.Deb

銀河系唯一のDebian専門誌

2013 年 11 月 16 日

特集1： wayland を動かす

特集2： tramp 環境構築

デビアン勉強会

目次

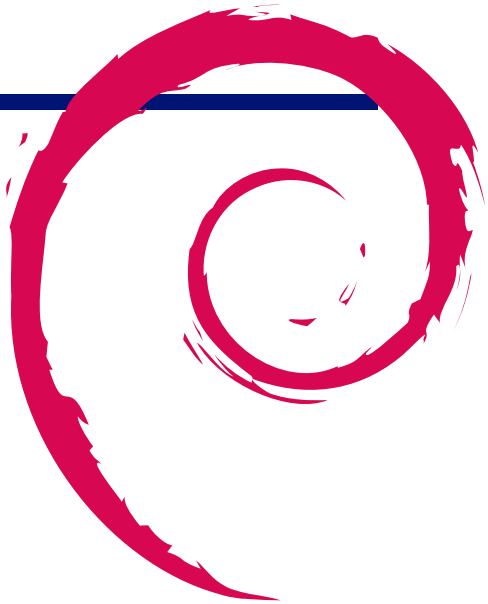
1 事前課題 2
- 1.1 野首 . . . 2
- 1.2 dictoss(杉本　典充) . . . 2
- 1.3 清野陽一 . . . 2
- 1.4 mtoshi . . . 2
- 1.5 まえだこうへい . . . 2
- 1.6 野島　貴英 . . . 2
- 1.7 上川純一 . . . 2
- 1.8 吉野 (yy_y_ja_jp) . . . 2

2 Debian Trivia Quiz 3

3 wayland を動かす 4
- 3.1 wayland . . . 4
- 3.2 weston の動いている様子 . . . 4
- 3.3 weston の対応出力デバイス . . 4
- 3.4 debian での weston 稼働のための下準備 . . . 4
- 3.5 その 1:X 上で動かしてみる . . 5
- 3.6 その 2:KMS/DRM で動かしてみる . . . 5
- 3.7 その 3.FrameBuffer デバイス上で動かしてみる . . . 6
- 3.8 weston のカスタマイズ . . . 8
- 3.9 weston の構造 . . . 8
- 3.10 その他 . . . 8
- 3.11 おわりに . . . 9

4 tramp 入門 10
- 4.1 emacs でリモートファイル編集する Tramp のすすめ . . . 10
- 4.2 リモートホスト側の設定 . . . 11
- 4.3 ssh の設定とチューニング . . . 11
- 4.4 tramp の.emacs での設定とチューニング . . . 12

5 索引 13



1 事前課題

上川 純一

今回の事前課題は以下です:

1. wayland vs mir についておもうことを語ってください。

この課題に対して提出いただいた内容は以下です。

1.1 野首

後発でありながら優位性のみられない Mir に意味があるとすれば、それは Canonical がコントロール可能な点だけなのかなという気がします。

1.2 dictoss(杉本 典充)

調べてきて Wayland が X.Org の後継として設計されて、Wayland から fork したものが Mir らしいということがわかった。過去との互換性をもっている Wayland のほうが debian としては採用しやすい分勝っているのかなと思う。

1.3 清野陽一

普段あまり意識したことがなかったので、これを気に勉強できればと思います。

1.4 mtoshi

さっぱり分かりません (涙)

1.5 まえだこうへい

名前しか耳にしたことがないので、さっぱり分からない。

1.6 野島 貴英

wayland も頑張った! mir は自分はよくわからんが、頑張ってる! X も負けてない! いやー、こちらの戦いは目がはなせませんネー。何事も競争相手がいるって良いことですネー。 mir との比較は検討したことないので、違いについて誰かよろしくおねがいします。いずれにしても、自分としては、組み込み含めて簡単に遊べそうだし、中身すっごいわかりやすい実装である、wayland としばらく戯れようと思ってます。

