NAIST

Text/Robotics/linux-command

Linuxコマンド

～ Linuxコマンド，プリンタ・エディタの使い方 ～

概要

研究で使用頻度の高いlinuxの基本操作についての説明

目次

基本用語

Linux

1991年に Linus によって開発された Unix 互換の OS.

kernel

LinuxOS の本体であるファイルを指す.

ディストリビューション

Linux という OS に様々なアプリケーションを附属して配付したもの. RedHat, FedoraCore, Vine, Debian, Ubuntu, Turbo, Suse, SlackWare, Plamo, Knoppix など, 開発元によって名前が異なり, Linux という OS を, 使えるシステムとして提供している. 各ディストリビューションはそれぞれ, 特徴のあるソフトウェア管理システムや GUI を提供しており, ディストリビューションの選択は個人の好みによるところが大きい. ちなみに, RedHat や Vine は販売をしているが, FTP を用いてフリーでダウンロードすることも可能である. 販売されている商用パッケージには, ATOK などといった商用ソフトが附属している。

(研究室では Ubuntu, Debian, RedHat が主流。)

モジュール (module)

Windows でいうところのデバイスドライバ. Linux LAN カードや SCSI ドライバをカーネルの動作中に組み込んだり取り外したりすることが可能である(ダイナミックローディングという). ちなみに, Linux カーネルは自分で構築することができるため, その再構築時にデバイスドライバをカーネル自身に最初から組み込むか, モジュールとして組み込むことが可能にするか, 使わないかを指定することができる.

モジュールの組み込みには **insmod** あるいは **modprobe** を使う. 現在組み込まれているモジュールの表示には **lsmod** を使う. モジュールを取り外すときには **rmmod** を使う.

ext3,ext2,swap

Linux におけるファイルシステム. 現在主流となっているファイルシステムは ext3 である. Windows とは違い swap をファイルではなく, パーティションとして確保する必要がある.

ext3 は, ext2 にジャーナルシステムを加え, 電源が急に落ちたときなどのデータ消去がおこりにくくなっている.

Linux のインストール後にファイルシステムを作る場合, **mkfs** を使う. mkfs は ext3,ext2,swap,vfat などに対応している. また, swap 領域の作成には **mkswap** コマンドを使う.

ブートローダ

PC に電源を入れたときに OS を立ち上げるためのソフト. **lilo** や **grub** がある. lilo や grub を使うことで Linux と Windows を同じパソコ

ンにインストールし，電源投入時にどちらを起動させるか選ぶことができる.

/bin (すらびん)

Linux で必要最低限のコマンドが置いてある. つまり，最も重要なコマンドである. たとえば vi や ls, rm, chown など.

/sbin (えすびん)

Linux を管理する上で最も重要なコマンドが置いてある. lsmod や mkfs, lilo, ifconfig など.

/home (ほーむ)

Unix 系 OS ではマルチユーザシステムであり，その各ユーザのファイルを置く場所. たとえば，ログイン名が tsuyo-s だと，通常，/home /tsuyo-s というディレクトリが作成され，そこにユーザのファイルを置くことになる.

/dev (すらでぶ)

Linux では，RS-232C やパラレルポート，USB，SCSI などといったものを，ファイルとして読み書きの操作を行うという考え方をしている. /dev 以下にはそういったデバイスにアクセスするためのファイルがある. たとえば，RS-232C からデータを読み込みたければ，通常のファイルと同様に /dev/ttyS0 を open すればいいというものである.

また，/dev/hda はプライマリマスターのハードディスクである. プライマリスレーブ，セカンダリマスター，セカンダリスレーブの順に hdb,hdc,hdd と続く. 近年主流になってきているSerialATAのハードディスクやUSBメモリなどの場合はSCSIデバイスとして認識されることもあり，その場合はsda,sdbとなる.

ハードディスクにはパーティションという概念があるが，これはhda1,hda2というようにデバイス名の次に番号をつく形式のファイルによって示される.

