

2007年冬号

あんどきゅめんとつど でびあん



東京エリアDebian勉強会著

目次

1	Debconf 参加報告	2
2	Debconf 2007 各種討議内容	6
3	OSC-Kansai 参加報告	12
4	将来の Debconf	16
5	cdn.debian.or.jp の紹介	18
6	Debian GNU/kFreeBSD のインストール	22
7	Exim 再発見	25
8	あなたの知らないかもしれない apt-xxx	28
9	DebTorrent に触ってみた	32
10	live-helper	34
11	Debian パッケージの作り方 (1) – 20 分で作る Debian パッケージ –	43
12	Debian パッケージの作り方 (2) – debian/rules を読む –	45
13	Debian パッケージの作り方 (3) – dpatch の使い方 –	47
14	Debian パッケージの作り方 (4) – ライブラリ編 –	49
15	HP ML350G5 Debian etch 動作確認	51
16	HP ML110G4 Debian etch 動作確認	57
17	Debian Weekly News trivia quiz	61
18	Debian Weekly News 問題回答	65
19	索引	66

『あんどきゅめんとっど でびあん』について

本書は、東京周辺で毎月行なわれている『東京エリア Debian 勉強会』で使用された資料・小ネタ・必殺技などを一冊にまとめたものです。収録範囲は勉強会第 29 回から第 34 回まで。内容は無保証、つっこみなどがあれば勉強会にて。

1 Debconf 参加報告

岩松 信洋

1.1 Debconf とは

2007 年度の Debconf は 6 月 13 日から 6 月 23 日まで、英国スコットランドのエジンバラで行われました。日本からは、上川 純一、矢吹 幸治、岩松 信洋が参加しました。

1.1.1 Debconf の歴史・経緯

Debian Conference <http://debconf7.debconf.org/> は Debian の開発者たちが一同に介するイベントです。通常顔をあわせることのないメンバーたちが一同に介し友好を深め、技術的な議論を戦わせます。過去の開催履歴を見ると表 1 のようになります。

表 1 歴代の Debconf 参加者推移

年	名前	場所	参加人数
2000	debconf 0	フランス ボルドー	
2001	debconf 1	フランス ボルドー	
2002	debconf 2	カナダ トロント	90 名
2003	debconf 3	ノルウェー オスロ	140 名
2004	debconf 4	ブラジル ポルトアレグレ	150 名
2005	debconf 5	フィンランド ヘルシンキ	200 名
2006	debconf 6	メキシコ オアスタベック	300 名
2007	debconf 7	英国スコットランド エジンバラ	約 400 名

1.1.2 Debconf 2007

2007 年度の Debconf の会場はエジンバラ大学の学生会館 Teviot を活用しました。専用のネットワーク回線をはりめぐらせ、無線 LAN ネットワークもはりめぐらせました。

また、Teviot は 夜 10 時に閉鎖する必要があったので、夜の会場 (night venue) というものも準備されました、現在売り物件となっている使われていない教会を使用し、ハックラボにしました。パイプオルガンなどがあり、風情がありました。パイプオルガンはもともと壊れていたのですが、Debconf の会期中に修復され、演奏会が催されました。

宿泊は会場から徒歩 5 分程度に位置する Budget Backpackers と Cowgate hostel という二つのホステルに分散して行いました。

1.2 スコットランド/エジンバラ

1.2.1 行き方

日本からエジンバラまでは、直行便がありません。パリ経由か、ロンドン経由等で一回トランジットが必要です。距離は約 10000km。飛行時間は約 14 時間かかります。上川、岩松組はパリの シャルル・ド・ゴール国際空港経由、矢吹はヒースロー経由で入国しました。

1.2.2 会場

会場は、エジンバラ大学の建物の一部である Teviot という名前の建物を借り切り、開催されました。

参加者はふたつのホテルに分散して宿泊していたのですが、それらのホテルから歩いて 10 分ほどのところにあります。



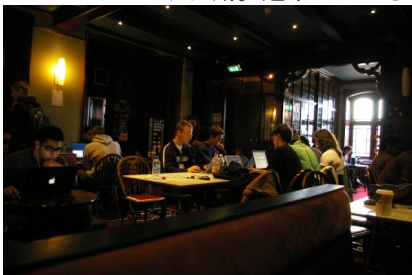
- Upper Talk Room: メイン用。250 人ほど入ることができます。



- Basement Talk Room
サブ用。50 人ほど入ることができます。
- Lower BoF Room
BOF 用。20 人ほど入ることができます。
- Upper BoF Room
BOF 用。20 人ほど入ることができます。
- Hacklab 1: ハック用。



- Hacklab 2: ハック用。通常はバーらしいです。



- Night venue: 廃墟と化した教会。パイプオルガンがあったりします。夜の 22 時以降は Teviot を使うことができないのでここを借りてみんなでハックしたり、話し合ったりしました。



1.3 スケジュール

16 日の Debian Day で Debian Conference は開始し、23 日まで毎日いろいろな予定がくまれていました。20 日だけはカンファレンス参加者で day-trip を実施しました。

スケジュールは ruby-on-rails で実装された pentabarf システム (図 2) で管理していました。スケジュールには随時変更がかり、IRC bot での通知がなかったら誰も状況においつけなかったでしょう。

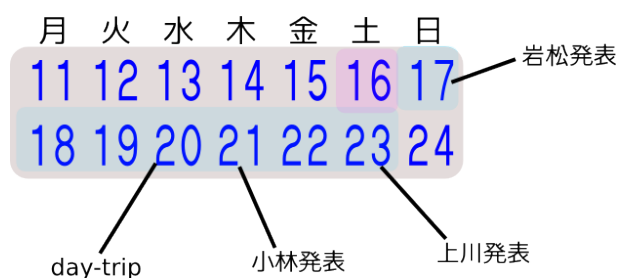


図 1 全体スケジュール

1.4 主となった討論

1.4.1 組込み系についての白熱した議論

ARM EABI の導入が大きなトピックです。日本から SuperH の話題ももっていきました。マインドシェアがおおきくなっているようです。また、組込関係の対応を Debian で行うための議論も行われました。New DPL の Sam Hovevar が組込み関係に興味があるようなので、いままで停滞していた組込み関係による成果のマージが加速するかもしれません。

1.4.2 バージョン管理システムとソースコード管理の話

git の利用方法のチュートリアルや、arch を例にとつての Debian ディストリビューションのフォークをメンテナンスするためのソースコード管理のやりかたについての紹介がありました。git などの普及により分散 SCM が普及し、ソースコードの管理のワークフローに影響しており、再考が必要だという風潮が見られました。

特に ubuntu でソースコード管理を見直しており、bazaar を中心としてブランチを dpatch のパッチファイルに変換したりするツールなどのインフラがととのい始めているということが大きいようです。

1.4.3 翻訳についての議論

翻訳関係の話が毎日行われました。毎日議論を重ね、議論した結果を毎晩ドキュメントを修正していました。また、時期リリースの lenny までに、翻訳のインフラやドキュメント整理を行う予定だそうです。

小林さんの翻訳関係のインフラに関するセッションがあったのですが、小林さんが来られなかったのが、上川さん



図 2 pentabarf 画面

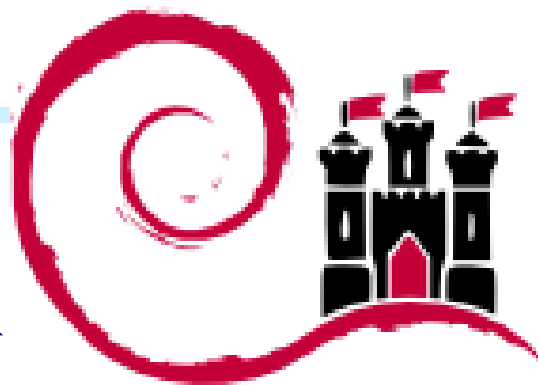
が代理で BOF を行いました。各国で使用されているツールの紹介などがありました。日本でも導入を検討をする必要がありそうです。

1.4.4 Daytrip

Debconf では一日、参加者で旅行をするというイベントがあります。今回の Debconf では Rotheway (Bute 島) でまったりとピクニックをしました。Rotheway への移動は、Edinburgh から Glasgow へ電車で移動し、Glasgow からさらに電車で Wemyss Bay へ移動します。Wemyss Bay は Rotheway 行き専用の舟着き場で、そこから船に乗って Rotheway に移動しました。

Rotheway の町は島で、特になにもないところです。ほとんどの建物は売出中で、裁判所の建物も売りに出ていました。財政がやばそうな感じです。建造物としては、教会や、バイキングの侵略の際に戦った城がありましたが、修復中でしかも工事は止まっていました。山の奥へ1時間ほど歩くと、湖があり、大抵の参加者はその湖でピクニックをしたり、ボードに乗って遊んでいたようです。





2 Debconf 2007 各種討議内容

岩松 信洋、矢吹 幸治、上川 純一

2007 年度の Debconf は 6 月 13 日から 6 月 23 日まで、スコットランド、エジンバラで開催されました。2007 年度の Debconf の討議内容を以下にまとめます。

2.1 6 月 16 日の発表内容

2.1.1 simple-CDD

CDD を簡単に準備できる仕組み。reprepro を使ってミラーを作成している。debpartial などとはうまくいかないことが多かったらしい。会場からはなぜ aptitude とかを利用しないのかという質問は出ましたが、そこまで検討していない、とのことでした。

--qemu オプションを指定したらイメージを作成して qemu でテストするところまでしてくれるそうです。build-simple-cdd コマンドを使えば簡単に CDD がつくれるそうです。

2.1.2 64studio

Debian ベースの音楽関係のソフトウェアを収録した CDD(Common Debian Distribution) です。各音源を組み合わせ、音を組み上げていく Jack というプログラムの説明などをしていました。イコライザーとして Jamin を利用し、出力の周波数特性をフリーハンドで変えることができるデモをしていました。また、PC のキーボードをつかって、パイプオルガンシミュレータ aeolus の (鍵盤の) キーボードを使って演奏するデモもしていました。

2.2 6 月 17 日の発表内容

2.2.1 Welcome talk

開会の挨拶。スポンサーと開催にかかわってくれた人たちへの感謝のコメントを行いました。非常にシンプルに終わった開会式でした。

2.2.2 About porting SuperH for Debian

Renesas 社製 CPU SuperH の Debian へのポーティングの話でした。SuperH を Debian にポーティングしている途中経過と現在発生している問題、および今後の課題について話し合いました。組込み関係の人が参加してくれていたが、みな ARM にかかわっているので直接の支援は難しそうな印象を受けました。

2.3 6月18日の発表内容

2.3.1 bugs.debian.org and debbugs

BTS の開発についての進捗報告でした。いろいろな機能が追加されているのですが、歴史的経緯で done 状態の遷移とバージョントラッキングで問題が解決しているかどうかという状態の遷移に整合性がとれないようになっているという話題がでました。この部分については互換性をなくしてでも解決してよいのではないかと、という議論をしました。また、SOAP インタフェースの新しい機能の紹介などもありました。

2.3.2 Embedded Debian

Debian の組み込み向けプロジェクト Emdebian の話です。今まで行ってきた方法の説明と結果を報告し、今後の方向性について話しをしました。現在の ftp-master 達はクロスビルドは受け付けてくれません。この問題を解決するためにパッケージにタグを付けたりして対策する予定とのことです。Embedded 用のツールも用意しており、これらを使ってクロスビルドできるようになりました。しかし、全てのパッケージはチェックできず

```
make check
```

や

```
make nodocs  
make nocheck
```

を使うように修正する必要があることを提案しました。その他の問題は後日行われた BOF で議論されました。

2.3.3 Debian Live

Debian のサブプロジェクトとして活動しています。ツールは Ubuntu の casper から fork して作成したものです。国際化も国コードやキーボードを入れるようにするみたいです。自分で会社を興して、キオスクシステムのために開発をしたとのこと。また、usb ブートなどもサポートしているので、コンピュータ本体に情報が残らなく、銀行などからも引合があると話していた。こちらから i18n に関連した質問をしたら、日本の市場に興味があるようで、あとで、すでに作ってあった日本用の CD イメージを見せてくれました。

<http://download.webconverger.com/i18n/jp/webc-2.21.jp.iso> からダウンロードできます。

2.3.4 Debian Armel Port

Armel の Debian へのポーティングの話でした。

2.3.5 OpenStreetMap

Free な地図をつくるためのプロジェクトの話です。Google Map がすでに存在していますが、DFSG-Free ではないので、GPS 等を使って、自由に扱える地図をつくるということが目的です。ライセンスは Creative Commons を採用しています。

<http://www.openstreetmap.org>

2.3.6 Wacky Ideas II

Wacky Ideas というのは、こんな凄いこと考え付いちゃったぜというのを、ディスカッションしてまともなものにしていく、ブレインストーミング系のセッションです。

口火をきったのは、Ian Jackson による upstream, ディストリビューション, 派生ディストリビューションで無駄なことしていないか? これらを含むような VCS (Version Control System) って作れないか? って話。

次は、Ubuntu の言語パックのように、翻訳 deb を作るべきかという話題でした。この方法は翻訳の部分が小さいのであればオーバーヘッドが大きすぎる。そのためパッケージをまとめて扱うことになるだろうということでした。極端

なやり方として Ubuntu は、(1 つで) 100MB のパッケージとして対処しているということです。

2.4 6月19日の発表内容

2.4.1 Debian virtualization support

Debian での仮想化ツールサポートの話です。あらゆる仮想化のシステムをサポートし、それらをサポートするための独自のツールを開発していることを紹介しました。また、仮想化ツールを使った、Debian パッケージのメンテナンスの話をし、ユーザー側だけでなく、開発者側から使い易くするために行っている活動をアピールしていました。

2.4.2 From Concept to Concrete

ハードウェアの企画、提案、そしてできるまでを、Daniel Silverstone と Vincent Sanders がコント風で行っていました。自分たちのハードウェア設計会社の紹介をしたかっただけなのかもしれないです。

2.4.3 PC Install and Backup Management

発表者が来れなかったので、集まったみんなでディスカッションして進めることになりました。まず、みんなどんなツールを使っているかという話題になりました。

unstable に、boxbackup というソフトウェアがあるから使っているという人から、mondo, rsync, cpio, tar などを使っている人もいました。また、サーバ上でシステムをバックアップするために lvm-snapshot を使わないといけないんだという人は、Mysql のデータベースの同期機能を利用していました。

つぎに話題は、d-i (Debian Installer) を使って、大量にインストールする話題になりました。preedit を使って同じようにインストールする手法や、FAI を使う方法などができました。

2.4.4 Debconf 9 Presentations

Debconf 9 のプレゼンテーションです。Debconf 8 は アルゼンチンに決定していますが、Debconf 9 はまだ未定です。このセッションでは、Debconf 9 の立候補地域によるプレゼンテーションを行いました。現在のところ、立候補地はスペインの Extremadura のみです。食事施設や宿泊施設が充実しており、プールもあるとのこと。この流れだと Debconf9 は Extremadura に決まりそうです。詳細は <http://wiki.debian.org/wiki/Extremadura> からみることができます。

2.4.5 Emdebian BOF

Debian の組込向けプロジェクトである Emdebian の BOF でした。uClibc のサポートおよび、busybox をベースにしたユーザーランドのサポートについて議論しました。既にサポートの体制を整えるようにしているが、uClibc をサポートしてしまうと、uClibc 用のバイナリを作成する必要があるので、バイナリの数が増えちゃいます。この問題をどのように解決していくか、今後議論を重ねていくとのこと。

Busybox の話は、busybox をベースにした場合、動くパッケージと動かないパッケージが出てしまいます。これは busybox をベースとして使用しているため発生する問題です。組込みでは busybox の base system を考える必要があるため、現在どのパッケージが動かないのかを調査している段階です。調査した結果は wiki にまとめられています。 <http://wiki.debian.org/EmdebianRootfs>

2.4.6 Keysigning Party

今年は、キーサインに集まった人達を 4 つのグループに分けて、各グループを大きめの部屋で分割してキーサインを行いました。全員がキーサインをするには巨大な場所が必要だったので分割したのでしょう。

キーサインには fingerprint を印刷した紙が必要です。そこに計算した sha1 の結果を書き込んで持参するのですが、その値 (sha1sum 出力) が読み上げられ、同じ結果が出たことが確認できた人だけが書名できます。

各人は国家の発行する公的な写真付きの証明書 (例えばパスポート) を見せ、fingerprint, sha1 の計算結果について

確認していきます。

分割したおかげか今年は 30 分程度で終わりました。

2.5 6 月 20 日の発表内容

2.5.1 Daytrip

Debconf では一日、参加者で旅行をするというイベントがあります。今回の Debconf では Rotheway (Bute 島) でまったりとピクニックをしました。Rotheway への移動は、Edinburgh から Glasgow へ電車で移動し、Glasgow からさらに電車で Wemyss Bay へ移動します。Wemyss Bay は Rotheway 行き専用の舟着き場で、そこから船に乗って Rotheway に移動しました。

