第3回 UNIXゼミ

ゼミ担当者 : 折戸 俊彦,荒久田 博士,鈴木 和徳 指導院生 片浦 哲平,谷口 義樹 : 開催日 : 2003年6月6日

ゼミ内容: 第2回 UNIX ゼミにおいて, UNIX 上で使用されているエディタである vi や簡単なファイ ル操作を行うための基本的なコマンドについて紹介した.第3回 UNIX ゼミでは, viの他によく用いら れる Emacs というエディタやファイル操作を行う際に便利なワイルドカードについて紹介する.

1 Emacs

前回紹介した vi の他に, UNIX 上ではよく Emacs という非常に高機能なエディタが使われる. Emacsは, Emacs Lisp というプログラミング言語を用いることで エディタの機能をさらに拡張することができるが,この 節では Emacs の基本的な使い方についてのみ紹介する.

1.1 Emacs でのキー操作

Emacs では,コマンド操作を行うときにはほかのエ ディタとは違ったキー操作をする.これ以降の説明では, Table 1のような表記ルールに従って基本となるファイ ル編集に必要となる操作を説明する.

Table 1	Emacs	でのモ	ー操作	
Ctrl = -	- を押し	かがら	' _v 'を押す	-

C-x	Ctrl キーを押しながら 'x' を押す
M-x	Alt キーを押しながら 'x' を押す
C-x C-c	Ctrl キーを押しながら 'x' を押し,次に
	Ctrl キーを押しながら 'c' を押す

1.2 Emacs でのファイル編集

ここでは, Emacs でのデフォルトのキー操作を説明 する.これらのキー操作は設定によって変更することも 可能である.

基本的には,\$emacsファイル名 と入力し,ファイ ルを編集・作成する.

Emacs でテキストを開くためには, "C-x C-f"と打 つ.そうすると,カーソルが画面の下の方に移るので, ここで編集したいファイル名を入力する.

編集したファイルを保存するためには"C-x C-s"と打 つ.保存するファイル名を変えたい場合は "C-x C-w" と打ち,その後にカーソルが画面の下の方に移るので, 保存したいファイル名を入力する.

コマンド操作をしようとして途中で操作を取り消した い場合は "C-g" と打つ.

文字を消去したい場合は消したい文字の上にカーソル を移動して "C-d" と打つ. DEL キーを押すと, カーソ

ルの前の文字が消去されるので注意する.

文字列を検索したい場合は , "C-s" を入力し , 続けて 検索したい文字列を入力すると,1文字ごとに入力文字 列に一致する文字列にカーソルが移動する. "C-r"とす ると,ファイルの先頭側に検索を実行する.

Emacs を終了するためには "C-x C-c" と打つ.この 際,未保存のファイルは保存するか聞かれるが,'y'を 押すと保存され, 'n'を押すと破棄することができる.

1.3 viとEmacsの比較

前項で紹介したように Emacs は非常に高機能なエディ タであり,わざわざ変わった操作を行わないと使えない viを使う必要性は感じられないかもしれない.しかし, UNIX では以下のような理由により viを使うことがある.

- Emacs は UNIX の標準エディタではないので,環 境によっては Emacs が入っていないこともある.vi はほぼすべての UNIX システムに標準で用意され ている.
- ちょっとした編集作業の場合は Emacs を起動する よりも, vi の方が処理が速い.

Table 2 に Emacs の編集操作の方法を掲載する.

Emacs	機能
C-x C-f	ファイルを読み込む
C-x C-s	ファイルを保存する
C-x C-w	ファイルを別名で保存する
C-g	コマンド操作を取り消す.
C-d	文字を消去する
C-s	ファイル末尾方向への検索
C-r	ファイル先頭方向への検索
C-x C-c	エディタを終了する
M-x replace-string	文字列の置き換え
M-g 50	50 行目にカーソル移動

Table 9 Emples のコマンド書

2 ワイルドカード

ワイルドカードを用いると一度に複数のファイルを扱うことが可能であることから,コマンドライン上で複数のファイルを扱う際には便利である.1つのコマンドで複数のファイルやディレクトリを同時に操作したい場合はワイルドカードと呼ばれる特殊な記号を用いる.複数のファイルを指定できるコマンドであれば,すべてのワイルドカードを使用可能である.ワイルドカードにはTable 3 のようなものがある.

