84

AIX update

October 2002

In this issue

© Xephon plc 2002

AIX Update

Published by

Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38342
From USA: 01144 1635 38342
E-mail: trevore@xephon.com

North American office

Xephon
PO Box 350100
Westminster, CO 80035-0100
USA
Telephone: 303 410 9344

Subscriptions and back-issues

A year's subscription to *AIX Update*, comprising twelve monthly issues, costs £180.00 in the UK; $275.00 in the USA and Canada; £186.00 in Europe; £192.00 in Australasia and Japan; and £190.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the November 1999 issue, are available separately to subscribers for £16.00 ($24.00) each including postage.

AIX Update on-line

Code from *AIX Update*, and complete issues in Acrobat PDF format, can be downloaded from our Web site at http://www.xephon.com/aix; you will need to supply a word from the printed issue.

Editors

Trevor Eddolls

Disclaimer

Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, scripts, and other contents of this journal before making any use of it.

Contributions

When Xephon is given copyright, articles published in *AIX Update* are paid for at the rate of £170 ($260) per 1000 words and £100 ($160) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of £50 ($80) per 100 lines. In addition, there is a flat fee of £30 ($50) per article. To find out more about contributing an article, without any obligation, please download a copy of our *Notes for Contributors* from www.xephon.com/nfc.

© Xephon plc 2002. All rights reserved. None of the text in this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior permission of the copyright owner. Subscribers are free to copy any code reproduced in this publication for use in their own installations, but may not sell such code or incorporate it in any commercial product. No part of this publication may be used for any form of advertising, sales promotion, or publicity without the written permission of the publisher. Copying permits are available from Xephon in the form of pressure-sensitive labels, for application to individual copies. A pack of 240 labels costs $36 (£24), giving a cost per copy of 15 cents (10 pence). To order, contact Xephon at any of the addresses above.

Printed in England.

awk – part 2

*This month we conclude a detailed look at **awk**, an input-driven Unix utility that can be used mainly for reporting purposes.*

SYSTEM FUNCTION

The **system( )** function executes a command supplied as an expression. It does not, however, make the result of the command available within the program for processing. It returns the status of the command that was executed. The script waits for the command to finish before continuing execution.

For example, create a directory before starting to process any input records:

```
BEGIN   {
         If  (    system ("mkdir  /usr/zamana/temp" )  !=  Ø  )
                  print   "Failed  to  create  directory"
                  exit
          }
```

Writing customized functions within nawk

The syntax for a customized function is as follows:

```
Function   name  (  parameter list )  {
                    Statements
                    return   expression  # optional
}
```

For example:

```
BEGIN   {
                    Display_Banner()
}
#
Function    Display_Banner  ( ) {
            Print     "awk program to validate input datafile"
}
```

Notes:

1 Place all the function definitions just after the *BEGIN* section,

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

although they can be placed anywhere.

2 These functions can be invoked from within the *BEGIN* section as well as from other parts of the program.

3 Function calls must be made within parentheses, even if there are no parameters.

4 When invoking a function, do not put any space between the function name and first parenthesis.

Executing awk programs

awk programs (or scripts) can be written directly on the terminal or placed in a text file.

The script written on the terminal will automatically be executed with the script end marker, which is the matching curly brace (}) .

For example:

```
awk  '{print $1, $2}'  < input_file_name>
```

If the script is placed in a file, it must be executed as follows:

```
awk  -f   <awk_script_file_name>  < input_file_name>
```

The command in the awk_script_file_name might look like:

```
{print $1,$2}
```

Notes:

1 With the **-f** option, **awk** will look for the file specified in *awk_script_file_name*.

2 Without the **-f** option, **awk** will look for the file *<awk_script_file_name>.awk*.

3 In both examples, awk is reading input from the file *<input_file_name>*.

4 Notice the syntax difference when **awk** commands are placed in the file:

– The **awk** command itself is not required

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

– Open and close quotes are not required.

PASSING PARAMETERS TO A SCRIPT

If no parameters are passed to the script, the script is executed as follows:

```
awk  -f   scriptfile   input_data_file
```

Suppose we want to pass two parameters called FROM_POS and TO_POS with values 100 and 200.

The command line will look as follows:

```
awk  -f  scriptfile  FROM_POS=100  TO_POS=200  input_data_file
```

Notes:

1 The parameters are passed as a pair of variables and their values. No spaces are allowed before and after the equals sign.

2 Inside the script, the variables FROM_POS and TO_POS are available and can be accessed like any **awk** variables.

3 An important restriction on command-line parameters is that they are not available in the *BEGIN* procedure because **awk** does not do anything until it reads a record, which happens after the *BEGIN* procedure is completed.

4 These parameters can be defined only on the command line after the scriptfile and before the input file name.

Shell wrapper for invoking awk

A shell script can be used to invoke the **awk** command with the required parameters.

The command line from the above example can be put into a shell script, awk_sh, as follows:

```
awk  -f  scriptfile  FROM_POS=$1   TO_POS=$2   input_data_file
```

and then invoke the shell script as follows:

```
awk_sh  100   200
```

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

PASSING SHELL VARIABLES INTO AN AWK PROGRAM

Shell variables can be passed into an **awk** program by:

```
#
#  define shell variables within the shell script
FNAME="ARIF"
LNAME="ZAMAN"
#  print these names from within awk program
#
#   define shell function to invoke nawk
PrintNames  ( )
{
   PARAM1=$1
   PARAM2=$2
 nawk  'BEGIN    {
                              FNAME=""
                              LNAME=""
                             }
           END  {
                        print   FNAME
                        print   LNAME
                    }'  FNAME=${PARAM1}  LNAME=${PARAM2}  <input_file>
}
```

Note: you must write a shell function to receive these parameters and assign these parameters to **awk** variables in the command line.