Rotheway の町は島で、特になにもないところです。ほとんどの建物は売出中で、裁判所の建物も売りに出ていました。財政がやばそうな感じです。建造物としては、教会や、バイキングの侵略の際に戦った城がありましたが、修復中でしかも工事は止まっていました。山の奥へ 1 時間ほど歩くと、湖があり、大抵の参加者はその湖でピクニックをしたり、ボードに乗って遊んでいたようです。



2.6 6 月 21 日の発表内容

2.6.1 Forking Debian every day

GNU arch でいかに SELINUX 版の Debian のポーティングを楽しめたか、というワークフロー紹介のセッションのはずでしたが、いろいろと技術的な障害があったようで、git で Debian のパッケージをメンテナンスするための方法をデモンストレーションするセッションに急遽変更されました。アプリケーションの各種機能を SCM のブランチの機能を利用して実装し、新しいバージョンがリリースされても SCM の機能が活用できる、という話題でした。

2.6.2 Quality assurance activities for localization

小林さんが提案を提出し通っていたのですが、なぜか Debconf7 に参加しなかったのと、報告・周知・対策を何も講じなかったため開催されるはめになりました。

急遽 IRC で現地と日本を結んで上川がセッションを行いました。フランス、ブラジルなどのチームでの翻訳の進め方やツールの使い方についてディスカッションを行いました。メーリングリストをスキャンしてくれるロボットツールがあり、日本翻訳チーム向けに使えるように調整してくれるとのことでした。

2.6.3 Debian ceilidh/Sun Drinks Reception

Sun Microsystems と Google のスポンサによるパーティでした。Sun が飲み物、Google がピザを提供してくれました。また、スコットランドの民謡の演奏家を呼び、大ホールに集まったパーティ参加者達でスコットランドのダンスを楽しみました。



2.7 6 月 22 日の発表内容

2.7.1 Derivatives Round Tables — Debian より派生したディストリビューション

Benjamin “Mako” Hill が司会を務める、Debian より派生したディストリビューションの関係者が集まったパネルセッションでした。ベネズエラで開発されている国の支援を受けたディストリビューション、スペインの Extremadura、Debian Edu, Ubuntu などが参加してきていました。予想どおり白熱しました。ディストリビューションからのフィードバックの部分が問題になっていました。BTS の共有などもト

ピックになっていました。

2.7.2 Proactive Bug Discovery

DPL である、Sam Hocevar のセッションでした。^{*1}ソースを全部チェックするのは非常にコストが高いため、クリティカルな部分だけは全部査読するが、ほとんどの部分は、ツールを使って機械的にチェックするだけでもかなりのことがわかるということでした。

ソースコードを正規表現でスキャンしたり、google のコードサーチエンジンでチェックしたり、コンパイラーにチェックさせたりという部分について語ってくれました。

2.8 6月23日の発表内容

2.8.1 debian-community.org

Debian コミュニティに対する問題意識から、<http://debian-community.org> の提案セッションが行われました。Ubuntu コミュニティの事例からとったものです。Debian 開発者になり活躍するまでの時間がかかるのが問題意識となっています。その問題を解決するべく、debian-community.org というサイトを作り、Debian コミュニティ活動すると宣言し、活動している間はメールの転送などのサービスを提供します。活動が一定期間止まったら、この活動リストより外されます。他にも planet や、wiki などの提供を行う予定があります。

問題点としては、これまでのローカルコミュニティとの整合性、debian.org 本体もコミュニティであること、新しいコミュニティを作ってドライブしていくだけの魅力がそこにあるかなどが話し合われました。

2.8.2 WTFM, again: Write The Fine Manual page

Debian package で頒布されているプログラムには man が付属していないといけないということが Debian Policy で決まっているが、nroff 形式の man は時代遅れです。man だけでなく、あらゆるフォーマットに対応したドキュメントを容易に作成するにはどうしたいのか話し、DocBook XML を使った場合の簡単なチュートリアルを行いました。また、man はあるが、Linux の man ではなく、Unix の man だったりすることがあるので、ユーザーに man を提供する際に注意すべき事などを話しました。

Debian package で頒布されているプログラムには man が付属していないといけないということが Debian Policy で決まっているが、norff 形式の man は時代遅れです。man だけでなく、あらゆるフォーマットに対応したドキュメントを容易に作成するにはどうしたいのか話し、DocBook XML を使った場合の簡単なチュートリアルを行いました。また、man はあるが、Linux の man ではなく、Unix の man だったりすることがあるので、ユーザーに man を提供する際に書いてある内容が本当に妥当なのか確認すべき事などを話しました。

2.8.3 pbuilder talk

上川 純一 が pbuilder, cowbuilder, qemubuilder についての議論を行いました。pbuilder を利用しているユーザーは非常に多いが、qemubuilder の利用者は数人もいなかったということがわかりました。また、マニュアルの存在を知らない 人が多数いました。

2.8.4 Lightning Talks

ライトニングトークは若干オーガナイズに失敗しており、最初計画していたメンバーがあまりいなかったため、好きな人が好きなことを語るという会になってしまいました。

2.8.5 Closing ceremony

最後のしめの挨拶がなされ、スポンサーに感謝したりしました。

^{*1} このセッションの英語は矢吹にはわかりやすかった。

2.9 講演以外のできごと

2.9.1 apt-listbugs 関連の討論

Don Armstrong がきており、bugs.debian.org の SOAP インタフェースを拡張した、といいました。apt-listbugs の実装を変更し、SOAP インタフェースを利用するようにし、現在サーバ側で生成しているインデックスファイルが、もう必要ないようにしました。現地で SOAP インタフェースのデバッグを実施し、実用になるようにしました。

また、debian-changelog-mode に以前パッチをおくってくれた Luca Capello と BTS の HTML をパースしているからだよ、という話をしたら、SOAP を emacs からつかうのはいやなので、apt-listbugs を使おうという話になり、apt-listbugs list コマンドを拡張して実装することになりました。しかし、それが実装されるまえに、vim のメンテナ Stefano Zacchioli がその話をうしろで聞いていて、その場で vim 用の debian/changelog での closes: 補完コードが実装されてしまいました。

2.9.2 QEMU 関連の討論

上川は qemu、qemubuilder 関連で熱く議論してまわりました。

Thiemo Seufer (QEMU mips ポートのメンテナで QEMU のコミッタ、および Debian の MIPS ポートのメンテナ) と議論しました。chroot 内部での qemu user emulation をいかに static link をしないで実施するのか、という点について議論し、環境変数を定義する必要があるね、ということで合意しました。

夕食の時間で Ottavio と議論し、qemubuilder の設計と、qemu system emulation ではなく qemu user emulation での実装について議論しました。

2.9.3 矢吹× grisu

apt に i18n 機能がマージされたのにまだサーバ側のインフラの整備が行われていません。矢吹は DDTP の担当者の Grisu と DDTP の展開について議論しているようでした。なんらかの成果がでるとよいですね。

2.10 来年の Debconf

来年の Debconf はアルゼンチンで 8 月に行われます。日本は夏ですが、アルゼンチンは冬です。冬期合宿になると思いますので、気合いをいれていかないと、ひどいことになるかもしれません。気を付けましょう。

3 OSC-Kansai 参加報告

山下 尊也

3.1 開催概要

関西 Debian 勉強会は、7 月 20,21 日に京都コンピュータ学院で開催されたオープンソースカンファレンス 2007 Kansai（以下、OSC Kansai）に 京都ならびに関西地方でも Debian のプレゼンスを向上させるため参加しました。また、7 月の関西 Debian 勉強会としての位置付けで、第 4 回 関西 Debian 勉強会としています。

1 日目は、背広族の方が多いと思っていましたが、そこまで多くなく、京都コンピュータ学院の生徒さんが多かったです。

2 日目は、関西 Debian 勉強会からブースへの協力して頂いた方が多かったので、入口の真っ正面である京都コンピュータ学院のシンボルでもある階段の下にブースを移動し、さらに多くの方がブースに足を運んでいただきました。



図 3 展示風景

3.2 セッション

今回、セッションは 13:00-13:45 という 45 分間しかありませんでしたが、25 人の方に参加して頂きました。以下の内容について行いました。

1. 関西 Debian 勉強会とは 山下 尊也
2. Debian.org / Debian JP / 関西 Debian 勉強会の関係 矢吹 幸治
3. Debconf 7 ミニ報告 + Debconf 日本開催について。 矢吹 幸治

講師は、DebianJP 会員であり、関西 Debian 勉強会について動いている私と矢吹さんが行いました。

私のセッションは、関西 Debian 勉強会とはという題で、OSC Kansai に参加して頂いてる方にも、関西 Debian 勉強会に今後参加して頂けるように、参加し易い勉強会をアピールしたかったので、いくつかの笑いをを入れて説明しました。第3回での「ブルースマン」さんのファイアーウォールフリーダムの画像や、なぜ、関西 Debian 勉強会のシンボルが「ほっけ」であるのかを説明すると、会場からは笑いが起こりました。

矢吹さんのセッションは、今まで、Debian JP と関西 Debian 勉強会との関係について述べる機会がなかったので、参加して頂いた方には、関係などが理解して頂きました。具体的には、8月12日(日)に神戸研究学園都市で行われた第5回関西 Debian 勉強会で、参加費について議論した際に、OSC Kansai で聴いていたため、分かり易かったとおっしゃって頂きました。ただ、今まで関西 Debian 勉強会では、このような関係について述べる機会が少なかったため、今後機会を増やす必要があると思いました。また、Debconf については、矢吹さんの Debconf で手に入れたお土産を景品にして、クイズを行いました。クイズ形式でしたが、手をあげて頂ける方が少なかったのが残念でした。関西国際空港もありますので、関西で開催出来そうな場所を教えていただけるように働きかけました。

3.3 ブース企画

3.3.1 リアル掲示板

今回、関西 Debian 勉強会では、みなさんの意見を付箋紙に書いてもらい、Debian についての意見を書いてもらいました。



図4 リアル掲示板

集まった意見は下記です。

- Debian Love IT! (・∀・)
- 宮原がんばれ debian もがんばれ Apache ユーザ会 安東
- 「あれ使いたい」が大抵見つかるのがよいです
- Geeklog.JP のサーバは debian で一す。
- Debian Love
- lilo.linux.or.jp も Debian で動いています by ohura
- Sumibi.org も Debian で動いています! by kiyoka
- 編集長は Debian ユーザ!
- もっと Debian!
- Debian かわいいよ Debian ♪
- 使い勝手がよい感じですよ。
- 周りに Debian にしろと言われ続け一年…そろそろ覚悟かなあ
- apt は使いますよ。でもマカーですが…

- apt-get !!
- python-mode emacs22 に対応してー (TAT) by わさび
- webwml の訳がんばります -Nov
- 並列計算環境もすぐに整います。
- あなた方が Debian を使えば、Debian は Windows を越えます! はっしー
- あーもー好きだー Debian
- 1998 年 Debian をインストール、ネットワークに繋ぎ、E メールできるも、グラフィックカードを正確に認識できず、LYNX のみでインターネットに接続。
- 2007 年 今、CD を入れ、コンソールに installguit と入力するだけで、WEB もメールも、動画も、音楽も、しほうだいになりました。万歳 Debian

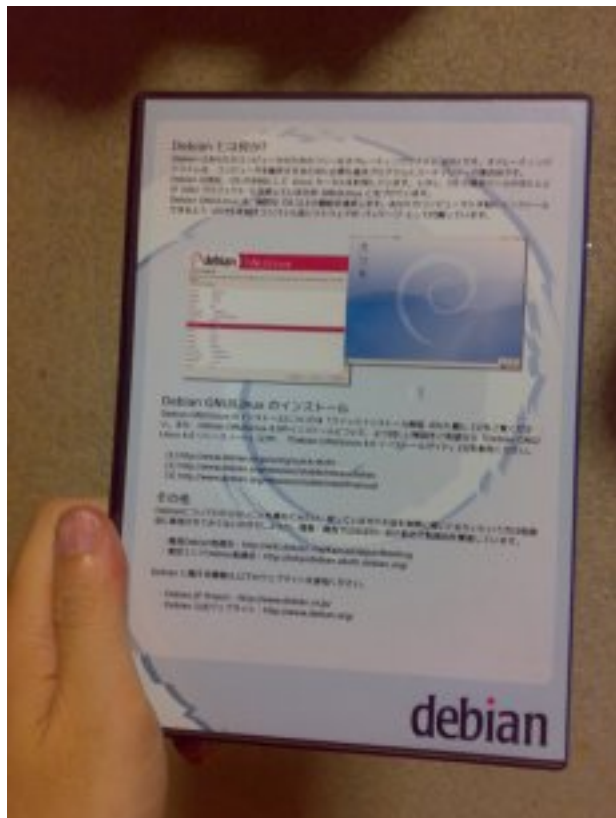
サーバでもデスクトップでも、本当にみなさん、Debian を愛してますね:)

個人的に気になったのは、python-mode についてですが、Emacs22 から python-mode は付属する形に変更されたみたいで、私の sid 上の Emacs22 では python-mode が動いています。

3.3.2 配布・販売物

以下のものを配布しました。

- フライヤー
- DVD



関西 Debian 勉強会では、OSC Kansai で配布を行うために、矢吹さんが提案して頂いたフライヤーを参考に、日本人向けに作り替えるために、のがたさんがデザインを担当し、かがさん、倉敷さんなどが文章を考えました。

また、DVD ジャケットについても、かがさんがデザインを担当し、武藤さんのインストールガイドの URL が書いてあったり、ジャケットが格好よかったので、家に飾りますとおっしゃられた方もいらっしゃいました。ただ、より多くの方に Debian について知ってもらうために、Debian を使っていらっしゃる方には周りの人に配布して下さいとお願いしました。



以下のものを販売しました。

- T シャツ
- ステッカー
- 「あんどきゅめんとてっど でびあん」 2006 年冬号の冊子

料金については、今年の 3 月に行われた OSC Spring と一緒にし、T シャツとシールは小室さん、「あんどきゅめんとてっど でびあん」の冊子は岩松さんに私の家へ送って頂き、委託販売と言う形で行いました。背景として、私が関西 Debian 勉強会に Debian T シャツを着ていくと、やはりグッズが欲しいなあと行っていただいたり、紙媒体で情報が欲しいって方がいらしゃったので、実現しました。

当初、僕が予想していたものよりも多くの売り上げがあり、やはり冊子でみたいと言う意見や、T シャツの黒色が欲しい、M サイズが欲しいなどの意見もありましたので、11 月 9,10 日に大阪南港 ATC で行われる関西オープンソースフォーラムでもグッズの販売を行いたいと思います。

売り上げですが、T シャツ (1 枚 2000 円) が 11 枚。冊子 (1 部 1000 円) が 7 部。ステッカー (1 枚 300 円) が 12 枚で、合計 32,600 円 (2000 円*11 枚 + 1000 円*7 部 + 300 円*12 枚) でした。ご協力頂いた方、本当にありがとうございました。

4 将来の Debconf

上川 純一

Debconf は来年はアルゼンチンですが、将来的には日本でも開催できるとよいですね。また、Debconf の開催内容をいかに有益に使えるか、考えてみましょう。

4.1 成果の活用

Debian Conference には複数の側面があります。成果はどうやってできるのでしょうか。

表 2 参加の成果

	参加した場合	参加しなかった場合
コード	合宿してコードがかける	
文書化	合宿して文書がかける	
議論	直接議論できる	
発表	セッションに参加して発表でき、発表をきくことができる。	

これを踏まえると、開催自体は重要ですが、参加にはおよびません。あなたも参加したくなってきたのではないですか？

4.2 日本開催

Debian Developer の中では Debian Conference を日本で開催したいと思っているメンバーがいます。日本で開催するとすれば、日本で開催するためのチームが必要です。日本で数度イベントを運営して円滑にすすめられるようにしておくことも必要でしょう。

日本での Debconf の検討の進捗については <http://wiki.debian.org/DebConfInJapan> で整理されています。

表 3 2005 年に実施した各種空港に到着するまでのコスト評価例

	フランス	アメリカ	南米
成田	828	809	1600
千歳	980	1197	2121
関西	736	809	1718
沖縄	1485	1197	4307

5 cdn.debian.or.jp の紹介

荒木 靖宏 (yasu@debian.or.jp, ar@debian.org)