Table 3	ワイルド カート	ヾの種類
Table 5	ノール・ハー	・レノイギデロ

記号	置き換え
*	0 文字以上の任意の文字列
?	任意の1文字
[]	[]内に指定された任意の1文字
[^]	[]内に指定されていない任意の1文字
[!]	[]内に指定されていない任意の1文字
{,}	「 ,」で区切られた文字列のいずれか

2.1 ワイルドカードの記述例

Table 4 ワイルドカードの記述例

記述例	意味
*tmp	tmp で終わるファイル.tmp も含む.
tmp	tmp という文字列を含むファイル
[a-z]*	a から z までの文字で始まるファイル
[-a-z]*	ハイフン,a~z から始まるファイル
[a-zA-Z]*	アルファベットで始まるファイル
[0-9]	ファイル名の中に数字が含まれているもの
[!0-9]*	数字で始まらないファイル
??	名前が2文字のファイル
??*	名前が2文字以上のファイル
tmp/*	tmp というディレクトリ下の全ファイル

Table 4 にワイルドカードの記述例を示す.アルファ ベット全体を指したい場合は [a-Z] と書かずに, [a-zA-Z] と書く必要がある.文字コードの大小で範囲を判断し ているため, [a-Z] とは [0x61-0x5A] のことであり, 範 囲内に何も含まない.逆に, [A-z] においては間に余計 なコードを含んでしまう.その他,注意すべき項目は {test1,test2}という記述である「,」の後にスペース を入れると「{test1,」と「test2}」の2引数と判断さ れてしまう.

以下に Table 3 または Table 4 の記号の使用例を示す. Fig. 1 は*の使用例である.ls ttya*のように使うと, ttya で始まるカレントディレクトリの全てのファイルを

表示することができる.

) O O kterm O								
	hello4 ttya0 ttya1 hello4	:/dev# ttya2 ttya3 :/dev#	ls ttya ttya4 ttya5 ∎	* ttya6 ttya7	ttya8 ttya9	ttyaa ttyab	ttyac ttyad	ttyae ttyaf	
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,									

Fig. 1 *の使用例

Fig. 2 は?の使用例である.ls ttya?のように使うと, ttya+「任意の一文字」のカレントディレクトリ内のファ イルを全て表示することができる.



Fig. 2 ?の使用例

Fig. 3 は [] の使用例である.ls tty[abcde]*のように 使うと, ttyの後に a,b,c,d,e のうちのどれかがくるカレ ントディレクトリ内のファイルを全て表示することがで きる.

000 kterm								
hellod:/dev#ls ttya0 ttya5 t ttya1 ttya5 t ttya2 ttya5 t ttya2 ttya5 t ttya2 ttya5 t ttya5 ttya6 t ttya6 ttya6 t ttya6 ttya6 t ttya7 ttyaf t hello4:/dev# ■	i tujBacde]* itub0 ttub8 ttuc0 itub0 ttub8 ttuc1 itub2 ttub8 ttuc1 itub2 ttub8 ttub0 itub4 ttub0 ttuc3 itub4 ttub0 ttuc4 itub5 ttub6 ttuc6 itub6 ttube ttuc6 itub7 ttubf ttuc7	ttyc8 ttyd0 tty ttyc9 ttyd1 tty ttyce ttyd2 tty ttycb ttyd2 tty ttycb ttyd4 tty ttycc ttyd4 tty ttycc ttyd6 tty ttyce ttyd6 tty ttycf ttyd7 tty	d8 ttye0 ttye8 d9 ttye1 ttye9 da ttye2 ttyee db ttye3 ttyeb dc ttye4 ttyec dd ttye5 ttyed de ttye6 ttyee df ttye7 ttyef					

Fig. 3 []の使用例

Fig. 4は [-] の使用例である.ls tty[a-e]*のように使う と, Fig. 3と同じ意味を表わし, tty に続いて a,b,c,d,e のうちのどれかがくるカレントディレクトリ内のファイ ルを全て表示することができる.つまり [abcde] は [a-e] と記述することができるといえる.

