第 0 回 福岡 Debian 勉強会資料

月刊 Deb専

日本唯一のDebian専門月刊誌

2012 年 2 月 18 日

特集1: XXXX

特集2: YYYY

1 Introduction

岩松 信洋

今月の Debian 勉強会へようこそ。これから Debian の世界にあしを踏み入れるという方も、すでにどっぷりとつかっているという方も、月に一回 Debian について語りませんか?

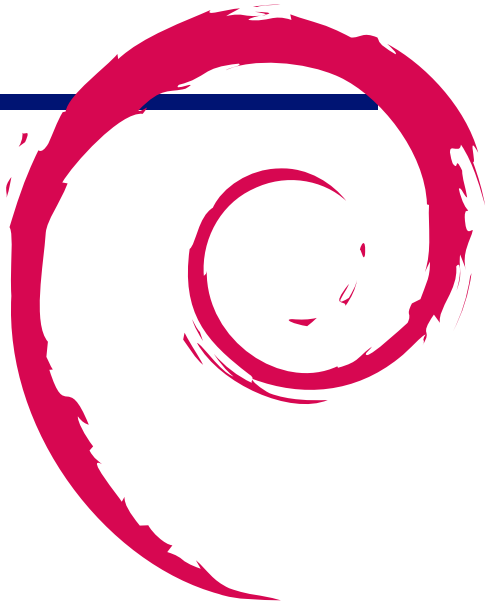
Debian 勉強会の目的は下記です。

- Debian Developer (開発者) の育成。
- 日本語での「開発に関する情報」を整理してまとめ、アップデートする。
- 場の提供。
  - 普段ばらばらな場所にいる人々が face-to-face で出会える場を提供する。
  - Debian のためになることを語る場を提供する。
  - Debian について語る場を提供する。

Debian の勉強会ということで究極的には参加者全員が Debian Package をがりがりと作るスーパーハッカーになった姿を妄想しています。情報の共有・活用を通して Debian の今後の能動的な展開への土台として、「場」としての空間を提供するのが目的です。

デビアン勉強会

目次



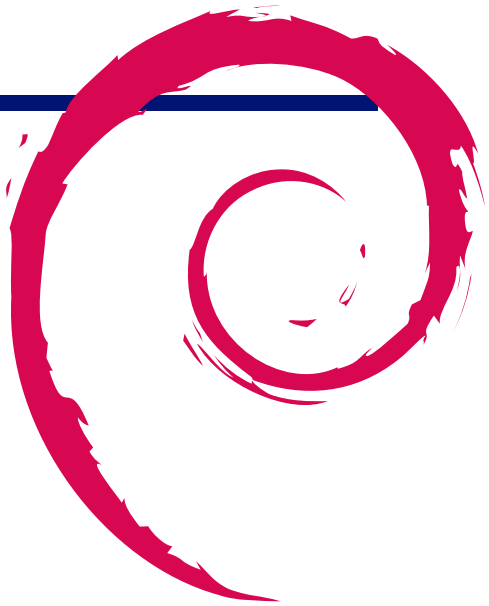
2 事前課題

岩松 信洋

今回の事前課題は以下です:

