第3回 UNIXゼミ

ゼミ担当者　：　谷口　義樹，永松　秀人，澤田　淳二
指導院生　　：　上川　純一，片浦　哲平
開催日　　：　2002 年 5 月 14 日

ゼミ内容: 本ゼミでは，研究活動での UNIX の使用において，最低限必要となる知識および操作のスキルの取得を目的とする．本研究室では最適化の研究に並列計算機を用いることが多いが，その際に，並列計算機の利用および管理のための LINUX の知識が必要不可欠となる．そこで，本ゼミではそれらの利用が可能となるように，LINUX 上での各操作を学ぶ．今回は，mount/unmount，proc，X-window-system，UNIX の考え方についてのゼミを行う．

1　UNIX という考え方

OS を使いこなすためには，その背後にある哲学を理解することが必要です．この章では UNIX という OS の考え方を紹介します．

1.1　9 つの定理

1. スモール・イズ・ビューティフル
   巨大で複雑なプログラムの開発者は，将来が予測可能で，そして現在とそう大きくは変わらないという勝手な思い込みを前提としている．一方，小さなプログラムの開発者が考えるのは，明日作られるものは今日作っているものとは違うということでしかない．

2. 一つのプログラムには一つのことをうまくやらせる
   最初の仕様から変更のないプロジェクトなど稀である．

3. できるだけ早く試作する
   試作によって何がうまくいくかが分かり，さらに重要なことには何がうまくいかないかが分かる．

4. 効率より移植性を優先する
   プログラムを速くすることに時間をかけないこと．最も効率のよい方法は，ほとんどの場合移植性に欠ける．

5. 数値データは ASCII フラットファイル[1]に保存する
   ASCII テキストは共通の交換形式．ASCII テキストは簡単に読めて編集できる．

6. ソフトウェアを梃子として使う
   よいプログラマはよいコードを書く．偉大なプログラマはよいコードを借りてくる．

7. シェルスクリプトによって梃子の効果と移植性を高める
   シェルスクリプトはCよりも移植性が高い．

8. 過度の対話的インタフェースを避ける
   拘束的プログラムはユーザーを人間と想定している．人間の限界によって動作を制限されるようなシステムは，潜在能力をフルに発揮できない．

9. すべてのプログラムをフィルタとして設計する
   データの合成ではなく，与えられたデータを選択的に通過させることに集中する．

1.2　総括

小さなプログラムというものは，人間にとって大きな「何か」よりも分かりやすい．つまり，理解しやすければ，当然保守も容易となるので，最終的には対費用効果が大きくなる．また，読み込み，実行，解放のいずれもすばやく行うことができ、効率が高まる．小さなプログラムは，他のツールと簡単に結合できる．単独ではたいしたことができなくても，他の小さなプログラムと組み合わさることによって，新しいアプリケーションが簡単に作成することもできる．

小さなプログラムを作成するためには，目的を絞らなければならない．すなわち，一つのことをうまくこなすことに専念しなければならない．大きくて複雑な単体型プログラムでは，他のアプリケーションとの共同作業に莫大な労力をささげなければならない．

世界のソフトウェアは絶えず進化し続けており，将来のニーズにまで対応するソフトウェアを作ることなど到底不可能である．できることといえば，今日のニーズに対応するソフトウェアを作ることである．開発者は設計仕様書の作成に何ヶ月も費やす無駄をやめ，早期に試作を作るべきである．試作が他人の手に早く届くことによって，システムはより良く変貌していく．変更が必要な場合では，すべてが完成してしまった最終段階において変更するよりも，万事が流動的な初期段階において変

[1]すべての数値データを ASCII 文字として格納することを意味する

更するほうがいいのは当たり前である．試作を作ってみれば，何がうまくいくのか，そしてより重要なことに，何がうまくいかないのかが早期に発見することができる．

ソフトウェアというものは，実は作るものではなく，成長していくものである．新しいハードウェア，新しいアーキテクチャは頻繁に現れる．ソフトウェアはそれらの新しい環境に対応していかなければならない．つまり，移植性の高いソフトウェアでなければならない．移植できるものは生き残り，それ以外のものは，やがて時代に取り残されることとなる．