Fig. 5は [!] の使用例である.ls tty[!a-z]*ように使う

000 kterm							
hello4:/dev#1s ttyj ttya0 ttya8 ttyb0 ttya1 ttya2 ttyb0 ttya2 ttyaa ttyb2 ttya3 ttya2 ttya3 ttya6 ttyb3 ttya4 ttya6 ttyb3 ttya4 ttya6 ttyb6 ttya7 ttyaf ttyb7 hello4:/dev#	lt ttyb5 ttyc0 ttyc5 ttyd0 ttyd5 ttye0 ttye5 ttyb9 ttyc1 ttyc9 ttyd1 ttyd9 ttye1 ttye9 ttyb9 ttyc2 ttyc3 ttyd2 ttyd3 ttye4 ttyb6 ttyc3 ttyc5 ttyd3 ttyd6 ttye3 ttyeb ttyb6 ttyc3 ttyc5 ttyd3 ttyd6 ttye4 ttyb6 ttyc5 ttyc4 ttyd5 ttyd4 ttye4 ttyb6 ttyc5 ttyc4 ttyd5 ttyd4 ttye6 ttyb6 ttyc5 ttyc4 ttyd5 ttyd4 ttye6 ttyb6 ttyc5 ttyc4 ttyd5 ttyd6 ttye6 ttyb6 ttyc5 ttyc4 ttyd5 ttyd6 ttye6 ttyb6 ttyc7 ttycf ttyd7 ttydf ttye7 ttyef						

Fig. 4 [-] の使用例

と, tty に続く文字が a~z でないカレントディレクトリ 内のファイルを全て表示することができる.

					kterm				
hello4:, tty0 tty10 tty11 tty12 tty13 tty14 tty15 tty16 tty16 tty16 tty18 tty18 tty18 tty18 tty19 hello4:,	/dev# tty2 tty20 tty21 tty22 tty23 tty24 tty25 tty25 tty27 tty28 tty29 tty29 tty29 /dev#	ls tty[tty30 tty31 tty32 tty34 tty35 tty36 tty36 tty37 tty38 tty39 tty4 tty40	!a-z]* tty41 tty42 tty43 tty44 tty45 tty46 tty47 tty46 tty47 tty49 tty50 tty51	tty52 tty53 tty55 tty55 tty55 tty57 tty58 tty58 tty56 tty60 tty61 tty62	tty63 tty7 tty8 tty8 ttyACM1 ttyACM1 ttyACM10 ttyACM12 ttyACM12 ttyACM13 ttyACM15	ttyACM2 ttyACM3 ttyACM4 ttyACM6 ttyACM7 ttyACM8 ttyACM8 ttyACM8 ttyACM8 ttyS0 ttyS1 ttyS2 ttyS3	ttyUSB0 ttyUSB1 ttyUSB10 ttyUSB12 ttyUSB13 ttyUSB14 ttyUSB14 ttyUSB2 ttyUSB2 ttyUSB3 ttyUSB3 ttyUSB3	ttyUSB6 ttyUSB7 ttyUSB8 ttyUSB9 ttyUSB9	

Fig. 5 [!] の使用例

Fig. 6 は [^] の使用例である.ls tty[^]*のように使う と ls tty[!a-z]*と同じことを表わし, tty に続く文字が a ~ z でないカレントディレクトリ内のファイルを全て表 示することができる.

000	000 kterm								
hellod:/dev# 1s tty tty0 tty2 tty30 tty1 tty20 tty31 tty10 tty21 tty32 tty11 tty20 tty33 tty11 tty22 tty33 tty12 tty23 tty34 tty13 tty24 tty35 tty14 tty26 tty37 tty15 tty26 tty37 tty15 tty28 tty38 tty17 tty38 tty29 tty4 tty18 tty29 tty4 hello4:/dev#	<pre>[^a-2]* tty41 tty52 tty63 tty42 tty53 tty7 tty43 tty54 tty6 tty44 tty55 tty96 tty45 tty66 tty9CM0 tty46 tty57 tty9CM1 tty47 tty58 tty9CM1 tty44 tty59 tty9CM1 tty44 tty60 tty9CM1 tty50 tty60 tty9CM1 tty50 tty60 tty9CM1 tty51 tty62 tty9CM15</pre>	ttyACM2 ttyUSB0 ttyACM3 ttyUSB1 ttyACM4 ttyUSB10 ttyACM5 ttyUSB10 ttyACM5 ttyUSB13 ttyACM5 ttyUSB13 ttyACM6 ttyUSB15 ttyACM6 ttyUSB15 ttyGCM5 ttyUSB15 ttyS1 ttyUSB15 ttyS2 ttyUSB4 ttyS3 ttyUSB5	ttyUSB6 ttyUSB7 ttyUSB8 ttyUSB9 ttyUSB9						