SAMPLE AWK PROGRAMS

File validation

Input file:

```
10 05 2002
20 CUSTOMER  3
30 111  I     Arif  Zaman
30 222  I     Ian  Davies
30 333 U     David  Varley
30  END
40 ADDRESS 3
50 111  I     69  The  Avenue,   Pinner,   Middlesex,  HA5  5BW
50 222  I     10  Manor Park,  Romford,   GT6 7TY
50 333  U    20  Park  Way,   Manor  Royal,    GH6  6TY
50 END
60  CUSTOMER_PREFERENCE  2
70  111  I     Y  N  Y  N
70  222  I      N  N  N  N
```


© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

```
70 END
```

Rules to validate:

1 File header must have three fields (Field1=Record Type (10), Field2= Processing_Month, and Field3= Processing_Year).

2 Transaction header type must be one of the following:

- Record Type = 20 CUSTOMER
- Record Type = 40 ADDRESS
- Record Type = 60 CUSTOMER_PREFERENCE.

3 Transaction record type must be one of the following:

- Record Type = 30 CUSTOMER
- Record Type = 50 ADDRESS
- Record Type = 70 CUSTOMER_PREFERENCE.

4 The record count in the transaction header record must match the actual number of records.

5 The record type must be either I (for insert) or U (for update).

6 For transaction type CUSTOMER_PREFERENCE, the flags in the third, fourth, fifth, and sixth fields must be Y or N.

Program

```
################################################################################
# Name     : validate_file.awk
# Overview : The awk script validate input file
# Usage    : nawk -f validate_file.awk \
#    PROCESSING_MONTH=<current monthg> PROCESSING_YEAR=<current year> \
#                      VALIDATION_MODE=<I|B> <input file name>
#
#  Notes       :1. The arguments to awk script are passed as  follows:
#               nawk -f <script file> param_name=param_value \
#                                     param_name=param_value ......\
#                                     < input file name >
#                2. The command line parameters are not available within
BEGIN
#                   section.
################################################################################
BEGIN {    FS=" "      # field separator
```

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

```
           #
           #
           no_header_record_fields=3
           no_transaction_header_record_fields=3
           FEC=1
           SEC=Ø
           record_count=Ø
           error_msg=""

       }
#  parse command line parameters
#  because these are not available with BEGIN section,
#  we've to parse them within this procedure
#  If an input file name is not provided, this section will not be
#  executed but awk will be waiting for input
NR == 1 {
   if ( ARGC !=  5  )
   {
   print "USAGE:nawk -f validate_file.awk PROCESSING_MONTH=<current
month>\\ \n          PROCESSING_YEAR=<current year
VALIDATION_MODE=<I|B>  <input file>"
    exit FEC
   }
   #
   if ( PROCESSING_MONTH == ""  )
   {
    print "ERROR:Must provide value for MONTH parameter"
    exit FEC
   }
   #
   if ( PROCESSING_YEAR == ""  )
   {
    print "ERROR:Must provide value for YEAR parameter"
    exit FEC
   }
   #
   if ( VALIDATION_MODE != "I" && VALIDATION_MODE != "B"   )
   {
    print "ERROR:Validation mode must be I or B"
    exit FEC
   }
}
# process file Header record
# compare first field in each record
$1 ==1Ø {
         # header record must have 3 fields
         if ( NF != no_header_record_fields )
            {
              if ( VALIDATION_MODE == "I" )
                {
```


© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

```
               print "ERROR:Header record ("$Ø") contains "NF " fields"
                  exit FEC
                }
              else
                {
                        error_msg=error_msg "Record No="NR " Record
("$Ø")"
                 error_msg=error_msg " contains "NF " fields\n"
                }
            }
           # file must be for current month
           if  ( $2 != PROCESSING_MONTH )
             {
              if ( VALIDATION_MODE == "I" )
               {
                      print "ERROR:Record No=" NR " Month in file
header "
                print "does not match  processing month"
                exit FEC
                }
              else
               {
                error_msg=error_msg  "Record No="NR " Month in file
header "
                error_msg=error_msg  "does not match processing
month\n"
                }
             }
            # file must be for current year
            if  ( $3 != PROCESSING_YEAR )
              {
               if ( VALIDATION_MODE == "I" )
                {
                 print "ERROR:Record No="NR " Year in file header "
                 print "does not match processing year"
                 exit FEC
                }
               else
                {
                   error_msg=error_msg  "Record No="NR " Year in file
header "
                error_msg=error_msg  "does not match processing year\n"
                }
               }
         }
# process transaction header record
# CUSTOMER
$1 ==2Ø {
         # transaction header record must have 3 fields
         if ( NF != no_transaction_header_record_fields )
```

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

```
            {
             if ( VALIDATION_MODE == "I" )
              {
    print "ERROR:Transaction header record ("$Ø") contains "NF" fields"
             exit FEC
              }
             else
              {
     error_msg=error_msg  "Record No="NR " Transaction header record ("
             error_msg=error_msg  $Ø") contains " NF" fields\n"
              }
            }
          # transaction type must be customer
          if ( $2 != "CUSTOMER" )
           {
            if ( VALIDATION_MODE == "I" )
              {
            print "ERROR:Record No="NR"
                        Transaction type ("$1") does not match CUSTOMER"
              exit FEC
              }
            else
              {
              error_msg=error_msg "Record No="NR " Transaction type ("$1
              error_msg=error_msg ") does not match CUSTOMER\n"
               }
           }
           # store no of transaction records
           no_of_records=$3
         }
# process CUSTOMER transaction records
$1 ==3Ø {
           #
           if ( $2 == "END"  )
            {
             # end of customer records
              if (  no_of_records != record_count )
                 {
                   if ( VALIDATION_MODE == "I" )
                    {
                   print "ERROR:Record mismatched for CUSTOMER records"
                     print "Record count in header ("no_of_records")"
              print "does not match records processed ("record_count ")"
                     exit FEC
                    }
                   else
                    {
          error_msg=error_msg "Record mismatched for CUSTOMER records\n"
          error_msg=error_msg "Record count in header ("no_of_records")"
          error_msg=error_msg
```


© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

```
                " does not match records processed ("record_count ")\n"
                   }
                 }
               else
                 {
                  # initialize counters
                   no_of_records=Ø
                   record_count=Ø
                  }
             }
           else
            {
              # increase the counter
              record_count++
              # record type must be U or I
              if ( $3 != "I" && $3 != "U"  )
                {
                 if ( VALIDATION_MODE == "I" )
                  {
                    print "ERROR:Wrong record type ("$3") for CUSTOMER"
                    exit FEC
                  }
                  else
                  {
                  error_msg=error_msg "Record No="NR
          error_msg=error_msg " Wrong record type ("$3") for CUSTOMER\n"
                  }
                }
             }
          #
         }
# process transaction header record
# ADDRESS
$1 ==4Ø {
          # transaction header record must have 3 fields
          if ( NF != no_transaction_header_record_fields )
             {
              if ( VALIDATION_MODE == "I" )
               {
          print "ERROR:Record No="NR " Transaction header record ("$Ø")"
                 print " contains "NF" fields"
                 exit FEC
               }
              else
               {
         error_msg=error_msg "Record No="NR " Transaction header record"
                 error_msg=error_msg " contains "NF" fields\n"
               }
              }
          # transaction type must be ADDRESS
```

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

```
          if ( $2 != "ADDRESS" )
           {
             if ( VALIDATION_MODE == "I" )
              {
          print "ERROR:Record No="NR " Transaction type ("$1") does not"
                print " match ADDRESS"
                exit FEC
              }
              else
               {
              error_msg=error_msg "Record No="NR " Transaction type ("$1
                 error_msg=error_msg ") does not match ADDRESS\n"
               }
           }
           # store no of transaction records
           no_of_records=$3
         }
# process ADDRESS transaction records
$1 ==5Ø {
           #
           if ( $2 == "END"  )
            {
             # end of address records
              if (  no_of_records != record_count )
                 {
                  if ( VALIDATION_MODE == "I" )
                   {
                     print "ERROR:Record mismatched for ADDRESS records"
                     exit FEC
                   }
                   else
                   {
          error_msg=error_msg "Record mismatched for ADDRESS records\n"
                   }
                 }
               else
                 {
                  # initialize counters
                   no_of_records=Ø
                   record_count=Ø
                  }
             }
           else
            {
              # increase the counter
              record_count++
              # record type must be U or I
              if ( $3 != "I" && $3 != "U"  )
                {
                  if ( VALIDATION_MODE == "I" )
```


© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

```
                {
            print "ERROR:Record No="NR " Wrong record type for ADDRESS"
                  exit FEC
                }
                else
                {
               error_msg=error_msg "Record No=" NR
               error_msg=error_msg " Wrong record type for ADDRESS\n"
                }
             }
          }
          #
         }
# process transaction header record
# CUSTOMER_PREFERENCES
$1 ==6Ø {
          # transaction header record must have 3 fields
          if ( NF != no_transaction_header_record_fields )
             {
              if ( VALIDATION_MODE == "I" )
               {
                print "ERROR:Record No="NR " Transaction header record "
                 print "contains " NF " fields"
                 exit FEC
               }
              else
               {
       error_msg=error_msg "Record No="NR " Transaction header record "
                error_msg=error_msg "contains " NF " fields\n"
               }
             }
           # transaction type must be CUSTOMER_PREFERENCES
           if ( $2 != "CUSTOMER_PREFERENCES" )
            {
              if ( VALIDATION_MODE == "I" )
               {
           print "ERROR:Record No=" NR "Transaction type does not match "
                 print "CUSTOMER_PREFERENCES"
                 exit FEC
               }
              else
               {
             error_msg=error_msg "Record No="NR " Transaction type does "
                 error_msg=error_msg "not match CUSTOMER_PREFERENCES\n"
               }
            }
           # store no of transaction records
           no_of_records=$3
         }
# process CUSTOMER_PREFERENCES transaction records
```

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

```
$1 ==7Ø {
          #
          if ( $2 == "END"  )
           {
            # end of address records
             if (  no_of_records != record_count )
                {
                 if ( VALIDATION_MODE == "I" )
                  {
                  print "ERROR:Record mismatched for
CUSTOMER_PREFERENCES"
                   exit FEC
                  }
                  else
                  {
                   error_msg=error_msg "Record mismatched for "
                   error_msg=error_msg "CUSTOMER_PREFERENCES\n"
                  }
                }
               else
                 {
                  # initialize counters
                   no_of_records=Ø
                   record_count=Ø
                  }
             }
          else
           {
             # increase the counter
             record_count++
             # record type must be U or I
             if ( $3 != "I" && $3 != "U"  )
               {
                if ( VALIDATION_MODE == "I" )
                 {
                   print "ERROR:Record No=" NR " Wrong record type for"
                   print " CUSTOMER_PREFERENCES"
                   exit FEC
                 }
                else
                 {
          error_msg=error_msg "Record No=" NR " Wrong record type for "
                  error_msg=error_msg "CUSTOMER_PREFERENCES\n"
                 }
               }
              # validate preference  flags
             if ( $4 != "Y" &&  $4 != "N"  )
               {
                if ( VALIDATION_MODE == "I" )
                 {
```


© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

```
      print "ERROR:Record No=" NR " Wrong value for "
      print "CUSTOMER_PREFERENCES flag "
      exit FEC
     }
    else
     {
    error_msg=error_msg "Record No=" NR " Wrong value for "
      error_msg=error_msg "CUSTOMER_PREFERENCES flag\n"
     }
  }
else if ( $5 != "Y" && $5 != "N"  )
  {
   if ( VALIDATION_MODE == "I" )
    {
     print "ERROR:Record No=" NR " Wrong value for "
     print "CUSTOMER_PREFERENCES flag "
     exit FEC
    }
   else
    {
   error_msg=error_msg "Record No=" NR " Wrong value for "
     error_msg=error_msg "CUSTOMER_PREFERENCES flag\n"
    }
  }
 #
else if ( $6 != "Y" && $6 != "N"  )
  {
   if ( VALIDATION_MODE == "I" )
    {
     print "ERROR:Record No=" NR " Wrong value for "
     print "CUSTOMER_PREFERENCES flag "
     exit FEC
    }
   else
    {
   error_msg=error_msg "Record No=" NR " Wrong value for "
     error_msg=error_msg "CUSTOMER_PREFERENCES flag\n"
    }
  }
 #
else if ( $7 != "Y" && $7 != "N"  )
  {
   if ( VALIDATION_MODE == "I" )
    {
     print "ERROR:Record No=" NR " Wrong value for "
     print "CUSTOMER_PREFERENCES flag "
     exit FEC
    }
   else
    {
```

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

```
                error_msg=error_msg "Record No=" NR " Wrong value for "
                  error_msg=error_msg "CUSTOMER_PREFERENCES flag\n"
                 }
               }
              #
            }
         #
        }
#
# display validation results
END {
# exit command from the body will not exit to the command line
       #
      if ( VALIDATION_MODE !=  "B" )
          exit FEC
      else
        {
          if ( error_msg == "" )
            print "No validation errors found"
          else
           {
             print "Validation Errors in " FILENAME
             print "————————————————"
             print error_msg
             exit SEC
           }
        }
 }
```

Arif Zaman
ETL Developer (UK)

© Xephon 2002

The if command

FORMAT OF THE IF COMMAND

The **if** command is a built-in shell construct that provides a mechanism for conditional execution of a command or list of commands. The decision is based on the exit status of the command that follows the keyword **if**.

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

With a **case** command you can make a decision based on the exit status of a command using the variable **$?**, whereas with the **if** command, the decision always depends on the exit status of the command itself, and you do not need to use **$?**.

There are several forms that the **if** command can take. The simplest is:

```
if condition_command
then
    then_body
fi
```

The word **if** introduces the command, and the keyword **fi** (which is **if** spelt backwards) indicates the end of the command. Both the *condition_command* and *then_body* may be single commands or pipelines, or lists of commands and/or pipelines.

For the moment, assume that the *condition_command* is a single command. To execute the **if** command the shell first runs *condition_command*. If this is successful and returns 0, the shell runs the commands in *then_body*. If *condition_command* returns a non-zero value, the commands in *then_body* are skipped and the shell either terminates or continues with the commands that follow the keyword **fi**.

Consider the following example:

```
$ vi mycmp

if cmp -s "$1" "$2"
then
     print They are the same
     exit
fi
print They are different
```

In the above example, the *condition_command* is **cmp -s $1 $2**. The **cmp** command compares two files and returns 0 if they are the same, and non-zero if the contents differ. The **-s** (for silent) option causes **cmp** not to print anything. The commands in the *then_body* will be executed only if the files specified by **$1** and **$2** are identical.

First copy one of the files in your current directory, and then run **mycmp** with:

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

```
$ cp file1 file2
$ mycmp file1 file2
```

They are the same.

Then try it with:

```
$ mycmp file1 diff_file
```

They are different.

This is not a particularly good example since if you run **mycmp** with, for example, **mycmp a a** (where **a** is a non-existent file name), then you will get the message 'They are different' since **cmp** will return a non-zero exit status because of the error condition it has experienced.

You will also note that we have enclosed the **$1** and **$2** arguments to **cmp** in quotes. By doing this we avoid getting an error message displayed by **cmp** whenever we run **mycmp** with just a single argument instead of the two it would normally expect. So the script is not perfect, but it is just a simple example of the use of the **if** command, not of good programming!

You do not have to format **if** commands in exactly the same way as the above example. You could instead place the commands in the *then_body* on the same line as the keyword **then**:

```
then print They are the same ; exit
```

It is customary to indent the commands in the *then_body* since it makes it easier to find the beginning and end of the **if** command. It is also possible to put the whole of the **if** command on one line, if it will fit. However, the keywords **if**, **then**, and **fi** will only be recognized by the shell if they are preceded by a command terminator, such as a semicolon. For example:

```
if cmp -s $1 $2; then print same; exit; fi
```

This style is not recommended since it is often difficult to determine where errors lie in shell scripts ordered in this fashion.

As mentioned earlier, the *condition_command* may actually be a pipeline, or a list of commands and pipelines. The exit status is then the exit status of the last command in the pipeline.

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

Consider the following pipeline:

```
$ who | grep wrong_name | wc -l
Ø
$ print $?
Ø
```

The exit status of this pipeline is 0 despite the fact that the **grep** command returns a non-zero value. Since **grep** does not find the word *wrong_name* in the **who** output, **grep** does not print anything and terminates with a non-zero exit status. This means that **wc** does not receive any input. However, **wc** does not consider this an error. It will display the number 0 to indicate that there were zero lines of input, and terminate with an exit status of 0.

When *condition_command* consists of a sequence of commands (or pipelines) that are separated by semicolons or that appear one per line, the exit status of the last command or pipeline determines whether or not *then_body* will be run.

