方法についてはまだ正式に決定されておらず、現在は trunk のみを利用しています。trunk ディレクトリの中は、まず、パッケージの頭文字一文字（大文字と小文字は区別する）で分類しています。それぞれのディレクトリの中に更にパッケージ名でディレクトリが作成され、各パッケージの SPEC ファイルが格納されています。

3.6.1 既存の VineSeed 向けパッケージをメンテナンスする場合

1. 項 5.1
2. SPEC ファイルのチェックアウト

```
$ svn co http://trac.vinelinux.org/repos/projects/specs/
    trunk/頭文字/パッケージ名
```

3. SPEC ファイルの修正
4. パッケージのビルド
5. 項 5.3
6. 修正のコミット

```
$ svn ci
```

ここで conflict が発生した場合は、他のメンテナと調整をしてください。

7. 項 5.4
8. 項 5.5

3.6.2　新たに VineSeed 向けパッケージを作成した場合

1. 項 5.1
2. SPEC ファイルの作成
3. パッケージのビルド
4. 項 5.3
5. 作成したパッケージの SPEC ファイルのみが入ったディレクトリをローカルコンピュータに作成
6. subversion リポジトリ上にそのパッケージ用のディレクトリを作成

```
$ svn mkdir --parents http://trac.vinelinux.org/repos/
    projects/specs/trunk/頭文字/パッケージ名
```

7.「3.6 - SPEC ファイルのバージョン管理について」で作成したディレクトリに移動し、svn リポジトリにインポート

```
$ svn import http://trac.vinelinux.org/repos/projects/specs/
    trunk/頭文字/パッケージ名
```

8. 項 5.4
9. 項 5.5

4　パッケージ作成環境の構築

4.1　環境設定

RPM パッケージの作成方法 `http://vinelinux.org/manuals/making-rpm.html` のパート I. 環境設定 `http://vinelinux.org/manuals/mr-setup.html` を読んで必要な環境設定を行ってください。

VineSeed での変更点

Vine Linux 5.x では、`~/.rpmmacros` で %_topdir をユーザ毎、設定しておりましたが、現在の VineSeed 環境では、`/usr/lib/rpm/macros` で同様の設定を行っており、ユーザ毎に設定する必要がなくなりました。 ■

4.2　vbootstrap パッケージのインストール

RPM パッケージの作成方法 `http://vinelinux.org/manuals/making-rpm.html` では、**rpmbuild** コマンドにより、バイナリパッケージを作成するように説明していますが、本書では、vbootstrap パッケージに含まれる **vbuilder** コマンドを利用してバイナリパッケージを作成します。

apt-get コマンドや Synaptic を利用して、vbootstrap パッケージをインストールしてください。

vbootstrap パッケージの詳細説明

vbootstrap は Vine Linux の基本システムを既存の Vine Linux システム上で作成するスクリプトです。rpm および apt を利用して、指定したディレクトリ以下に基本の rpm パッケージをインストールし、chroot できるようにします。

vbuilder は vbootstrap を利用して chroot 環境の構築し、その chroot の中で

パッケージをビルドします。パッケージが正しい BuildRequires 依存関係を持っているかを確認するために有用なシステムです。 ■

5　パッケージ作成から公開までの流れ

5.1　パッケージを作成・更新する前に

5.1.1　パッケージの存在を確認する

パッケージを作成する前に既にパッケージが存在しないか、確認をしてください。コマンドを使用してパッケージを検索する場合は、次のようにします。

```
# apt-get update
# apt-cache search packagename | sort
```

複数のバージョンを共存させる場合などパッケージ名にメジャーバージョンが含まれる場合があるので注意してください。また、pkgconfig のように実際の配布名 pkg-config と異なっているパッケージもあるので注意が必要です。パッケージの検索時に表示されるパッケージの概要説明も参考にしてください。

5.1.2　既存パッケージの場合（整理中）

既存パッケージを更新する場合は、パッケージのカテゴリに応じて以下の承認プロセスを踏む必要があります。この際、項 3.5 も念頭に置いてください。

セキュリティ問題の修正である場合

セキュリティ問題の修正である場合、メンテナの承認は不要（通知のみ）とします。

また、バグトラッキングシステムは、Security Watch Team `http://security.vinelinux.org/` のもの（非公開）を使用します。 ■

■main のパッケージの場合

1. メンテナである場合は、「5.1 パッケージを作成・更新する前に」へ進む。
2. パッケージを更新したい理由とその方法について開発者用メーリングリストに投稿する。
3. 1 週間以内にメンテナから異議が出た場合は、作業を中止し、対応について調整する。
4. パッチによる更新の場合は、項 5.2 へ進む。
5. バージョンアップが伴う場合は、BTS `http://bts.vinelinux.org/` に新規レポートを作成
6. 開発者用メーリングリストに提案し、合意を取る。
7. パッケージを作成・動作確認し、proposed-updates 用のディレクトリにアップロードする。
8. 動作確認レポートを「5.1 パッケージを作成・更新する前に」へのリプライとして登録してもらう。
9. 全てのアーキテクチャでの複数の動作確認レポートがあるか、「5.1 パッケージを作成・更新する前に」から 1 ヶ月経過した場合、正式にリリースする。

