MRTG を利用したネットワークの障害検知

03H017 大西由之

目次

1 はじめに 1

2 システム 2
2.1 システムの目的 2
2.2 本研究で提案するシステムの概要 2

3 システムの構成 3
3.1 MRTG ログデータの取得 3
3.2 基準値ファイルの作成 4
3.3 日付の取得 4
3.4 ログデータの選択 4
3.5 管理者へのメイル送信 5
3.6 Web ブラウザへの表示 6

4 考察 7
4.1 本システムの利点 7
4.2 本システムの課題 7

5 まとめ 8

付録 A プログラムのソース 10
A.1 ネットワークの状態を Web ブラウザへ表示するプログラム 10
A.2 メイル送信の判定プログラム 15
A.3 メイル送信プログラム 19
A.4 crontab による定期実行 19

1　はじめに

今日では、様々なネットワークを利用したシステムがある。そして、それらのシステムは常時稼働していることが理想である。しかし、常に異常のないシステムを作成することは困難である。そのため、ネットワークの障害を早期発見することが重要である。

これを実現するには、ネットワークの状態を常に監視することが必要である。ネットワークの監視を行うことで、万が一障害が発生した時、その障害の発生をすぐに知ることが可能となり、障害への素早い対応が可能となる。

しかし、人間が常にネットワークの状態を監視し続けることは困難である。そこで、ネットワークの監視を自動的に行い、異常が発生した場合に、管理者へ通知するシステムが必要となる。

研究室内では、ネットワーク監視ツールとして、MRTG[*1]が利用されている。SNMP[*2]によって取得されたトラフィックデータを、MRTG によって、グラフ化することで、視覚的にネットワークを監視することが出来るツールである。このグラフを見ることで、管理者はネットワークの障害発生の有無を知ることができ、異常を発見した場合にすぐに対応することが可能になる。

下の図は実際にグラフ化されたデータの例である。

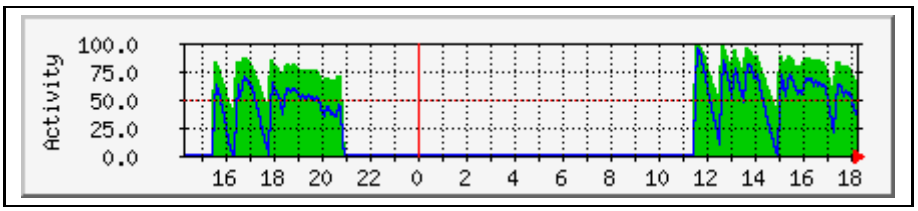

図 1　MRTG によってグラフ化されたデータ

しかし管理者が障害発生を確認するには、MRTG のグラフの変化を見て判断しなければならない上に、常時グラフの変化を見つづけるのは困難である。そこで、今回の研究では、ネットワークの障害検知を自動化し、ネットワークの障害を、早期発見する事を可能にするシステムを作成することが可能になると考えている。

[*1] Multi Router Traffic Grapher の略。様々な数値をグラフ化することが可能なソフトウェアである。これを利用してネットワークの監視が可能になる。

[*2] Simple Network Management Protocol の略。ルータなどのトラフィックデータを数値化するプロトコル。

2　システム

2.1　システムの目的

今回のシステムを作成することによって、障害検知から対応までの時間短縮と、システム管理者の、常時監視の負担を軽減することを目的とする。

2.2　本研究で提案するシステムの概要

SNMP によって取得されたトラフィックデータは、MRTG のログディレクトリに格納される。その格納されたデータを元にしてグラフ化されている。今回の研究では、MRTG のログディレクトリに格納されているログファイルを元にして、ネットワークの障害検知を行う。