5.1 CDN とは

Content Delivery Network (CDN) はウェブコンテンツ配置および配送方法として Akamai 社によりサービスされ広く知られることになった。当初から一部の人気の高いサーバへのトラフィック集中によるサーバ停止の回避、海外のリッチコンテンツ取得の高速化、トラフィック分散によるネットワークおよびサーバの利用平準化などの理由で広く受け入れられた。

CDN という用語自体は WWW に限ることなく、一般にコンテンツを取得するための配送手段や方法全体を指す場合がある。たとえば、Winny や Bittorrent などのコンテンツを取得するために特別に設計されたプロトコルを用いて、P2P ネットワークを構成するような手法も含まれる。

5.2 Debian における CDN の現状

5.2.1 利用法とユーザから見た動作

cdn.debian.or.jp では Debian でインストール時から広く deb ファイルの入手に使われる apt で使える CDN として設計し、運用している。そのため、Debian における CDN の利用法は至極簡単である。/etc/apt/source.list に記述する APT リポジトリとして、

```
deb http://cdn.debian.or.jp/debian/ stable main contrib non-free
deb-src http://cdn.debian.or.jp/debian/ stable main contrib non-free
```

以上のように指定するだけでユーザは今までとなんら変わることなく apt コマンドを使用できる。サービス時の手順と構成は以下のようになる。(図 5)

1. ユーザが apt-get コマンドを行うと cdn.debian.or.jp を DNS で問い合わせる
2. cdn.debian.or.jp を管理する DNS はサーバ候補 (surrogate) 選択する
3. 選択結果を DNS のリプライとして返す
4. apt は cdn.debian.or.jp として Surrogate C を使用する

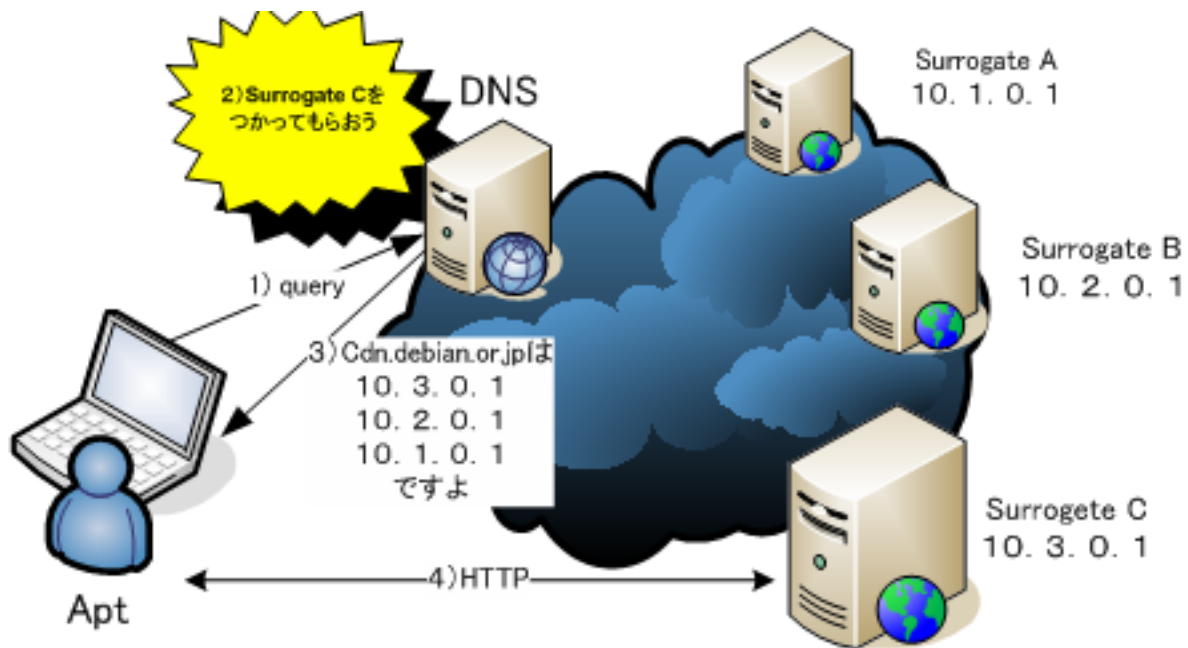


図5 ユーザから見た cdn.debian.or.jp の動作

5.2.2 cdn.debian.or.jp のシステムと動作

CDN システムが完全に動作しユーザから使用されるためには、システムが完全なファイルを提供すること、システムが安定して動作すること、そして CDN を使った場合に高速に動作していることが求められる。

提供ファイルの完全性

このために以下二点を満たさねばならない。

- 個々のファイルがコンテンツ提供者たる deb ファイル配布元と同一であること
- apt-get update の結果取得するファイル群がどの Surrogate でも入手できること

前者については、deb はそのファイルの md5 値、sha1 値とともに配布され、ユーザが使用する apt で確認後に利用されるため CDN を使用した場合でも問題にならない。

後者についてはユーザが apt-get update を行ったときに接続する Surrogate と apt-get dist-upgrade を行ったときに接続する Surrogate は同一であるとは限らないため、DNS が Surrogate として返すサーバが保持するファイルは同一である必要がある。cdn.debian.or.jp では Debian プロジェクトで一般に行われている方法と同様に、rsync プロトコルを用い、push ミラーを行っている (図2)。そのため、cdn.debian.or.jp のサロゲート内で最上流にあるサーバとミラーが同一であることを 2 分毎に rsync ミラー終了時に作成されるスタンプファイルを確認して、同一でないサーバはサロゲート候補から一時的に除外している。

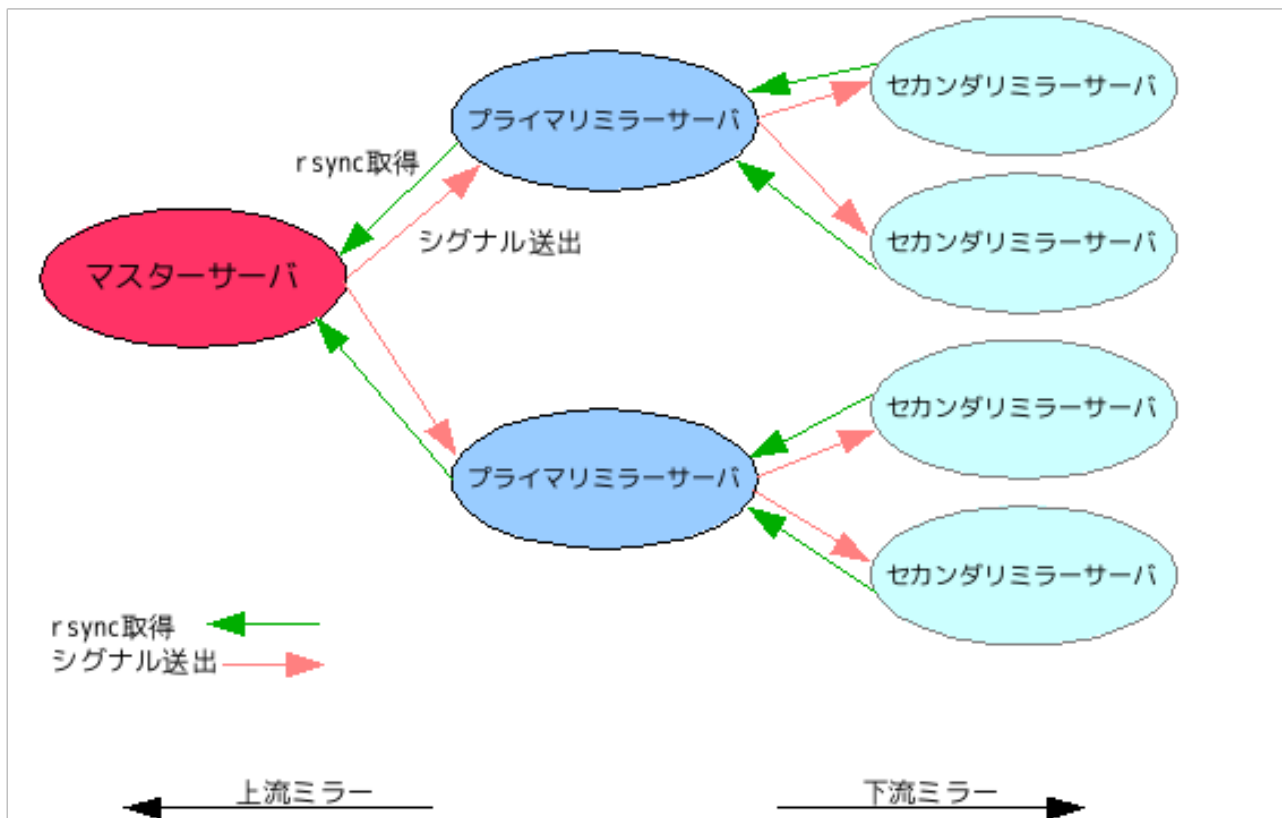


図6 Debian サロゲートの rsync によるミラー

システムの安定動作

先に述べたように、ユーザは CDN を使用する際には DNS を最初に使用するため、DNS の安定運用がカギとなる。そのため、cdn.debian.or.jp を管理する DNS サーバはまったく独立に動作するサーバで行っている。

また、サーバの動作を確認は、5 秒以内に HTTP のレスポンスを返さないサーバはサロゲート候補から一時的に除外している。

高速動作

cdn.debian.or.jp では DNS で問い合わせされるとサロゲートリストとして複数の IP アドレスを返す。この IP アドレスはラウンドロビンで選択しているわけではなく、サーバキャパシティやネットワーク速度を考慮し、設定している。

5.3 将来の展望

ここまで説明してきた、cdn.debian.or.jp の動作には改善すべき点が多数存在する。改善の展望としていくつか挙げる。

5.4 apt-get コマンドの HTTP REDIRECT

apt-get コマンドは HTTP REDIRECT に対応していない。

HTTP REDIRECT は、いったん HTTP GET などで接続してきたクライアントに対して、新たにそのリソースが存在する URL を通知するものである。この仕組みをうまくつけた CDN として、Coral Content Distribution Network (Coral CDN) がある。Coral CDN はサロゲート間で P2P によるファイル配置し、そのインターフェース

として、HTTP を使用し、しかも使用にはインターネットから取得可能なファイルであれば制限をかけていない。さらに、Apache を使った一時配布サーバでは HTTP REDIRECT をつかって Coral CDN に誘導することも推奨されている。ただし、現状で、Coral CDN を使うために、

```
deb http://cdn.debian.or.jp.nyud.net:8090/debian/ stable main contrib non-free
```

を指定することも可能だが、少なくとも日本においては Coral CDN を担うサロゲートが存在しないこともあって非常に低速である。ただし韓国や中国では広くつかわれており、将来の拡張に使用したい。

5.4.1 IP アドレスの位置情報を使用したサーバ選択

global に CDN を展開する場合には地理的に近いサーバ群からある程度絞込むのが有効である。現在、GeoIP など無料で IP と地理情報のマッピング提供者が現れており、この活用は `cdn.debian.or.jp` の次の拡張として最有力だと考えている。

5.4.2 apt の P2P 対応

現在、<http://wiki.debian.org/DebTorrent> や <http://www.cs.sfu.ca/~camerond/personal/GoogleSoCDebian.html> で apt の Bittorrent 対応が進められており、有力な候補である。ただし、Bittorrent プロトコルをクライアントで直接使うものであり、ネットワーク利用ポリシーとの競合や install 時に利用可能なのかなど今後検証すべき問題も多い。

5.5 おわりに

いつでも必要なソフトウェアやコンテンツを安価に入手する手段として CDN はこれからも様々な発展を続けると考える。Debian は deb の安定入手手段の有無がシステムの信頼性を左右するシステムであり、CDN の広範な活用が今後ますます求められるようになると考える。

6 Debian GNU/kFreeBSD のインストール

上川 純一

6.1 はじめに

最近めっさり話題の Debian GNU/kFreeBSD を qemu でインストールしてみました。

6.2 CD イメージの取得

Debian GNU/kFreeBSD のページ <http://www.debian.org/ports/kfreebsd-gnu/> からリンクをたどり、今回は <http://glibc-bsd.alioth.debian.org/install-cd/kfreebsd-i386/20070313/> から `debian-20070313-kfreebsd-i386-install.iso` を取得しました。

<http://glibc-bsd.alioth.debian.org/doc/> に文書があります。

6.3 qemu の準備

まず、ディスクイメージを作成します。

```
qemu-img create -f qcow f.cow 4G
```

6.4 インストーラの起動

qemu で ISO イメージから起動します。具体的なコマンドラインはこのようになります。(筆者のシステムは amd64 アーキテクチャのため、kqemu を活用するために、`qemu-system-x86_64` を利用しています。i386 であれば、`qemu` コマンドをかわりに利用します。)

```
qemu-system-x86_64 -hda f.cow \  
-cdrom debian-20070313-kfreebsd-i386-install.iso \  
-m 256 -boot d
```

Express を選択、適当にパーティションを切ってみて、FreeBSD のブートローダを利用してみました。マニュアルにしたがって適当に答えていきます。インストール対象は Minimal を選択して、インストールを続行します。

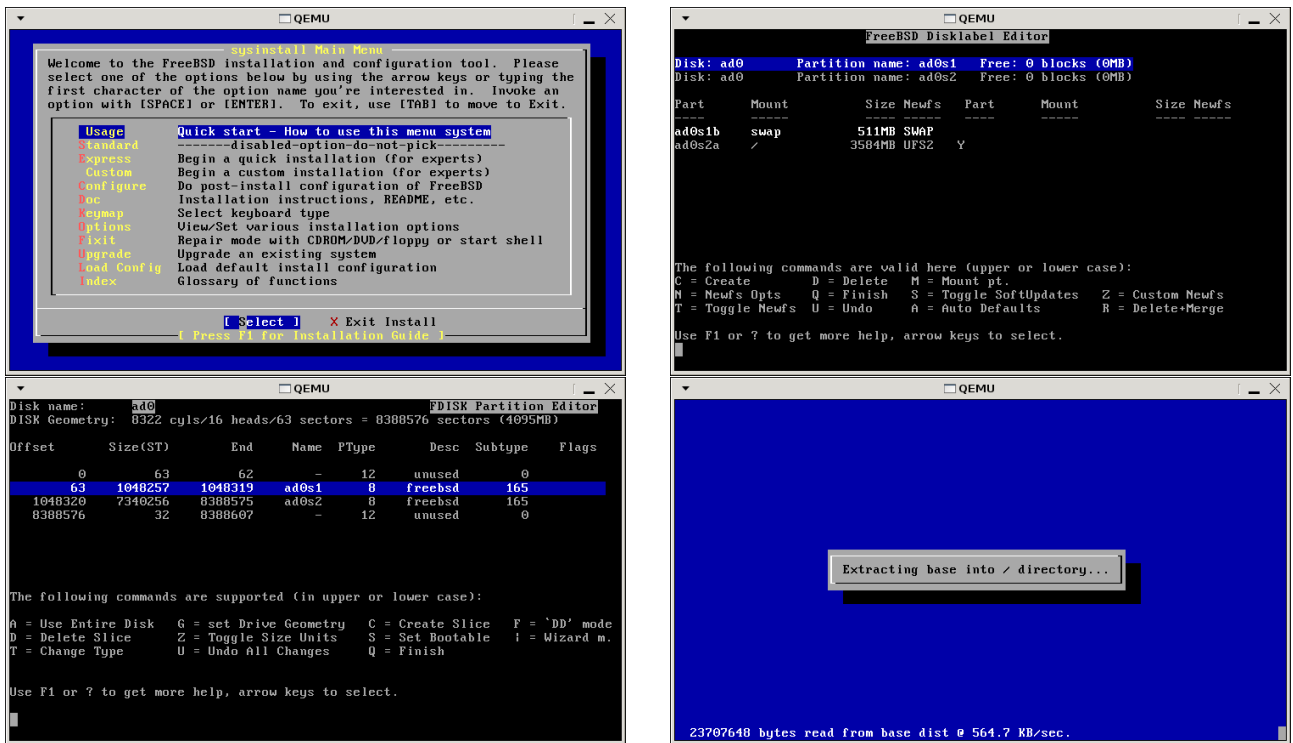


図7 Debian GNU/kFreeBSD インストール画面

しばらく待つと alt-f3 で画面を切り替えると表示されます。debconf の質問に答つつインストールがつづきます。

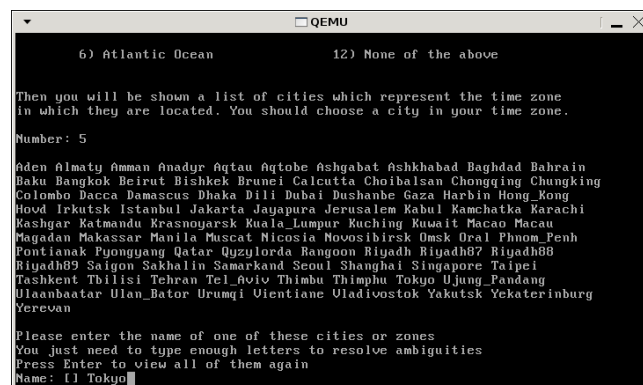


図8 初期パッケージインストール・設定中

最後にリブートをするように指示されるので、そこで qemu を一旦終了します。
ここで、qemu を HDD イメージから起動するようにして実行します。

```
qemu-system-x86_64 -hda f.cow -m 256
```