■plus のパッケージの場合

1. メンテナである場合は、「5.1 パッケージを作成・更新する前に」へ進む。
2. パッケージを更新したい理由とその方法について開発者用メーリングリストに投稿する。
3. 1 週間以内にメンテナから異議が出た場合は、作業を中止し、対応について調整する。
4. パッチによる更新の場合は、項 5.2 へ進む。
5. 開発者用メーリングリストに提案し、合意を取る。
6. 項 5.2 へ進む。

5.1.3　新規パッケージの場合

項 5.1.1 の結果、該当するパッケージが見つからなかった場合もまず開発者用メーリングリストで作業開始の旨を連絡してください。この際、パッケージの概要や開発元、ライセンスなどを明記してください。場合によっては、作成に異議がある場合もありますのでその後の返信に注意してください。

5.2　パッケージの作成・更新

5.2.1　SPEC ファイルの準備

パッケージの設計書である SPEC ファイルを準備します。

既存パッケージを更新する場合は、

```
$ apt-get source パッケージ名
```

などとして既存の SPEC ファイルを取得してください。(VineSeed の場合は、項 3.6 を参考に SPEC ファイルを取得してください。)

新規パッケージの場合は、SPEC ファイルを新たに作成します。この時、ファイル名は パッケージ名`-vl.spec` の形式にしてください。

SPEC ファイルの詳細については、RPM パッケージの作成方法 `http://vinelinux.org/manuals/making-rpm.html` の第 5 章 SPEC ファイルの記述 `http://vinelinux.org/manuals/make-spec.html` を参照してください。

5.2.2　ソースやパッチの配置

パッケージの作成に必要なソースやパッチを `%_topdir/SOURCES/` に配置してください。項 5.2.1 で **apt-get source** を利用した場合は、更新前のソースやパッチは既に配置されているはずです。

新たにパッチを作成した場合やソースのバージョンを更新する場合は、それらを先ほどのディレクトリに配置してください。

5.2.3 ソース RPM の作成

SPEC ファイルを元にソース RPM を作成します。以下のコマンドを実行してください。

```
$ rpmbuild -bs SPECファイル名
```

GnuPG による署名をするように設定している場合は、パスフレーズを求められますので入力してください。

```
パスフレーズの入力:
```

パスフレーズに問題がなければ、次のように表示されます。

```
パスフレーズは正常です。
署名の作成中: 1005
```

特に問題がなければ、書き込み完了の案内が表示されます。

5.2.4 バイナリ RPM の作成

まず、次のようにして chroot 環境（仮想のファイルシステム）に Vine Linux の基本システムをインストールします。（chroot を利用する関係上、vbuilder の実行には root 権限が必要となります。）

```
# vbuilder --version 5.1 --arch i386 clean build
```

5.1 や i386 といったオプション値は、ターゲット環境に併せて変更してください。この作業には、しばらく時間がかかります。

```
Making a build farm for 5.1_i386 done.
```

のように表示されたら、次のようにしてソース RPM から、バイナリ RPM を作成します。

```
# vbuilder --version 5.1 --arch i386 --unionfs clean build-rpm /
    path/to/hoge.src.rpm
```

特に問題がなければ、作成されたパッケージが、%_topdir/vbuilder/ 以下にコピーされます。(vbuilder の出力をよく確認してください。)

ビルドが途中で失敗した場合は、項 5.2.1 に戻って原因を修正してください。(BuildRequires の修正など)

vbuilder の詳細は、http://trac.vinelinux.org/wiki/VineBootstrap を参照してください。

5.3　パッケージのテスト

パッケージを作成したあとは、実環境にインストールしてみて一通り問題なく動作するか、試してください。

パッケージをインストールするには、次のコマンドを実行します。/path/to の部分は、パッケージファイルまでのパス、arch の部分は i386 や x86_64 といった使用環境にあったアーキテクチャ名に置き換えてください。

```
# rpm -Uvh /path/to/packagename.arch.rpm
```

パッケージがインストールできたら、メニューや端末エミュレータから、アプリケーションを実行してください。ライブラリであれば、依存するアプリケーションを実行します。

通常のアプリケーションであれば、全てのメニューを一通り使用して問題がないかを確認します。GUI のアプリケーションであっても端末エミュレータから、実行してみると不具合がある場合に端末に警告メッセージなどが表示される場合があります。特に実行時に必要なライブラリの漏れなどがないか確認してください。

不具合を発見した場合、可能であれば、開発元の BTS などに同様の不具合が報告されていないか、確認し、Patch が提供されていれば、それを適用するようにパッケージを再作成してください。

また、アンインストールのテストを行って以下のような問題がないか確認してください。

- %preun、%postun で指定したスクリプトが問題なく動作するか?