/dev/null や /dev/zero といった変わったデバイスもある. 前者は何を書き込んでも無視されるデバイス，後者はどれだけ読み込んでも0x00が帰ってくるデバイスである.

/etc (すらえとせ)

設定ファイルやデーモンの起動スクリプトが入っている.

-/etc/init.d
起動時に呼び出されるスクリプト群がある. ここにあるスクリプトを使うことで，任意のデーモンの再起動や停止を行うこともできる.

/proc

システム状態に関する情報が入っている. このフォルダとその中身はLinuxにより自動的に生成される.

例:/proc/version:カーネルのバージョン情報が書かれている

/var

一時的な情報やログファイルなどが入っている

-/var/log/0
各種ログファイルがある. logrotateなどがある場合，0.log.1といったファイル名で過去のファイルが保存されている.
-/var/run/0
起動中のデーモンのPIDが書かれたファイルなどがある
-/var/mail
一部のシステムではここにメールの本文や添付ファイルが格納されることがある. その場合，/varの容量を圧迫することがある.

パーミッション

root

Linux 管理人のこと. すべてのファイルにアクセス可能であるが, 逆に言えば消してはいけないファイルを消す危険性があり, 普段は, root でログインして作業するべきではない. ユーザでログインして root 権限を持つには, su と入力し, root パスワードを入れる. su - と入力してrootになった場合, 少し意味が違う. 最近では, root権限が必要となる作業はsuを使ってrootになっておこなうのではなく, sudoコマンドを用いて一般ユーザのまま行わせるようになっている. sudoは実行できるユーザやコマンドが限られており, 任意のユーザで実行できてしまうsuより安全性が高くなっている. sudoを用いる場合は, root権限を実行して行いたいコマンドの前にsudoをつける.

```
例 : sudo vi /etc/host.conf
```

ユーザ, グループ

各ファイル, ディレクトリは所有者(user), 所有グループ(group)をそれぞれ持つ. Unix 系 OS では, user, group と次に述べるパーミッションによってファイルのアクセス権限を管理している.

パーミッション

ファイルのアクセス権限のこと. ``-rwxrwxrwx`` , ``-rw-r--r--`` , ``drwx--x--x`` のように表される. Unix 系システムでは, r(read), w(write), x(execute)の3つと左から順に自分自身, 自分のグループ, 全ユーザのアクセス権限となる. また, 2進数の考え方から 777, 644, 711 のように表す場合もある. ただし, いくらパーミッションを設定したところで, root 権限の前には効力がない. パーミッションの変更には chmod, ユーザの変更には chown, グループの変更には chgrp コマンドを使う. 頭についている d はディレクトリという意味だ. 後述するシンボリックリンクの場合, lrwxrwxrwx とする. また, パーミッションには r,w,x のほかに t,s というものもあるが自習.

基本コマンド

man

```
man［コマンド名］
```

コマンドのマニュアルを見ることができる. また, C言語の関数のリファレンスにもなる. コマンドのヘルプは

```
［コマンド名］--help
```

で大抵見ることができるが, 更に詳しい情報を知りたいときには man を使う.

ls

MS-DOS でいうところの dirコマンド.

lsで今いるディレクトリのファイル一覧を表示. ただし, 『.』で始まるファイル(.bashrc)などは隠しファイルにあたるので表示されない. 全てのファイルを表示したい場合は,

```
ls -a
```

詳しく表示したい場合,

```
ls -l
```

を使う.

ls -la や ls -a を実行すればわかるが, 各ディレクトリには『.』というファイルと『..』というファイルが存在する. 『.』はそのディレクトリ自身を指し, 『..』はその上の階層にあるディレクトリを指す.

pwd

現在いるディレクトリのパスを表示する.

cd

ディレクトリを移動する. 前述したように **cd ..** でひとつ上のディレクトリに移動できる. また, 最も上位階層は/(スラッシュ)で表され, **cd /** で移動できる. 便利な機能として, **cd ~** で自分のホームディレクトリ, **cd ~ユーザ名** でそのユーザのホームに移動できる. なお, 当然ながらパーミッションによって許可されていないディレクトリには移動することはできない.

mkdir

ディレクトリを作る.

rm

ファイルを消す. 一度消したファイルはもとには戻らないので注意.

```
rm［ファイル名］
```

ディレクトリを消すときは

```
rm -r［ディレクトリ名］
```

r は再帰的という意味のオプション.