- MRTG のログデータの取得
  h-ps011 サーバにある MRTG のログデータを取得し、ネットワークの障害検知に利用する。
- 基準となるログデータファイルの作成
  まず、MRTG のログディレクトリから、基準となるログデータファイルを作成する。また、平日・土日祝日によって、研究室内のマシンの利用率が異なるため、それぞれ基準となるログデータファイルを作成する。
- 日常のログデータの読み込み
  MRTG のログディレクトリに格納されているデータファイルを読み込む。そして、最後に更新されたデータを抜き出す。
- 日付の判定
  平日・土日祝日の判定を行う。そして、その結果に対応した基準となるログファイルのデータを読み込む。
- ログファイルの比較と判定
  基準となるログデータファイルと、日常のログデータファイルを比較する。比較した結果、基準データと日常のデータに大きな差があった場合は、障害が発生したという判定をする。
- 警告メイルの送信
  障害発生の判定が出た場合、管理者宛てに警告メイルを送信するプログラムを作成する。これにより、常にグラフの変化を確認することなく、ネットワークの障害を知ることが出来る。
- Web ブラウザへの表示
  Web ブラウザ上に、障害発生の有無を表示させるプログラムを作成する。表示方法としては、正常に稼働している場合は、緑色で表示、異常が発生した場合の表示は、赤色の表示をさせる。
  これにより、Web ブラウザ上でグラフをの変化を確認することなく、障害発生を知ることが出来る。

これらを実現することにより、ネットワーク監視システムの向上が可能になると考えている。

3　システムの構成

3.1　MRTG ログデータの取得

MRTG のログデータは、h-ps011 サーバ内にある、MRTG のログディレクトリに、ログファイルとして格納されている。このログファイルには、UNIX TIME やネットワークのトラフィックデータが書き込まれている。

今回の研究では、このネットワークのトラフィックデータを元にネットワークの障害検知を行った。

下の図は MRTG のログファイルのデータの中身である。今回の実験では配列の三つめのネットワークのトラフィック量を数値化したデータを用いて実験を行った。

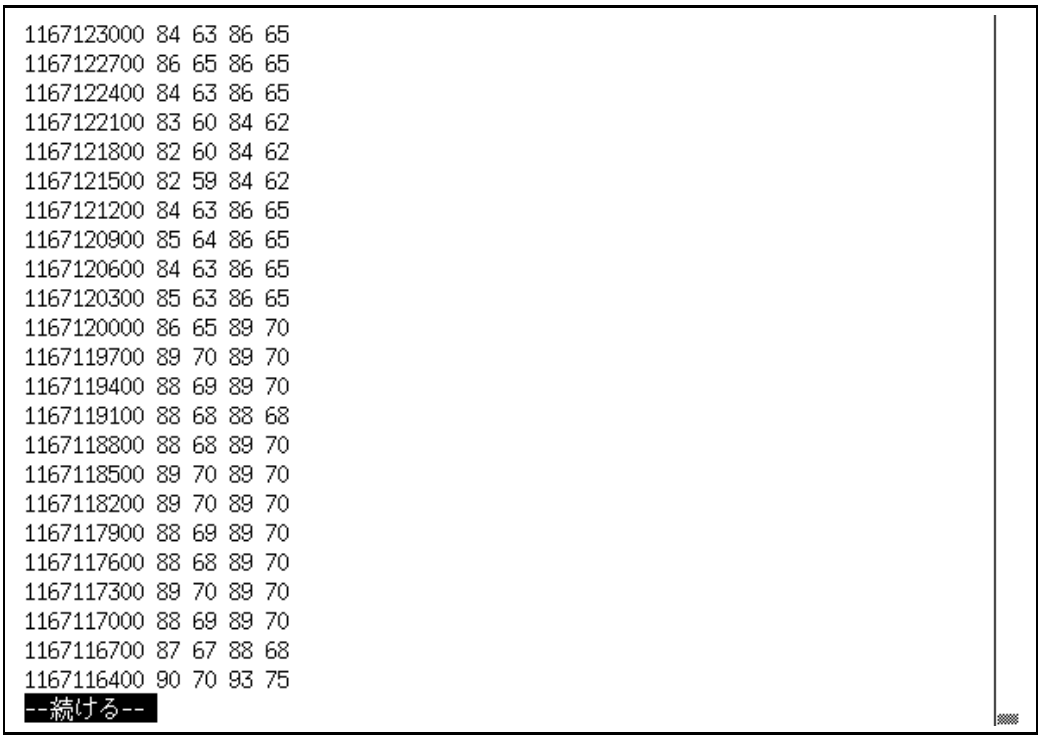

図 2　MRTG のログファイルのデータ

3.2　基準値ファイルの作成

ログデータは、平日・土日祝日によってそれぞれマシンの利用率が異なる為、別々に基準となるデータファイルが必要となる。

よって、平日・土曜・日曜祝日ごとに、基準となるデータファイルを作成し、基準値を入力しておく。

3.3　日付の取得

基準値となるログデータのファイルを選択するには、現在の日付の情報が必要である。

まず、現在の年月日、及び曜日を取得し、その日付を元に、ログデータと比較するための基準値ファイルの選択を行う。

また、祝日の判定は、最初にテキストファイルに祝日の年月日一覧を記入しておき、祝日テーブルを作成する。この祝日テーブルの中に、先ほど取得した年月日と、一致するものがあるかどうかを調べるプログラムを作成する。一致した場合には、祝日という判定をする。