- 不要なディレクトリが残らないか?

依存するパッケージが多く、実際にアンインストールすることが現実的でない場合は、rpm や apt-get でシミュレーションを行います。

パッケージのアンインストールを行う-e とともに--test と-vv を組み合わせて使用します。rpm では、アンインストールのテストは一般ユーザ権限でも実行できます。

```
$ rpm -e --test -vv packagename
```

例 1　rpm によるアンインストールのシミュレート

apt-get で実際には、アンインストールを行わずシミュレートのみ実行する場合は、--simulate を利用します。rpm と違い、シミュレートであっても root 権限が必要となります。

```
# apt-get --simulate remove packagename
```

例 2　apt-get によるアンインストールのシミュレート

5.4　パッケージのアップロード

パッケージのアップロード先はお問い合わせください

現在のところ、パッケージのアップロード先は、開発者以外へ非公開としています。

アップロード先のアドレス、アカウント情報については、Vine@vinelinux.org mailto:Vine@vinelinux.org までお問い合わせください。 ■

パッケージのアップロードに先だって GnuPG による署名を行います。例 3 のようにして作成されたパッケージに署名してください。

```
$ rpm --addsign パッケージファイル名
```

例 3　パッケージへの署名

なお、vbuilder によるパッケージのビルド時に--sign オプションを付加することにより、署名を同時に行うことも可能です。

念のため、アップロードする全てのパッケージファイルに対して例 4 のようにしてパッケージへの署名の検証を行ってください。

この例では、アップロードするパッケージを一つのディレクトリに集めた場合を想定しています。

```
$ rpm -K *.rpm
gnome-panel-2.28.0-4vl6.src.rpm: (sha1) dsa sha1 md5 gpg OK
gnome-panel-2.28.0-4vl6.x86_64.rpm: (sha1) dsa sha1 md5 gpg OK
```

アップロードするパッケージファイル全てに gpg という文字列が含まれていることを確認してください。

例 4　パッケージへの署名の検証

署名に問題がなければ、lftp などの ftp クライアントを用いて、パッケージをアップロードします。アップロード先のディレクトリ構成は以下のようになっています。

- VineLinux
  - 5（Vine Linux 5.x 向けの main カテゴリに属するパッケージの proposed-updates 用）
- VinePlus
  - 4.0（Vine Linux 4.2 向けの plus カテゴリ等に属するパッケージアップロード用）
    - nonfree（同 nonfree カテゴリ等に属するパッケージアップロード用）
  - 5（Vine Linux 5.x 向けの plus カテゴリ等に属するパッケージアップロー

ド用）

* nonfree（同 nonfree カテゴリ等に属するパッケージアップロード用）

- VineSeed（nonfree を除く VineSeed 向けパッケージアップロード用）
  - nonfree（VineSeed 向け nonfree パッケージアップロード用）

署名を忘れた場合の通知について

パッケージへの署名を行わずにパッケージをアップロードした場合、apt リポジトリへの登録が拒否されます。

この場合、開発者用メーリングリストに「[VPMIRROR] NOT SIGNED PACKAGE UPLOADED」という件名のメールが自動送信されます。このメールに拒否されたパッケージファイル名が含まれていますので署名を行った上で再度、アップロードしてください。 ■

誤ったパッケージをアップロードした場合

誤ったパッケージをアップロードした場合、自分で削除することができません。

誤ったパッケージのアップロードに気づいた場合は、速やかに開発者用メーリングリストに削除依頼を投稿してください。 ■

5.5 パッケージ作成・更新のアナウンス

パッケージをアップロードしたら、開発者用メーリングリストにその旨を投稿します。

メールに含ませる内容

件名　upload: パッケージ名-バージョン-リリース番号

本文

- パッケージの Summary または Description

- アップロードしたファイルのリスト
- 更新内容（既存パッケージの場合）
- 依頼事項（必要により）
  - パッケージのテスト
  - 他のアーキテクチャ用パッケージのビルド

5.6　バグ報告への対応

時として BTS http://bts.vinelinux.org/ やメーリングリストに自分が作成したパッケージのバグが報告される場合があります。このような場合は、以下の手順を参考に対処してください。

1. レポートの担当者・状態を修正
   自分で対処する場合は、状態を「割当済み」に変更し、担当者を自分にしてください。
   時間がとれないなど何らかの理由で対処することが難しい場合は、他の開発者に相談してください。
   もし、バグ報告がメーリングリストに直接投稿されていた場合は、BTS http://bts.vinelinux.org/ で新たにレポートを作成します。
2. バグの再現性を確認
   まず、その問題が自分の環境で再現するかを確認します。
   再現しない場合、バグ報告者に具体的な操作方法や設定について改めて確認します。可能であれば、他の開発者の環境でも再現性を確認してもらいます。
   バグ報告者の操作や設定が明らかに間違っている場合は、その対処方法を明示した上でレポートの状態を「却下」にしてください。
3. バグの切り分け
   バグの再現性が確認できたならば、そのバグがパッケージングミスであるのか、パッチが原因であるのか、開発元のソースに由来するものなのか切り分