You should give careful thought to using sequences of commands or pipelines in the *condition_command* part of an **if** construct, since you may not achieve your intended results.

THE ELSE KEYWORD

There is an additional keyword that you can use with the **if** command. The **else** keyword introduces an alternative sequence of commands which will be run instead of the *then_body* if the *condition_command* is not successful.

The general form of an **if** command with the **else** keyword is:

```
if condition_command
then
     then_body
else
     else_body
fi
```

Using this structure, **mycmp** can be rewritten as:

```
if cmp -s "$1" "$2"
then
      print They are the same
```

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

```
else
      print They are different
fi
```

THE ELIF KEYWORD

The indenting of **if** structures makes a program much more readable. If the first command in the *body* of the **if** structure is also an **if** command, then the commands in its *body* will also be indented, and so on. If there are several levels of nesting, however, the commands may be indented all the way to the right hand side of the screen, and your script often becomes hard to read when lines of code are continued onto the next physical line of the screen.

The alternative to nesting the **if** commands is to use the **elif** keyword, which will make your script easier to read and the logic easier to follow; **elif** is short for **else if**, although you cannot use this as an alternative since you will get a syntax error.

The general form of the command is:

```
if condition_command_1
then
      body_1
elif condition_command_2
then
      body_2
.
.
elif condition_command_n
then
      body_n
else
      else_body
fi
```

The shell will execute this kind of **if** command by running each of the *condition_commands* in order, until one is successful. The shell then runs the commands in the associated *body*, and this completes the execution of the entire structure.

If none of the *condition_commands* are successful, the shell runs the commands in the *else_body*. The execution of this type of command may involve running one, two, or all of the *condition_commands*, but

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

one, and only one, of the bodies will be run.

MODIFICATIONS TO LVMAN

Now that we are familiar with the **if** construction, let us make some modifications to the **lvman** script – see pages 14-16, *AIX Update* Issue 83, September 2002.

Changes to sections in the previous version are shown in *italics*.

```
#!/bin/ksh
# Script name: lvman
# Usage: lvman {[-v VGname]|[-p PVname]}
#################################################################
# Version History
#################################################################
#-----------------------------------------------
# Function: f_dsp_usage
# Displays usage messages
#-----------------------------------------------
f_dsp_usage()
{
    print "Usage: $(basename $0) {[-v VGname] | [-p PVname]}"
    print "Where:"
    print "\t-v VGname specifies a single volume group"
    print "\t-p PVname specifies a single physical volume"
    print "Note: Use either the -v OR -p option"
}
#-----------------------------------------------
# Function: f_chk_valid
# Arguments: $1 - volume group or physical volume
# Checks the volume group or physical volume name is valid
#-----------------------------------------------
f_chk_valid()
{
    DEV=$1
    lsattr -El $DEV >/dev/null 2>&1
    # lsattr returns 0 for valid device,
    # or 255 for non valid device
    case $? in
    0)
      return 0 ;;
    *)
      return 1 ;;
    esac
}
#-----------------------------------------------
# Function: f_get_vg_space
```

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

```
# Arguments: $1 - volume group name
# Gets the total and free space of the volume group
#-----------------------------------------------
f_get_vg_space()
{
    VG=$1
    # Get total space and free space
    TOTAL=$(lsvg $VG | grep "TOTAL PPs" | cut -f2 -d "(" |
      tr ' ' '\t' | cut -f1)
    FREE=$(lsvg $VG | grep "FREE PPs" | cut -f2 -d "(" |
      tr ' ' '\t' | cut -f1)
    eval ${VG}_LVNUM=$(lsvg -l $VG | tail +3 | wc -l | tr -d " ")
    # Print output
    printf "%-20s %-15s %-15s\n" "Volume Group" "Total Size" \
        "Free Space"
    printf "%-20s %-15s %-15s \n" $VG "$TOTAL MB" \
        "$FREE MB"
    eval print Number of LVs in $VG = '$'${VG}_LVNUM
}
#-----------------------------------------------
# Function: f_get_pv_space
# Arguments: $1 - physical volume name
# Gets the total and free space on a physical volume
#-----------------------------------------------
f_get_pv_space()
{
    PV=$1
    #
 # Get total space and free space
    #
    TOTAL=$(lspv $PV | grep "TOTAL PPs" | cut -f2 -d "(" |
      tr ' ' '\t' | cut -f1)
    FREE=$(lspv $PV | grep "FREE PPs" | cut -f2 -d "(" |
      tr ' ' '\t' | cut -f1)
    eval ${PV}_LVNUM=$(lspv -l $PV | tail +3 | wc -l | tr -d " ")
    # Print output
    printf "%-20s %-15s %-15s\n" "Physical Volume" "Total Size" \
"Free Space"
    printf "%-20s %-15s %-15s \n" $PV "$TOTAL MB" \
"$FREE MB"
    eval print Number of LVs on $PV = '$'${PV}_LVNUM
}
#####################################################
# Main section
#####################################################
getopts :v:p: opt
case $opt in
v)
    VG=$OPTARG
    if f_chk_valid $VG
```


© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

```
    then
      f_get_vg_space $VG
    else
      print $VG is not a valid volume group
      exit 1
    fi
    ;;
p)
    PV=$OPTARG
    if f_chk_valid $PV
    then
      f_get_pv_space $PV
    else
      print $PV is not a valid physical volume
      exit 2
    fi
    ;;
*)
    f_dsp_usage
    exit 3
    ;;
esac
```

Although the **case** statements within the main section of the script are perfectly adequate to achieve our objectives, you will find that, for simple tests, programmers tend to use the **if** command in preference to **case**, which is more useful when you expect a large number of values for a variable or output from a particular command, or where you want to use the powerful pattern matching features of **case** and so eliminate the need for multiple **if** statements.