Fig. 6 [[^]] の使用例

Fig. 7 は,の使用例である.ls *{USB,tec}*のように 使うと USB, または tec の文字列を含むカレントディレ クトリ内のファイルを全て表示することができる.

2.2 ワイルドカードの仕組み

ワイルドカードはシェルが展開する.つまり,コマン ドの引数にワイルドカードを含むファイルを指定した場 合,コマンドが引数を処理する前に,シェルがワイルド カードを解釈し,複数の引数に展開した上でコマンドに

000		kterm			0
hello4:/dev# 1s *{USB,tec}* fb0autodetect fb1autodetect fb1autodetect fb5autodetect fb2autodetect fb5autodetect fb3autodetect fb7autodetect hello4:/dev# [ttyUSB0 ttyUSB1 ttyUSB10 ttyUSB11	ttyUSB12 ttyUSB13 ttyUSB13 ttyUSB14 ttyUSB15	ttyUSB2 ttyUSB3 ttyUSB4 ttyUSB5	ttyUSB6 ttyUSB7 ttyUSB8 ttyUSB9 ttyUSB9	0

Fig. 7 {,} の使用例

受け渡す.

2.3 ワイルドカード使用上の注意

cp のような「複数のファイルを引数とし,その並び 順に意味があるコマンド」の場合や,findのようなコマ ンド自身もワイルドカードが処理できる場合には思わぬ 動作をする可能性がある.

cp での失敗例

test で始まるファイルを別のディレクトリにコピー するつもりで「cp test*」したとする.このディレ クトリに test1 と test2 があった場合,コマンドは シェルによって「cp test1 test2」に展開される.つ まり,test1を test2 にコピーするという処理になっ てしまう.

find での失敗例

test で始まるファイルを指定したい場合「find.name test*」としても実行できない.正規表現に マッチするカレントディレクトリ内の全ファイルを 探したい場合は「find.-name"test*"」する必要 がある.

3 プロセスに関するコマンド

第1回 UNIX ゼミではプロセスとジョブという内容 があったが,今回,プロセスに関する基本的なコマンド を紹介する.Table 5 にコマンドを示す.

Table 5	プロセスに関するコマンド	表
---------	--------------	---

コマンド	機能
ps	実行中プロセスの情報を表示
top	プロセスの CPU 占有率を表示
kill	プロセスを停止させる

ps または top コマンドでプロセス ID を調べ, kill [プロセス ID] とすることでプロセスを殺すことがで きる.

kazunoris@su	uzukik:~\$ ps	
PID TTY	TIME CMD	
1797 pts/0	00:00:00 bash	
1805 pts/0	00:00:01 emacs	
1816 pts/0	00:00:00 ps	
kazunoris@su	uzukik:~\$ kill 1805	
PID TTY	TIME CMD	
1797 pts/0	00:00:00 bash	
1816 pts/0	00:00:00 ps	

Table 6 に ps のよく使われるオプションを紹介する.

	Table 6	コマンド	ps のオプション
--	---------	------	-----------

オプション	機能
-a	全てのプロセスの状態を表示
-u	ユーザ名を含めてプロセスを表示
-x	制御端末を持たないものを含めて表示

これらのオプションを使った例を次ページに一部示す. また top で CPU の占有率を調べ,占有率の高いプロ セスを殺したりもする.また kill コマンドを実行しても 終了できない場合がある.そのときは kill -9 のオプショ ンを使うことで強制終了することができる.

top の使用例を次ページに一部示す. CPU 占有率の 高い順に表示され,自動的に情報が更新される.

4 マウントとは

Windows にはドライブという概念があり, Cドライ ブ, Dドライブのように呼ぶが, Linux にはドライブと いう概念はない. すべてルートディレクトリを出発点と してアクセスできるようになっている. そのため, CD-ROM や複数のハードディスクを扱う場合, マウントと いうしくみを用いる.