けます。
バグが開発元のソースに由来するものであった場合、可能であれば、既知のバグでないか確認の上、開発元にバグ報告を行ってください。(この際、修正パッチを提供すると喜ばれるかもしれません。)

4. パッケージの更新
   SPEC ファイルやパッチに問題がある場合、修正してください。
   また、開発元から修正パッチを入手することができた場合は、そのパッチが適用できるように SPEC ファイルを修正してください。
   この際、リリース番号を +1 してパッケージを作成し、テストの上、パッケージをアップロードしてください。
   パッケージアップロードの旨を入力し、レポートの状態を「確認待ち」に変更します。
5. バグ報告者のバグ修正確認報告
   バグ報告者からバグの修正を確認できた旨の報告があった場合、リリース済み Vine Linux 向けの main パッケージであった場合は、状態を「errata 待ち」にします。それ以外のパッケージである場合は、状態を「完了」とします。
6. errata `http://vinelinux.org/errata.html` の発行
   リリース済み Vine Linux 向けの main パッケージであった場合は、Project Vine に errata の発行を依頼してください。
   errata の発行が確認できたら、レポートの状態を「完了」とします。

BTS への投稿はに転送されます

BTS へバグ等が報告された場合、開発者用メーリングリストにもその内容とレポートへのリンクが送信されます。この際、[VineSeed:メールの通し番号] の後に [BTS プロジェクト識別子:バグ ID] のような文字列が付加されます。開発者用メーリングリストでバグ等の報告に気づいた場合は、開発者用メーリングリストへの直接の投稿なのか、BTS `http://bts.vinelinux.org/` への投稿なのかに注意してください。 ■

表 4　BTS プロジェクト識別子の種類

BTS プロジェクト識別子	目的
VineLinux	Vine Linux に関するバグの報告を受け付けます。
wishes	Vine Linux に関する要望を受け付けます。

付録 A　gpg コマンドの使用方法

A.1　GnuPG 鍵対の生成

端末でコマンド **gpg --gen-key** を実行します。

```
$ gpg --gen-key

gpg (GnuPG) 1.4.6; Copyright (C) 2006 Free Software Foundation,
    Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

すきな鍵の種類を選択してください:
(1) DSAとElGamal (既定)
(2) DSA (署名のみ)
(5) RSA (署名のみ)
どれにしますか? 1
```

ここでは既定の 1 を選択してください。

```
DSA鍵対は1024ビットになります。
ELG-E keys may be between 1024 and 4096 bits long.
どの鍵長にしますか? (2048)
```

ここでも既定の 2048 以上を推奨します。通常は既定でかまいません。

```
要求された鍵長は2048ビット
鍵の有効期限を決めてください。
           0 = 無期限
```

```
         <n>  = 有効期限 n 日間
         <n>w = 有効期限 n 週間
         <n>m = 有効期限 n か月間
         <n>y = 有効期限 n 年間
鍵の有効期間は? (0)
```

ここでは今作成している鍵の有効期限を設定します。通常は永久に有効となる既定値 0 で構いません。

```
鍵は無期限です
これでいいですか? (y/N) y
```

ここまで問題なければ y で続行してください。

```
あなたの鍵を同定するためにユーザIDが必要です。
このソフトは本名、コメント、電子メール・アドレスから
次の書式でユーザIDを構成します:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

本名: Your Name
電子メール・アドレス: name@example.com
コメント:
```

鍵のユーザ ID となる本名、電子メール・アドレス、コメントを聞かれますので、答えてください。半角英数です。メールアドレスは有効なものを入力してください。コメントは不要であれば入力しなくても構いません。

```
次のユーザIDを選択しました:
    "Your Name <name@example.com>"
名前(N)、コメント(C)、電子メール(E)の変更、またはOK(O)か終了(Q)?
    O
```