If we had been familiar with **if** when we started to write our program, we would most probably have used it, rather than **case**, in certain parts of the script – we have modified the main section to reflect this.

The changes we have made are relatively simple. For example, for the **-v** option we now use the **if** statement to check whether the return value from **f_chk_valid $VG** is *true* (0) or otherwise. If it is *true* then we know the volume group exists and so we display the space utilization output. Our **f_chk_valid** function can now return any value other than 0 when the volume group does not exist.

Tonto Kowalski
Guru (UAE)
© Xephon 2002

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

Problems with name resolution

START CONFIGURATION

Our normal AIX system network configuration consists of an empty (not set) NSORDER environment parameter. The resolver settings put in */etc/netsvc.conf* are:

```
cat /etc/netsvc.conf
hosts=local,bind
```

To find the right nameserver and append its domain we set up *resolv.conf* with:

```
cat /etc/resolv.conf
domain test
nameserver 172.26.133.93
```

In this configuration, a request for host aix19 will be a local search in the */etc/hosts* for aix19.test. If this is not successful, the resolver tries to get an answer from the nameserver at 172.26.133.93.

THE PROBLEM

Each time we were disconnected from the Internet, we found that the telnet, ftp, and other IP applications had a very, very, long timeout until the loginprompt popped up, when they connected to an internal host. However, ping and nslookup would resolve the hostname immediately.

We observed this on all our AIX 4.3.*x* and 5.*x* systems, but not on the AIX 4.2.*x* boxes.

At this point we could not understand why a hostname look-up served by an internal dns server took so long.

To look deeper into the problem we put an additional option in the *resolv.conf* file, options debug, and tried with **telnet aix19** to connect once more.

Below is sample output from this session. The interesting points are

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

the first request of the AAAA record for *aix19.test* and the second with *aix19* without domain. After the eight query-timeouts, the resolver requests the *A* record of *aix19.test* and gets the answer.

Debug output session 1:

```
telnet aix19
;; res_setoptions(" debug
", "conf")..
;;      debug
;; res_nquerydomain(aix19, test, 1, 28)
;; res_query(aix19.test, 1, 28)
;; res_nmkquery(QUERY, aix19.test, IN, AAAA)
;; res_send()
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36132
;; flags: rd; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0
;;      aix19.test, type = AAAA, class = IN
;; Querying server (# 1) address = 172.26.133.93
;; timeout
;; Querying server (# 1) address = 172.26.133.93
;; timeout
;; Querying server (# 1) address = 172.26.133.93
;; timeout
;; Querying server (# 1) address = 172.26.133.93
;; timeout
;; res_query: send error
;; res_nquerydomain(aix19, <Nil>, 1, 28)
;; res_query(aix19, 1, 28)
;; res_nmkquery(QUERY, aix19, IN, AAAA)
;; res_send()
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36133
;; flags: rd; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0
;;      aix19, type = AAAA, class = IN
;; Querying server (# 1) address = 172.26.133.93
;; timeout
;; Querying server (# 1) address = 172.26.133.93
;; timeout
;; Querying server (# 1) address = 172.26.133.93
;; timeout
;; Querying server (# 1) address = 172.26.133.93
;; timeout
;; res_query: send error
;; res_nquerydomain(aix19, test, 1, 1)
;; res_query(aix19.test, 1, 1)
;; res_nmkquery(QUERY, aix19.test, IN, A)
;; res_send()
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36134
;; flags: rd; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0
;;      aix19.test, type = A, class = IN
```

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

```
;; Querying server (# 1) address = 172.26.133.93
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36134
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1
;;      aix19.test, type = A, class = IN
aix19.test.             16w3d17h46m39s IN A  172.26.133.47
test.                   16w3d17h46m39s IN NS  dnssrv.test.
dnssrv.test.            16w3d17h46m39s IN A  172.26.133.93
Trying...
Connected to aix19.test.
```

After checking that the resolver first asks for an IPV6 address, we searched on the Internet and found a useful option to fix this. We put the string ‘hosts=local,bind4’ in the */etc/netsvc.conf* file and tried again – and immediately got a good result.

Debug output session 2:

```
telnet aix19
;; res_setoptions(" debug
", "conf")..
;;      debug
;; res_nquerydomain(aix19, test, 1, 1)
;; res_query(aix19.test, 1, 1)
;; res_nmkquery(QUERY, aix19.test, IN, A)
;; res_send()
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53451
;; flags: rd; QUERY: 1, ANSWER: Ø, AUTHORITY: Ø, ADDITIONAL: Ø
;;      aix19.test, type = A, class = IN
;; Querying server (# 1) address = 172.26.133.93
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53451
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1
;;      aix19.test, type = A, class = IN
aix19.test.             16w3d17h46m39s IN A  172.26.133.47
test.                   16w3d17h46m39s IN NS  dnssrv.test.
dnssrv.test.            16w3d17h46m39s IN A  172.26.133.93
Trying...
Connected to aix19.test.
```

CONCLUSION

Even a set-up that works well in normal situations can fail in unpredictable circumstances. Look at the debug and trace output after operating system upgrades or maintenance level implementations if the normal system conditions are still in place.