1. 2012 年の勉強会のテーマとして各月なにをするべきか提案してください。
2. 自分がどのテーマを担当したいか提案してください。

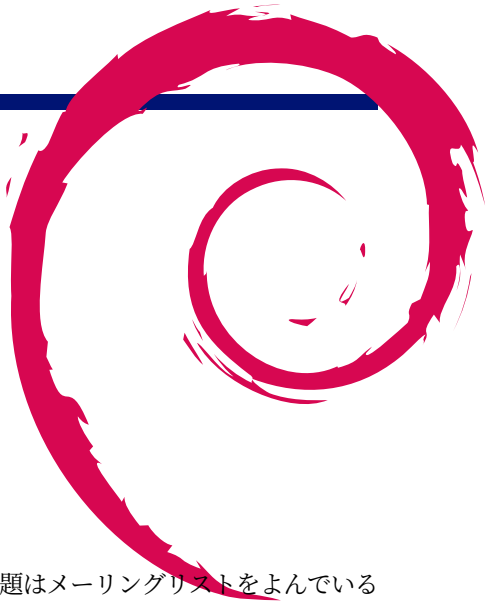
この課題に対して提出いただいた内容は以下です。



3 最近の Debian 関連のミーティング報告

岩松 信洋

3.1 東京エリア Debian 勉強会 84 回目報告

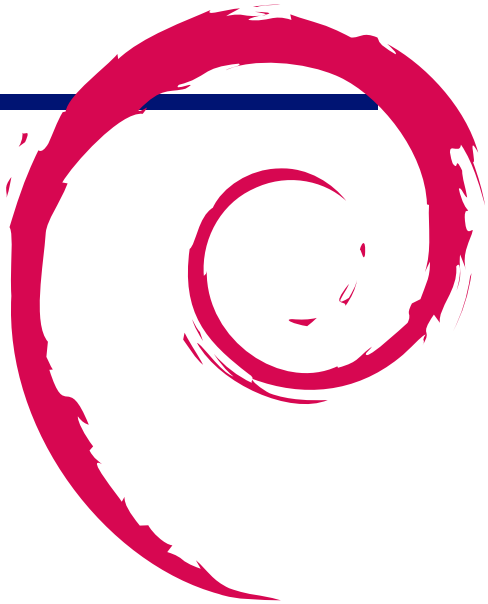


4 Debian Trivia Quiz

岩松 信洋

ところで、みなさん Debian 関連の話題においついていますか? Debian 関連の話題はメーリングリストをよんでいると追跡できます。ただよんでいるだけではりあいがないので、理解度のテストをします。特に一人だけでは意味がわからないところもあるかも知れません。みんなで一緒に読んでみましょう。

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



5 Dynamic Kernel Module Support Framework

岩松 信洋

5.1 Dynamic Kernel Module Support Framework とは

Dynamic Kernel Module Support Framework（ 以下、 DKMS）は、マシンにインストールされているカーネル毎に自動的にドライバモジュールをビルドし管理する仕組みです。

通常、 Linux カーネルが更新されたとき、 Linux カーネルで提供されていないサードパーティ のドライバはその Linux カーネル用にデバイスドライバを再ビルドおよびインストールを行う必要があります。しかし DKMS を使ってサードパーティ のデバイスドライバを管理している場合、新しい Linux カーネルで起動された時に対応したデバイスドライバがないことをチェックし、ドライバモジュールをビルドして（ 自動インストールが設定されていれば）インストールします。もちろんそのドライバは Linux カーネルに対応している必要がありますが、ドライバがメンテナンスされていれば、半自動的に新しいカーネルへ移行できるようになります。

DKMS は DELL `http://linux.dell.com/dkms/` で開発されており、 Debian や Ubuntu などののディストリビューションで利用できるようになっています。

ちなみに Fedora は Kmods2 `http://rpmfusion.org/Packaging/KernelModules/Kmods2` と呼ばれる別のフレームワークを利用しています。 Arch Linux と Gentoo も DKMS サポートパッケージもあるが、独自の実装が中心のようです。

今回は DKMS の仕組みと DKMS に対応したパッケージの作成方法、 Debian の DKMS 対応状況について説明します。

5.2 DKMS の仕組み

DKMS の仕組みを理解するために簡単なデバイスドライバを使って説明します。ドライバをロードするとカーネルデバッグメッセージに「 Hello, world! from hello_init」、ドライバをアンロードすると「 Good bye! from hello_exit」と出力するものです。以下に実際のソースコード、 Makefile、実行結果とディレクトリ構成を示します。

ドライバソースコード:

```
#include <linux/init.h>
#include <linux/module.h>

static int __init hello_init(void)
{
  pr_info("Hello, world! from %s\n", __func__);
  return 0;
}

static void __exit hello_exit(void)
{
  pr_info("Good bye! from %s\n", __func__);
}

module_init(hello_init);
module_exit(hello_exit);
MODULE_LICENSE("GPL v2");
```

Makefile:

```
obj-m:= src/
```

src/Makefile:

```
obj-m:= hello.o
```

ドライバのコンパイルと実行結果:

```
$ make -C /lib/modules/‘uname -r‘/build M=‘pwd‘ modules
$ sudo insmod hello.ko
$ dmesg | tail -1
[5963787.044134] Hello, world! from hello_init
$ sudo rmmod hello.ko
$ dmesg | tail -1
[5963798.420368] Good bye! from hello_exit
```

ディレクトリ構成:

```
hello-0.0.1
├── Makefile
├── README
├── dkms.conf
└── src
    ├── Makefile
    └── hello.c
```

dkms.conf:

```
PACKAGE_NAME="hello"
PACKAGE_VERSION="0.0.1"
BUILT_MODULE_NAME="$PACKAGE_NAME"
BUILT_MODULE_LOCATION="src"
DEST_MODULE_LOCATION="/updates"
AUTOINSTALL="yes"
```