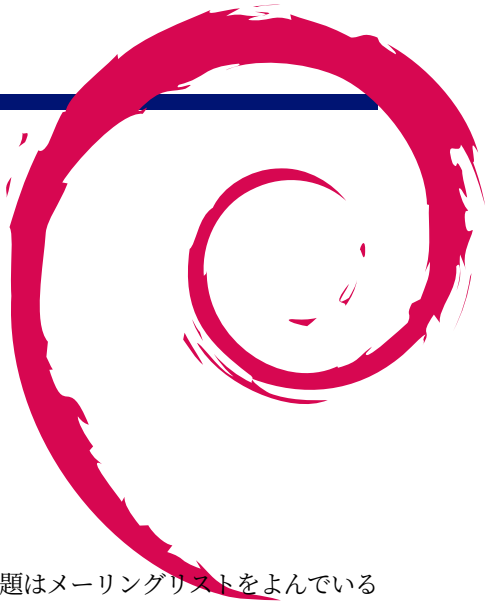
※ mir は調べてないよ?

1.7 上川純一

x11 のプロトコルは改良の余地があると思うので頑張って開発が進み競争があるのは好ましいと思います。

1.8 吉野 (yy_y_ja_jp)

Wayland になってほしいです.



2 Debian Trivia Quiz

上川純一

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` に投稿された内容などからです。

問題 1. alioth になにがおきたか

☐ A 懸賞があたった

☐ B RAID のハードディスクが 2 個壊れた

☐ C 新アーキテクチャに移行した

問題 2. DSA が DPL の承認なく使える予算はいくらか

☐ A $ 0

☐ B $ 100

☐ C $ 400

問題 3. Jessie のフリーズはいつか

☐ A 2013 年 11 月 5 日

☐ B 2014 年 11 月 5 日

☐ C 2015 年 11 月 5 日

問題 4. policy 3.9.5.0 によるとバイナリパッケージ内のファイル名のエンコーディングはなにか

☐ A UTF-8

☐ B Latin-1

☐ C sjis

3 wayland を動かす

野島 貴英

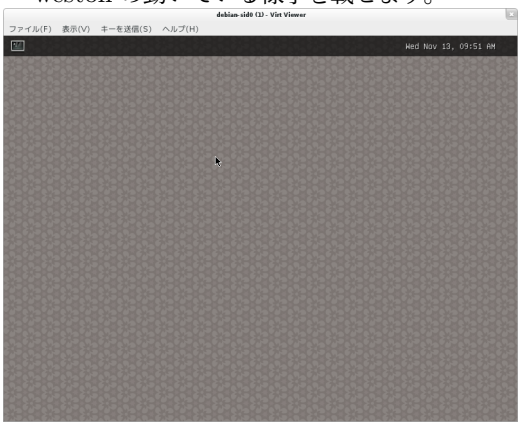
3.1 wayland

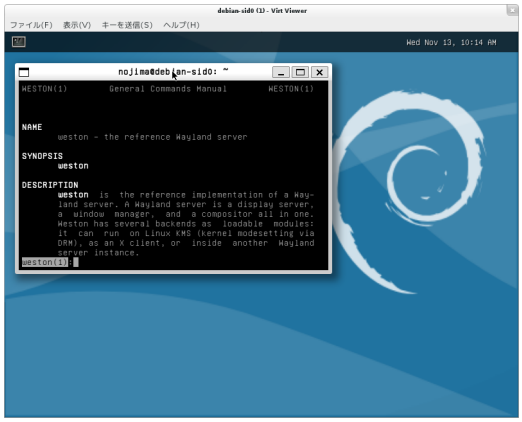
wayland とは、 Kristian Høgsberg さんが中心となって作っているディスプレイサーバーのプロトコルのことです。wayland プロトコルを扱えるディスプレイサーバーとして weston があります。