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

A short summary of the resolver-specific issues that I found on my system with AIX 4.3.3 ml=10 installed are described below:

- The IX76289 address is the problem discussed in this article. You can get the original output of the fixes/bugs abstract and symptom text with:

```
instfix  -va -i -k ARPA#
```

- IX85934 – traceroute ignores */etc/netsvc*, NSORDER should be fixed with bos.net.tcp.client:4.3.2.3.
- IX76289 – hostname look-ups in AIX 4.3.0 take a long time because of a query of record AAAA timing out. The fix creates the ability to do IPV4, IPV6, or both look-ups. The filesets needed are bos.adt.prof:4.3.1.1, bos.rte.control:4.3.1.1, bos.rte.libc:4.3.1.1, and bos.rte.shell:4.3.1.1.
- IX88329 – problems with netstat and resolver options should be fixed with bos.adt.prof:4.3.2.5 and bos.rte.libc:4.3.2.5.
- IY01106 – problems with reverse look-ups and resolver options should be fixed with bos.adt.prof:4.3.2.10 and bos.rte.libc:4.3.2.10.
- IY03450 bind4,local4 reverse look-ups from rexec,rsh – problems should be fixed with bos.adt.prof:4.3.3.10 and bos.rte.libc:4.3.3.10.
- IX74707 */etc/netsvc.conf* parsing – the problem with spaces should be fixed with bos.adt.prof:4.3.0.6, bos.rte.control:4.3.0.2, bos.rte.libc:4.3.0.6, and bos.rte.shell:4.3.0.2 .
- IX84501 */etc/hosts* table, IPV4 addresses, telnet, ftp – should be fixed with bos.adt.prof:4.3.2.3 and bos.rte.libc:4.3.2.3.

Imhotep
Unix System Administrator (Austria)
© Xephon 2002

Code from individual articles of *AIX Update* and complete issues in PDF format can be accessed on our Web site at:

http://www.xephon.com/aix

You will be asked to enter a word from the printed issue.

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

Error mail notification

Are you responsible for serveral AIX machines? Is checking all the error logs on all machines a very time-consuming procedure every day? Do you want to be notified automatically by your favourite e-mail system outside AIX (eg Lotus Notes) when hardware or software errors are written to the error log?

There is a way to reduce the amount of time and resources required to monitor the error log as well as reduce the risk of missing important and vital messages. To achieve this you can use the ODM-class **errnotify** (Object Data Manager). The error notification object class is located in the */etc/objrepos/errnotify* file.

Each time an error occurs, the error notification daemon checks the error notification class, **errnotify**, for any that match certain criteria. If the daemon finds a match, the notify method (en_method) is run. You can specify a notify method that is executed when an error log entry is written. This can be a shell script or any kind of command string; for instance you can write a shell script to send an e-mail. Details of this will be covered later. First some details about the error notification object class, **errnotify**.

An example of the error notification object class, **errnotify**, is shown below:

```
errnotify:
    en_pid = 0
    en_name = "DISK_ERR4"
    en_persistenceflg = 1
    en_label = "DISK_ERR4"
    en_crcid = 0
    en_class = ""
    en_type = ""
    en_alertflg = ""
    en_resource = ""
    en_rtype = ""
    en_rclass = ""
    en_symptom = ""
en_method = "echo DISK problem on host $(hostname) >
/tmp/err_notify.txt.$$;/user/err_notify.sh /tmp/err_notify.txt.$$; rm /
tmp/err_notify.txt.$$"
```


© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

Below is a list of the descriptors, what they do, and any possible values:

- en_pid – specifies a process ID for use in identifying the error notification object – *Process-ID (numeric)*.
- en_name – unique identifier of the object – *Name (text string)*.
- en_persistenceflg – defines whether the error notification object is removed when the system restarts. If descriptor is 0 the object is removed when the system restarts, if descriptor is 1 the object persists through boot – *0* (non-persistent), *1* (persistent).
- en_label – specifies the label as defined in the **errpt -t** output – *Label (text string)*.
- en_crcid – specifies the error identifier associated with a particular error. Possible values can be viewed using the **errpt -t** command – *Identifier (text string)*.
- en_class – specifies the class of an error. If not specified or defined as a null string, all classes of error will be matched – *H* (hardware errors), *S* (software errors), *O* (operator messages generated by the errlogger command), *U* (undetermined).
- en_type – severity of error log entries – *INFO:* informational, *PEND:* impending loss of availability, *PERM:* permanent error, *PERF:* performance degradation, *TEMP:* temporary error, *UNKN:* unknown error, *TRUE:* matches alertable error.
- en_alertflg – specifies whether an error is alertable. Used by alert agents in network management applications – *TRUE:* alertable, *FALSE:* not alertable.
- en_resource – specifies the name of the failing resource. Only used for hardware errors. The resource name is the name of the device – *Resource (text string)*.
- en_rtype – specifies the type of the failing resource. Only used for hardware errors. The resource type is the device type as specified in the PdDv – *Device type (text string)*.
- en_rclass – specifies the class of the failing resource. Only used for hardware errors. The resource type is the device type as specified

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

in the PdDv – *Device class (text string).*

- en_symptom – a symptom string will be generated when value is set to TRUE – *TRUE:* symptom string will be generated, *FALSE:* no symptom string will be generated.
- en_method – specifies the action. The action is user-programmable. This can be a shell script or a command string that is executed when an error matching the selection criteria of this error notification object is detected. The daemon uses the **sh -c** command to execute the notify method. A number of parameters can be passed to the specific program (see below) – *Text string*. Note that the maximum value of en_method may not exceed 256 characters.

A list of parameters and their descriptions is shown below:

- $1 – error log entry sequence number.
- $2 – error log entry error ID.
- $3 – error log entry class.
- $4 – error log entry type.
- $5 – error log entry alert flags.
- $6 – error log entry resource name.
- $7 – error log entry resource type.
- $8 – error log entry resource class.
- $9 – error log entry error label.

All possible values for en_label, en_type, en_class, and en_crcid, together with a short description, can be found using the **errpt -t** command.

Note that if a descriptor used to match part of an error log entry is not included in the object class, or if its value is a null string, that descriptor will match all possible values.