すべてのソフトウェアというものは，命令とデータとから構成されている．ソフトウェアの移植性を考える場合には，データの移植性もあわせて考えなければならない．データは ASCII フラットファイルとして保存する．ASCII テキストは，最良の形式でないにしろ，間違いなく最も一般的な形式である．データをテキスト形式で保存することによって，他システムへの移植をきわめて簡単なものとしている．

「良いプログラマはよいコードを書く．偉大なプログラマは良いコードを借りてくる」，すなわち，新しいアプリケーションを開発するには，既存のソフトウェアのコードを再利用するべきである．新しいアプリケーションを一から開発するのはいいことである．しかし，誰かがすでにやっていることを改めてやり直すのは，時間の無駄である．今日，世界に存在するソフトウェアは偉大な共有資産である．

ソフトウェアの梃子の効果をいっそう大きくするために，できるだけシェルスクリプトを使う．シェルスクリプトは他人の努力の結果を自分の目標の達成に向けて組み込むことができる．シェルスクリプトで使うコードの大部分は自分で書くのではない．すでに他人が書いてくれているものを利用するだけでよいのである．シェルスクリプトを使えば，世界中の人々の仕事を合成することができ，小さな努力で大きな成果をあげられる．

小さなプログラムのコレクションが手元にあれば，シェルスクリプトを簡単に作れる．しかし，ユーザーにいちいち応答を要求するようなプログラムは，あまり役に立たない．拘束的ユーザーインタフェースは避けるべきである．拘束的プログラムは，どこかで人間と対話することを想定している．そのため，これをシェルスクリプトに組み込むことは非常に難しい．

すべてのプログラムは，簡単なものでも複雑なものでも，何らかの形式のデータを入力として受け入れ，何らかの形式のデータを出力として生成する．すなわち，すべてのプログラムはフィルタである．プログラムはデータを作らず，データを造るのは人間である．プログラムはデータを一つの形式から別の形式へと変換するだけである．

2 proc ファイルシステム

proc ファイルシステムとは，OS の実行に必要な様々な内部情報を一般のファイルと同様に参照，一部は書き換えが可能なもので，これを用いることで様々な情報を取得可能です．通常，proc ファイルシステムは/proc にマウントされていています．

proc ファイルシステムには先程述べましたように OS の様々な内部情報が書き込まれています．たとえば，ハードディスクのパーティションの情報を見たい場合は，次のコマンドを入力します．

```
junjis@mikilab:~$ cat /proc/partitions
major minor  #blocks  name

  22     0   80418240 hdc
  22     1   80413326 hdc1
   3     0   40209120 hda
   3     1     995998 hda1
   3     2   39206632 hda2
   3    64  120627360 hdb
   3    65  120624021 hdb1
```

このほかにも，proc ファイルシステムには，CPU の情報（cpuinfo），メモリの情報（meminfo），システムの負荷状況（loadavg）など，様々な情報が格納されています．ファイル名を見れば，そのファイルにどのようなことが記述されているかということは想像できると思いますが，数字の名前の付いたディレクトリに関しては理解しにくいと思いますので，別に説明します．

この数字のディレクトリは各プロセス情報に対応しています．ps コマンドを打つと PID というものが表示されますが，この PID とディレクトリの数字がそれぞれ対応しています．このディレクトリの中には，プロセスを起動した際のコマンドラインオプションやプロセスのメモリ割り当て，現在のプロセスの状態といったように様々な情報が格納されています．

3 マウントについて

3.1 mount と umount

物理的なディスクを UNIX 上で用いるには，そのディスクをマウントする必要があります．たとえば，Windows 用にフォーマットされたフロッピーディスクを/floppy にマウントしたい場合は，

```
mount -t vfat /dev/fd0 /floppy
```

のように入力します．これは，「このファイルシステムで」，「このデバイスを」「このディレクトリに」マウントするように，ということを意味しています．

通常，マウントコマンドはスーパーユーザで実行する必要があります．一般ユーザーがマウントを行おうとすると，権限がないために拒否されます．

必要がなくなったディスクは umount コマンドでアンマウントする必要があります．先ほどマウントしたフロッピーをアンマウントする場合は，

```
umount /floppy
```