5.2.1 DKMS 全体の流れ

DKMS の全体の流れは以下のようになっています。各要素は DKMS で提供されているコマンドとスクリプトになっています。

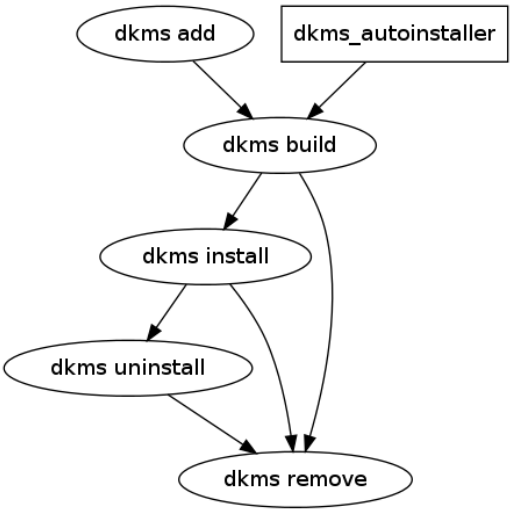


図 1　DKMS 全体の流れ

dkms_autoinstaller はシステム起動時に呼ばれる[*1]スクリプトです。既に DKMS に登録(dkms add) されているドライバモジュールの中で、現在起動しているカーネル用にドライバモジュールがインストール(dkms install) されていないものがあれば、ドライバモジュールのビルドとインストール(dkms build / dkms install) が実行されます。

5.2.2 モジュールソースコードの配置位置

DKMS 対応モジュールのソースコードは /usr/src/ドライバモジュール名-バージョン/ 以下に配置する必要があります。先のドライバの場合、 /usr/src/hello-0.0.1/ 以下に配置します。

5.2.3 DKMS 設定ファイル

DKMS は、各ドライバパッケージ毎に設定ファイルを持つ必要があります。この設定ファイルでは、どのようなドライバが提供、ビルド、インストールされるのか記述されている必要があります。以下によく利用する設定項目を示します。

BUILT_MODULE_LOCATION や BUILT_MODULE_NAME などは配列を使ってドライバモジュール毎に設定できます。 BUILT_MODULE_NAME 配列の番号は各要素の番号に紐づいているので、処理されるときは各々の番号が参

[*1] ディストリビューションによって異なる

表 1　DKMS 設定ファイル項目

設定項目	内容
PACKAGE_NAME	パッケージ名
PACKAGE_VERSION	パッケージバージョン
CLEAN	キャッシュなどの一時ファイルを削除するためのコマンドを指定する。指定しない場合には ”make clean” が実行される。
MAKE	モジュールをビルドするコマンドを指定する。
BUILT_MODULE_NAME	ドライバモジュールの名前を指定する(拡張子は必要なし）。
BUILT_MODULE_LOCATION	ビルドされたモジュールが作成されるディレクトリまでの相対パス
DEST_MODULE_LOCATION	モジュールをインストールするディレクトリを指定する。
AUTOINSTALL	モジュールを自動的にインストールするか、”yes” か ”no” を指定する。
REMAKE_INITRD	ドライバインストール時に INITRD イメージを再構成するか、”yes” か ”no” を指定する。
PATCH	適用したいパッチを指定する。
PATCH_MATCH	パッチを適用するバージョンの指定する。 Linux カーネルが 2.6.38 と 2.6.39 の場合にパッチを適用したい場合には ”2.6.(38—39)” と指定する。

照されます。例えば、一つのドライバパッケージから foo と bar の 2 つのデバイスドライバが提供される場合、以下のように設定することができます。

```
( 省略)
BUILT\_MODULE\_NAME[0] = "foo"
BUILT\_MODULE\_LOCATION[0]  = "foo_build"
DEST\_MODULE\_LOCATION[0] = "/update"
BUILT\_MODULE\_NAME[1] = "bar"
BUILT\_MODULE\_LOCATION[1]  = "bar_build"
DEST\_MODULE\_LOCATION[1] = "/update"
( 省略)
```

これにより、ドライバ foo のビルドは foo_build ディレクトリで行われ、 update ディレクトリ以下にインストール、ドライバ bar のビルドは bar_build ディレクトリで行われ、 update ディレクトリ以下にインストールされます。