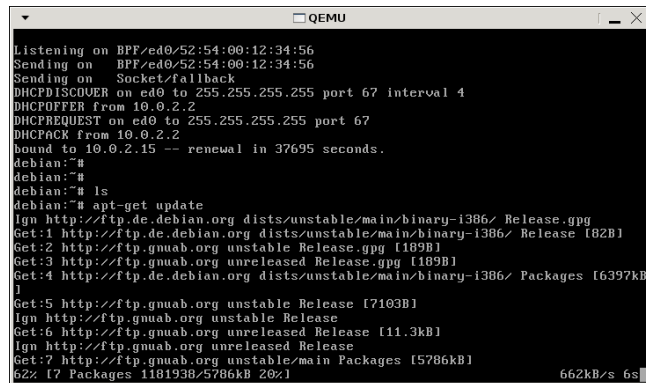
起動すると、なぜだか root filesystem が read-only だからといろいろと失敗します。fsck が必要な場合の起動に不都合があるようです。一旦 root でログインし、reboot コマンドでリブートしてみると root filesystem を正常にマウントすることが出来るようです。

また、/etc/network/interfaces がまったく設定されていない状態なので、ネットワークが使えない状態で起動してきますが dhclient を実行すれば IP を取得して稼働することも可能です。

```
dhclient ed0
```

6.5 動いた！

これで無事に Debian GNU/kFreeBSD の稼働が確認できました。まだまだ完成度が至らない点が多いので、デバッグしほうだいです。



```

Listening on BPF/ed0:52:54:00:12:34:56
Sending on BPF/ed0:52:54:00:12:34:56
Sending on Socket/fallback
DHCPDISCOVER on ed0 to 255.255.255.255 port 67 interval 4
DHCPOFFER from 10.0.2.2
DHCPREQUEST on ed0 to 255.255.255.255 port 67
DHCPACK from 10.0.2.2
bound to 10.0.2.15 -- renewal in 37695 seconds.
debian:~#
debian:~#
debian:~# ls
debian:~# apt-get update
Ign http://ftp.de.debian.org dists/unstable/main/binary-i386/ Release.gpg
Get:1 http://ftp.de.debian.org dists/unstable/main/binary-i386/ Release [82B]
Get:2 http://ftp.gnuab.org unstable Release.gpg [189B]
Get:3 http://ftp.gnuab.org unreleased Release.gpg [189B]
Get:4 http://ftp.de.debian.org dists/unstable/main/binary-i386/ Packages [6397kB]
Get:5 http://ftp.gnuab.org unstable Release [7103B]
Ign http://ftp.gnuab.org unstable Release
Get:6 http://ftp.gnuab.org unreleased Release [11.3kB]
Ign http://ftp.gnuab.org unreleased Release
Get:7 http://ftp.gnuab.org unstable/main Packages [5786kB]
62% [7 Packages 1181938/5786kB 20%] 662kB/s 6s
```

図9 Debian GNU/kFreeBSD で apt-get してみました

7 Exim 再発見

小室 文

7.1 intro

私達の日常ではメールは欠かせないコミュニケーションツールです。それを実現する為に、導入される MTA (Mail Transfer Agent) は、どのソフトを使ったら一番コストパフォーマンスが良いか、またどうしたら会社、グループ、個人のニーズに答える事が出来るか、システム管理者は日夜頭を悩ましているはずで。そして一度導入したメールサーバーの仕組みは、なかなか別の仕組みに乗り換えるのは、時間と費用がかかり、問題があったとしてもなかなか移行しづらいという現状もあり、MTA を決めるのは人生でなかなか訪れない大決心の一つである事は明白です。MTA の一つ、Exim は Debian の Default MTA と呼ばれながらも、今や Postfix の人気にすっかり影を潜めて過去の栄光に甘んじているのが現実です。今日は Exim の歴史、利点・不利点、導入方法を御紹介します。皆さんの MTA に選ばれなかったとしても、Exim はこういうパッケージなのか！と知って頂ければ幸いです。

7.2 Exim とは

Exim^{*2} とは、Unix もしくは Unix like な OS の上で動く Mail Transfer Agent です。Cygwin を使えば Windows の上でも動かす事が出来ます。Exim はケンブリッジ大学で 1995 年に Philip Hazel さんによって開発されました。

7.3 Exim の歴史

ケンブリッジ大学では、複数の MTA が動いていて (Sendmail, Smail, PP? などなど)、そんな環境にうんざりしてたがどうかは不明ですが、Philip Hazel さんは Smail を拡張して MTA を大学のニーズに合わせて作ろうと試みました。しかし残念ながら、Smail 拡張はあっさり諦め、Hazel さんはスクラッチから MTA を作ろうと試みる事にしました。その作業が、彼が所属した Computer Science の仲間知れわたり、FTP サーバーを立てられ、配布されるようになりました。書いている途中で配布がされるようになった為、正式な Exim 0 もしくは 1 はリリースしていません (少なくとも Hazel さんはリリース出来なかったと思っているようです)。

元々 Sendmail, Smail を触っていた Hazel さんは Sendmail に代わる MTA を作ろうと Exim を設計されています。

^{*2} Exim の名前の由来は EXperimental Internet Mailer (Exim)

1995 年 ? 月	Exim 開発開始
1995 年 11 月	同僚が FTP サーバをつくりパッケージを配布始める。クチコミで広がる
1998 年夏	Perl の正規表現ライブラリーがなかったので作る (PCRE)
1999 年 3 月	コードネーム Slink の Debian 2.1 リリース。Exim をデフォルト MTA として起用する
1999 年 9 月	ケンブリッジ大学で Exim の講座を開始する。
2000 年 8 月	コードネーム potato の Debian 2.2 リリース
2002 年 7 月	コードネーム woody の Debian 3.0 リリース
2004 年 5 月	ケンブリッジ大学が Exim を正式にサポートすると宣言
2005 年 6 月	コードネーム sarge の Debian 3.1 リリース
2007 年 4 月	コードネーム etch の Debian 4.0 リリース

7.4 Exim と Debian の関係

7.4.1 現在の Exim のメンテナー

現在の Exim のメンテナーは

- Andreas Metzler ametzler@debian.org
- Marc Haber mh@debian.org

です。Slink で Exim が搭載されるようになる前から、Sendmail や Smail から Exim へ移行を試みる人が沢山いたようです。日本ではマニュアルの日本語化があまり進まず (現在も小数の人達が作業しているのみ)、Slink で Exim が搭載されたので、移行した、という人が多かったようです。

7.4.2 なぜ Exim は Debian の Default MTA になったのか !?

1996 年 09 月	Tim Cutt が Debian で Exim をパッケージとして提供する為に作業を始める
1997 年 05 月	Exim が unstable に入る by David Sewell
1999 年 03 月	Slink で Exim をデフォルト MTA として起用する。

当時は Postfix、Sendmail、Smail などがあったりしましたが、ライセンス問題^{*3}、機能の充実度合などがあり、Debian の Default MTA になったようです。

7.5 Exim と他の MTA の相違点

MTA	ライセンス形態	メイン製作者	リリース状態
qmail	DJB ライセンス	D. J. Bernstein	1997 年に出したっきり
Postfix	IBM Public License	Wietse Zweitze Venema	1997 年にリリース後、都度都度にリリース
Exim	GPL	Philip Hazel	1995 年にリリース後、都度都度リリース

- qmail
 - 1997 年からリリースされていない。IPv6 に対応してない。ライセンス形態、導入が難しい
 - 大量メールを配信する場合、セキュリティー面
- Postfix
 - Exim ほど機能の実装がない
 - 移行が簡単、コミュニティがアクティブ、日本でも使っているユーザーが多い (本も多い)
- Exim
 - 日本語のマニュアルがない
 - コミュニティーがアクティブ、Debian の Default MTA、ライセンス、ドキュメントが豊富 (英語)

^{*3} ライセンス形態：GNU General Public License (GPL) ver2

7.6 Exim の設定方法

7.6.1 既存パッケージ一覧

現在の Debian の stable である etch には以下の Exim 向けパッケージが用意されています。

exim4	Exim 4 を簡単にインストールする為にメタパッケージ
exim4-base	全 Exim 4 パッケージ支援パッケージ
exim4-config	Exim 4 設定用パッケージ
exim4-daemon-heavy	追加機能 (exiscan-acl 含む) を搭載しているデーモンパッケージ
exim4-daemon-light	簡易機能を搭載しているデーモンパッケージ
exim4-daemon-light-dbg	debug 用
exim4-dbg	debug 用
exim4-dev	ヘッダーファイル用の Exim 4 パッケージ
exim4-doc-html	Exim 4 の html 形式のドキュメントパッケージ
exim4-doc-info	Exim 4 の info 形式のドキュメントパッケージ

7.6.2 インストール方法

```
aptitude install exim4 exim4-base exim4-config exim4-daemon-heavy
```

か

```
aptitude install exim4 exim4-base exim4-config exim4-daemon-light
```

を実行します。

exim-config によって必要な情報入力はプロンプトが出るので指示に従います。

1. 設定ファイルを分割するか、否か
2. メールの扱いについて (送信サーバの決定)
3. メールドメイン名について
4. 受けつけるメール送信元の IP 制限の決定 (IPv6 対応)
5. Virtual Domain の下準備
6. オープンメールの設定
7. ローカルネットワークからの送信の決定
8. ダイアルアップ時の DNS look up の決定
9. メール形式の決定 (mbox/Maildir)

7.7 Exim のこれから

Philip Hazel さんは 2007 年 2 月 8 日付で、9 月末に Exim から引退したいと申し出ましたが、まだ決着はついていません。すでに Exim を Hazel さんと同等に知っているメンテナーが世界中にいますので、肝はそれを統括するマネージャーのような人を立てる、という事が大事になってくると思われます。

メインで動いている人達は Exim の新機能や、Exim をどのように運営、更新をしていくか Exim-future@exim.org という場所で会議をしようと試みています。

Debian では、Exim 5 がリリースされるタイミングが現在もなお不透明な為、例え Lenny が近い未来リリースされる事があっても、その時に Exim 5 が間に合うとは保障出来ません。日本では私がとりあえずマニュアルの翻訳をしようと Exim ユーザー会を作ってみました。

<https://sourceforge.jp/projects/exim-jp>

8 あなたの知らないかもしれない apt-xxx

岩松 信洋

8.1 はじめに

Debian ユーザーは apt がないと生きていけません。apt-get はみんなが知っているコマンドですが、apt にはいろいろなコマンドが存在します。今回はあまり知られていない apt-xxx について調べてみました。

8.2 レベル 小

勝手にレベルをつけていますが、レベル小 は一般ユーザの方なら知っておいて損はないというものです。Debian 上での生活を楽にしてくれるかもしれません。

8.2.1 apt-key

apt の GPG 鍵リングを制御するフロントエンドです。2006 年から secure apt が導入された。secure apt は Debian アーカイブの信頼性を上げるため導入されたのですが、これには GPG が使われており、一般ユーザーにはちょっと難しいかもしれません。しかし、secure apt の GPG のキーは毎年変更されるので、今後使うことがあるかもしれません。また、GUI で操作したい人のために gui-apt-key^{*4} パッケージがあります。

8.2.2 apt-spy

ネットワークの情報から最適な apt-line を生成することができるツールです。いまは cdn.debian.or.jp があるのでどうでもいいかんじです。

8.2.3 auto-apt

auto-apt は世間では検索用のツールになっています。^{*5}

```
% auto-apt upate
% auto-apt search stdio.h
usr/include/stlport/stdio.h      libdevel/libstlport5.1-dev
usr/include/fcgi_stdio.h         libdevel/libfcgi-dev
usr/include/H5FDstdio.h          libdevel/libhdf5-lam-dev,libdevel/libhdf5-mpich-dev,libdevel/libhdf5-serial-dev
usr/include/stdio.h              libdevel/libc6-dev
```

しかし、auto-apt は コマンド実行時に足りないファイルをパッケージをから探しだし、インストールしてくれるツールだったりします。

```
# auto-apt run ./configure
```

^{*4} <http://packages.debian.org/unstable/admin/gui-apt-key>

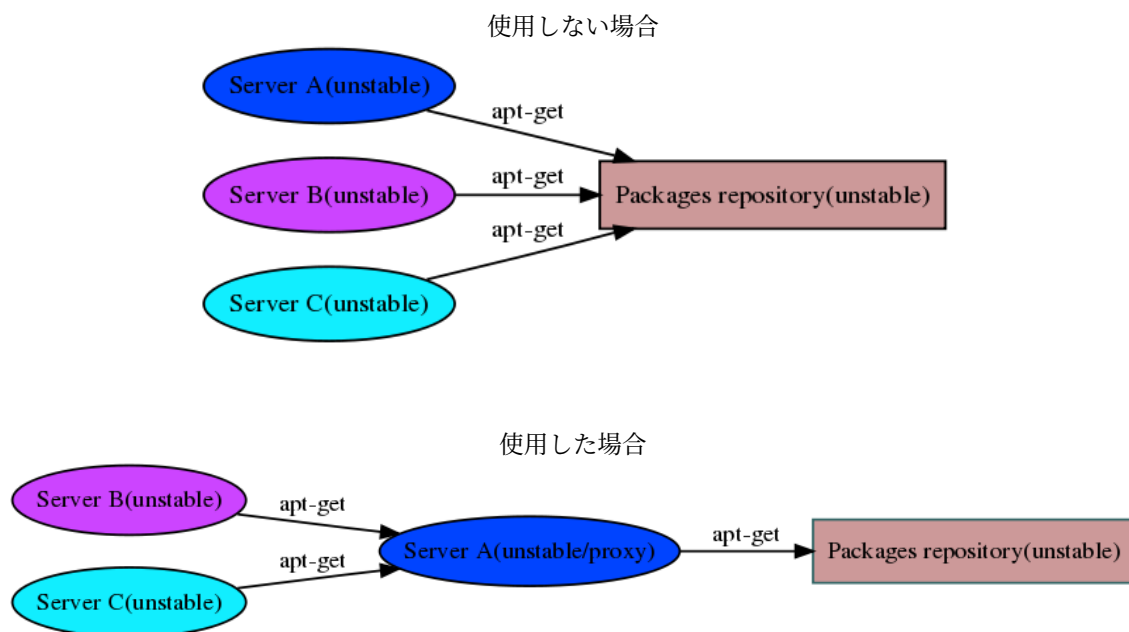
^{*5} ファイル検索用として apt-file というコマンドがあります。

8.2.4 cron-apt

apt-get update / apt-get upgrade を cron で書いている人を見かけますが、cron-apt を使えば、ログに apt の実行結果を残した設定が可能になります。似たような名前でも apticron^{*6} というパッケージがありますが、これは aptitude の cron-apt 版ではなくセキュリティアップデート情報をメールで送信してくれるツールです。

8.2.5 apt-proxy / apt-cacher

Debian パッケージのキャッシングプロキシを構築するパッケージです。例えば、家の中でマシンが数台あり、すべて sid だったとしましょう。特に設定を行ってない場合、各マシンは apt-get 毎に ミラーサーバーから Debian パッケージを取得します。これは無駄なので、1 台だけ Debian パッケージをミラーサーバーから取得し、キャッシュし、他のマシンは対象のキャッシュしているパッケージを使ってアップデートを行うようにします。これを実現するためのパッケージが apt-proxy / apt-cacher です。



8.3 レベル 中

一般ユーザーは知っていてもあまり役に立たないと思われる apt-xxx。

8.3.1 apt-ftparchive

Sources.gz / Packages.gz などのパッケージ情報用ファイルを作成するためのツール。自分で作ったパッケージを apt-line として公開したいときに使います。

```
% apt-ftparchive packages . | gzip -9 > Packages.gz
% apt-ftparchive sources . | gzip -9 > Sources.gz
% apt-ftparchive release . > Release
```

8.3.2 apt-sortpkgs

Packages ファイル および Sources ファイルをソートします。apt-ftparchive で作成したものはソートされていなかったりするので、アルファベット順にソートするときに使います。

^{*6} <http://packages.debian.org/unstable/admin/apticron>

```
% apt-sortpkgs Packages > Packages.sort
```