マウントとは,別のディレクトリツリーをルートディ レクトリ以下のディレクトリツリーに結合することを 言う.例えば,CD-ROMドライブを/mnt/cdromとい うディレクトリにマウントした場合,CD-ROMの中に ある「test」というディレクトリは「/mnt/cdrom/test」 として扱うことができる.こうすることで,ユーザー はファイルの物理的な場所を意識せず,統一した方法で ファイルを扱うことができる.

4.1 マウント

ファイルシステムをマウントするときは mount コマ ンドを用いる.mount コマンドは「mount オプション デバイス名 マウントポイント」の書式でスーパーユー ザのみが使用できる.

mount -t iso9660 /dev/cdrom /mut/cdrom

ただし,/etc/fstab で設定するとその設定に従ったマ ウント方法であれば,一般ユーザでもマウントできるよ うになる.一般ユーザが行う場合は「mount デバイス 名(またはマウントポイント)」のような書式になる.

\$ mount /floppy

「 mount 」のみで実行した場合は現在のマウント状況 が表示される.一般ユーザでも実行できる.

orito@hello4:~\$ mount
/dev/hda3 on / type ext2
proc on /proc type proc
devpts on /dev/pts type devpts

4.2 アンマウント

FD, CD-ROM などの取はずし可能なデバイスは,マ ウントを解除 (アンマウント) してから取り出す.アン マウントは umount コマンドを用いる.

umount /floppy

/etc/fstabの設定で,一般ユーザでもマウント可能に なっている場合は,一般ユーザでアンマウントすること ができる.

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ある.

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 を実行する.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る.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する.

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. 9 に示する.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

suzukik@mi	kilak	o:~\$ p	os aux								
USER	PID	%CPU	%mem	VSZ	RSS	TTY	ST	AT	START	TIME	COMMAND
root	1	0.0	0.0	1272	408	?	S		May19	0:04	init [2]
root	2	0.0	0.0	0	0	?	SW		May19	0:00	[keventd]
root	3	0.0	0.0	0	0	?	SWI	N	May19	0:00	[ksoftirqd_CPU0]
root	4	0.0	0.0	0	0	?	Z		May19	0:15	[kswapd <defunct></defunct>
root	5	0.0	0.0	0	0	?	SW		May19	0:00	[bdflush]

Fig. 8 オプションつきの ps コマンド (一部)

17:46:5 222 proc CPU state Mem: Swap: 1	up 9 days esses: 220 es: 0.6% 513280K tot 595980K tot	s, 8 slee user cal, cal,	:08, 2 ping, , 1, 4991 300	25 use 1 rur .0% sy 184Κ ι)00Κ ι	ers, 2 nning, ystem, used, used,	load 1 zon 0.0 14 1565	averag mbie, 0% nic 096K : 980K :	ge: 0.0 0 stop ce, 98 free, free,	7, 0.08, 0.06 pped 8.4% idle 60520K buffers 259048K cached
PID US	ER PRI	NI	SIZE	RSS	SHARE	STAT	%CPU	%MEM	TIME COMMAND
11624 su	UKIK I/	0	1064	1064	748	R a	0.7	0.2	0:00 top
25794 eg	ami 9	0	1836	1808	1464	S	0.1	0.3	0:00 sshd
1 ro	ot 8	0	448	408	388	S	0.0	0.0	0:04 init
2 ro	ot 9	0	0	0	0	SW	0.0	0.0	0:00 keventd
3 ro	ot 19	19	0	0	0	SWN	0.0	0.0	0:00 ksoftirqd_CPU0
4 ro	ot 9	0	0	0	0	Z	0.0	0.0	0:15 kswapd <defunct></defunct>

Fig. 9 top コマンドの使用例(一部)

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.

となっている.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る.

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する.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 が把握できる.

NICごとの統計情報を表示したものを Fig. 10 に示す. 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

を意味している.

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 実行する.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 できる.

プロトコルレベルでの統計量表示したものを Fig. 11 に示す.

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ある.