ディレクトリの中が空の時は

```
rmdir［ディレクトリ名］
```

でも消せる.

cp

ファイルをコピーする

```
cp［コピー元］［コピー先］
```

ディレクトリのコピーは

```
cp -r［コピー元］［コピー先］
```

mv

ファイル, ディレクトリの移動. ファイル名の変更にも使われる

```
mv［コピー元］［コピー先］
```

ln

リンクの作成. シンボリックリンク(ショートカットみたいなもの)の作成には,

```
ln -s［元ファイル］［シンボリックリンク先］
```

基本コマンド2

less/more

less(or more) はテキストファイルを見るページャー. 見るだけで編集することはできない.

```
less［ファイル名］
```

cat

cat はファイルの内容を標準出力に表示するもの. 短いファイルの中身を見るには less よりも便利である.

```
cat［ファイル名］
```

grep

grep はファイルから特定の文字を検索するもの. main.c というファイルから printf という文字列を検索するには

```
grep printf main.c
```

.cという拡張子から printf という文字列を検索するには

```
grep printf *.c
```

というように使います. * はワイルドカードという.

リダイレクト，パイプ

リダイレクト

リダイレクト(＞)は，標準出力されるものをファイルに保存する記号である. たとえば，そのディレクトリにあるファイル名を a.txt に保存したい場合，

```
ls > a.txt
```

とすると，コンソールには何も表示されず，a.txt というファイルに ls の結果が入る. ただし，『＞』はファイルの内容を上書きしてしまうので注意が必要である. ファイルの末尾に追加する場合は，

```
ls >> a.txt
```

のようにする. コンソールに表示されるものはこうやってファイルに保存することができる.

パイプ

パイプ(｜)は，コマンドの標準出力を次のコマンドの標準入力につなぐというもので，

```
ls -l |less
rpm -qa |grep emacs
```

のように使うことができる.

ジョブ・プロセス管理

jobs,fg,bg

プログラムは & を付けて実行することでバックグラウンドで実行可能である & を付けない場合はフォアグラウンドで実行されていることになる. ::

```
emacs &
emacs
```

フォアグラウンドで実行されているプログラムは Ctrl+c で強制終了させることが可能である. また，Ctrl+z により一時中断され，jobsコマンドによりバックグラウンドで実行中のプログラムや中断中のプログラムを確認できる.

```
jobs
```

中断中のプログラムは fg コマンド や bg コマンドにより再開させることが可能である. fg はフォアグラウンドで，bg はバックグラウンドで実行される.

```
fg
```

```
bg
```

ps

現在実行中のプロセスの状態を確認する.

```
ps
ps -aux |grep emacs
```

kill

プログラムを強制終了させる. Ctrl+c で強制終了させることができなかった時などは kill を使用する.

```
kill 1900 (軽度)
kill -9 1900 (重度)
```

ＴＡＢ補完

Kterm などのターミナル上で bash や tcsh , zsh などのシェルを利用している場合, TABキーを押すことでコマンド名やファイル名を補完することができる. TAB補完を使うと入力が早くなるだけでなく, 誤入力も防げる.

例::

```
less /pr <TAB>
  ↓
less /proc/
less /proc/cpu <TAB>
  ↓
less /proc/cpuinfo
```

文字コード

最近のUNIX系OSの多くの文字コードは UTF-8 である. 一昔前では EUC-JPが使われていた. WindowsXP は SJIS である. ちなみにメールは標準は JIS であるが, 一般的にメーラが変換してくれる.

文字コードの変換は

```
nkf -e -Lu filename > new_filename
nkf --unix filename > new_filename
nkf --windows filename > new_filename
```

などとする.