8.3.3 apt-extracttemplates

Debian パッケージから設定とテンプレート情報を抽出するためのツールです。

```
% wget http://http.us.debian.org/debian/pool/main/x/xorg/xserver-xorg_7.3~rc1_all.deb
% apt-extracttemplates xserver-xorg_7.3~rc1_all.deb
% ls
xserver-xorg.config.34261
xserver-xorg.template.34260
```

8.3.4 apt-build

Debian で提供されているバイナリパッケージはあまり最適化されていません。人によっては自分の環境に合わせてチューニングしたり、製品に組み込んだりする場合があります。apt-build は apt-get する感覚で環境に合わせてバイナリを作成をサポートするツールです。Debian パッケージをリビルドする場合は

```
% apt-get update
% apt-get source hello
% cd hello-x.x
% debuild -us -uc
% sudo dpkg-i ../hello_xxxx.deb
```

という手順を踏みますが、apt-build の場合は

```
% apt-get update
% apt-build install hello
```

だけです。apt-build の設定ファイルは

```
/etc/apt/apt-build.conf
```

にあり、

```
build-dir = /var/cache/apt-build/build
repository-dir = /var/cache/apt-build/repository
Olevel = -O3
march = -march=pentium2
mcpu = -mcpu=pentium2
options =
```

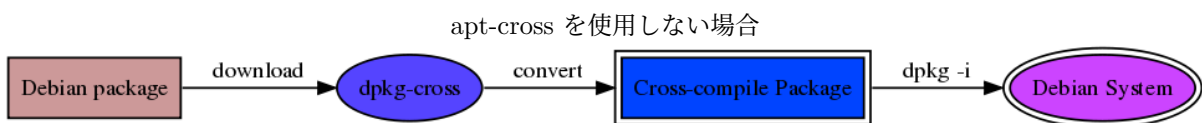
という設定になっています。例えば、自分の使っているマシンが i686 ではなく、Crusoe の場合には

```
Olevel =
-O2 -fomit-frame-pointer -fno-strict-aliasing -fno-common \
-pipe +-mpreferred-stack-boundary=2 -march=i686 -malign-functions=0 \
-malign-jumps=0 -malign-loops=0
```

とすればよいでしょう。

8.3.5 apt-cross

Debian package を cross 環境で使用できるように変換してインストールしています。いままでは ダウンロードした Debian package を dpkg-cross で変換してインストールしていましたが、apt-cross を使うことによって、手作業が減らすことができます。





8.3.6 apt-transport-https

apt は ftp/http で取得できるのですが、このパッケージを使うことによって https 経由で apt を行うことができます。

8.4 レベル 高

知っていても使わないだろうと思われる apt-xxx. 話のネタにはなるかもしれない。

8.4.1 apt-zip

リムーバブルメディアの ZIP からの apt をサポートするためのツールです。同じようなツールで apt-cdrom がありますが、違いがいまいちわかりません。

8.4.2 aptsh

apt の操作をができる Shell.

```
% sudo aptsh
```

で shell が aptsh になります。試しに、

```
% ls
```

を実行すると、インストール可能なパッケージ一覧が表示されます。

8.5 まとめ

今回はさわりだけを説明しました。今回の紹介で気になる apt-xxx がありましたら詳細を説明していきたいと思っています。

9 DebTorrent に触ってみた

山下 尊也

9.1 DebTorrent とは

DebTorrent は、Debian パッケージを配布するためのプロトコルです。BitTorrent という P2P の技術を利用しています。具体的には、proxy で apt に渡しているみたい。

BitTorrent の利点は、HTTP ではパッケージのダウンロードが集中したときに、速度の低下などが考えられますが、協力してダウンロードする事により、より多くのユーザに高速にファイルを配布出来ます。

調べてみたら、岩松さんが 2007 年の 1 月に東京エリア Debian 勉強会で apt-torrent の話をしていたのですが、詳しく調べると、DebTorrent と apt-torrent は違うみたいです。また、apt-torrent は 2007 年 9 月 15 日現在、公式パッケージではありません。

```
% aptitude show debtorrent
パッケージ: debtorrent
新規: yes
状態: インストールされていません
バージョン: 0.1.4.1
優先度: 任意
セクション: net
メンテナ: Cameron Dale <camrdale@gmail.com>
展開サイズ: 1217k
依存: python, python-support (>= 0.2), adduser
推奨: python-crypto, apt-transport-debtorrent
提案: python-psyc
提供: python-debtorrent
説明: bittorrent proxy for downloading Debian packages
DebTorrent is a proxy for downloading Debian packages files with APT. It will
download any needed packages from other DebTorrent peers in a bittorrent-like
manner, and so reduce the strain on the Debian mirrors.

The DebTorrent client runs as a daemon, automatically started on bootup, and
listens for requests from APT for files. Any non-package files are downloaded
and served to APT similarly to other proxying software (e.g. apt-proxy,
apt-cacher, and approx). The configuration is very simple, and only involves
prepending a server and port to your current sources.list files (similar to
apt-cacher).

When downloading package files, the DebTorrent client will try to use any other
DebTorrent clients it can find to download from. This will use the uploading
bandwidth of other peers, while reducing the demand on the Debian mirror
network. However, if a package cannot be found on any peers, DebTorrent will
fall back to downloading from a mirror to ensure all packages are downloaded.

Homepage: http://debtorrent.alieth.debian.org/
```

9.2 DebTorrent を用いてダウンロード

1. DebTorrent をインストールします。aptitude install debtorrent
2. /etc/apt/sources.list を修正します。localhost:9988/cdn.debian.or.jp のようにする。
ポートはフォルトでは、9988 を利用しています。/etc/debtorrent/debtorrent-client.conf を変更する事で、ポート番号を変える事が出来ます。
3. 後は、aptitude と一緒に、aptitude update
4. aptitude install *packagename* でインストール出来ます。

```
# cat /etc/apt/sources.list
deb http://localhost:9988/cdn.debian.or.jp/debian/ sid main contrib non-free
```

9.3 実際にやってみる

残念ながら、Python 2.3 以上でないと動かないため、etch での利用は難しいと思います。また、Q&A を読んでみると、experimental を 2007 年 9 月 15 日現在、サポートしていないようです。

```
# time apt-get source linux-source-2.6.22
Reading package lists... Done
Building dependency tree
Reading state information... Done
Need to get 58.3MB of source archives.
Get:1 http://localhost sid/main linux-2.6 2.6.22-4 (dsc) [4832B]
Get:2 http://localhost sid/main linux-2.6 2.6.22-4 (tar) [57.4MB]
Get:3 http://localhost sid/main linux-2.6 2.6.22-4 (diff) [889kB]
Fetched 58.3MB in 26s (2165kB/s)
sh: dpkg-source: command not found
Unpack command 'dpkg-source -x linux-2.6_2.6.22-4.dsc' failed.
Check if the 'dpkg-dev' package is installed.
E: Child process failed

real    0m28.620s
user    0m7.280s
sys     0m0.728s
#
```

実際に、この様にパッケージがある場所が見つかれば良いのですが、見つからない場合は、いつまで経ってもダウンロードが始まりません。プロセスが死んだのかと不安になります。

しかし、見つかってしまえば結構な速度が出ます。4 回やった結果ですが最短は 18 秒。20 秒、28 秒、26 秒と毎秒 2MB ほど出るとなると嬉しくなります。

ちなみに、私は普段 cdn.debian.or.jp を使っていますが、同じ事を debtorrent を使わずにやってみると、49 秒かかりました。

ただ、今回の例では大きなファイルについてやっているのでも、小さなファイルで実行したり、小さなファイルをたくさんダウンロードするとなると、http の方が速度が出ます。試しに、safe-upgrade をしてみましたが、ダウンロードに時間がかなりの時間がかかり、話になりませんでした。

9.4 まとめ

私が考えたのは、security.debian.org は、stable や testing のユーザにいち早くセキュリティ修正されたパッケージを配っています。その際に、パッケージの更新があれば、世界中のユーザからのアクセス集中により繋がりにくい状況に陥ったり、結果としてサーバダウンに繋がる事もあるので、DebTorrent はこのような状況で使われたら良いのではないかと思ったのがきっかけでした。

ただ、今の stable には依存関係で導入が難しい事を考えると実際には、勧める事は難しくなります。また、パッケージを送る側になる事も難しいでしょう。しかし、おもしろい試みなので、今後も注目して、開発に期待したいと思います。

10 live-helper

岩松 信洋

10.1 live-helper とは

live-helper は Debian の Live-CD / USBboot イメージを作成するためのツールです。作成するためのスクリプトが各種用意され、Live-CD の仕組みを知らない人でも容易に Live-CD を作成することができます。

10.2 コマンド

live-helper は lh_xxx という形式でコマンドが提供されています。たくさんのコマンド^{*7}が用意されていますが、基本的に使用するののは、

- lh.config
- lh.build
- lh.clean

です。他のコマンドは細かい設定を行うためや、内部処理で使用されます。

10.3 雛型の作成

まず、lh.config コマンドを使って、Live-CD の雛形を作成します。

```
% lh.config
```

lh.config 実行時にオプションを指定することにより、さまざまな設定を行うことが可能です。また、これらのオプションは環境変数としても指定することができます。例えば、**-union-filesystem** というオプションがあるのですが、この場合は

```
LH_UNION_FILESYSTEM
```

として、設定することが可能です。

10.4 作成された設定ファイルの説明

lh.config を実行した後、config ディレクトリ以下に以下のディレクトリが作成されます。

^{*7} 2007/11/02 現在で、67 個

ディレクトリ名	説明
binary	作成される Live-CD イメージに関する設定が書かれています。
bootstrap	Live-CD 作成環境に関する設定が書かれています。
chroot	Live-CD のユーザーランドに関する設定が書かれています。
common	live-helper の基本設定が書かれています。
source	source イメージに関する設定が書かれています。

これらの設定はテキストファイルになっています。なので、適当なエディタで編集可能ですが、実際には **lh_config** コマンドを使い、設定ファイルを書き換えます。

10.5 細かい設定方法

live-helper は雛形を作成し、作成された各設定ファイルに追記することによって細かいカスタマイズが可能になっています。以下にカスタマイズの方法について説明します。

10.5.1 アーキテクチャの指定

lh_config の -a オプションを使用することによって、作成する Live-CD イメージのアーキテクチャを指定することができます。

```
% lh_config -a アーキテクチャ名
```

10.5.2 ディストリビューションの指定

lh_config -d オプションを使用することにより、作成する Live-CD のディストリビューションを指定することができます。

```
% lh_config -d ディストリビューション名
```

10.5.3 言語の指定

lh_config -l オプションを使用することにより、Live-CD で使用するデフォルトの言語を設定することができます。例えば、デフォルトの言語を日本語にする場合は

```
% lh_config -l ja
```

とします。

10.5.4 ブートローダーの指定

lh_config の --bootloader オプションを指定することにより、Live-CD で使用するブートローダーを指定することができます。

```
% lh_config --bootloader grub
```

指定することができる ブートローダーは以下の 3 つです。

- grub
- syslinux
- yaboot

10.5.5 作成するイメージの指定

live-helper では、

- iso
ISO9660 イメージ
- net
NET ブート用イメージ
- tar
tar 形式でまとめたもの
- usb-hdd
USB メモリや USB-HDD で起動することができるイメージ

のイメージを作成することができます。

lh_config の `--binary-images`^{*8} オプションを指定することにより、作成するイメージを指定することができます。

```
% lh_config --binary-images iso
```

10.5.6 bootstrap および live-image 内で使用する apt-line の変更方法

live-helper で使用する apt-line は

- Live-CD 作成に使用する apt-line
- Live-CD 内の `/etc/apt/sources.list` に書き込まれる apt-line

の 2 種類があります。

これらは特に指定しない場合、

```
http://ftp.debian.org/debian/
```

が指定されています。

変更する場合は **lh_config** コマンドを利用して変更します。その他に以下の オプションを指定することによって各々の apt-line を変更することができます。

- `--mirror-bootstrap-security URL`
Live-CD 作成時に使用する セキュリティアップデート向け apt-line を設定します。
- `--mirror-bootstrap URL`
Live-CD 作成時に使用する apt-line を設定します。
- `-m | --mirror-binary-security URL`
Live-CD 内で設定される、セキュリティアップデート向け apt-line を設定します。
- `--mirror-binary URL`
Live-CD 内で設定される、 apt-line を設定します。

10.5.7 Debian パッケージの追加

Live-CD に Debian Project で配布されているパッケージを追加したい場合は、

```
lh_config --packages "パッケージ名"
```

を実行します。パッケージ名のところは、スペースで区切り、複数パッケージを指定することが可能です。

^{*8} `-b` でも可能

この方法でパッケージの追加を行った場合、再度実行してしまうと、上書きされてしまうためパッケージの追加ができません。^{*9}

複数パッケージの追加や、パッケージの種類によって管理したい場合は

```
config/chroot_local-packageslists/
```

ディレクトリに適当なファイルを作成し、パッケージ名を列挙します。例えば、 bluetooth ^{*10} メタパッケージを追加したい場合は、

```
% cat config/chroot_local-packageslists/bluetooth
# bluetooth packages
bluetooth
```

として、パッケージ名をファイルに列挙します。

ファイル名は管理しやすい名前しておくといいいでしょう。

10.5.8 オリジナル Debian パッケージの追加

自分で作成した Debian パッケージや オリジナルのパッチを当てた Debian パッケージは以下の方法で Live-CD に追加することができます。

まず、自分で作成したパッケージ用のレポジトリを作成します。次に

```
config/chroot_sources/適当なファイル名.bootstrap
```

を作成し、作成した レポジトリを apt-line として追記します。

```
config/chroot_local-packageslists/
```

に適当なファイルを作成し、追加したいパッケージ名を書きます。

これにより、Live-CD 作成時に 作成した apt-line からパッケージがダウンロードされ、インストールされます。作成した apt-line を Live-CD にも追加する場合は

```
config/chroot_sources/適当なファイル名.binary
```

を作成し、作成した レポジトリを apt-line として追記します。

10.5.9 apt-line を使わない Debian Package の追加方法

```
config/chroot_local-packages/
```

ディレクトリに Debian パッケージをコピーします。コピーしておくことにより、自動的に Live-CD 内にインストールされます。依存関係は解決してないので、必要なパッケージは別途 apt を使ってインストールするように設定しておく必要があります。

10.5.10 すでに用意されているパッケージリスト

live-helper ではまとまった環境がパッケージリストとして用意されています。これらを使用することによって、ある程度容易に Live-CD を作成することができるようになっています。パッケージリストは

```
/usr/share/live-helper/lists/
```

にあり、これらを利用することが可能です。これらのリストを利用するには、lh_config オプションの `-packages-lists` `-p` を使用します。gnome-desktop ベースの Live-CD を作成する場合、

^{*9} エディタで編集すれば対応できますが。

^{*10} <http://packages.debian.org/sid/bluetooth>

```
% lh_config -p gnome-desktop
```

とします。

10.6 ホスト名を変更する

Live-CD のホスト名を変更するには、lh_config の `-hostname` オプションを使用します。

```
% lh_config --hostname myhostname
```

を実行し、設定したいホスト名を指定します。

10.7 ユーザー名を変更する

Live-CD に新しいユーザを追加するには、lh_config の `-username` オプションを使用します。

```
% lh_config --username myname
```

を実行し、追加したいユーザ名を指定します。パスワードは **live** になっています。

10.7.1 パッケージ化されていないソフトウェアの追加方法

アイコンや簡単なスクリプトをパッケージ化せず、Live-CD にインストールしたい場合があります。この場合は、

```
chroot_local-includes
```

ディレクトリに `content` ディレクトリを作成し、この中にファイルを追加します。例えば、`usr/bin/` に `hello_world` というプログラムを追加したい場合は

```
chroot_local-includes/content/usr/bin/hello_world
```

にコピーします。

10.7.2 カーネル用パッケージの追加

Debian で提供されているカーネル用パッケージを Live-CD に追加するには lh_config の `-linux-packages` オプションを使用します。

```
% lh_config --linux-packages "追加したいパッケージ名"
```

追記ができず、上書きになってしまうため、パッケージを追加したい場合は

```
config/chroot
```

ファイルの

```
LH_LINUX_PACKAGES
```

の部分を編集するとよいでしょう。

10.7.3 フック機能

Live-CD イメージ作成に処理を入れたいときに使用します。

```
config/chroot_local-hooks
```

ディレクトリにシェルスクリプトを入れることによって動作します。

例えば、`bluetooth` パッケージを Live-CD 内にインストールし、起動時に有効にしたい場合は

```
% cat config/chroot_local-hooks/enable-bluetooth.sh
#!/bin/sh -x
sed -ie 's/^BLUETOOTH_ENABLED=.* /BLUETOOTH_ENABLED=1/' /etc/default/bluetooth
```