従来からある Unix で有名なディスプレイサーバーのプロトコルとしては、 X プロトコルがあり、 X プロトコルを扱えるディスプレイサーバーに X があります。しかしながら、 X は 1984 年頃の設計から始まって、ずっとハードの進化にあわせてつぎはぎしてきたため、いろいろ実装と機能に無理が生じています。 wayland は、 X に比べて、圧倒的に洗練された設計で、シンプルに、プロトコルとディスプレイサーバーを実現したものとなります [1]。

3.2 weston の動いている様子

weston の動いている様子を載せます。





3.3 weston の対応出力デバイス

weston の対応できる出力デバイスは表 3.3 となります。

3.4 debian での weston 稼働のための下準備

debian で weston を動作させるには、以下の下準備が必要です。

Step 1. weston の導入をします。

```
# aptitude install weston
```

Step 2. systemd パッケージを導入するなどして、環境変数 XDG_RUNTIME_DIR が設定されるようにします。

項番	出力デバイス	バックエンド名	パッケージに搭載済	備考
1	DRM/KMS	drm-backend.so	○	
2	FrameBuffer	fbdev-backend.so	○	
3	X	x11-backend.so	○	
4	Wayland	wayland-backend.so	○	
5	Headless	headless-backend.so	○	
5	Rassberry Pi	rpi-backend.so	×	
6	RDP	rdp-backend.so	×	

表 1　出力デバイスと weston の対応状況

```
# aptitude install systemd
```

Step 3.　weston-launch にユーザを加え、ログインしなおします。

```
# usermod -a -G weston-launch <your-login-id>
... 重要: この後ログインしなおす...
```

Step 4.　環境変数 XDG_RUNTIME_DIR が設定されているか確かめます。なお、何らかの理由で、 systemd パッケージ付属の systemd-logind が動作しない等の理由により、 XDG_RUNTIME_DIR が設定されていないのであれば、 export XDG_RUNTIME_DIR=/tmp などして書き込みが可能なディレクトリを指定しておきます。

```
[1] systemd-logind が無事動いている場合↓
$ env| fgrep XDG_RUNTIME_DIR
XDG_RUNTIME_DIR=/run/user/1000

[2] systemd-logind が何らかの理由で動作していない場合↓
$ env| fgrep XDG_RUNTIME_DIR
... なにも表示されない...
$ export XDG_RUNTIME_DIR=/tmp
```

3.5　その 1:X 上で動かしてみる

一番簡単に動かせます。 X のターミナルソフト上で、

```
weston
```

とするだけで、 x11-backend.so が読み込まれ、 weston の窓が開き、 weston が動作を開始します。
停止する時は、 weston を起動しているターミナルで、 Ctrl-C を押下します。

3.5.1　X 上で動かしてみる時の注意点

何故か、手元の nvidia 製グラフィクスカードにて、 non-free の nvidia ドライバを使っていると、画面真っ黒のウィンドウが開いてしまいました。–use-pixmap を指定して weston を起動して、 weston が EGL 等のハードウェアアクセラレーションを利用しないようにしても同様だったりします。一方で、 intel 製のグラフィクスチップを載せているノート PC 上では問題なく動作しました。

こちらの原因はまだつかめていません。

3.6　その 2:KMS/DRM で動かしてみる

intel のグラフィックスチップが搭載されている PC(例: ノート PC とか）であれば、 linux の KMS/DRM ドライバで動作します。また、 AMD/Nvidia のグラフィクスチップであれば、 linux の KMS/DRM ドライバの radeon/nouveau ドライバで動作すると思われますが、自分は未評価です。どなたか radeon/nouveau で動作した方がいらっしゃれば教えてください。

Step 1. グラフィカルなログイン画面が出ているようであれば、こちらを停止させます。

例: gdm3 が立ち上がっている場合の止め方

```
Ctrl-Alt-F1 等を押下してコンソールに切り替える
$ su
# service gdm3 stop
```

Step 2. KMS/DRM ドライバが有効であることを確かめます。

```
$ lsmod | egrep '(i915|radeon|nouveau)'
...(i915/radeon/nouveau のどれかの文字列が出れば OK)...
```