なお、日曜と祝日のネットワークの状態は同じ扱いとする。

3.4　ログデータの選択

取得した日付から、曜日を判定し、平日の場合は平日の基準ファイル、土曜日の場合は土曜日の基準ファイル、, 日曜・祝日の場合は日曜・祝日の基準ファイルを、それぞれ選択し、読み込む。

3.5　管理者へのメイル送信

障害が発生した場合、対応するにはその情報をすぐに管理者が知る必要がある。そこで、障害が発生した場合に管理者へメイルを送信するプログラムを作成した。

作成した基準値ファイルと、現在のログデータを比較する。ネットワークの状態を比較し、エラーが出た場合には、名前をテキストファイルに入力していく。そして、すべての判定が終わった後に、そのテキストファイルに入力された内容を管理者にメイル送信する。これにより、ネットワークのどの部分に障害が発生したかどうかをしることができる。

また、異常がない場合にはメイルが送信されないように、テキストファイルの中身が空の場合は送信しないようにし、プログラムが実行される度にテキストファイルを削除するようにした。

このプログラムは定期的に実行する必要があるので、crontab を使用して、5 分毎に実行させるようにした。

下の図はメイルが送信された場合の受信画面である。

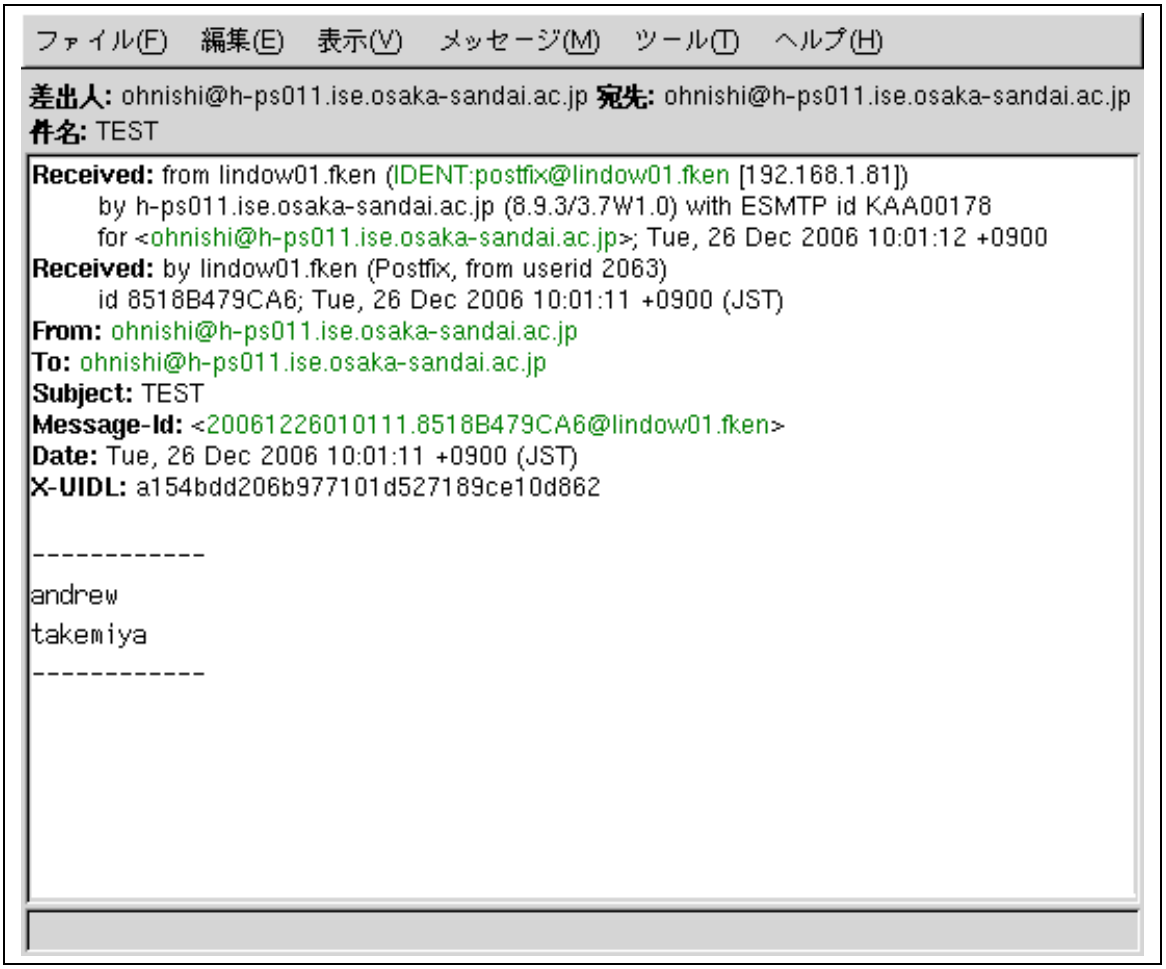

図 3　障害発生時のメイル受信画面

3.6　Web ブラウザへの表示

管理者が忙しい場合、メイルを見る余裕がない場合がある。この時、メイルを見てから障害の発生に気づいたのでは時間がかかる。そこで、Web ブラウザに、障害発生の有無を表示させておくことで、障害が発生したかどうかを、ブラウザ上ですぐに確認することが出来る。

基準値ファイルと、ログデータを比較する。そして、正常だと判断した場合には、「OK」という表示をし、基準値ファイルよりも大きくなった場合は「WARNING」と表示する。

下の二つの図は、ネットワークの状態が正常な場合と、障害が発生した場合の表示である。文字だけでは、見にくいので、色でも識別できるように、正常な場合は緑色、障害が発生した場合は赤色で表示させるようにした。今回は実験のため、基準値を変更して表示させた。

name	status
andrew	warning
uetsuji	ok
ohnishi	ok
oguri	ok
kashio	ok
takemiya	warning
tadaaki	ok
nakamae	ok
hajime	ok
fujikawa	ok
mikane	ok