提示されたもので問題なければ "O" で続行してください。

```
秘密鍵を保護するためにパスフレーズがいります。
```

パスフレーズを聞かれますので、この鍵に対するパスフレーズを設定してください。パスワードとは違い長めの文章が望まれます。入力後、表示のとおり乱数を生成しますので、しばらく時間がかかります。

```
次のユーザIDを選択しました:
gpg: このセッションでgpgエージェントは無効です
今から長い乱数を生成します。キーボードを打つとか、マウスを動かす
とか、ディスクにアクセスするとかの他のことをすると、乱数生成子で
乱雑さの大きないい乱数を生成しやすくなるので、お勧めします。
+++++++++++++++.+++++.+++++++++++++++++++++++++++++++++.++++++++++(
    以下省略)
今から長い乱数を生成します。キーボードを打つとか、マウスを動かす
とか、ディスクにアクセスするとかの他のことをすると、乱数生成子で
乱雑さの大きないい乱数を生成しやすくなるので、お勧めします。
++++++++++++++++++.++++++++++++++++++++++++++..+++++++++++++++(
    以下省略)
gpg: key 4BF0CEAC marked as ultimately trusted
公開鍵と秘密鍵を作り、署名しました。

gpg: 信用データベースの検査
gpg: 3 marginal(s) needed, 1 complete(s) needed, classic trust
    model
gpg: depth: 0  valid:   3  signed:  24  trust: 0-, 0q, 0n, 0m, 0f
    , 3u
gpg: depth: 1  valid:  24  signed:  47  trust: 20-, 0q, 0n, 0m, 4
    f, 0u
gpg: depth: 2  valid:  17  signed:  19  trust: 17-, 0q, 0n, 0m, 0
    f, 0u
pub   1024D/4BF0CEAC 2008-10-01
Key fingerprint = 7307 7E27 C02A 4B3C 1062  E4B2 0C6D C326 4BF0
    CEAC
uid                  Your Name <name@example.com>
sub   2048g/1FBBC15E 2008-10-01
```

以上で、GnuPG の鍵対が生成できました。

A.2　鍵束内の鍵一覧を表示

gpg --list-key で自分の持っている鍵束（keyring）を表示することができます。この中には自分自身の公開鍵だけでなく、（登録してあれば）他人の公開鍵も含まれます。鍵束は~/.gnupg/pubring.gpg に保存されています。

```
$ gpg --list-key
pub   1024D/4BF0CEAC 2008-10-01
uid                  Your Name <name@example.com>
sub   2048g/1FBBC15E 2008-10-01
```

A.3 鍵の Fingerprint（電子指紋）を表示

gpg --fingerprint ユーザID|鍵ID でそのユーザ ID または鍵 ID の fingerprint を表示することができます。GnuPG 公開鍵のハッシュ値で、公開鍵の正当性をチェックするために使用します。鍵 ID は項 A.2 の例では、0x4BF0CEAC となります。ID を指定しない場合は鍵束内のすべての鍵の fingerprint が表示されます。

```
$ gpg --fingerprint 0x4BF0CEAC
pub   1024D/4BF0CEAC 2008-10-01
Key fingerprint = 7307 7E27 C02A 4B3C 1062  E4B2 0C6D C326 4BF0
    CEAC
uid                  Your Name <name@example.com>
sub   2048g/1FBBC15E 2008-10-01
```

A.4 公開鍵のエキスポート

公開鍵を渡すためには鍵束から特定の鍵だけ取り出す必要があります。通常は取扱いがしやすいように以下のようにアスキー形式でとりだします。

```
$ gpg -o pubkey.asc --export -a 0x4BF0CEAC
$ cat pubkey.asc
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.6 (GNU/Linux)

mQGiBEjjIOgRBAC0gD6KZW83WTzj/m4MUrbrRALkgUr0jVIN1+NsipLFhgkT70vJ
Theqh7rwy0ujLecjp+5wY/uh/BfftkKylLiaERWxO8j8wBNa+337LZeRBqOEzndm
22P8m4rUrDSzj5i1It3+GhpYE0OC0HA5GyuzbLzLvLkBy4rrjGpUXbQU+wCg5fcE
S7jii059P25wnfFU53Lz+JcEAK4HaYHS4aZlFAUsKZCj9vzY+4a7m1MbOQhL4WcH
8DfFL7y/8EXzTIDVF8US8i+A48QkhSw58Jeg1LNhiuQYoqVKn9NtLembMyun93Af
(途中省略)
EDkY0dQclHefP1XN/fyTDsVjU8+MAvbYieqj/7rSlgVqAKw4qSXYXqPtIj0YhQyG
```

```
UIt2AxOB0YpEJ7bN6mIbbJQwdOvIiEkEGBECAAkFAkjjIO0CGwwACgkQDG3DJkvw
zqzrgwCeO8YaYOER+1uQoj8PdEiCQ7AGPj8AoNQYhMJkMIEapRHIFiCKHMumkNvY
=gs65
-----END PGP PUBLIC KEY BLOCK-----
```

A.5　公開鍵サーバへの登録

hkp プロトコルに対応しているサーバであれば、次のコマンドで登録することができます。

```
$ gpg --keyserver pgp.mit.edu --send-keys 0x4BF0CEAC
gpg: 鍵0x4BF0CEACをhkpサーバーpgp.mit.eduへ送信
```

A.6　参考リンク

- The GNU Privacy Guard - GnuPG.org `http://www.gnupg.org/`
- GNU Privacy Guard 講座 `http://gnupg.hclippr.com/`