Step 3 weston を動かします。

```
$ weston-launch
```

3.6.1 KMS/DRM 上で動かしてみる時の注意点

linux の KMS/DRM ドライバは、 i915/radeon/nouveau 以外は動作するかどうかは正直やってみないと分かりません。

例えば、 debian にて、仮想化技術の KVM では cirrus チップセット用の KMS/DRM ドライバが virt-viewer の元で使えるのですが、 weston は egl 側 cirrus 未対応によるエラーもしくは、 Segfault で落ちてしまうためどうにも動作しませんでした。こちらも原因が自分では正確につかめていません。

3.7 その 3.FrameBuffer デバイス上で動かしてみる

linux の FrameBuffer デバイス上で動かしてみます。今のところ仮想化技術の KVM 上で動かすのに便利です。ここでは、実際に KVM を使って動かします。

なお、大変残念なことに、現在の debian sid で導入できる weston パッケージのバージョン 1.3.0 では、仮想端末制御に関する upstream 側のバグのために、 FrameBuffer デバイスでは動作しません。幸い、こちらのバグが修正された、weston-1.3.1 がリリースされましたので、ここでは、このバージョンを簡易的に debian パッケージ化して導入します。

Step 1. weston を動かす対象の debian sid を KVM を使ってインストールします。なお、 KVM ホスト OS 側の debian 機の br0 はインターネットに接続できるように設定されているものとします。 br0 のセットアップについて、詳しくは [2] を参照してください。

```
# aptitude install libvirt-bin virtinst
# qemu-img create -f raw /var/lib/libvirt/images/debian-sid0 10G
# wget http://cdimage.debian.or.jp/7.2.0/multi-arch/iso-cd/debian-7.2.0-amd64-i386-netinst.iso
# virt-install --connect=qemu:///system -n debian-sid0 --ram 512  --cdrom /home/yours/debian-7.2.0-amd64-i386-netinst.iso \
  --disk /var/lib/libvirt/images/debian-sid0,bus=virtio,size=10,format=raw,cache=writeback \
  --bridge=br0,model=virtio --vnc --hvm --accelerate
```

なお、インストールの際の設定は表 2 のとおりを仮定します。

Step 2. インストール完了したら、すぐに debian sid へアップグレードします。

```
debian-sid0 にログインの後、
debian-sid0 $ su
debian-sid0 # cat > /etc/apt/sources.list
deb http://ftp.jp.debian.org/debian/ sid main contrib non-free
deb-src http://ftp.jp.debian.org/debian/ sid main contrib non-free
<ctrl+d>を押下
debian-sid0 # aptitude update;aptitude full-upgrade;aptitude clean
```

Step 3. weston-1.3.1-1 パッケージを作って導入します。

項番	設定項目	指定内容	備考
1	select a language	Japanese	
2	場所の選択	日本	
3	ネットワークの設定	手動設定。IP:192.168.0.2, Netmask:255.255.255.0, Gateway:192.168.0.1, Host 名:debian-sid0	
4	パッケージマネージャの設定	ミラー:日本, ミラーサイト:ftp.jp.debian.org, HTTP プロキシ: 空欄	
5	ソフトウェアの選択	SSH サーバー、標準システムユーティリティのみ選択。あとはすべて解除。	