印刷

Unix 系 OS では印刷する場合, 基本的には ps ファイルに変換してからプリンタに送る必要がある. 以下が印刷方法である.

ps ファイルの印刷

```
lpr -Pプリンタ名 ファイル名.ps
```

キューに登録されている印刷ジョブの確認

```
lpq -Pプリンタ名
```

ps ファイルを読む

```
gv ファイル名.ps
```

テキストファイルの印刷

```
a2ps ファイル名.txt | lpr -Pプリンタ名
```

ユーザ管理

useradd, adduser

ユーザの追加を行う.

```
useradd hoge
```

userdel

ユーザの削除を行う.

```
userdel hoge
```

passwd

パスワードの変更を行う.

```
passwd
```

PCの停止・再起動

停止

root権限になり以下のコマンドを実行する.

```
shutdown -h now
```

または, ::

```
halt
```

最近では一般ユーザでも使用できるようになっている場合がある.

再起動

root権限になり以下のコマンドを実行する.

```
shutdown -r now
```

または,

```
reboot
```

停止コマンドと同様に, 最近では一般ユーザでも使用できるようになっている場合がある.

ソフトウェアのインストール

ソースからのインストール

Windowsと違って, Unix 系のソフトウェアはソースファイルが配付され, それをコンパイルすることによって実行形式のファイルを作るということが普通である. 最近ではバイナリ配付も行われているため, ソフトウェアの導入が容易になったが, ソースからソフトウェアを導入する方法について紹介する.

1. 解凍

tar	ディレクトリを1ファイルに
gzip	.gz の圧縮
gunzip	解凍
zip	.zip の圧縮
unzip	解凍
bzip2	.bz の圧縮
bunzip2	解凍
lha	.lzh 形式の圧縮/解凍
compress	.Z の圧縮
uncompress	解凍

2. tar ボール作成/展開

tar は，1つのディレクトリを1つのファイルとして扱うようにまとめるコマンドである.

```
tar -cvf hoge.tar ディレクトリ名
```

また，.tar のファイルを展開するには，

```
tar -xvf hoge.tar
```

である.

3. tar.gz(or tgz) の展開

tar.gz で配付されているファイルの解凍は，

```
gunzip hoge.tar.gz
tar -xvf hoge.tar
```

でも良いが，tar の z オプションを使って，

```
tar zxvf hoge.tar.gz
```

で解凍できる.

4. ドキュメントを見る .tar を解凍するとディレクトリが作成され，そのなかに INSTALL や README というファイルが存在することが多い． そのファイルはインストールの仕方や使い方が書いてあるため，事前に読んでインストールする(大抵の場合，英語だが難しくはないはず).

5. configure 解凍したディレクトリに configure という実行形式のファイルがある場合が多い． これは，自分の環境にあわせてコンパイルを行うためにマシンを調べるスクリプトコマンドである.

```
./configure
```

で実行する.

もし，コンパイルに必要なライブラリなどがなかった場合には， configure はエラーを出力して終了する． その場合，エラーを読み，必要なライブラリをインストールするなどして，もう一度 configure する必要がる.

6. make

configure に成功すれば，

```
make
```

でコンパイルを開始できる.

エラーが出ずに make が終わると, 実行形式が作成される.

7. make install 実行形式ができると, そのファイルを /usr/bin か, /usr/local/bin にインストールしなければならない(通常, /usr/local/bin).

インストールを行うには, root 権限になった上で,

```
make install
```

を実行する. 以上でインストール完了である.

8. ldconfig もし, インストールしたものが, ライブラリであった場合は,

```
ldconfig
```

を実行する.

ldconfig は最新の共有ライブラリに対して必要なリンクを作成したり, ライブラリをキャッシュしたりするコマンドである.

rpm

1. rpm のインストール

rpm は RedHat 系 OS 用のバイナリパッケージである. .i385.rpm は Intel386 系の CPU 互換のバイナリパッケージで Intel/AMD/VIA を問わずインストールできる. また, i686.rpm や, .athlon.rpm といったものもあり, これは新しい CPU の命令セットなどをフル活用して性能を向上させるためにその CPU 向けにカスタマイズされたものである.

rpm パッケージのインストールは(当然ながら root権限で),

```
rpm -ivh  dvipdfm-0.13.2c-0vl8.i386.rpm
```

のように行う. ライブラリなどの依存関係などがなければインストールできる.