のように，マウント時に指定したディレクトリを指定することでアンマウントできます．アンマウントしないと，ディスクに正しく情報が書き込まれないので，ディスクが壊れてしまう可能性があります．アンマウントするディレクトリにいる状態でアンマウントしようとするとエラーになってアンマウントできないので，アンマウントするディレクトリから出てからアンマウントするようにしてください．

3.2 /etc/fstab について

/etc ディレクトリに，fstab というファイルがあります．このファイルには，Fig. 1 のように,「どのデバイス」が「どのファイルシステム」で「どのディレクトリ」にマウントされるか，といったことが記述されています．また，それ以外にも各種のオプションを指定することができます．オプションで noauto とつけないと，起動時に自動的にマウントされるようになります．また，user をつけると，一般ユーザからでもマウントすることができるようになります．

このファイルに記述があれば，ディスクをマウントする場合にいちいちファイルシステムやマウント位置を指定しなくても，

```
mount /cdrom
```

と，マウントするディレクトリ位置を指定するだけでディスクをマウントすることができるようになります．

4 X-window-system

UNIX 系 OS は，以前はサーバー用途で使うのに優れているとされてきましたが，最近ではクライアントマシンとしても優れた環境を提供してくれます．ここでは，一般的に用いられている GUI 環境である X Window System を取り上げます．X-window-system には，クライアント／サーバーモデル，ネットワーク透過性という２つの特徴があります．

4.1 X server/X client

X Window System は，それぞれ X server/X client と呼ばれる，2 種類のプログラムが連携して動作しています．ファイルマネージャ，エディタ，Web ブラウザなど，一般に X 上で使っている「プログラム」は，X client です．X server は，画面の描画と，キーボード，マウスなどの入力デバイスからの入力を，X client に伝えるのが仕事です．

4.2 ネットワーク透過性

X client と X server のやり取りには，ネットワークを用いるようになっています．つまり，X client と X server の間にはネットワークがあればいいわけで，必ずしも同じ機械で動いている必要は在りません．もちろん同一のワークステーションで X client と X server を動かすこともできますし，UNIX の動いているワークステーションから，別の UNIX のワークステーションにログインして，ログインした先のワークステーションのデスクトップを表示させることもできるわけです．隣の部屋にあるワークステーションのアプリケーションを，自分の席で利用したりすることができるようになります．

4.3 実際の作業

現在 192.168.6.145 というローカルホストにいて，そこで X server が起動されているとします．ここで xeyes を実行すると，当然 xeyes のウィンドウが画面に表示されます．この状況で実行されているプログラムは，X client (xeyes) と X server です．xeyes が「マウスの位置を教えろ」「円を描け」と X server に要求し，X server はそのリクエストに従ってマウスの情報を渡したり，画面に円を描いたりしています．

では他のホストで X client を実行してみます．まず他のホストにログインする必要があるが，その前に，

```
% xhost +
```

としておきます[2]．そして ssh で forte(202.23.147.73) にログインします．

```
% ssh yoshiki@forte.doshisha.ac.jp
```

ここで xeyes を実行すると，

```
% xeyes
cannot open display.
```

とエラーになってしまいます．これは，
forte 側でウィンドウを開こうとして失敗したからです．192.168.6.145 にウィンドウを開くように指示しなければなりません．

```
% export DISPLAY=192.168.6.145:0.0
```

これで xeyes が実際にリクエストを出す X server は，forte ではなく，192.168.6.145 になります．

[2] X サーバへのアクセスの許可・不許可を設定

```
# /etc/fstab: static file system information.
#
# <file system> <mount point> <type> <options>                    <dump> <pass>
/dev/hda2       /             ext2   defaults,errors=remount-ro 0      1
/dev/hda1       none          swap   sw                         0      0
proc            /proc         proc   defaults                   0      0
/dev/hdc1       /home         ext2   defaults                   0      2
/dev/hdb1       /backup       ext2   defaults                   0      2
/dev/hdd        /cdrom        iso9660 ro
```

Fig. 1　/etc/fstab の中身

```
% xeyes
```

とすると xeyes が表示されるはずです．この状況では，xeyes という X client は forte で動いていますが，xeyes は 192.168.6.145 の X server に対して「円を描け」などと要求しています．