表 2　インストーラで選択する項目

```
debian-sid0 # aptitude build-dep weston/sid
debian-sid0 # aptitude install libxcb-composite0-dev fakeroot
debian-sid0 # exit
debian-sid0 $ mkdir weston weston-work
debian-sid0 $ cd weston
debian-sid0 $ apt-get source weston/sid
debian-sid0 $ cd ../weston-work
debian-sid0 $ wget -O weston_1.3.1.orig.tar.gz \
  http://cgit.freedesktop.org/wayland/weston/snapshot/weston-1.3.1.tar.gz
debian-sid0 $ tar xzf weston_1.3.1.orig.tar.gz
debian-sid0 $ cd weston-1.3.1
debian-sid0 $ tar xzf ../../weston/weston_1.3.0-1.debian.tar.gz
debian-sid0 $ cd debian;
debian-sid0 $ patch -p1 <<__HERE
--- debian.org/changelog        2013-10-11 20:04:50.000000000 +0900
+++ debian/changelog    2013-11-12 14:48:36.219299000 +0900
@@ -1,3 +1,9 @@
+weston (1.3.1-1) unstable; urgency=low
+
+  * update to upstream
+
+ -- your name <foo@bar.com>  Fri, 11 Nov 2013 12:34:56 +0900
+
 weston (1.3.0-1) unstable; urgency=low

    [ Sven Joachim ]
diff -ru debian.org/control debian/control
--- debian.org/control  2013-10-11 19:58:17.000000000 +0900
+++ debian/control      2013-11-12 14:49:32.347299000 +0900
@@ -35,6 +35,7 @@
   libpam0g-dev,
   libvpx-dev,
   libsystemd-login-dev,
+ libxcb-composite0-dev,
 Standards-Version: 3.9.4
 Homepage: http://wayland.freedesktop.org/
 Vcs-Git: git://anonscm.debian.org/pkg-xorg/wayland/weston
__HERE
debian-sid0 $ cd ..
debian-sid0 $ env DEB_BUILD_OPTIONS='noopt nostrip' \
   dpkg-buildpackage -rfakeroot -us -uc 2>&1 | tee ../build.log
debian-sid0 $ cd ..
debian-sid0 $ su
debian-sid0 # dpkg -i ./weston_1.3.1-1_amd64.deb
```

Step 3.　3.4 章の Step 2.〜Step 4. に記載の下準備をしておきます。(Step 1. は先ほど構築した weston パッケージが導入済みですので不要です)

Step 4.　FrameBuffer のセットアップをします。なお weston を動かすためには、color depth は 24bit でなければなりません。

```
debian-sid0 # aptitude install fbset
debian-sid0 # modprobe cirrusfb
debian-sid0 # fbset -g 800 600 800 600 24
```

Step 5.　一般ユーザになり、weston を起動します。

```
debian-sid0 # exit
debian-sid0 $ weston-launch -- --backend=fbdev-backend.so --log=weston.log
```

3.8 weston のカスタマイズ

weston はデフォルトのままだと少し寂しい画面なので、ちょっとカスタマイズしてみます。カスタマイズは$ HOME/.config/weston.ini にいろいろ記載すると、いろいろとカスタマイズできます。ここでは壁紙を入れ、ウィンドウポップアップがアニメーションするようなカスタマイズをしてみます。

```
$ cat >.config/weston.ini   <<__HERE
[shell]
background-image=/usr/share/images/desktop-base/spacefun-grub.png
background-type=tile
locking=true
animation=zoom
binding-modifier=ctrl

[launcher]
icon=/usr/share/weston/terminal.png
path=/usr/bin/weston-terminal
__HERE
```