パッケージのアップデートは,

```
rpm -Uvh  dvipdfm-0.13.2c-0vl8.i386.rpm
```

とするか, ::

```
rpm -Fvh  dvipdfm-0.13.2c-0vl8.i386.rpm
```

とする.

2. src.rpm rpm パッケージをインストールするときには, OS の配布元が出している自分の OS のバージョンに合ったものを利用すれば間違いないが, 自分の OS に対応していると銘打っていないものもしばしば存在する.

このような場合, rpm パッケージをインストールすることで, 動くこともあるが, 動かないことやインストールできないことがある. その場合, ソースファイルを rpm パッケージとしてまとめた .src.rpm という拡張子のファイルを利用し, 自分でバイナリ rpm ファイル(.i386.rpm)を作り, それをインストールするとうまくいくことが多い. .src.rpm から .i386.rpm を作る作業を rebuild という.

リビルドの方法は,

```
rpm --rebuild  dvipdfm-0.13.2c-0vl8.src.rpm
```

や,

```
rpmbuild --rebuild  dvipdfm-0.13.2c-0vl8.src.rpm
```

という方法があり, 環境によって異なる. リビルドすると ~/rpm/RPMS/i386/ や /usr/src/redhat/RPMS/i386/ あたりに rpm ができるので, それを上述の方法でインストールする. ちなみに, src.rpm をそのままインストールすると ~/rpm/SOURCES/ や /usr/src/redhat/SOURCES/ あたりにソースがインストールされる.

```
rpm -ivh  dvipdfm-0.13.2c-0vl8.src.rpm
```

3. rpm のアンインストール

rpm は依存関係の問題がなければ，アンインストールすることができる．アンインストールは，

```
rpm -e パッケージ名
```

である．パッケージ名は rpm ファイルの .i386.rpm を除いた部分のことである．パッケージ名の検索は，

```
rpm -q dvipdfm
```

のように行う．

apt

aptとはDebian系(ubuntuやKNOPPIXなど)で用いられるパッケージ管理システムの名称である． 依存性自動解決やパッケージの自動ダウンロードなど，高度な機能を持つ． X-Window用GUIとしてSynaptic，CUIとしてaptitudeを備える． 1．インストール

```
aptitude install［パッケージ名］
例：aptitude install apache
```

パッケージ名は部分一致検索が適用されるため，厳密なバージョンなどは必要ない． ただし，該当するパッケージが複数ある場合は選択するように指示される．

なお，apacheとhttpdのように同等のパッケージなのに名前が異なるものがある．ほとんどの場合はどちらかしか有効ではないため，片側でヒットしない場合はもう一方を試してみると良い．

2．パッケージリストのアップデート

```
aptitude update
```

パッケージリストはローカルに保存しているため，インストール済みのパッケージを更新しようとしたり，リリースされたばかりのものをインストールしようとする場合はパッケージリストのアップデートが必要となる． 上記のコマンドを入力することで，

```
/etc/apt/sources.lst
```

に記述されたダウンロードサイトよりパッケージリストを入手し，アップデートを行う． このコマンドではインストールされているパッケージ自体のアップデートは行われない．

3．インストール済みのパッケージのアップデート

```
aptitude safe-upgrade
```

このコマンドを実行すると，アップデートされるパッケージの一覧が表示され，適用するかどうかを聞かれる． そこでyを選択すると，実際のアップデート作業に入る． 依存関係にあるパッケージも，aptを通じてインストールされたものは自動的に解決される．

Ｅｍａｃｓ エディタの使い方

『いーまっくす』と読む． ちょっとしたメモを取ったり，プログラムを書いたり，修士論文を書いたりと様々な用途に使えるエディタである． ここでは触れないが，非常に奥が深い．

別に vi や jed や nano を使ってもらっても構わない．これは，好みの問題である． ここでは Emacs の説明をする．

プログラムの起動

kterm などから

```
emacs &
```

と入力すると起動する． & はバックグラウンドと言う意味．付けたり付けなかったりを試してみれば違いは分かると思う．

```
emacs ファイル名 &
```

と入力するとファイルを開く．

```
emacs -nw
```

とすると, ターミナル上に Emacs が起動. この時は & は付けないこと. nw は *no window* の略.