というようなファイルを作成しておくと、Live-CD 内でデフォルトで有効になっています。

10.7.4 CDROM 起動時の splash 画面を変更する

```
% lh_config --syslinux-splash FILENAME
```

として、ファイル名を指定します。

10.7.5 GRUB 起動時の splash 画面を変更する

```
% lh_config --grub-splash FILENAME
```

として、ファイル名を指定します

10.7.6 インタラクティブモード

live-helper では設定したあと、自動的に各イメージが作成されます。イメージ作成途中で、操作をしたいとき、インタラクティブモードに設定しておくことにより手動で細かい設定をすることが可能になります。

```
% lh_config --interactive enable
```

10.8 イメージの作成

作成した設定でイメージを作成するためには

```
# lh_build
```

を実行します。実行すると、イメージの作成を開始します。再度イメージを作成する場合は、

```
#lh_clean
```

を実行し、キャッシュをクリアしてから行います。

10.9 GUI を使った作成方法

live-helper は基本的に提供されているスクリプトを駆使して、イメージを作成しますが、GUI で作成するためのフロントエンドとして、live-magic というものが提供されています。簡単な使い方を説明します。

10.9.1 live-magic の起動

```
% sudo apt-get install live-magic
```

でインストールし、起動します。

起動した直後の画面を右に示します。



10.9.2 基本システムの選択

次に基本システムを選択します。選択可能なシステムは以下の通りです。

- Gnome
- KDE
- XFCE
- non-Desktop
- システム復旧用



図 11 live-magic 基本システムの選択

10.9.3 作成イメージの選択

次に作成するイメージを選択します。選択可能なイメージは以下の通りです。

- CD-ROM イメージ
- HDD イメージ
- NFS イメージ

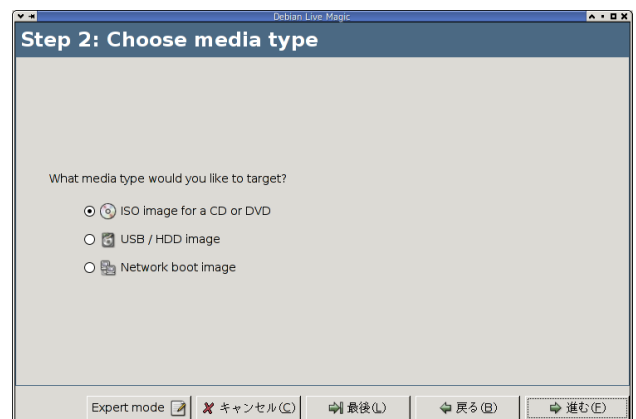


図 12 live-magic 作成イメージの選択

10.9.4 アーキテクチャの選択

次に対象のアーキテクチャを選択します。選択可能なアーキテクチャは以下の通りです。

- i386
- powerpc
- amd64

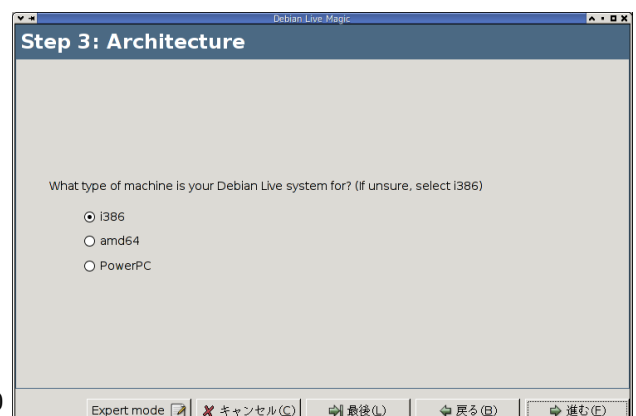


図 13 live-magic アーキテクチャの選択

10.9.5 ミラーサーバーの選択

次にミラーサーバーを選択します。

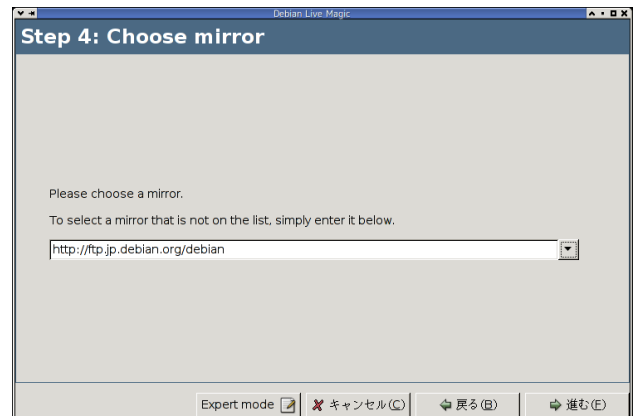


図 14 live-magic ミラーサーバーの選択

10.9.6 イメージ作成実行

適用ボタンを押すと、イメージ作成を実行します。

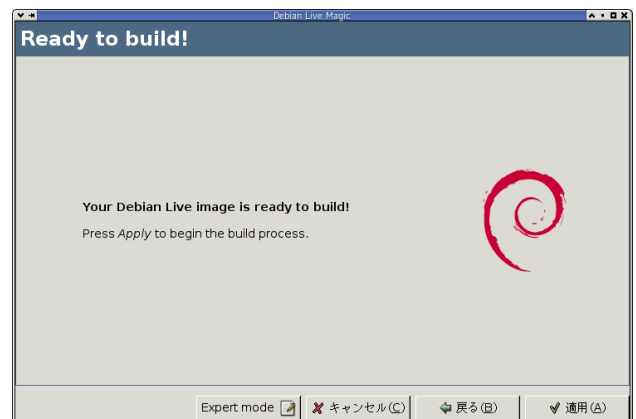


図 15 live-magic イメージ作成実行

10.9.7 root パスワード要求

一般ユーザーで実行した場合、root のパスワードを要求されます。

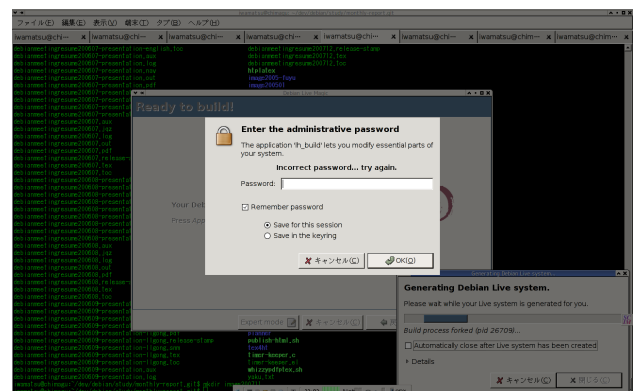


図 16 live-magic root パスワード要求画面

10.9.8 作成中画面

イメージ作成中はプログレスバーが表示され、途中経過を確認することができます。

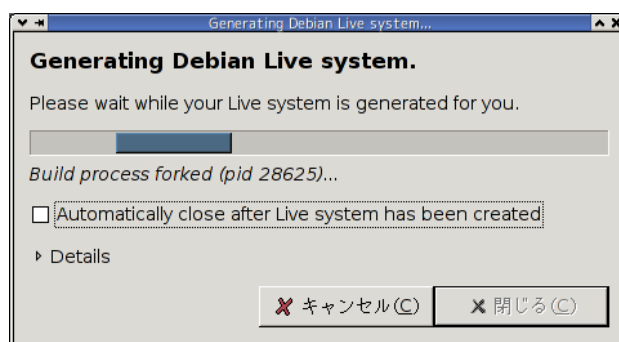


図 17 live-magic 作成中画面

10.9.9 デバッグ方法

動作確認やデバッグのために、毎回作成した CD/DVD ISO イメージを 焼いていると時間もお金もかかりますので、作成した ISO イメージのデバッグ方法について簡単に説明したいと思います。

デバッグ方法はいろいろありますが、自分は qemu^{*11}を使って動作確認しています。

```
# apt-get update
# apt-get install qemu
% qemu -cdrom binary.iso
```

qemu を使ったエミュレーション環境でデバッグすることにより、時間や CD-R 代の節約にもなります。他の方法としては、VMWare, VirtualPC を使ったデバッグ方法なども考えられます。

10.10 live-helper を使ってみて

10.10.1 Live-CD からのインストールのサポート

現状、Live-CD からのインストールがサポートされていません。HDD イメージを作成することはできますが、一般ユーザーには敷居が高いと考えています。Live-CD が気に入ったなら、動作している PC にインストールが容易にできるようになればユーザーは増えるのではないのでしょうか。

10.10.2 日本語環境のサポート

live-helper には、ある程度の環境をまとめたものとして、packages-selections というものが提供されています。これに日本語環境もサポートに入れることによって、日本語対応の Live-CD 作成が容易になると思います。

10.10.3 live-magic の日本語化

live-helper のフロントエンドである live-magic が英語のままなので、国際化をしたいところです。

10.11 その他の情報

live-helper の情報は以下のサイトから得ることができます。

1. live-helper 公式サイト <http://debian-live.alioth.debian.org/>
2. wiki.debian.org <http://wiki.debian.org>

^{*11} <http://packages.debian.org/sid/qemu>

11 Debian パッケージの作り方 (1) – 20 分で作る Debian パッケージ –

大浦 真 (Debian Project)

11.1 はじめに

テーマ 人は何分で Debian パッケージを作ることができるのか。(説明をしながら)

題材 GNU Hello (<http://www.gnu.org/software/hello/>)

11.2 手順

1. 必要なパッケージのインストール。
 - **build-essential**
 - そのソフトウェアのビルドに必要なパッケージ。
 - **dh-make**、**debhelper**、**devscripts**、**fakeroot**、**lintian/linda**
2. パッケージを作りたいソフトウェアのアーカイブを用意し展開。
3. **dh-make** でソースツリーの雛型を作成。
4. ひとまず **debuild** でパッケージを作ってみる。
5. うまくいかない場合は、ログや **./debian/rules** ファイルを確認。
6. うまくできた場合は、**./debian/** 以下のファイルを確認。
 - **./debian/changelog**: Debian パッケージの更新履歴。
 - **./debian/control**: Debian パッケージの情報。(依存関係や説明文)
 - **./debian/copyright**: ソフトウェアのライセンスの説明。
 - **./debian/*.ex**: 必要になるかもしれないファイルの雛型。必要なければそのまま削除。
7. **lintian** コマンドの出力を確認し、修正。
8. インストール。
 - 生成されるファイル
 - ***orig.tar.gz** オリジナルのソースアーカイブ。
 - ***diff.gz** オリジナルのソースと Debian ソースパッケージとの間の差分。
 - ***dsc** Debian ソースパッケージの情報。
 - ***i386.deb** バイナリパッケージ。
 - ***i386.changes** Debian パッケージの情報と最新の変更履歴。
 - ***i386.build** **debuild** のログ。

11.3 参考文献

- 「入門 Debian パッケージ」(やまだあきら [著]、鵜飼 文敏 [監修]) 技術評論社 (ISBN-10: 477412768X)
- Debian Project の Web ページ内の「Debian 開発者のコーナー」(<http://www.jp.debian.org/devel/>)。
特に、「Debian ポリシーマニュアル」「デベロッパーズリファレンス」「新規メンテナのためのガイド」

12 Debian パッケージの作り方 (2) – debian/rules を読む –

大浦 真 (Debian Project)

12.1 はじめに

テーマ debian/rules ファイルを読んでもみる。

題材 hello-debhelper パッケージのソースパッケージ。debhelper の簡単なサンプルになっている。

12.2 debian/rules とは

- Debian パッケージを作るための手続きを記述したファイル。
 - makefile になっている。
 - 一行目は `#!/usr/bin/make -f` となっていなければならない。
 - 最低限、`clean`、`binary`、`binary-arch`、`binary-indep`、`build` の五つのターゲットを含んでいなければならない。
 - 必須のターゲット以外は自由に利用していい。
- build** ソースパッケージの設定とコンパイルを行うためのターゲット
- binary**、**binary-arch**、**binary-indep** バイナリパッケージを作るためのターゲット。`binary-arch` は、アーキテクチャに依存したパッケージ、`binary-indep` は、アーキテクチャに依存しないパッケージを作る。`binary` はその両方のパッケージを作る。
- clean** バイナリパッケージの作成後、ソースパッケージを元の状態に戻すためのターゲット。
- install** ソースパッケージのコンパイル後、規定の位置にファイルを置くためのターゲット。`build` ターゲットで作成されたファイルは一旦、`debian/$(package)/` 以下にインストールされる。
- configure** ソースパッケージの設定を行うためのターゲット。
- get-orig-source** オリジナルのソースパッケージを取得するためのターゲット。あまり知られていない。
- それぞれのターゲットを実行するための手助けをする `debhelper` という一連のスクリプトがある。
 - 全て `dh_` で始まる名前の `perl` スクリプト。
 - `debhelper(7)` にスクリプトの一覧がある。
- dh_clean** ソースパッケージの中のいらないファイルを削除する。
- dh_installdirs** `debian/$(package)/` 以下に必要なディレクトリを作成する。
- dh_installdocs**、**dh_installchangelog** `usr/share/doc/` 以下にドキュメントや `changelog` をインストール。
- などなど。

12.3 参考文献

- Debian Policy (<http://www.jp.debian.org/doc/debian-policy/>) 「4.9 Main building script: debian/rules」

13 Debian パッケージの作り方 (3) – dpatch の使い方 –

大浦真 (Debian Project)

13.1 はじめに

テーマ Debian パッケージの patch 管理システムである dpatch を試してみる。

素材 hello-debhelper パッケージのソースパッケージ。メッセージを変更する patch を作ってみる。

13.2 Debian パッケージと patch

- Debian のソースパッケージは三つのファイルから出来ている。
 - *.orig.tar.gz オリジナルのソースアーカイブ。
 - *.diff.gz オリジナルのソースと Debian ソースパッケージとの間の差分。
 - *.dsc Debian ソースパッケージの情報。
- debian/ 以下のファイルも含めて、オリジナルとの差分は全て.diff.gz にまとめられるので、Debian 側で patch を複数あてている場合は管理が面倒。
- dpatch を使うと、patch を分割して管理することができる。オリジナルが更新された時の追従も簡単。
- debian/patches/ 以下で patch をまとめて管理。
- 移行に .orig.tar.gz の修正は不要。

13.3 dpatch の使い方

- debian/rules の修正
 - 冒頭に include /usr/share/dpatch/dpatch.make を追加。
 - build ターゲットで patch ターゲットが、clean ターゲットで unpatch ターゲットが実行されるように変更する。

```
#!/usr/bin/make -f
...
include /usr/share/dpatch/dpatch.make
clean: unpatch
...
install: build
...
build: patch
...
```

- debian/control の Build-Depends: に dpatch を追加。
- dpatch-convert-diffgz を使うと .diff.gz の中の debian/* 以外のファイルに対する patch を抽出できる。
- dpatch-edit-patch で patch を作成。

```
$ dpatch-edit-patch 01_hello_japan
( ... 自動的に /tmp 以下にコピーが作られ、新しいシェルが起動する。 )
( ... ソースツリーを修正。 )
$ exit
$ ( ... debian/patches/ 以下に patch ができる。 )
```

既存の patch の編集もできる。

- できた patch (上例では、debian/patches/01_hello_japan.dpatch) に適当なコメントを書いておく。
- patch を debian/patches/00list に追加。ファイル名の .dpatch を除いた文字列を追加。
- patch がきちんと適用できるかどうかは、fakeroot ./debian/rules patch、fakeroot ./debian/rules unpatch で確認できる。
- dpatch-list-patch で patch の一覧を確認できる。

13.4 類似のシステム

- dbs: オリジナルのソースが .tar.gz の形でそのまま .orig.tar.gz に含まれる。移行には、.orig.tar.gz の修正が必要。
- quilt

13.5 参考文献

- dpatch の利用方法 (<http://www.netfort.gr.jp/~dancer/column/dpatch.html.ja>)
- dpatch(1), dpatch.make(7), /usr/share/doc/dpatch/examples/rules/rules.new.dh.gz
- 「入門 Debian パッケージ」(やまだあきら [著]、鵜飼 文敏 [監修]) 技術評論社 (ISBN-10: 477412768X)

14 Debian パッケージの作り方 (4) – ライブラリ編 –

大浦 真 (Debian Project)

14.1 はじめに

テーマ ライブラリの Debian パッケージ作成を実演してみる。普通のパッケージよりも少し複雑。

素材 libhello (“Hello, library world.” と表示する関数 hello を提供するライブラリ)