yada	ok

図 4　障害発生時の Web ブラウザへの表示

4　考察

4.1　本システムの利点

- 警告メイルの送信による障害発生の早期発見
  管理者へメイルを送信することによって、ネットワークの障害へのすみやかな対応が可能になる。
- Web ブラウザへの表示によるネットワークの状態確認の簡易化
  Web ブラウザへ、現在のネットワークの状態を表示することによって、管理者が作業中で忙しい場合でも、MRTG のグラフを見て状態の変化を確認する手間が省け、Web ブラウザを表示し、緑色の表示がされているか確認するだけで、ネットワークの状態を知ることができる。

4.2　本システムの課題

- プログラムの修正
  現在のプログラムでは、Web ブラウザへの表示のプログラムと、メイル送信のプログラムの、前半部分の処理が同じである。この同一部分の処理を 1 つのプログラムにまとめることで、処理の高速化と、負荷の軽減が可能であると思われる。

5　まとめ

今回の研究では、MRTG に格納されるログディレクトリ内のデータを利用して、基準となるデータとの比較を行うことにより、ットワークの障害検知を行った。警告メイルを送信することで、管理者が常時グラフの監視をする必要がなくなった。

また、ネットワークの状態を表示することにより、管理者がグラフの変化を確認する事なく、状態を確認することが可能になった。

これらのことにより、管理者への常時監視の負担はなくなり、障害発生の早期発見が可能になった。

謝辞

本研究を進めていく上で、御指導及び御協力を頂いた大垣 斉講師・藤井 信夫教授・fken.a4w メイルリストのメンバーの方々、そして情報システム工学科の卒研生の方々に深く感謝の意を表します。

参考文献

[1] MRTG 日本公式サイト, [http://www.mrtg.jp/doc/]

[2] Linux 管理者への道, [http://www.atmarkit.co.jp/flinux/rensai/root08/root08a.html]

[3] Perl 講座, [http://www.iis.u-tokyo.ac.jp/ susaki/perl/index.html]

[4] SAK Streets, [http://homepage2.nifty.com/sak/]

[5] open Perl 講座, [http://www.rfs.jp/sb/perl/05/open.html]
(URL は、2006 年 12 月 26 日現在)