日本語入力

日本語入力をする場合は環境によって, 操作が違う場合がある.

```
Shift+Space       または,
C-¥
```

これらを覚えておけば日本語入力はできると思う.

```
C- : Ctrl を押しながら.
```

文字変換中の操作

```
C-o 変換範囲を右に一文字拡げる
C-i 変換範囲を左に一文字縮める
C-f 変換対象を右に移動する
C-b 変換対象を左に移動する
```

ショートカット一覧

特殊表記

```
C-x : Ctrl を押しながら, x を押す.
M-x : Alt を押しながら, x を押す. または,
  ESC を押してから, x を押す.
```

基本操作：

C-x C-f	ファイルを開く
C-x C-s	上書保存
C-x C-w	別名保存
C-x C-c	Emacs の終了

カーソル移動：

C-f	→ へ1文字移動 (forward)
C-b	← へ1文字移動 (before)
C-p	↑ へ1文字移動 (pullup)
C-n	↓ へ1文字移動 (next)
M-f	→ へ1単語移動
M-b	← へ1単語移動
C-a	行頭へ移動
C-e	行末へ移動
M-<	ファイルの先頭に移動
M->	ファイルの末尾に移動

切り取り,削除 / コピー / 貼り付け :

C-d	カーソル位置にある文字を削除
C-k	カーソル位置から行末まで Cut
M-d	カーソル位置から単語の末尾まで Cut
M-Del	カーソル位置から単語の先頭まで Cut
C-Space	領域の先頭をマーク
C-@	同上
C-w	マーク位置からカーソル位置まで Cut
M-w	マーク位置からカーソル位置まで Copy
C-y	C-k, C-w, M-w で選択した部分を Paste

訂正操作 :

C-g	キャンセル
C-x u	アンドゥ

|C-\|アンドゥ

バッファ操作 :

C-x C-b	バッファリスト一覧
C-x b	編集バッファの切り替え
C-x o	二つのバッファ間のカーソル移動
C-x 0	カレントのバッファ表示を閉じる
C-x 1	カレント以外のバッファを閉じる
C-x 2	バッファを上下二つに分割
C-x 3	バッファを左右二つに分割

検索,置換 :

C-s	文字列の前方検索
C-r	文字列の後方検索
M-%	文字の置換

これら以外にも色々ある.

ｎｅｔｗｏｒｋ関連コマンド

リモートログイン

研究室サーバにアクセスする場合はsshを使う。

telnet

```
telnet [サーバ名]
```

パスワードは平文で送られる.

rlogin

```
rlogin [サーバ名]
```

```
rlogin -l [ユーザ名] [サーバ名]
```

パスワードは平文で送られる.

ssh

```
ssh [サーバ名]
ssh [ユーザ名]@[サーバ名]
ssh -l [ユーザ名] [サーバ名]
```

パスワードは暗号化される.

ファイル転送

ftp

```
ftp [サーバ名]
```

パスワードは平文で送られる.

sftp

```
sftp [サーバ名]
```

パスワードは暗号化される.

scp

```
scp [コピー元] [コピー先]
scp [ユーザ名]@[サーバ名]:[コピー元]  [ユーザ名]@[サーバ名]:[コピー先]
```

パスワードは暗号化される.

network設定

ifconfig

IPアドレスなどを調べる

```
/sbin/ifconfig
```

eth0のネットワークを起動する

```
/sbin/ifup eth0
```

eth0のネットワークを停止する

```
/sbin/ifdown eth0
```

ping

ネットワークに接続されているか調べる

```
ping [IPアドレス]
ping [サーバ名]
```

特殊なアドレスとホスト

```
127.0.0.1**  :自分を表すIPアドレス
localhost**  :自分を表すホスト名
```

network設定ファイル

Linuxでのネットワーク設定は，ディストリビューション毎にかなりの差が存在する．ここでは現在研究室にてよく用いられるubuntuでの設定の方法を説明する．なお，X-Windowが起動している環境の場合はGUIでの設定が可能なのでそちらを利用する方が望ましい．