14.2 基本的な手順

1. 必要なパッケージのインストール。
 - **build-essential**
 - そのソフトウェアのビルドに必要なパッケージ。
 - **dh-make**、**debhelper**、**devscripts**、**fakeroot**、**lintian/linda**
2. パッケージを作りたいソフトウェアのアーカイブを用意し展開。
3. **dh_make** でソースツリーの雛型を作成。
4. ひとまず **debuild** でパッケージを作ってみる。
5. うまいいかない場合は、ログや **debian/rules** ファイルを確認。
6. うまくできた場合は、**lintian** の出力や **debian/** 以下のファイルを確認し、さらに修正。

14.3 ライブラリパッケージの作り方

- ライブラリパッケージは、共有ライブラリ本体が含まれるパッケージ (**libhello0**) と開発用のファイルが含まれるパッケージ (**libhello-dev**) に分かれている。
- ライブラリの場合も、**dh_make** で雛型を作ることができるがそのままではパッケージができない。
- **debian/control** の修正
 - **dh_make** の指示通り、**‘libhelloBROKEN’** となっている部分を修正。この場合は、**‘libhello0’** にする。
- **debian/rules** の修正
 - **‘\$(MAKE) distclean’** の行を **‘[! -f Makefile] || \$(MAKE) distclean’** に変更。
 - **dh_install** に **‘-sourcedir=debian/tmp’** というオプションを付ける必要がある^{*12}。
 - **dh_makeshlibs** を有効にする。-V オプションを付ける。有効にすると、適切な **postinst/postrm** スクリプトが用意される。(install 時と remove 時に **ldconfig** が実行される。)

^{*12} かつては **dh_movefiles** というものがあったが、現在は使われていない。

- その他
 - libhello1.dirs と libhello1.install を libhello0 に名前を変える。
 - libhello-dev.install から **pkg-config** 関係を削除。
 - 不要な debian/*.ex を削除。

14.4 共有ライブラリの名前

- 共有ライブラリは、三つの名前を持つ。

soname (/usr/lib/libhello.so.0): 実行ファイルが参照する名前。ライブラリのインターフェイスが変更されると末尾の数字が変わる。Debian ではこの数字をパッケージ名に付加する。

real name (/usr/lib/libhello.so.0.0.0): ライブラリの実体。soname はこのファイルへのシンボリックリンクになっている。

linker name (/usr/lib/libhello.so): ライブラリを利用するプログラムをコンパイルするときに参照する名前。

14.5 参考文献

- <http://www.linux.or.jp/JF/JFdocs/Program-Library-HOWTO/> 「Program Library HOWTO」(JF)
- <http://www.netfort.gr.jp/~dancer/column/libpkg-guide/libpkg-guide.html> 「Debian Library Packaging guide」

14.6 サンプル

```

libhello.c

#include <stdio.h>

void hello(void) {
    printf("Hello, library world.\n");
}

```

```

libhello.h

void hello(void);

```

```

demo_use.c

#include "libhello.h"

int main(void) {
    hello();
    return 0;
}

```


15 HP ML350G5 Debian etch 動作確認

上川 純一

15.1 サーバ

HP ML350G5 は Intel Xeon CPU を搭載している Debian 4.0 (etch) が稼働することがハードウェアベンダ、および Debian プロジェクトの有志によって確認されているサーバ^{*13} ^{*14} です。今回利用したサーバには 300GB の SAS ディスクが 6 本搭載されていました。

15.2 インストール前の準備

サーバの起動時に「F8」を押し、ハードウェア RAID 機能の設定を行います。今回は一つの RAID5 のボリューム (約 1.5TB) として OS に見せる設定にしました。

15.3 インストール

Debian installer でインストールします。今回は Debian GNU/Linux 4.0r1 DVD の 1 枚目を利用しました。ML350G5 は em64t 対応の Xeon のため、i386 でも amd64 でも動作します。今回は amd64 をインストールしました。

15.4 デバイスの認識

各種デバイスは簡単に稼働します。グラフィックカードは ati ドライバで動作します。debconf で自動設定した値で適切な設定ファイルが出力されます。

```
# dpkg-reconfigure xserver-xorg
```

15.5 USB ストレージデバイスの認識

手元にあった「Green house GH-UFD2GR」の USB メモリで動作確認したところ、認識しました。

dmesg は次のようになりました：

^{*13} HP の ProLiant の Debian GNU/Linux 対応ページ: <http://www.hp.com/go/debian>

^{*14} Debian の ProLiant on Debian ページ: <http://wiki.debian.org/HP/ProLiant>

```

usb 6-6: new high speed USB device using ehci_hcd and address 4
usb 6-6: configuration #1 chosen from 1 choice
Initializing USB Mass Storage driver...
scsi0 : SCSI emulation for USB Mass Storage devices
usbcore: registered new driver usb-storage
USB Mass Storage support registered.
usb-storage: device found at 4
usb-storage: waiting for device to settle before scanning
Vendor: USBDisk Model: FlashDisk Rev: 1.00
Type: Direct-Access ANSI SCSI revision: 02
usb-storage: device scan complete
SCSI device sda: 4005000 512-byte hdwr sectors (2051 MB)
sda: Write Protect is off
sda: Mode Sense: 0b 00 00 08
sda: assuming drive cache: write through
SCSI device sda: 4005000 512-byte hdwr sectors (2051 MB)
sda: Write Protect is off
sda: Mode Sense: 0b 00 00 08
sda: assuming drive cache: write through
sda: sda1
sd 0:0:0:0: Attached scsi removable disk sda
FAT: utf8 is not a recommended IO charset for FAT filesystems, filesystem will be case sensitive!

```

df で確認しても認識されているのがわかります：

```

# df -h /media/usbdisk/
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        2.0G   76M  1.9G   4% /media/usbdisk
# df -T /media/usbdisk/
Filesystem      Type  1K-blocks      Used Available Use% Mounted on
/dev/sda1       vfat   2001888       76864   1925024   4% /media/usbdisk

```

15.6 USB webcam の認識

PlayStation2 用 USB カメラ EyeToy を動作させてみました。ov51x-jpeg のデバイスドライバを認識させるためのドライバは残念ながら Debian 4.0 には含まれていないので、最新のドライバをバックポートしてみました。



図 18 EyeToy カメラ

まず、インストールされたカーネルでコンパイルするために必要なパッケージをインストールします。

```

apt-get install linux-headers-2.6.18-5-amd64 linux-kbuild-2.6.18 \
linux-source-2.6.18

```

ov51x-jpeg のソースを取得して make コマンドでビルドできます。

```
debian:/home/hoge/ov51x-jpeg# make
make -C /lib/modules/2.6.18-5-amd64/build M=/home/hoge/ov51x-jpeg modules
make[1]: Entering directory '/usr/src/linux-headers-2.6.18-5-amd64'
CC [M] /home/hoge/ov51x-jpeg/ov51x-jpeg-core.o
CC [M] /home/hoge/ov51x-jpeg/ov511-decomp.o
CC [M] /home/hoge/ov51x-jpeg/ov518-decomp.o
CC [M] /home/hoge/ov51x-jpeg/ov519-decomp.o
LD [M] /home/hoge/ov51x-jpeg/ov51x-jpeg.o
Building modules, stage 2.
MODPOST
CC /home/hoge/ov51x-jpeg/ov51x-jpeg.mod.o
LD [M] /home/hoge/ov51x-jpeg/ov51x-jpeg.ko
make[1]: Leaving directory '/usr/src/linux-headers-2.6.18-5-amd64'
```

生成されたモジュールをあわせて insmod すれば ekiga で画面が見れるようになります。

```
insmod v4l1-compat.ko
insmod v4l2-common.ko
insmod videodev.ko
insmod ov51x-jpeg.ko
```

15.7 謝辞

OSC Tokyo/Fall 2007 Debian ブースにて本作業を行いました。サーバは日本 HP からお借りしました。



図 19 検証時のブースの様子

15.8 付録

15.8.1 デバイス接続

```
# lspci
00:00.0 Host bridge: Intel Corporation 5000Z Chipset Memory Controller Hub (rev b1)
00:02.0 PCI bridge: Intel Corporation 5000 Series Chipset PCI Express x8 Port 2-3 (rev b1)
00:03.0 PCI bridge: Intel Corporation 5000 Series Chipset PCI Express x4 Port 3 (rev b1)
00:04.0 PCI bridge: Intel Corporation 5000 Series Chipset PCI Express x4 Port 4 (rev b1)
00:05.0 PCI bridge: Intel Corporation 5000 Series Chipset PCI Express x4 Port 5 (rev b1)
00:10.0 Host bridge: Intel Corporation 5000 Series Chipset Error Reporting Registers (rev b1)
00:10.1 Host bridge: Intel Corporation 5000 Series Chipset Error Reporting Registers (rev b1)
00:10.2 Host bridge: Intel Corporation 5000 Series Chipset Error Reporting Registers (rev b1)
00:11.0 Host bridge: Intel Corporation 5000 Series Chipset Reserved Registers (rev b1)
00:13.0 Host bridge: Intel Corporation 5000 Series Chipset Reserved Registers (rev b1)
00:15.0 Host bridge: Intel Corporation 5000 Series Chipset FBD Registers (rev b1)
00:16.0 Host bridge: Intel Corporation 5000 Series Chipset FBD Registers (rev b1)
00:1c.0 PCI bridge: Intel Corporation 631xESB/632xESB/3100 Chipset PCI Express Root Port 1 (rev 09)
00:1d.0 USB Controller: Intel Corporation 631xESB/632xESB/3100 Chipset UHCI USB Controller #1 (rev 09)
00:1d.1 USB Controller: Intel Corporation 631xESB/632xESB/3100 Chipset UHCI USB Controller #2 (rev 09)
00:1d.2 USB Controller: Intel Corporation 631xESB/632xESB/3100 Chipset UHCI USB Controller #3 (rev 09)
00:1d.3 USB Controller: Intel Corporation 631xESB/632xESB/3100 Chipset UHCI USB Controller #4 (rev 09)
00:1d.7 USB Controller: Intel Corporation 631xESB/632xESB/3100 Chipset EHCI USB2 Controller (rev 09)
00:1e.0 PCI bridge: Intel Corporation 82801 PCI Bridge (rev d9)
00:1f.0 ISA bridge: Intel Corporation 631xESB/632xESB/3100 Chipset LPC Interface Controller (rev 09)
00:1f.1 IDE interface: Intel Corporation 631xESB/632xESB IDE Controller (rev 09)
01:03.0 VGA compatible controller: ATI Technologies Inc ES1000 (rev 02)
01:04.0 System peripheral: Compaq Computer Corporation Integrated Lights Out Controller (rev 03)
01:04.2 System peripheral: Compaq Computer Corporation Integrated Lights Out Processor (rev 03)
01:04.4 USB Controller: Hewlett-Packard Company Unknown device 3300
01:04.6 Serial bus controller [0c07]: Hewlett-Packard Company Unknown device 3302
02:00.0 PCI bridge: Broadcom EPB PCI-Express to PCI-X Bridge (rev c3)
03:00.0 Ethernet controller: Broadcom Corporation NetXtreme II BCM5708 Gigabit Ethernet (rev 12)
04:00.0 PCI bridge: Intel Corporation 6311ESB/6321ESB PCI Express Upstream Port (rev 01)
04:00.3 PCI bridge: Intel Corporation 6311ESB/6321ESB PCI Express to PCI-X Bridge (rev 01)
05:00.0 PCI bridge: Intel Corporation 6311ESB/6321ESB PCI Express Downstream Port E1 (rev 01)
05:01.0 PCI bridge: Intel Corporation 6311ESB/6321ESB PCI Express Downstream Port E2 (rev 01)
12:00.0 PCI bridge: Broadcom EPB PCI-Express to PCI-X Bridge (rev b4)
13:04.0 PCI bridge: Broadcom HT1000 PCI/PCI-X bridge (rev b2)
13:08.0 RAID bus controller: Hewlett-Packard Company Unknown device 3238

# lspci -n
00:00.0 0600: 8086:25d0 (rev b1)
00:02.0 0604: 8086:25f7 (rev b1)
00:03.0 0604: 8086:25e3 (rev b1)
00:04.0 0604: 8086:25e4 (rev b1)
00:05.0 0604: 8086:25e5 (rev b1)
00:10.0 0600: 8086:25f0 (rev b1)
00:10.1 0600: 8086:25f0 (rev b1)
00:10.2 0600: 8086:25f0 (rev b1)
00:11.0 0600: 8086:25f1 (rev b1)
00:13.0 0600: 8086:25f3 (rev b1)
00:15.0 0600: 8086:25f5 (rev b1)
00:16.0 0600: 8086:25f6 (rev b1)
00:1c.0 0604: 8086:2690 (rev 09)
00:1d.0 0c03: 8086:2688 (rev 09)
00:1d.1 0c03: 8086:2689 (rev 09)
00:1d.2 0c03: 8086:268a (rev 09)
00:1d.3 0c03: 8086:268b (rev 09)
00:1d.7 0c03: 8086:268c (rev 09)
00:1e.0 0604: 8086:244e (rev d9)
00:1f.0 0601: 8086:2670 (rev 09)
00:1f.1 0101: 8086:269e (rev 09)
01:03.0 0300: 1002:515e (rev 02)
01:04.0 0880: 0e11:b203 (rev 03)
01:04.2 0880: 0e11:b204 (rev 03)
01:04.4 0c03: 103c:3300
01:04.6 0c07: 103c:3302
02:00.0 0604: 1166:0103 (rev c3)
03:00.0 0200: 14e4:164c (rev 12)
04:00.0 0604: 8086:3500 (rev 01)
04:00.3 0604: 8086:350c (rev 01)
05:00.0 0604: 8086:3510 (rev 01)
05:01.0 0604: 8086:3514 (rev 01)
12:00.0 0604: 1166:0103 (rev b4)
13:04.0 0604: 1166:0104 (rev b2)
13:08.0 0104: 103c:3238
```

15.8.2 xorg.conf

debconf で自動生成された設定です。

```

# /etc/X11/xorg.conf (xorg X Window System server configuration file)
#
# This file was generated by dexconf, the Debian X Configuration tool, using
# values from the debconf database.
#
# Edit this file with caution, and see the /etc/X11/xorg.conf manual page.
# (Type "man /etc/X11/xorg.conf" at the shell prompt.)
#
# This file is automatically updated on xserver-xorg package upgrades *only*
# if it has not been modified since the last upgrade of the xserver-xorg
# package.
#
# If you have edited this file but would like it to be automatically updated
# again, run the following command:
#   sudo dpkg-reconfigure -phigh xserver-xorg

Section "Files"
    FontPath        "/usr/share/fonts/X11/misc"
    FontPath        "/usr/X11R6/lib/X11/fonts/misc"
    FontPath        "/usr/share/fonts/X11/cyrillic"
    FontPath        "/usr/X11R6/lib/X11/fonts/cyrillic"
    FontPath        "/usr/share/fonts/X11/100dpi:unscaled"
    FontPath        "/usr/X11R6/lib/X11/fonts/100dpi:unscaled"
    FontPath        "/usr/share/fonts/X11/75dpi:unscaled"
    FontPath        "/usr/X11R6/lib/X11/fonts/75dpi:unscaled"
    FontPath        "/usr/share/fonts/X11/Type1"
    FontPath        "/usr/X11R6/lib/X11/fonts/Type1"
    FontPath        "/usr/share/fonts/X11/100dpi"
    FontPath        "/usr/X11R6/lib/X11/fonts/100dpi"
    FontPath        "/usr/share/fonts/X11/75dpi"
    FontPath        "/usr/X11R6/lib/X11/fonts/75dpi"
    # path to defoma fonts
    FontPath        "/var/lib/defoma/x-ttcidfont-conf.d/dirs/TrueType"
EndSection

Section "Module"
    Load            "bitmap"
    Load            "ddc"
    Load            "dri"
    Load            "extmod"
    Load            "freetype"
    Load            "glx"
    Load            "int10"
    Load            "vbe"
EndSection

Section "InputDevice"
    Identifier       "Generic Keyboard"
    Driver           "kbd"
    Option           "CoreKeyboard"
    Option           "XkbRules"            "xorg"
    Option           "XkbModel"           "pc104"
    Option           "XkbLayout"          "us"
    Option           "XkbOptions"          "ctrl:nocaps"
EndSection

Section "InputDevice"
    Identifier       "Configured Mouse"
    Driver           "mouse"
    Option           "CorePointer"
    Option           "Device"              "/dev/input/mice"
    Option           "Protocol"            "ImPS/2"
    Option           "Emulate3Buttons"     "true"
EndSection

Section "Device"
    Identifier       "ATI Technologies Inc ES1000"
    Driver           "ati"
    BusID            "PCI:1:3:0"
EndSection

Section "Monitor"
    Identifier       "17 ANALOG MO"
    Option           "DPMS"
    HorizSync        28-50
    VertRefresh       43-75
EndSection