3.9 weston の構造

図 1 に wayland アプリケーションと weston の構造を載せます。

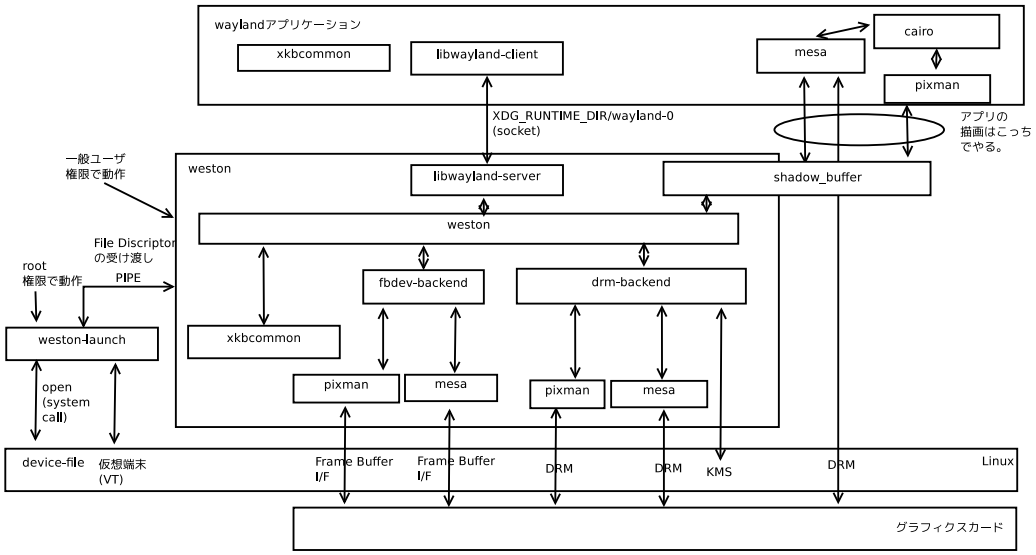


図 1　wayland アプリケーションと weston

特徴として、

- 基本的にアプリケーション側も含めて、ローカルシステムで描画を行うことを前提にした作りになっています。
- cairo/mesa/xkbcommon 等を最大限利用して、 weston 自体は非常にシンプルな設計になっています。
- プロセスのユーザ権限を制限するため、 weston-lanch には root 権限が最低限必要な部分のみの操作を、 weston は一般ユーザでの権限による動作をするように、権限が分割されています。なお、現在の実装では weston-launch は操作・オープン済みの端末の FileDiscriptor と、デバイスファイルの File Descriptor、 weston-launch へのシグナルを、 weston の求めに応じて引き渡す機能が実装されています。

3.10 その他

debian で利用できる weston/wayland にはちょっとしたクセがあります。

- Xwayland が動作しません。これは weston が起動する X に wayland 用の I/F を搭載するパッチが摘要されてい

なければならない（ X 起動時に-wayland オプションが使えなければならない）のですが、こちらは未だ X のパッケージに含められていない為となります。なお、 2013/10 頃にこちらの debian パッケージ用のパッチが debian-x の ML に流れていました。

- gtk 等の有名なツールキットは wayland 未対応の状態です。

3.11 おわりに

次期ディスプレーサーバーになるかもしれない weston と wayland について、いろいろ試してみました。プログラム本体は非常に小さいプログラムですし、発展途上ですので、いじってみようという方は、今ならたやすく改造/解析し放題の旬な時です。 Hack の対象にぜひ。

参考文献

[1] Daniel Stone,“The real story behind Wayland and X”,linux.conf.au 2013, http://people.freedesktop.org/~daniels/lca2013-wayland-x11.pdf, http://www.youtube.com/watch?v=RIctzAQOe44

[2] 野島 貴英,「 Debian 開発者の KDE 環境あれこれ」, 第 85 回東京エリア Debian 勉強会資料, http://tokyodebian.alioth.debian.org/pdf/debianmeetingresume201202.pdf