5.2.4　DKMS で提供されるコマンド

DKMS は dkms コマンドで操作します。以下に利用頻度が高いコマンドを紹介します。

1. dkms status

   DKMS で提供されている モジュールのステータスを表示します。

   ```
   $ dkms status
   v4l2loopback, 0.5.0, 3.1.0-1-amd64, x86_64: installed
   v4l2loopback, 0.5.0, 3.2.0-1-amd64, x86_64: installed
   virtualbox, 4.1.8, 3.1.0-1-amd64, x86_64: installed
   virtualbox, 4.1.8, 3.2.0-1-amd64, x86_64: installed
   ```

2. dkms add

   モジュールのソースコードを管理対象に追加します。実行する時に -m オプションで追加するドライバパッケージ名を、-v オプションでバージョンを指定します。

   ```
   $ sudo dkms add -m hello -v 0.0.1

   Creating symlink /var/lib/dkms/hello/0.0.1/source ->
                    /usr/src/hello-0.0.1

   DKMS: add completed.
   $ dkms status
   hello, 0.0.1: added
   ```

実行すると/var/lib/dkms 以下に シンボリックリンクを張り、 DKMS が管理するドライバデータベースに登録されます。

3. dksm build

   管理対象になっているモジュールをビルドします。実行する時に -m オプションで追加するドライバパッケージ名を、-v オプションでバージョンを指定します。

   ドライバは /var/lib/dkms/ドライバ名/ドライババージョン/BUILT_MODULE_LOCATION 以下でビルドされます。作成されたモジュールとビルドログは /var/lib/dkms/ドライバ名/ドライババージョン/カーネルバージョン/アーキテクチャ 以下に置かれます。

```
$ sudo dkms build -m hello -v 0.0.1

Kernel preparation unnecessary for this kernel.  Skipping...

Building module:
cleaning build area....
make KERNELRELEASE=3.0.0-1-amd64 -C /lib/modules/3.0.0-1-amd64/build M=/var/lib/dkms/hello/0.0.1/build....
cleaning build area....
```

4. dksm install

   dkms build コマンドによってビルドされたモジュールを dkms.conf の DEST_MODULE_LOCATION で指定されている場所にインストールします。例えば Linux カーネル 3.2.0-1-amd64 を使っていて、

```
DEST_MODULE_LOCATION[0]="/updates"
```

   と指定されている場合には /lib/modules/3.2.0-1-amd64/updates/dkms/ 以下にインストールされます。

```
$ sudo dkms install -m hello -v 0.0.1

hello:
Running module version sanity check.
 - Original module
   - No original module exists within this kernel
 - Installation
   - Installing to /lib/modules/3.0.0-1-amd64/updates/dkms/

depmod.........

DKMS: install completed.
$ dkms status
hello, 0.0.1: installed
$ sudo modprobe -l | grep hello.ko
updates/dkms/hello.ko
```

5. dkms uninstall

   モジュールをアンインストールします。実行する時に -m オプションで追加するドライバパッケージ名を、-v オプションでバージョンを指定します。また全てのカーネルから削除する場合には –all オプション、特定のカーネルモジュールのみを削除する場合には -k オプションでカーネルバージョンを指定します。

```
$ sudo dkms uninstall -m hello -v 0.0.1 -k 3.0.0-1-amd64

-------- Uninstall Beginning --------
Module:  hello
Version: 0.0.1
Kernel:  3.0.0-1-amd64 (x86_64)
-------------------------------------
( 中略)

depmod....

DKMS: uninstall completed.
```

   *2

6. dkms remove

   DKMS の管理対象から外します。 uninstall と同様のオプションが必要です。ドライバモジュールがアンインストールされていない場合、アンインストールを実行してから、管理対象から外れるようになっています。

*2 この資料を作成している時点では、「 dkms uninstall」は動作しません。一部のチェックが未実装なため、 uninstall の処理まで行われないためです。 http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=659672

```
$ dkms status
hello, 0.0.1, 3.0.0-1-amd64, x86_64: installed
virtualbox, 4.1.6, 3.0.0-1-amd64, x86_64: installed
$ sudo dkms remove  -m hello -v 0.0.1 --all

-------- Uninstall Beginning --------
Module:  hello
Version: 0.0.1
Kernel:  3.0.0-1-amd64 (x86_64)
-------------------------------------
( 中略)

depmod....

DKMS: uninstall completed.

------------------------------
Deleting module version: 0.0.1
completely from the DKMS tree.
------------------------------
Done.
$ dkms status
```