```

```

Section "Screen"
    Identifier      "Default Screen"
    Device          "ATI Technologies Inc ES1000"
    Monitor         "17 ANALOG M0"
    DefaultDepth    24
    SubSection "Display"
        Depth      1
        Modes      "1280x1024" "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Depth      4
        Modes      "1280x1024" "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Depth      8
        Modes      "1280x1024" "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Depth      15
        Modes      "1280x1024" "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Depth      16
        Modes      "1280x1024" "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Depth      24
        Modes      "1280x1024" "1024x768" "800x600" "640x480"
    EndSubSection
EndSection

Section "ServerLayout"
    Identifier      "Default Layout"
    Screen          "Default Screen"
    InputDevice     "Generic Keyboard"
    InputDevice     "Configured Mouse"
EndSection

Section "DRI"
    Mode           0666
EndSection

```

16 HP ML110G4 Debian etch 動作確認

上川 純一

16.1 サーバ

HP の ML110G4 は Celeron CPU を搭載したサーバモデルです。今回利用したサーバは SATA 接続のディスクを搭載していました。このハードウェア上で Debian etch 4.0r1 が動作することは有志により確認されています^{*15}。ベンダーとしては特に Debian の動作確認はしていないようです。

16.2 インストール前の準備

特に必要ないようです。

16.3 インストール

Debian installer でインストールします。今回は Debian GNU/Linux 4.0r1 DVD の 1 枚目を利用しました。i386 のインストール CD を利用しました。

16.4 デバイスの認識

グラフィックデバイスを自動認識できず、VESA モードで動作します。MGA のカードなので、mga ドライバで動作します。メモリが少ないため、デフォルトでは 640x480x24 で動作するため若干画面が狭いです。色を 16bit に減らすと 1024x768x16 で動作しました。

SZ:	Pixels	Physical	Refresh
*0	1024 x 768	(271mm x 203mm)	*60
1	800 x 600	(271mm x 203mm)	75 72 60 56
2	640 x 480	(271mm x 203mm)	75 73 60
3	832 x 624	(271mm x 203mm)	75
4	640 x 400	(271mm x 203mm)	60
5	640 x 384	(271mm x 203mm)	60
6	576 x 384	(271mm x 203mm)	55
7	512 x 384	(271mm x 203mm)	60
8	416 x 312	(271mm x 203mm)	75
9	400 x 300	(271mm x 203mm)	75 72 60 56
10	320 x 240	(271mm x 203mm)	75 73 60

Current rotation - normal
Current reflection - none
Rotations possible - normal
Reflections possible - none

16.5 謝辞

OSC Tokyo/Fall 2007 Debian ブースにて本作業を行いました。サーバはびぎねっとからお借りしました。

^{*15} Debian の ProLiant on Debian ページ: <http://wiki.debian.org/HP/ProLiant>

16.6 付録

16.6.1 デバイス接続

```
# lspci
00:00.0 Host bridge: Intel Corporation E7230 Memory Controller Hub (rev c0)
00:1c.0 PCI bridge: Intel Corporation 82801G (ICH7 Family) PCI Express Port 1 (rev 01)
00:1c.4 PCI bridge: Intel Corporation 82801GR/GH/GHM (ICH7 Family) PCI Express Port 5 (rev 01)
00:1c.5 PCI bridge: Intel Corporation 82801GR/GH/GHM (ICH7 Family) PCI Express Port 6 (rev 01)
00:1d.0 USB Controller: Intel Corporation 82801G (ICH7 Family) USB UHCI #1 (rev 01)
00:1d.1 USB Controller: Intel Corporation 82801G (ICH7 Family) USB UHCI #2 (rev 01)
00:1d.2 USB Controller: Intel Corporation 82801G (ICH7 Family) USB UHCI #3 (rev 01)
00:1d.3 USB Controller: Intel Corporation 82801G (ICH7 Family) USB UHCI #4 (rev 01)
00:1d.7 USB Controller: Intel Corporation 82801G (ICH7 Family) USB2 EHCI Controller (rev 01)
00:1e.0 PCI bridge: Intel Corporation 82801 PCI Bridge (rev e1)
00:1f.0 ISA bridge: Intel Corporation 82801GB/GR (ICH7 Family) LPC Interface Bridge (rev 01)
00:1f.1 IDE interface: Intel Corporation 82801G (ICH7 Family) IDE Controller (rev 01)
00:1f.2 RAID bus controller: Intel Corporation 82801GR/GH (ICH7 Family) Serial ATA Storage Controller RAID (rev 01)
03:00.0 VGA compatible controller: Matrox Graphics, Inc. MGA G200e [Pilot] ServerEngines (SEP1) (rev 02)
04:00.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5721 Gigabit Ethernet PCI Express (rev 21)
# lspci -n
00:00.0 0600: 8086:2778 (rev c0)
00:1c.0 0604: 8086:27d0 (rev 01)
00:1c.4 0604: 8086:27e0 (rev 01)
00:1c.5 0604: 8086:27e2 (rev 01)
00:1d.0 0c03: 8086:27c8 (rev 01)
00:1d.1 0c03: 8086:27c9 (rev 01)
00:1d.2 0c03: 8086:27ca (rev 01)
00:1d.3 0c03: 8086:27cb (rev 01)
00:1d.7 0c03: 8086:27cc (rev 01)
00:1e.0 0604: 8086:244e (rev e1)
00:1f.0 0601: 8086:27b8 (rev 01)
00:1f.1 0101: 8086:27df (rev 01)
00:1f.2 0104: 8086:27c3 (rev 01)
03:00.0 0300: 102b:0522 (rev 02)
04:00.0 0200: 14e4:1659 (rev 21)
```

16.6.2 xorg.conf

debconf で自動生成された設定に手をいれています。色を 16bit にしているのと、driver を mga に明示的に指定しています。


```

# /etc/X11/xorg.conf (xorg X Window System server configuration file)
#
# This file was generated by dexconf, the Debian X Configuration tool, using
# values from the debconf database.
#
# Edit this file with caution, and see the /etc/X11/xorg.conf manual page.
# (Type "man /etc/X11/xorg.conf" at the shell prompt.)
#
# This file is automatically updated on xserver-xorg package upgrades *only*
# if it has not been modified since the last upgrade of the xserver-xorg
# package.
#
# If you have edited this file but would like it to be automatically updated
# again, run the following command:
#   sudo dpkg-reconfigure -phigh xserver-xorg

Section "Files"
    FontPath            "/usr/share/fonts/X11/misc"
    FontPath            "/usr/X11R6/lib/X11/fonts/misc"
    FontPath            "/usr/share/fonts/X11/cyrillic"
    FontPath            "/usr/X11R6/lib/X11/fonts/cyrillic"
    FontPath            "/usr/share/fonts/X11/100dpi:unscaled"
    FontPath            "/usr/X11R6/lib/X11/fonts/100dpi:unscaled"
    FontPath            "/usr/share/fonts/X11/75dpi:unscaled"
    FontPath            "/usr/X11R6/lib/X11/fonts/75dpi:unscaled"
    FontPath            "/usr/share/fonts/X11/Type1"
    FontPath            "/usr/X11R6/lib/X11/fonts/Type1"
    FontPath            "/usr/share/fonts/X11/100dpi"
    FontPath            "/usr/X11R6/lib/X11/fonts/100dpi"
    FontPath            "/usr/share/fonts/X11/75dpi"
    FontPath            "/usr/X11R6/lib/X11/fonts/75dpi"
    # path to defoma fonts
    FontPath            "/var/lib/defoma/x-ttcidfont-conf.d/dirs/TrueType"
EndSection

Section "Module"
    Load               "bitmap"
    Load               "ddc"
    Load               "dri"
    Load               "extmod"
    Load               "freetype"
    Load               "glx"
    Load               "int10"
    Load               "vbe"
EndSection

Section "InputDevice"
    Identifier          "Generic Keyboard"
    Driver              "kbd"
    Option              "CoreKeyboard"
    Option              "XkbRules"            "xorg"
    Option              "XkbModel"           "jp106"
    Option              "XkbLayout"          "j"
EndSection

Section "InputDevice"
    Identifier          "Configured Mouse"
    Driver              "mouse"
    Option              "CorePointer"
    Option              "Device"              "/dev/input/mice"
    Option              "Protocol"            "ImPS/2"
    Option              "Emulate3Buttons"     "true"
EndSection

Section "Device"
    Identifier          "Generic Video Card"
    Driver              "mga"
    BusID               "PCI:3:0:0"
EndSection

Section "Monitor"
    Identifier          "RDT153EM"
    Option              "DPMS"
    HorizSync           28-50
    VertRefresh          43-75
EndSection

```

```

Section "Screen"
    Identifier      "Default Screen"
    Device          "Generic Video Card"
    Monitor         "RDT153EM"
    DefaultDepth    16
    SubSection "Display"
        Depth      1
        Modes       "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Depth      4
        Modes       "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Depth      8
        Modes       "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Depth      15
        Modes       "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Depth      16
        Modes       "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Depth      24
        Modes       "1024x768" "800x600" "640x480"
    EndSubSection
EndSection

Section "ServerLayout"
    Identifier      "Default Layout"
    Screen          "Default Screen"
    InputDevice     "Generic Keyboard"
    InputDevice     "Configured Mouse"
EndSection

Section "DRI"
    Mode            0666
EndSection

```

17 Debian Weekly News trivia quiz

上川 純一

ところで、Debian Weekly News (DWN) は読んでいますか？Debian 界限でおきていることについて書いている Debian Weekly News。毎回読んでいるといろいろと分かって来ますが、一人で読んでいても、解説が少ないので、意味がわからないところもあるかも知れません。みんなで DWN を読んでみましょう。

漫然と読むだけではおもしろくないので、DWN の記事から出題した以下の質問にこたえてみてください。後で内容は解説します。

17.1 2007 年 6 号

<http://www.debian.org/News/weekly/2007/06/> にある 7 月 3 日版です。

問題 1. André Luiz Rodrigues Ferreira が宣言したのはどのウェブサイトか

- ☐ A Debian art
- ☐ B Debian pop
- ☐ C Debian tart

問題 2. Jülich で行われた会議で lenny のリリースプロセスについて何をすることが決まったか

- ☐ A 秘密のプロセスにのっとり、今後リリースがどうなっているかは非公開にする
- ☐ B 毎月か二ヶ月に一回の最終週にリリース状況についてのメールを出す
- ☐ C 安全保障のため今後は Debian Developer でないとリリースの状況がわからないようにする

問題 3. Lucas Nussbaum は毎月何をすると発表したか

- ☐ A 深刻な問題のあるパッケージを順番にのっとる
- ☐ B 深刻な問題のあるパッケージを管理している人に罰ゲームをさせる
- ☐ C 深刻な問題のあるパッケージについて通知するメールを自動で送付

問題 4. Alexander Wirt は何を発表したか

- ☐ A backports.org が sid に対応
- ☐ B backports.org が lenny に対応
- ☐ C backports.org が etch に対応

問題 5. Martin Michlmyr が挑戦しているのは何か

- ☐ A Debian を gcc 4.2 でビルドできるようにする
- ☐ B Debian を全部 C++ におきかえる
- ☐ C Debian を全部 ruby におきかえる

17.2 問題

今回の出題範囲は http://www.debian.org/vote/2007/vote_003 にある投票結果と、<http://lists.debian.org/debian-devel-announce/> にある最近のアナウンス文書です。

問題 6. Debian Maintainers の提案は何をするものか

- ☐ A 気に入らない Debian Developer を投票により追放する
- ☐ B Debian Developer より制限された権限をもつ Debian Maintainers を定義する
- ☐ C Debian Developer の品質を改善する

問題 7. Bits from the DPL: FTP assistants, DM, APT, sharing patches で Sam Hocever が主張したのは

- ☐ A パッチを共有しよう
- ☐ B もう会長としての仕事は終わった
- ☐ C Debian としては Ubuntu の殲滅が目標

問題 8. apt-get install の仕組みで大きな変化が発生したのは何か

- ☐ A Suggests をデフォルトでインストールするようになった
- ☐ B Recommends をデフォルトでインストールするようになった
- ☐ C Depends を無視するようになった

問題 9. lenny のリリースゴールに入っているのはどれか

- ☐ A Debian の市場シェア 40% 以上の獲得
- ☐ B debian/rules が国際化対応
- ☐ C debian/changelog と debian/control は UTF-8

問題 10. sparc32 になにがおきたか

- ☐ A 次のリリースではサポートされなくなる
- ☐ B 急にユーザが増えたので開発者を募集している
- ☐ C arm とバイナリ互換になった

17.3 問題

今回の出題範囲は debian-devel@lists.debian.org に投稿された内容からです。

問題 11. Albert Einstein が作った Debian ベースのディストリビューションは何か

- ☐ A ice linux
- ☐ B fantasy linux
- ☐ C fire linux

問題 12. そしてこの Albert Einstein が debian-devel で質問した内容は何でしょう

- ☐ A なぜ Internet Explorer が Debian にないのですか。
- ☐ B なぜ Opera が Debian にないのですか。
- ☐ C なぜ Safari が Debian にないのですか。

問題 13. Luk Claes が RC バグについて提案したのはどのような内容か

- ☐ A RC バグが出たパッケージのメンテナへのペナルティを考える提案
- ☐ B RC バグの 0-day NMU についての提案
- ☐ C RC バグをいかにして無視するか、という HowTo.

問題 14. packages.debian.org にいろいろ新機能が追加されました。どのような機能が追加されましたか

- ☐ A メールフォワード機能
- ☐ B カルマ付加機能
- ☐ C Web からパッケージ乗っ取り機能

17.4 問題

今回の出題範囲は `debian-devel-announce@lists.debian.org` に投稿された内容からです。

問題 15. 10/4 にアナウンスがあった alioth のサービスに追加されたものは？

- ☐ A VSS サポート
- ☐ B darcs サポート
- ☐ C p4 サポート

問題 16. DebianGis チームは何をするチームか？

- ☐ A Gis のパッケージのメンテナンス
- ☐ B Debian を Gis でのっとるプロジェクト
- ☐ C 人間関係をギスギスしてみる

問題 17. testing security のメールの仕組みで何が変わったか

- ☐ A unstable から testing へのマイグレーションでセキュリティーバグが修正されてもアナウンスされるようにした
- ☐ B 昨年度 Debian testing security team が CVE を 5500 も処理したことが自慢できるようになった
- ☐ C SMTP プロトコルのハンドシェイクが変わった

問題 18. Debconf8 の日程は

- ☐ A 5 月 1 日から 5 月 10 日
- ☐ B 8 月 2 日から 8 月 17 日
- ☐ C 12 月 24 日から 1 月 1 日

問題 19. `http://security-tracker.debian.net/tracker/` で何が見れるか

- ☐ A 手元のマシンが脆弱化どうかの試験
- ☐ B セキュリティーについての入門
- ☐ C セキュリティーバグの現在の状態

問題 20. Debian System Administrator として新しく任命されたのは誰か

- ☐ A Sven Luther
- ☐ B Phil Hands
- ☐ C Peter Palfrader

問題 21. ries.debian.org (ftp-master) はどれくらい停止していたか

☐ A 11 月 5 日から 11 月 12 日

☐ B 11 月 5 日から 11 月 30 日

☐ C 11 月 1 日から 11 月 5 日

18 Debian Weekly News 問題回答

上川 純一

Debian Weekly News の問題回答です。あなたは何問
わかりましたか？

1. A
2. B
3. C
4. C
5. A
6. B
7. A
8. B
9. C
10. B

11. A
12. B
13. B
14. A
15. B
16. A
17. A
18. B
19. C
20. C
21. A



19 索引

Akamai, 18
apt-build, 30
apt-cacher, 29
apt-cross, 30
apt-extracttemplates, 30
apt-ftpparchive, 29
apt-key, 28
apt-proxy, 29
apt-sortpkgs, 29
apt-spy, 28
apt-transport-https, 31
apt-xxx, 28
apt-zip, 31
aptsh, 31

Bittorrent, 18

cdn.debian.or.jp, 18
Content Delivery Network, 18
cron-apt, 29

Debconf, 2, 6, 16
Debconf2007, 2, 6
Debian GNU/kFreeBSD, 22
debian/rules, 45
Debian JP, 12, 18
dpatch, 47

exim, 25

FreeBSD, 22

live-helper, 34

ML110G4, 57
ML350G5, 51

Open Source Conference, 12

Winny, 18

関西 Debian 勉強会, 12



あんどきゅめんとてっど でびあん 2007 年冬号

2007 年 12 月 31 日 初版第 1 刷発行
東京エリア Debian 勉強会（編集・印刷・発行）