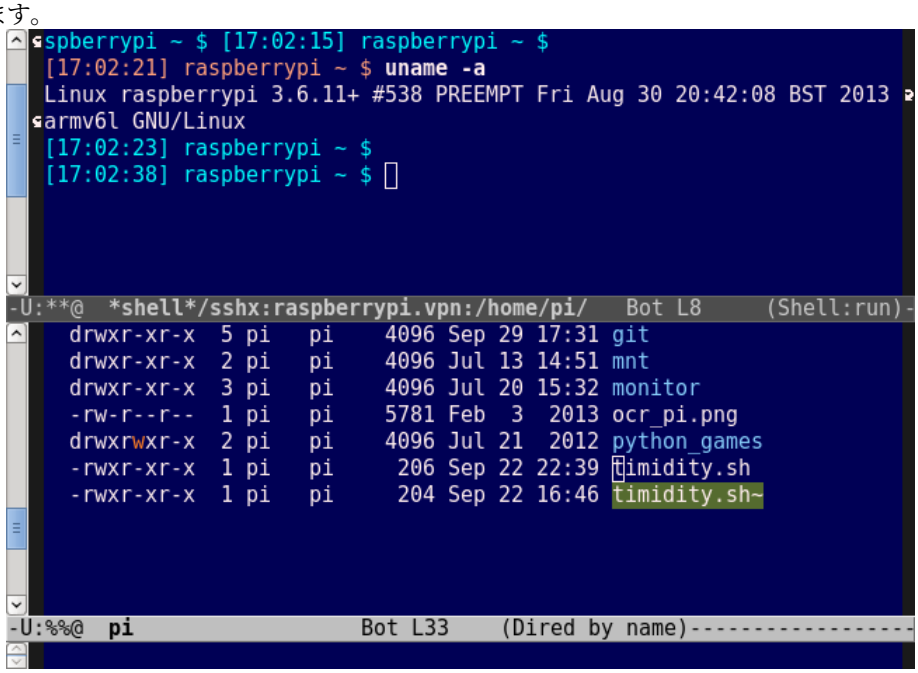
4 tramp 入門

上川純一

4.1 emacs でリモートファイル編集する Tramp のすすめ

リモートサーバでファイルの編集はどうしていますか? mosh を使っている? sshfs を使っている? いろいろあると思いますが mosh を使うとローカルのファイルと扱いが変わってしまい、リモートの vi だったり emacs を使って編集することになり、設定の同期がほぼ不可能になります。一方 sshfs だとローカルファイルのように扱うことができるのですが、root などの別のユーザ(root?) としてファイルの編集ができにくかったりしますし、リモートホストでコマンドを実行しようとすると別途 ssh でログインして作業することになり、 ssh セッションをひらきながら sshfs で実行し、ローカルのパスとリモートのパスの違いを意識しながら作業することになります。

emacs ユーザの方々に朗報です。 tramp とは `/sshx:hostname:path/to/file` という特殊なファイル名を指定すると透過的に ssh を使ってリモートのファイルを取得して編集できるようにしてくれる仕組みです。また、 dired でリモートのディレクトリのファイル管理もローカルと変わらないように行えます。さらに便利なのは `M-x shell`, `M-x compile` などリモートでコマンドを実行するコマンドを利用すると hostname に ssh でログインして path/to/file をカレントディレクトリとした状態でコマンドを実行してくれるところです。個人的には mosh などを利用してシェルのコマンドを実行するより、 emacs のローカルバッファでコマンドラインを編集して確定時にリモートに送信するスタイルが気に入っています。



4.2 リモートホスト側の設定

ssh で接続される側のの設定ですが、 tramp はシェルのセッションの入出力を利用し、機械的にプロンプトを検出するので、プロンプト(PS1 など）をカスタマイズしていると動かない場合があります。 tramp のために `.bashrc` はシンプルにするといいかもしれません。

例えば、 raspbian のデフォルトは色がつきまくってたりしてファンシー過ぎてうまく動きませんでした。

4.3 ssh の設定とチューニング

ローカルの ssh の設定ファイル `~/.ssh/config` に設定したほうがよいものを紹介します。マニュアルは `man ssh_config`[2] で参照できます。

4.3.1 ControlMaster で接続を再利用

手元ではこんな設定にしています。

```
ControlMaster auto
ControlPersist 120
ControlPath ~/tmp/ssh-%r@%h:%p
```

ControlPersist に指定した秒数間、 ssh で接続する際にデーモンプロセスが起動して、コネクションを張りっぱなしにし、コネクションを再利用してくれます。

手元では、 ControlPath を明示的に指定しています。ホームディレクトリ以下の tmp 以下にしているのはファイル名が予想可能になるので衝突を防ぐためです。

ControlMaster を Auto にしておくとすでにコネクションが開いていない場合はデーモンをたちあげて、もしコネクションが開いている場合にはそれを利用するようになります。