付録 A　プログラムのソース

A.1　ネットワークの状態を Web ブラウザへ表示するプログラム

```
#!/usr/bin/perl

$afile="/home/2006/ohnishi/public_html/cgi-bin/sat.log";
$bfile="/home/2006/ohnishi/public_html/cgi-bin/sun-hol.log";
$cfile="/home/2006/ohnishi/public_html/cgi-bin/day.log";

use POSIX 'strftime';

my $now = strftime "%Y,%m,%d", localtime(time);
my $wday = strftime "%w", localtime(time);
my $hizuke = strftime "%c", localtime(time);
$holiday = "/home/2006/ohnishi/public_html/cgi-bin/holiday.txt";


open(FILE, $holiday) or die "$!";

while($line = <FILE>){
    if ($line =~ m/$now/){
        $hantei = 0
        }
}
if($wday == 0){
        $hantei = 0
        }
if($wday == 6){
        $hantei = 1
        }

sub file{

open (FILE, $inputfile) or die "$!";

@a = <FILE>;

$b = $a[0];

$y = (split(/\s+/,$b))[2];
```

```
close (FILE);

}


if($hantei == 1){

    $files = $afile;
    open (FILE, $files) or "$!";


}elsif($hantei == 0){

    $files = $bfile;
    open (FILE, $files) or "$!";

}else{

    $files = $cfile;
    open (FILE, $files) or "$!";
}

@f = <FILE>;

$g = $f[0];

$n = (split(/\s+/,$g))[2];

close (FILE);

sub hikaku{

if($y < $n){
    $word = "ok";
    $color = "#00ff33";
}else{
    $word = "warning";
    $color = "#ff0000";
}

}

sub t{
```

```
    print <<"EOL";
<tr>
<td align="center">
$name
</td>
<td align="center" bgcolor="$color">
<font size ="3px" color="white">$word</font>
</td>
</tr>

EOL
}

print <<"EOL";
Content-type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html lang="ja">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=EUC-JP">
<meta http-equiv="Refresh" content="300">
<title>status($hizuke)</title>
</head>
<body>
<hr>
<h1 align="center">Network Status</h1>
<hr>
<p align="center">$now</p><br>

<table width="200" border="1" align="center">

<tr>
<td align="center" bgcolor="gray">name</td>
<td align="center" bgcolor="gray">status</td>
</tr>

EOL

$inputfile="/home/2006/ohnishi/tmp/rwho.andrew.log";
&file;
&hikaku;
$name = 'andrew';
&t;
```

```
$inputfile="/home/2006/ohnishi/tmp/rwho.uetsuji.log";
&file;
&hikaku;
$name = 'uetsuji';
&t;

$inputfile="/home/2006/ohnishi/tmp/rwho.ohnishi.log";
&file;
&hikaku;
$name = 'ohnishi';
&t;

$inputfile="/home/2006/ohnishi/tmp/rwho.oguri.log";
&file;
&hikaku;
$name = 'oguri';
&t;

$inputfile="/home/2006/ohnishi/tmp/rwho.kashio.log";
&file;
&hikaku;
$name = 'kashio';
&t;

$inputfile="/home/2006/ohnishi/tmp/rwho.takemiya.log";
&file;
&hikaku;
$name = 'takemiya';
&t;

$inputfile="/home/2006/ohnishi/tmp/rwho.tadaaki.log";
&file;
&hikaku;
$name = 'tadaaki';
&t;

$inputfile="/home/2006/ohnishi/tmp/rwho.nakamae.log";
&file;
&hikaku;
$name = 'nakamae';
&t;
```

```
$inputfile="/home/2006/ohnishi/tmp/rwho.hajime.log";
&file;
&hikaku;
$name = 'hajime';
&t;

$inputfile="/home/2006/ohnishi/tmp/rwho.fujikawa.log";
&file;
&hikaku;
$name = 'fujikawa';
&t;

$inputfile="/home/2006/ohnishi/tmp/rwho.mikane.log";
&file;
&hikaku;
$name = 'mikane';
&t;

$inputfile="/home/2006/ohnishi/tmp/rwho.yada.log";
&file;
&hikaku;
$name = 'yada';
&t;

    print <<"EOL";

</table>

</body>
</html>

EOL
```