しかし，この方法は，IP による認証のみでユーザの認証が全くないため，セキュリティ上問題があります．これを解消するためには SSH のユーザ認証を用いる方法があり，この方法によって安全性が高まります．

```
% ssh -X forte.doshisha.ac.jp
% xeyes
```

実際にこのようにコマンドを実行すれば，export DISPLAY しなくても X のフォワーディングが可能になります．

4.4　Windows における X server

Windows での X server として代表的なものとして，アステック・プロダクツ社の「ASTEC-X」があります[3]．これを用いることにより，UNIX で動いている X のアプリケーションを，Windows で動いている X server で動かすことができます (Fig. 2)．しかし，この方法では，ユーザ認証はできないので，先ほどの export DISPLAY 同様，セキュリティ上問題が発生してしまいます．

5　netstat コマンド

netstat コマンドを用いると，ネットワークへの接続状況を表示する，ネットワークに流れたパケットの統計を取る，自分のマシンのルーティングテーブルを表示するといったようにネットワークに関する様々な情報を取得することが可能です．

[3]http://www.astec.co.jp/products/ASTECX/astecx.html

Fig. 2　xeyes on Windows

5.1　接続状況の表示

ネットワークへの接続状況を表示するためには，-a オプションをつけて netstat を実行します．こうすることで，現在有効なインターネット接続，UNIX ドメインソケットなどの接続状況が表示されます．インターネット接続だけを表示したい場合は，“-A inet” のように，表示するプロトコルの種類を指定します．

実際に，表示するプロトコルをインターネット接続のみにして接続状況を表示したものを Fig. 3 に示します．この表示の意味は，左から，

- プロトコルの種類
- 受信されなかったバイト数
- 送信されなかったバイト数
- 接続元 (自分自身) の IP アドレスとポート番号

  IP アドレスからホスト名がわかる場合はホスト名が，ポート番号からサービス名がわかる場合はサー

ビス名が表示される．-n オプションをつけると，ホスト名とサービス名の解決は行われない．

- 接続先の IP アドレスとポート番号
- 接続状態

  ESTABLISHED は接続されている状態，LISTEN は接続要求待ちの状態である．

となっています．このなかで，*（ホスト名を解決しない場合は，0.0.0.0）となっているものは，アドレスは自分が持つインターフェースの任意のもの，ポート番号は任意の値を用いることができるということを意味しています．

5.2 NIC ごとの統計量表示

NIC(Network Interface Card) ごとに，どれだけのパケットがやりとりされているかを調べるためには，-i オプションをつけて実行します．こうすることで，データリンク層レベルでどれだけのパケットがやりとりされたかが把握できます．

NIC ごとの統計情報を表示したものを Fig. 4 に示します．この表示は左から順に，

- インターフェース名
- 最大転送単位
- メトリック数（ネットワーク間の距離）
- 受信関連の統計値

  順に、正常パケット数，エラーパケット数，破棄パケット数，オーバーロード パケット数を示す．
- 送信または転送関連の統計値
- 各種フラグ

を意味しています．

5.3 プロトコルレベルでの統計量表示

プロトコルレベルでどれだけのパケットがやりとりされているかを調べるためには，-s オプションをつけて実行します．そうすると，IP や ICMP，TCP や UDP といったプロトコルレベルでどれだけのパケットが受信されたかやどれだけのパケットが受信エラーになったか，どれだけのパケットが送信されたかといったことが把握できます．

プロトコルレベルでの統計量表示したものを Fig. 5 に示します．

5.4 ルーティングテーブルの表示

ある IP パケットがある場合に，送信先アドレスの情報から次にどの IP アドレスにパケットを送信すればいいかを記述したものがルーティングテーブルです．

ルーティングテーブルを表示するには，-r オプションをつけて実行します．これによって，どの IP アドレスが書かれているとどの IP アドレスにパケットが送られるか，ということが確認できます．

ルーティングテーブルを表示したものを Fig. 6 に示します．この表示は左から，

- 送り先の IP アドレス
- 転送先の IP アドレス
- ネットマスク
- 各種フラグ
- 最大セグメントサイズ
- ウインドウサイズ
- 初期ラウンドトリップ時間
- インターフェイス名