付録 B　GnuPG の GNOME フロントエンド Seahorse の使い方

VinePlus の seahorse を使うと GnuPG 鍵に関するほとんどの操作をマウス操作で行うことができます。

seahorse をインストールするとアプリケーション → アクセサリ → パスワードと暗号鍵で起動することができます。

スクリーンショットは Vine Linux 5.1 のものを使用しています

付録付録 B で使用しているスクリーンショットは、Vine Linux 5.1 向けの seahorse のものです。

VineSeed 用の seahorse とは、若干、違いますが基本的な操作方法は同じです。 ■

B.1 GnuPG 鍵対の生成

1. 初めて Seahorse を起動した場合は、図 2 のように初回起動時のオプションが表示されますので新規ボタンをクリックします。

図 2　初回起動時の Seahorse

もし、初回起動時のオプションが表示されない場合でもファイル → 新規（**Ctrl-N**）で GnuPG 鍵対の生成を始めることができます。

2. 図 3 のようなダイアログが表示されるので PGP 鍵を選択して続行をクリックします。
3. 氏名と E-メールアドレスを半角で入力します。コメントは、不要であれば入力しなくても構いません。

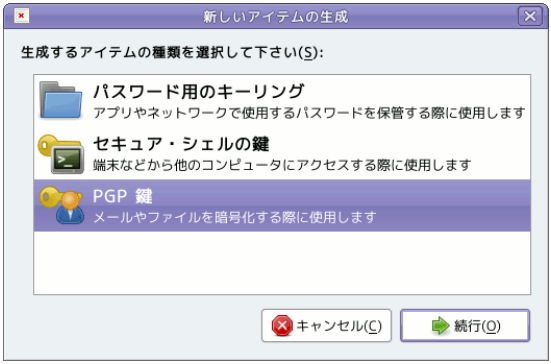

図 3　新しいアイテムの生成

図 4　新しい PGP 鍵

拡張オプションを展開すると暗号化の種類・鍵の長さ・有効期限の変更が行えますが、通常はデフォルトのままで構いません。

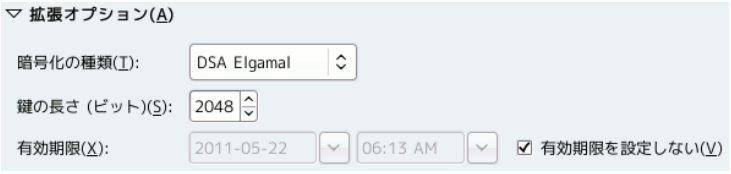

図 5　拡張オプション

入力した内容を確認のうえ、生成をクリックしてください。

4. パスフレーズを 2 回入力します。パスワードを考える時と同様、破られにくいものを考えてください。パスワードと違い、半角スペースを使用することができます。

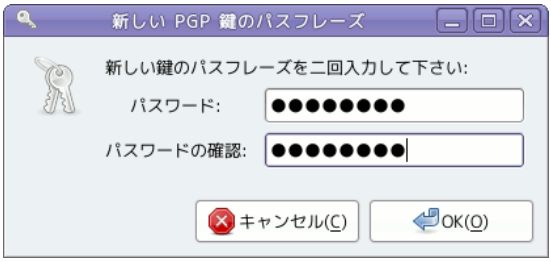

図 6　新しい PGP 鍵のパスフレーズ

入力が完了したら、OK をクリックします。

5. 鍵の生成には、しばらく時間がかかります。

図 7　鍵の生成中

鍵の生成が終了すると個人の鍵の一覧に追加されます。

B.2　鍵の Fingerprint（電子指紋）を表示

鍵の Fingerprint（電子指紋）など詳細を確認したい場合は、確認したい鍵を選択し、以下のいずれかを実行します。

- ツールバーのプロパティボタン

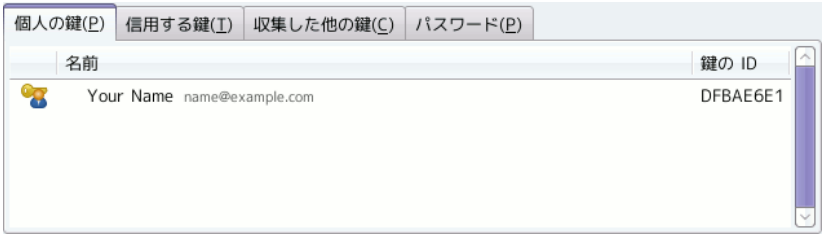

図 8　個人の鍵

- 鍵を右クリックしてプロパティ

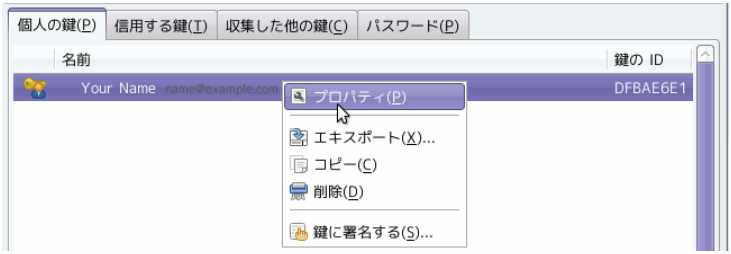

図 9　ポップアップメニューからのプロパティの選択

すると図 9 のようなダイアログが表示されます。

鍵の Fingerprint（電子指紋）は、詳細のページで確認することができます。

指紋と書かれた所にある次のような文字列が、Fingerprint（電子指紋）です。

```
7362 2350 47E0 8941 D689
A1F5 F66D AA9C DFBA E6E1
```