リモートで true コマンドを実行する速度を計測してみたところ随分高速になり (図 3)、 ping time の二倍くらいになってくれるようです。

command	time(ms)
ping	271
ssh connection sharing on	544
ssh connection sharing off	4070

表 3　ping test against wagner.debian.org

4.3.2 keep-alive

tramp はネットワーク接続の切断を ssh プロセスの生死で検出します。 ssh はネットワーク接続がきれたりしてもすぐにはそれを検出しないので、キープアライブを送信するようにします。

```
ServerAliveInterval 3
ServerAliveCountMax 5
```

本当はパソコンがサスペンドから復活して新しいネットワーク接続につながったら ssh プロセスに再接続してほしいのですが、その方法をまだ発見できていません。

4.3.3 その他の設定

Mac の Rendezvous とか Linux の Avahi とか設定しているとホスト名.local という名前で名前解決ができるようになりますが、そうすると IP アドレスは DHCP だと一定ではないため、 IP に対しての鍵が違うと怒られます。その場合はホスト IP をチェックしないようにするとよいでしょう。

```
Host *.local
 CheckHostIP no
```

あとデフォルトではいろいろな鍵を試したりする設定になっていますが、ついでに鍵を指定して公開鍵認証にしておくとよいんじゃないでしょうか。

```
Host tekitouna.vpn
 Hostname そのホストの IP アドレス
 IdentityFile ~/.ssh/ssh-keygen で作ったファイルのパス
 IdentitiesOnly yes
```

4.4 tramp の.emacs での設定とチューニング

.emacs にほとんど設定しなくてもデフォルトの設定でそれなりにうごきます。マニュアルは info 形式で提供されています [1]。

tramp では/sudo:root@hostname:/ のような形式でリモートホスト上で sudo で別ユーザになってからファイルにアクセスするように指定することが可能です。ただ、そのためにはリモートホストへの到達手段を指定する必要があります。たとえば、ローカルネットワークにあるホスト *.local で sudo をサポートしたいという要望であれば、次のような設定を追記しておきます。

```
(add-to-list 'tramp-default-proxies-alist
     '("\\.local\\'" "\\`root\\'" "/ssh:%h:"))
```

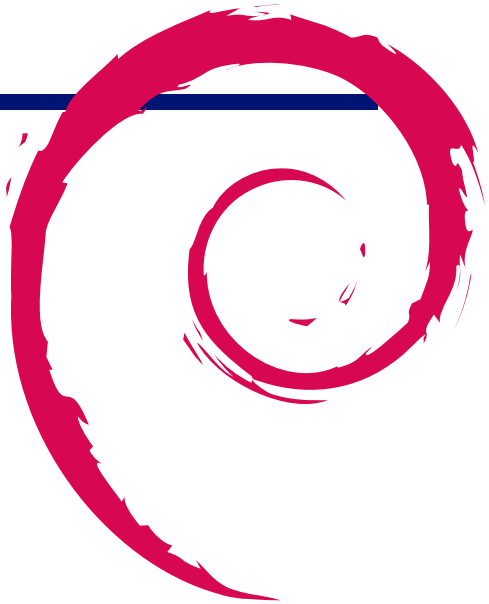
あと、デフォルトの設定だとリモートアクセスしているというメッセージがステータスに表示されるのですが、正直邪魔なのでそれを消すのもよいかと思います。

```
(custom-set-variables '(tramp-verbose 1))
```

参考文献

[1] “TRAMP User Manual”, info page, http://www.gnu.org/software/emacs/manual/html_node/tramp/index.html#Top

[2] “ssh_config(5)” manual page,



5 索引

emacs, 10

ssh, 11
ssh_config, 11

tramp, 10

wayland, 4
weston, 4

Debian 勉強会資料

2013 年 11 月 16 日　　初版第 1 刷発行

東京エリア Debian 勉強会（ 編集・ 印刷・ 発行）