ルーティングテーブルを表示するには,--rオプション をつけて実行する.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る.ルーティングテーブルを

suzul	kik@mikila	ab:~\$ r	netstat -aA inet		
Acti	ve Interne	et conr	nections (servers and est	tablished)	
Prot	o Recv-Q S	send-Q	Local Address	Foreign Address	State
тср	0	0	*:time	*:*	LISIEN
тср	0	0	*:pop3	*:*	LISIEN
tcp	0	0	*:WWW	*:*	LISIEN
tep	0	0	*:0470	*:* *·*	LISIEN
tcp	0	0	*.5511	*.* *·*	LISIEN
tep	0	0	*.Smup	*.* 202 22 142 72,00501	ETADI TOUED
tcp	1	0	mikilab dochicha · 36/31	zuz.zu.140.70.20021	CLOSE WATT
tcp	1	0	mikilah doshisha a.non3	mikilab doshisha	TIME WATT
tcp	0	0	mikilah doshisha acissh	202 23 147 71.3454	FSTARI ISHED
tcp	0	0	mikilab doshisha acissh	202.23.143.73.1068	ESTABLISHED
tcp	0	0	mikilab doshisha ac:uuu	1068-5-197-194 oc 47360	TIME WATT
tcp	0	0	mikilah doshisha a:non3	mikilab doshisha :46405	TIME WATT
tcp	0	0	mikilab doshisha acissh	202 23 143 73.29230	ESTABLISHED
tcp	Ő	Ő	mikilab.doshisha.ac:ssh	202.23.147.71:4156	ESTABLISHED
tcp	1	0	mikilab doshisha :40408	mikilab doshisha ·40405	CLOSE WAIT
tcp	0	0	localhost:pop3	localhost:46408	TIME WAIT
tcp	Ő	Ő	mikilab.doshisha.ac:ssh	FLA1Aae158 kng mes:1047	ESTABLISHED
tcp	Ő	Ő	localhost:pop3	localhost:46411	TIME WAIT
tcp	0	Õ	mikilah doshisha actsh	202 23 147 71.54262	ESTABLISHED
tcp	Ő	Ő	mikilab doshisha ac:ssh	n3030-inad02dajan:64835	ESTABLISHED
tcp	Ő	Ő	mikilab doshisha ac:ssh	202 23 147 71.54263	ESTABLISHED
tcp	0	Õ	mikilab doshisha a:pop3	mikilab.doshisha.:46414	TIME WAIT
tcp	0	Õ	mikilab doshisha ac:ssh	202, 23, 143, 73:28390	ESTABLISHED
tcp	0	Õ	mikilab.doshisha.a:pop3	mikilab.doshisha.:46416	TIME WAIT
tcp	1	Ő	mikilab.doshisha.:55372	mikilab doshisha :55369	CLOSE WATT
tcp	- 1	Õ	mikilab doshisha :39566	mikilab.doshisha.:39563	CLOSE WAIT
tcp	0	Õ	mikilab.doshisha.ac:ssh	202.23.143.73:eklogin	ESTABLISHED
tcp	0	0	mikilab.doshisha.ac:ssh	202.23.143.73:32825	ESTABLISHED
tcp	1	0	mikilab.doshisha.:42552	mikilab.doshisha.:42549	CLOSE WATT
tcp	0	Õ	mikilab.doshisha.ac:ssh	202.23.143.73:1596	ESTABLISHED
tcp	1	0	mikilab.doshisha.:39570	mikilab.doshisha.:39567	CLOSE WAIT
tcp	- 1	0	mikilab.doshisha.:50198	mikilab.doshisha.:50195	CLOSE WATT
tcp	0	0	mikilab.doshisha.ac:ssh	202.23.147.71:4589	ESTABLISHED
tcp	1	0	mikilab.doshisha.:55364	mikilab.doshisha.:55361	CLOSE WAIT
tcp	1	0	mikilab.doshisha.:39555	mikilab.doshisha.:39552	CLOSE WAIT
tcp	0	0	mikilab.doshisha.ac:ssh	kutf21910260203.ge:3020	ESTABLISHED
tcp	0	0	mikilab.doshisha.ac:ssh	202.23.143.73:1396	ESTABLISHED
tcp	1	0	mikilab.doshisha.:55368	mikilab.doshisha.:55365	CLOSE WAIT
tcp	0	0	mikilab.doshisha.ac:www	gate.toyama-u.jp:3558	TIME_WAIT
tcp	0	0	mikilab.doshisha.ac:ssh	202.23.143.73:1032	ESTABLISHED
tcp	1	0	mikilab.doshisha.:40123	mikilab.doshisha.:40120	CLOSE_WAIT
tcp	0	19824	mikilab.doshisha.ac:www	p6e41f6.freedc01.a:1851	ESTABLISHED
tcp	1	0	mikilab.doshisha.:44686		CLOSE_WAIT
tcp	0	0	mikilab.doshisha.ac:ssh	202.23.143.73:1037	ESTABLISHED
tcp	1	0	mikilab.doshisha.:45784	mikilab.doshisha.:45781	CLOSE_WAIT
tcp	0	0	mikilab.doshisha.ac:ssh	202.23.143.73:1167	ESTABLISHED
tcp	0	0	mikilab.doshisha.ac:ssh	202.23.143.73:28123	ESTABLISHED
tcp	1	0	mikilab.doshisha.:45769	mikilab.doshisha.:45766	CLOSE_WAIT
tcp	0	0	<pre>mikilab.doshisha.ac:ssh</pre>	202.23.143.73:2524	ESTABLISHED
tcp	0	0	mikilab.doshisha.ac:ssh	202.23.147.71:2370	ESTABLISHED
tcp	1	0	mikilab.doshisha.:45774	mikilab.doshisha.:45771	CLOSE_WAIT
tcp	0	0	<pre>mikilab.doshisha.ac:ssh</pre>	202.23.143.73:2065	ESTABLISHED
tcp	0	0	<pre>mikilab.doshisha.ac:ssh</pre>	202.23.147.71:1030	ESTABLISHED
tcp	1	0	mikilab.doshisha.:45778	mikilab.doshisha.:45775	CLOSE_WAIT
tcp	1	0	mikilab.doshisha.:37554	mikilab.doshisha.:37551	CLOSE_WAIT
tcp	0	0	$\tt mikilab.doshisha.ac:ssh$	202.23.147.71:1029	ESTABLISHED
tcp	0	0	<pre>mikilab.doshisha.ac:ssh</pre>	202.23.143.73:1047	ESTABLISHED
udp	0	0	${\tt mikilab.doshisha.ac:ntp}$	*:*	
udp	0	0	localhost:ntp	*:*	
udp	0	0	*:ntp	*:*	