この文字列は、マウスでドラッグすることにより図 11 のように選択状態にすることができ、右クリックで表示されるポップアップメニューから、コピーすることが可能です。

図 10　鍵のプロパティ（所有者）

B.3　公開鍵のエキスポート

公開鍵をエキスポートするには、エキスポートしたい鍵を選択し、以下のいずれかを実行します。

- ファイル → エキスポート
- ツールバーのエキスポートボタン
- 鍵を右クリックしてエキスポート

すると図 13 のようなダイアログが表示されるのでファイル名と保存先のフォルダを選択し、保存ボタンを押してください。（ファイル名にデフォルトで空白が

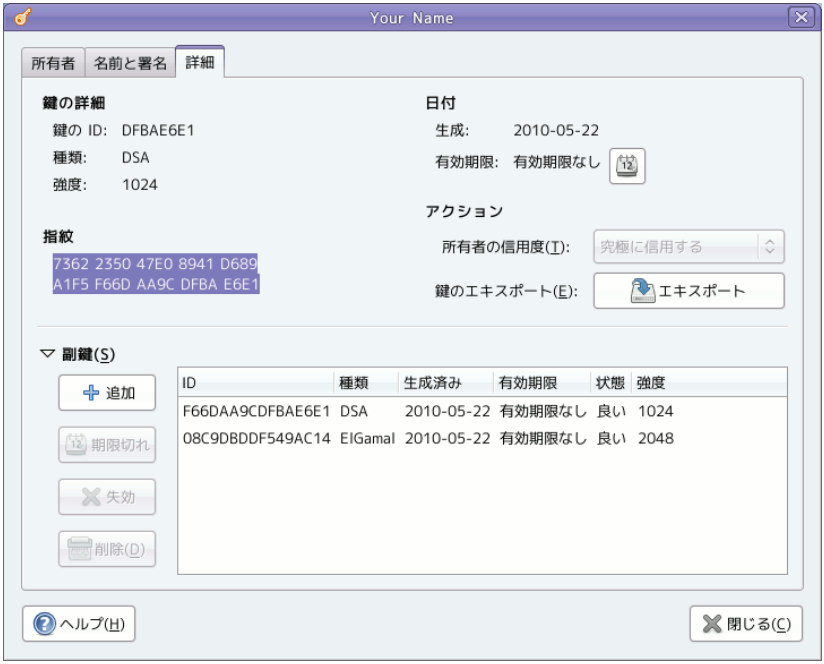

図 11　鍵のプロパティ（詳細）

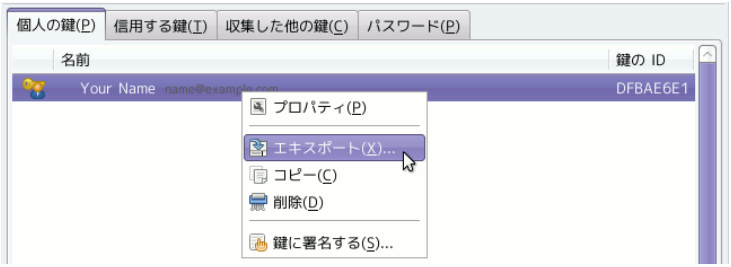

図 12　ポップアップメニューからのエキスポートの選択

入ってしまいますが、空白を含めないようにしてください。）

図 13　公開鍵のエキスポート

B.4　公開鍵サーバへの登録

公開鍵サーバへ鍵を登録するには、以下の手順を実行します。

1. 公開鍵サーバへ鍵を登録するには、鍵を選択し、メニューからリモート → 鍵の公開と同期を実行します。
2. 図 14 のような画面が表示されるので鍵サーバボタンを押します。

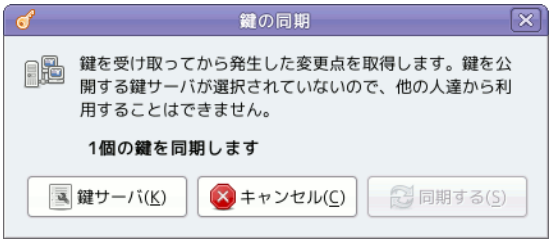

図 14　鍵の同期

3. 鍵の公開先のドロップダウンリストから、任意の鍵サーバを選択します。
4. 閉じるボタンを押します。
5. 同期するボタンを押します。
6. 公開鍵サーバとの通信が終了するまで図 17 のように進捗状況が表示されます。
   特にエラーメッセージが表示されなければ、登録完了です。