/etc/network/interfaces

各インタフェースの設定を記述する．lo(ループバックインタフェース)の設定

```
auto lo
iface lo inet loopback
```

有線LANの設定例(固定IPアドレスの場合)

```
iface eth0 inet static
address 163.221.139.225
netmask 255.255.255.0
gateway 163.221.139.1
auto eth0
```

有線LANの設定例(DHCPの場合)

```
iface eth0 inet dhcp
auto eth0
```

/etc/resolv.conf

DNSの設定を記述する．DHCP環境の場合は自動的に書き換えられるため，設定の必要は無い．NAISTにおける一般的な設定

```
search naist.jp
nameserver 192.168.8.11
nameserver 192.168.8.12
domain naist.jp
```

/etc/hosts

ホスト名リスト::

```
127.0.0.1     : mypc localhost.localdomain localhost
192.168.0.2   : friend friend.localdomain
```

127の行は消しては行けない．色々と動かなくなります．ただし，ホスト名を書き足すのはOK．

ここに，IPアドレスとホスト名を記述しておくと，自分で決めた呼び名でアクセスできる．

network設定スクリプト

/etc/init.d/network

上述のファイルを変更した場合，ネットワーク機能を再起動して新しい設定を読み込ませる必要がある．

```
./network start    : ネットワークを起動する．
./network stop     : ネットワークを停止する．
./network restart  : ネットワークを再起動する．
```

おまけ

Linux に触れる上で参考になる本，サイトである．また google 先生とはお友達になることを推奨．

- UNIX USER

    - 月刊誌 (研究室で年刊購読)
- Linux WORLD
    - 月刊誌
- Linuxコマンドスーパーリファレンス
    - 伊藤真人, 小巻賢二郎, 田谷文彦, 前田雄一郎 (著)
    - ISBN4-7973-0771-4
    - SOFTBANK Publishing
    - 2,000円
- Linux Personal Workstation/Linux パーソナルワークステーション
    - http://www.lain.org/winglab/Works/book.html
    - Linux(UNIX)の本1冊分のpdfファイルが(今なら)無料でダウンロード可能.
- @IT Linux Tips
    - http://www.atmarkit.co.jp/flinux/rensai/linuxtips/tipsindex.html
- IT media Tips for Linux
    - http://www.itmedia.co.jp/help/tips/linux/
- IT Pro Linux
    - http://itpro.nikkeibp.co.jp/lin-os/index.html

なぜコマンドラインを使うのか.

以下はひとつの意見にすぎないため, 自分なりに考えること. これは, GUI(グラフィカルユーザーインターフェース) vs CUI(キャラクタユーザーインターフェース)の違いを見て行くと考えやすい. GUIは基本的にマウスを利用するため, 直感的に操作可能であるために非常に 使いやすい. しかし, このマウスはキーボードに比べ, コンピュータに伝えられる情報量が非常に小さい. なぜならば, x,y,右クリック,左クリック程度の情報しか伝えられないためである. それに対し, キーボードは, 単一時間あたりの情報量が非常に大きい. (慣れによるが. .) そのため, コマンドラインで操作すること(キーボードショートカット等を含む) に慣れてしまうとマウス操作を要するGUIよりも素早く, より確実な操作が可能になる. そのため, 早くからコマンドラインを多用するLinuxを学び, 慣れておくことは, 今後コンピュータを扱って行く上でも プラスになると考える. 最近のLinuxは, GUIが充実しているため, 初心者でも触りやすくなってきている. しかし, コマンドを知り, コマンドラインでUnixやLinuxを触ってみてはいかがだろうか?

Last-modified: 2013-02-12 (火) 11:53:59 (1084d)