その他、 rpm パッケージを作成する mkrpm オプションや Debian パッケージを作成する mkdeb オプションなどが用意されています。

5.3 DKMS 対応 Debian パッケージの作り方

上記での説明のように、 DKMS 対応モジュールのソースを規定のディレクトリに展開しておくと DKMS は処理を行います。 Debian の場合もパッケージ化するときはインストールした時に規定のディレクトリに展開するようにしておきます。パッケージ名などに関してはルールが決まっており、 DKMS 対応パッケージは-dkms というサフィックスがついています。これはパッケージ名がわかりやすいという理由と、 DKMS 用の debhelper コマンド、 dh_dkms コマンドが -dkms サフィックスのついたパッケージに対して処理を行うためです。 debian/パッケージ名.dkms または debian/dkms を dkms.conf に変換して、適切な dkms 用ディレクトリにコピーし、 postinst と prerm に DKMS 用の処理を追加します。 dh_dkms コマンドは dkms パッケージで提供されています。 debhelper 7 以降で利用する場合には、アドオンとして読み込ませる必要があります。

debian/hello-dkms.dkms:

```
PACKAGE_NAME="hello"
PACKAGE_VERSION="#MODULE_VERSION#"
BUILT_MODULE_NAME[0]="$PACKAGE_NAME"
BUILT_MODULE_LOCATION="src"
DEST_MODULE_LOCATION[0]="/updates"
AUTOINSTALL="yes"
```

DKMS 用 debian/rules:

```
#!/usr/bin/make -f

VERSION := $(shell dpkg-parsechangelog | sed -nr '/^Version:/s/Version: (.*:)?(.*)-(.*)/\2/p')

%:
    dh $@ --with dkms
override_dh_install:
    dh_install src/* usr/src/hello-$(VERSION)/src/
    dh_install Makefile usr/src/hello-$(VERSION)/
override_dh_dkms:
    dh_dkms -V $(VERSION)

override_dh_auto_configure override_dh_auto_build override_dh_auto_test override_dh_auto_install override_dh_auto_clean:
```

debian/rules ファイル内で override_dh_auto_configure や override_dh_auto_build を呼び出している理由は Makefile がある場合、 make が実行されてしまうのでこれを抑制するためです。

5.3.1 Debian でのドライバモジュールビルドのタイミング

Debian では DKMS ドライバパッケージがインストールされた後と、カーネルヘッダ または カーネルイメージパッケージがインストール(更新) された後、 dkms が起動してドライバモジュールのコンパイルが行われるようになっています。これらは以下のファイルで処理されます。

```
/etc/kernel/postinst.d/dkms
/etc/kernel/header_postinst.d/dkms
```

5.3.2 Debian の DKMS 対応状況

主要なドライバの殆どは DKMS と module-assistant（ m-a）の両方に対応しています。開発があまり活発ではないドライバは m-a のみをサポートした状態が多いようです。新しくパッケージ化されたドライバはほとんどが両方に対応しています。 DKMS のほうがメリットが多いため、 m-a から DKMS に切り替えているユーザも多いようです。

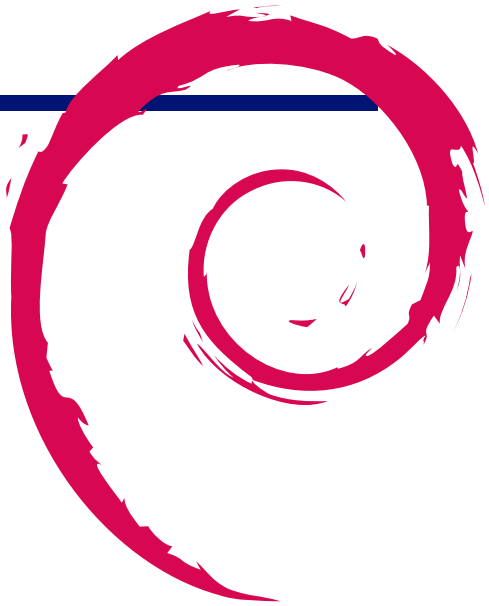
5.4 module-assistant との違い

Debian には他のドライバパッケージ管理機構に module-assistant（ m-a）があります。 DKMS と m-a の違いには以下のようになっています。