A.2　メイル送信の判定プログラム

```
#!/usr/bin/perl

$afile="/home/2006/ohnishi/public_html/cgi-bin/sat.log";
$bfile="/home/2006/ohnishi/public_html/cgi-bin/sun-hol.log";
$cfile="/home/2006/ohnishi/public_html/cgi-bin/day.log";
$erorr = '/home/2006/ohnishi/public_html/cgi-bin/erorr.txt';

use POSIX 'strftime';

my $now = strftime "%Y,%m,%d", localtime(time);
my $wday = strftime "%w", localtime(time);
my $hizuke = strftime "%c", localtime(time);
$holiday = "/home/2006/ohnishi/public_html/cgi-bin/holiday.txt";


open(FILE, $holiday) or die "$!";

while($line = <FILE>){
    if ($line =~ m/$now/){
        $hantei = 0
        }
}
if($wday == 0){
        $hantei = 0
        }
if($wday == 6){
        $hantei = 1
        }

sub file{

open (FILE, $inputfile) or die "$!";

@a = <FILE>;

$b = $a[0];

$y = (split(/\s+/,$b))[2];

close (FILE);
```

```
}


if($hantei == 1){

    $files = $afile;
    open (FILE, $files) or "$!";


}elsif($hantei == 0){

    $files = $bfile;
    open (FILE, $files) or "$!";

}else{

    $files = $cfile;
    open (FILE, $files) or "$!";
}

@f = <FILE>;

$g = $f[0];

$n = (split(/\s+/,$g))[2];

close (FILE);

sub bun{

if($y > $n){
    open(OUT, ">> $erorr");
    print OUT "$name \n";
    close(OUT);
}

}

$inputfile="/home/2006/ohnishi/tmp/rwho.andrew.log";
$name = 'andrew';
&file;
&bun;
```

```
$inputfile="/home/2006/ohnishi/tmp/rwho.uetsuji.log";
$name = 'uetsuji';
&file;
&bun;


$inputfile="/home/2006/ohnishi/tmp/rwho.ohnishi.log";
$name = 'ohnishi';
&file;
&bun;


$inputfile="/home/2006/ohnishi/tmp/rwho.oguri.log";
$name = 'oguri';
&file;
&bun;


$inputfile="/home/2006/ohnishi/tmp/rwho.kashio.log";
$name = 'kashio';
&file;
&bun;


$inputfile="/home/2006/ohnishi/tmp/rwho.takemiya.log";
$name = 'takemiya';
&file;
&bun;


$inputfile="/home/2006/ohnishi/tmp/rwho.tadaaki.log";
$name = 'tadaaki';
&file;
&bun;


$inputfile="/home/2006/ohnishi/tmp/rwho.nakamae.log";
$name = 'nakamae';
&file;
&bun;
```

```
$inputfile="/home/2006/ohnishi/tmp/rwho.hajime.log";
$name = 'hajime';
&file;
&bun;


$inputfile="/home/2006/ohnishi/tmp/rwho.fujikawa.log";
$name = 'fujikawa';
&file;
&bun;


$inputfile="/home/2006/ohnishi/tmp/rwho.mikane.log";
$name = 'mikane';
&file;
&bun;


$inputfile="/home/2006/ohnishi/tmp/rwho.yada.log";
$name = 'yada';
&file;
&bun;

if(-s $erorr){
    require '/home/2006/ohnishi/public_html/cgi-bin/sendmail.pl';
}

unlink $erorr;
```

A.3 メイル送信プログラム

```
#!/usr/bin/perl
  require "/home/2006/ohnishi/public_html/cgi-bin/jcode.pl";

  $file = '/home/2006/ohnishi/public_html/cgi-bin/erorr.txt';
  $sendmail = '/usr/lib/sendmail';
  $from = 'ohnishi@h-ps011.ise.osaka-sandai.ac.jp';
  $to = 'ohnishi@h-ps011.ise.osaka-sandai.ac.jp';
  $subject = 'WARNING';




  &jcode::convert(\$subject, "jis");
  &jcode::convert(\$msg, "jis");

  open(MAIL, "| $sendmail -t -f'$from'") or die 'error';

print MAIL "From: $from\n";
print MAIL "To: $to\n";
print MAIL "Subject: $subject\n\n";
print MAIL "------------\n";
open(FILE, $file);

while(<FILE>){
    print MAIL;
}
close(FILE);
print MAIL "------------\n";
close(FILE);

close(MAIL);
```

A.4 crontab による定期実行

```
0,5,10,15,20,25,30,35,40,45,50,55 * * * *
/home/2006/ohnishi/public_html/cgi-bin/mail.pl
```