を意味しています．

参考文献

1) 「UNIX という考え方」，Mike Gancarz 著，芳尾桂監修，オーム社，2001

```
junjis@mikilab:~$ netstat -aA inet
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 *:pop3                  *:*                     LISTEN
tcp        0      0 *:www                   *:*                     LISTEN
tcp        0      0 *:ftp                   *:*                     LISTEN
tcp        0      0 *:ssh                   *:*                     LISTEN
tcp        0      0 *:smtp                  *:*                     LISTEN
tcp        0      0 localhost:pop3          localhost:49091         ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.147.71:3129      ESTABLISHED
tcp        0      0 mikilab.doshisha.a:pop3 mikilab.doshisha.:49092 TIME_WAIT
tcp        0      0 mikilab.doshisha.a:pop3 mikilab.doshisha.:49093 TIME_WAIT
tcp        0      0 mikilab.doshisha.ac:ssh L024245.ppp.dion.n:1040 ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:1583      ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.147.71:3059      ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.147.71:3121      ESTABLISHED
tcp        1  67161 mikilab.doshisha.ac:www proxy3.sagam1.kn.:53226 LAST_ACK
tcp        0      0 mikilab.doshisha.:45432 arashi.debian.or.j:7586 ESTABLISHED
tcp        0      0 localhost:49091         localhost:pop3          ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:www 202.108.250.207:50505   TIME_WAIT
tcp        0     52 mikilab.doshisha.ac:ssh p53-dn04inage.chi:49152 ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:32882     ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:32883     ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:58612     ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh tky-gw2.celartem.c:1312 ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh YahooBB21814117404:3720 ESTABLISHED
tcp        0    156 mikilab.doshisha.ac:ssh 202.23.143.73:1100      ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh YahooBB21814117404:3726 ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:www 202.108.250.207:50489   TIME_WAIT
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.147.220:55288    ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh YahooBB21814117404:3010 ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:32769     ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:32770     ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:49924     ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:1029      ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:www 202.108.250.207:50481   TIME_WAIT
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.147.71:1044      ESTABLISHED
tcp        0      0 mikilab.doshisha.:45431 netfort.gr.jp:ssh       ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh n00038.max004.phs.:4203 ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:37725     ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh dsl07-052.kcn.ne.:60017 ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:32798     ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.147.71:1037      ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:32799     ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:1040      ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:www 202.108.250.207:50469   TIME_WAIT
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:2515      ESTABLISHED
tcp        0      0 mikilab.doshisha.ac:ssh 202.23.143.73:1044      ESTABLISHED
udp        0      0 mikilab.doshisha.ac:ntp *:*
udp        0      0 localhost:ntp           *:*
udp        0      0 *:ntp                   *:*
```

Fig. 3　ネットワーク接続状況の表示

```
junjis@mikilab:~$ netstat -i
Kernel Interface table
Iface   MTU Met    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0   1500   0 13783855      0      0      0 16308906      0      0      0 BRU
lo    16436   0  3852748      0      0      0  3852748      0      0      0 LRU
```

Fig. 4　各 NIC でのパケット送受信統計

```
junjis@mikilab:~$ netstat -s
Ip:
    17607780 total packets received
    0 forwarded
    0 incoming packets discarded
    17353723 incoming packets delivered
    20160526 requests sent out
    2 reassemblies required
    1 packets reassembled ok
Icmp:
    151993 ICMP messages received
    22759 input ICMP message failed.
    ICMP input histogram:
        destination unreachable: 47803
        echo requests: 81261
        echo replies: 178
    128397 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        destination unreachable: 47136
        echo replies: 81261
Tcp:
    199597 active connections openings
    0 passive connection openings
    4915 failed connection attempts
    0 connection resets received
    52 connections established
    16514405 segments received
    19803949 segments send out
    193860 segments retransmited
    498 bad segments received.
    9161 resets sent
Udp:
    180173 packets received
    47136 packets to unknown port received.
    0 packet receive errors
    227933 packets sent
```

Fig. 5　各プロトコルでのパケット送受信統計

```
junjis@mikilab:~$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
127.0.0.0       *               255.255.255.0   U        40 0          0 lo
localnet        *               255.255.255.0   U        40 0          0 eth0
localnet        *               255.255.255.0   U        40 0          0 eth0
default         202.23.156.1    0.0.0.0         UG       40 0          0 eth0
```

Fig. 6　ルーティングテーブルの表示