Fig. 10 ネットワーク接続状況の表示

suzuki	ik@miki	lab	:~\$ nets	tat -i							
Kernel	l Inter	face	e table								
Iface	MTU	Met	RX-OK	RX-ERR	RX-DRP	RX-OVR	TX-OK	TX-ERR	TX-DRP	TX-OVR	Flg
eth0	1500	0	9483877	0	0	01	1376806	0	0	0	BMRU
lo	16436	0	1778212	0	0	0	1778212	0	0	0	LRU

Fig. 11 各 NIC でのパケット送受信統計

```
suzukik@mikilab:~$ netstat -s
Ip:
    11261932 total packets received
    0 forwarded
    0 incoming packets discarded
    11104119 incoming packets delivered
    13155266 requests sent out
Icmp:
    7472 ICMP messages received
    6860 input ICMP message failed.
    ICMP input histogram:
        destination unreachable: 433
        echo requests: 22
        echo replies: 159
    8251 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        destination unreachable: 8229
        echo replies: 22
Tcp:
    146942 active connections openings
    0 passive connection openings
    2 failed connection attempts
    0 connection resets received
    55 connections established
    10401384 segments received
    12460838 segments send out
    65641 segments retransmited
    91 bad segments received.
    69070 resets sent
Udp:
    660784 packets received
    8229 packets to unknown port received.
    0 packet receive errors
    686004 packets sent
```

Fig. 12 各プロトコルでのパケット送受信統計

suzukik@mikila	ab:~\$ netstat -r				
Kernel IP rout	ing table				
Destination	Gateway	Genmask	Flags	MSS Window	irtt Iface
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. 13 ルーティングテーブルの表示

表示したものを Fig. 12 に示す.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

を意味している.

参考文献

- Linux のすべて
 西村めぐみ,株式会社 日本実業出版社
- 2) UNIX シェルプログラミングBruce Blinn , SOFT BANK