図 15　設定

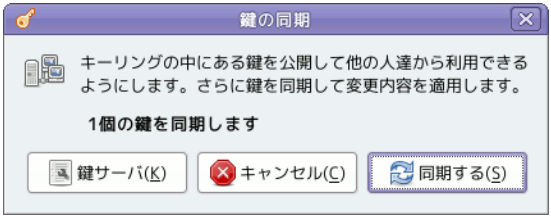

図 16　鍵の同期（公開先設定後）

付録 C　VineSeed 環境の構築要領

VineSeed はアルファ版です

VineSeed はアルファ版であり、様々な問題があることを御了解ください。

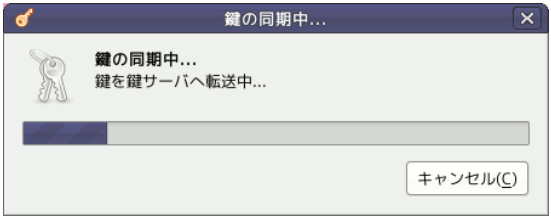

図 17　鍵の同期中...

特にリリースバージョンからのアップグレードや、逆にリリースバージョンへのアップグレードに支障がでる可能性があります。またパッケージ構成もダイナミックに変わる可能性があります。正常にインストール出来ない場合や、アップグレードで重要なパッケージが動かなくなる可能性もあります。単純に利用のみの目的ではお使いにならないことを推奨します。 ■

以下に VineSeed 環境構築の一例を記述します。この例では、コンソールベースでの作業がメインとなります。日本語出力を抑制するために、適宜、言語を変更するなどしてください。

```
$ export LANG="C"
```

また、ほとんどのコマンドの実行に root 権限を必要とします。

1. Vine Linux の最新リリースバージョンを「ベースシステム」でインストールします。(パッケージグループの選択の際には、「基本構成」を選択します。) インストール後、初回起動時にはネットワークデバイスが無効となっているため、nfs 関連サービスの起動に失敗し、ログインプロンプトが出るまで時間がかかります。(Vine Linux 5.2 で解消される予定)
2. ログイン後、ネットワークを起動します。

   ```
   # /etc/init.d/network start
   ```

   次回起動時以降もネットワークが起動するように以下のコマンドを実行します。

```
# /sbin/chkconfig network on
```

3. apt-get コマンドを利用して Errata を適用します。

```
# apt-get update
# apt-get upgrade
```

4. 再起動後、システムが正常に動作するのを確認します。
5. APT によるパッケージの取得先を修正するため、ディレクトリ /etc/apt/sources.list.d/ 以下の全てのファイルを vim 等のエディタなどで編集します。

   例えば、Vine Linux 5.1 の /etc/apt/sources.list.d/main.list には、次のような行が含まれています。

```
# (master)
rpm     [vine] http://updates.vinelinux.org/apt 5.1/$(ARCH)
    main updates
rpm-src [vine] http://updates.vinelinux.org/apt 5.1/$(ARCH)
    main updates
```

   #で始まる行はコメントとして扱われ、設定には影響しません。

   この例では、「5.1/$(ARCH)」となっている部分を全て「VineSeed/$(ARCH)」に変更します。

   これら設定ファイルの詳細は、次のようにして sources.list のマニュアルを参照してください。

```
$ man sources.list
```

6. APT を更新します。

```
# apt-get update
# apt-get install apt libxml2
```

7. グラフィカル環境が必要な場合は、以下の例を参考にしてインストールしてください。

```
# apt-get install task-xorg-x11 task-gnome
```

task-gnome の部分は、GNOME を使う場合です。
好みの日本語入力システムもインストールしておくと良いでしょう。

8. システムをアップグレードします。

```
# apt-get update
# apt-get dist-upgrade
```

9.「付録 D VineSeed 環境の構築要領」で

```
E: Unable to correct problems, you have held broken packages
    .
```

のようなメッセージが出た場合は、個別にパッケージをインストールします。

```
# apt-get install package1 package2 ...
```

10.「付録 D VineSeed 環境の構築要領」でグラフィカル環境をインストールした場合は、/etc/inittab を修正します。

```
# cp -p /etc/inittab /etc/inittab.org
# cp -p /etc/inittab.sysv /etc/inittab
```

必要に応じ、デフォルトのランレベルを編集してください。

11. システムを再起動します。

依存関係を解決できない場合には

「付録 D VineSeed 環境の構築要領」のような状況に遭遇した場合、BTS http://bts.vinelinux.org/ で同様の問題が報告されていないか確認の上、新規レポートを作成してください。

新規レポートを作成する際は、分かる範囲で以下のような情報を含めてください。

- 使用しているコンピュータの詳細（CPU、メモリ容量など）
- 入力したコマンド・操作の流れ
- 具体的なエラーメッセージと前後の出力

付録 D　開発を継続できなくなった時には

事情により開発を継続できなくなった場合には、開発者用メーリングリストに連絡をお願いします。

特に理由を説明する必要はありませんが、メンテナンスを担当していたパッケージやドキュメントを付記してください。