- m-a は自動ビルド機構がない。
  カーネルが更新されると、手動で m-a を実行する必要があります。
- DKMS はドライバ用のバイナリパッケージを作らない。
  m-a は カーネル毎バイナリパッケージを作り、それをインストールします。
- m-a は起動しているカーネルのみのドライバをビルドするが、 DKMS はインストールされているカーネルをビルドできる。

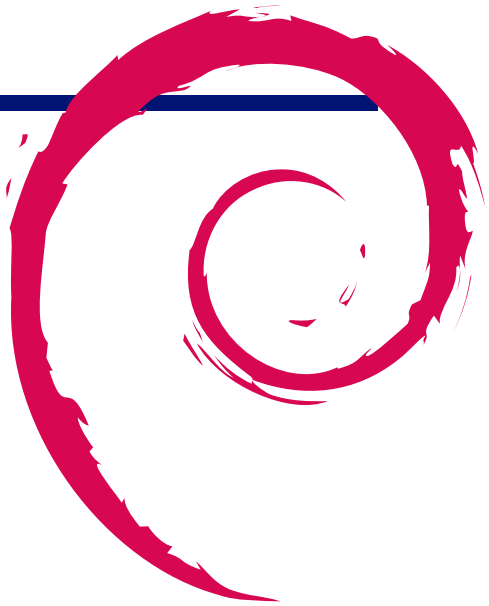
5.5 まとめ

今回は DKMS の基本的な仕組みと、 Debian での DKMS 対応パッケージ作成方法について説明しました。今までは m-a が主流だったのですが DKMS もサポートしているドライバパッケージも増え始め、徐々に DKMS に移行しつつあるように感じました。 Debian パッケージ用ツールが用意されているため、対応パッケージ作成も難しくないと思います。今後ドライバパッケージを作成する場合には、 DKMS をサポートしてみてはいかがでしょうか。



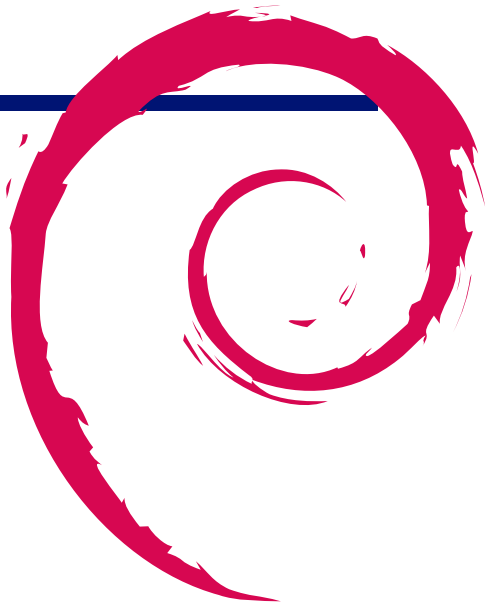
6 Debian サーバ量産工場 - 無限ポチポチ on Debian

山田 泰資



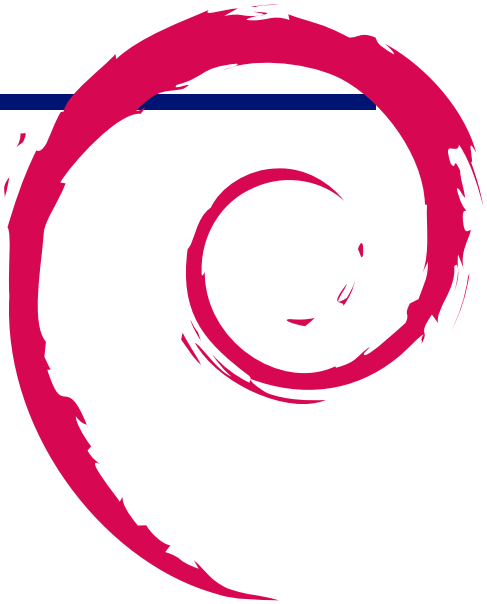
7 ついにねんがんの HCA を手に入れたぞ! - InfiniBand on Debian

山田 泰資



8 ホビーマイコン開発 on Debian

山田 泰資



9 Debian JP Project & 祝 Debian サーバ in 福岡

山田 泰資

Debian 勉強会資料

2012 年 2 月 18 日　　初版第 1 刷発行

福岡 Debian 勉強会（ 編集・ 印刷・ 発行）