HOW TO ADD AN ERROR NOTIFICATION OBJECT

To add an error notification object that sends you an e-mail when a

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

filesystem becomes full, just create a file */tmp/odmadd_errnotify.txt* with the following content:

```
errnotify:
    en_pid = Ø
    en_name = "JFS_FS_FULL"
    en_persistenceflg = 1
    en_label = "JFS_FS_FULL"
    en_crcid = Ø
    en_class = ""
    en_type = ""
    en_alertflg = ""
    en_resource = ""
    en_rtype = ""
    en_rclass = ""
    en_symptom = ""
en_method = "echo JFS FULL on host $(hostname) >
/tmp/err_notify.txt.$$;/user/err_notify.sh /tmp/err_notify.txt.$$;rm /
tmp/err_notify.txt.$$"
```

In the above, all the commands specified in the en_method descriptor will be run each time any Journalled File System (JFS) is running out of space.

After saving the file run the **odmadd /tmp/odmadd_errnotify.txt** command. This adds the object to the ODM class, **errnotify**. To verify that the object was installed correctly, run the command **odmget -q ,en_name= JFS_FS_FULL , errnotify**, and the contents of the object will be displayed. To remove this object, execute the command **odmdelete -q ,en_name= JFS_FS_FULL , -o errnotify**, and the object will be deleted from the ODM.

Script:

```
#!/usr/bin/sh
# define variables
ERR_REC=/user/scripts/errnotify_recipients
                                          # file holding all recipients
if [ -a $ERR_REC ]
then
      for j in ,cat $ERR_REC |  grep "^[a-zA-Z]",
do
            mail -s"'errpt-message from $(hostname)'" $j < $1
              done
fi
```

File holding all recipients:

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

```
# mail recipients for error notification
# to exclude a person, just comment out the line using #
#
# format: name@domain
#
#
aix.admin1@domain
#aix.admin1@domain
```

The script is rather simple. The en_method creates a temporary file with some short error information and the hostname where the error occurred. It reads a file, which holds all recipients, so more than one person can be notified. Only lines with an alphanumeric character in the first position are processed. In order to distinguish errors from different machines, the hostname is passed in the subject-field. So after receiving an e-mail you can check the error log using **errpt -a** for further investigatation of the error.

OTHER NECESSARY TASKS

In order to have AIX send the error notifications to your favourite e-mail system you must configure the local AIX mailing system. You need to adjust */etc/sendmail.cfg* to your local configuration.

More information about error notification can be found in Chapter 4 of the IBM manual *General Programming Concepts: Writing and Debugging Programs*.

Robert Schuster
System Administrator (Germany)

© Xephon 2002

Leaving? You don't have to give up *AIX Update*

You don't have to lose your subscription when you move to another location – let us know your new address, and the name of your successor at your current address, and we will send *AIX Update* to both of you, for the duration of your subscription. There is no charge for the additional copies.

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

Using vmtune

INTRODUCTION

Over the years we have found the **vmtune** command to be very helpful for AIX tuning, but it can be quite problematic in use. The command is primarily responsible for displaying and adjusting the Virtual Memory Manager (VMM) parameters. However, incorrect use can cause system failure or other irregularities. So users need to be cautious when using it. This article is not a guide to using **vmtune**, because there are plenty available; however, it does provide an overview of some of the issues that can cause systems problems if they are not addressed.

The **vmtune** command can be found in */usr/samples/kernel*. It is part of the *bos.adt.samples* fileset, which is installed from the AIX base installation media.

A root user using **vmtune** can dynamically change kernel parameters such as VMM page replacement, persistent file reads and writes, file system buffer structures (bufstructs), LVM buffers, raw input/output, paging space parameters, page deletes, and the memory pinning parameters.

USEFUL FLAGS

The most useful **vmtune** flags are shown below:

- **-a** – displays the current statistic counters.
- **-c Nmclust** – indicates the number of 16KB clusters processed by write behind.
- **-d [0 |1]** – enables and disables deferred paging space allocation.
- **-f MinFree** – the minimum number of frames on the free list.
- **-F MaxFree** – the number of frames on the free list at which page stealing is to stop.
- **-g LargePageSize** – the size in bytes of the hardware-supported

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

large pages used for the implementation for the **shmget( )** system call with the SHM_LGPAGE flag.

- **-k NpsKill** – the number of free paging-space pages at which AIX begins killing processes.
- **-M MaxPin** – the maximum percentage of real memory that can be pinned.
- **-n uid** – indicates that processes with a user ID less than **uid** should *not* be killed when paging space is low.
- **-N Pd_Npages** – indicates the number of pages that should be deleted in one chunk from RAM when a file is deleted.
- **-p minperm** – the point below which file pages are protected from the repage algorithm.
- **-P MaxPerm** – the point above which the page stealing algorithm steals only file pages.
- **-t maxclien** – the maximum percentage of RAM that can be used for cacheing client pages.
- **-u lvm_Bufcnt** – indicates the number of Logical Volume Manager (LVM) buffers for raw physical I/Os.
- **-U unixfile** – indicates the name of the AIX file to patch for the **-m**, **-v,** and **-C** flags.
- **-v framesets** – shows the number of real memory free lists (framesets) per memory pool.
- **-w NpsWarn** – indicates the number of free paging-space pages at which AIX begins sending the SIGDANGER signal to processes.
- **-W MaxRandWrt** –indicates a threshold (in 4KB pages) for random writes to accumulate in RAM before these pages are **msync**ed to disk via a write-behind algorithm.

PRECAUTIONS

As mentioned earlier **vmtune** needs to be used with caution. The following issues can result in problems.

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

Version control

The **vmtune** command is AIX version specific. The **vmtune** code that accompanies each release of the operating system is tailored specifically to the VMM in that release. This is why the **vmtune** command is located in the samples directory. It is essential that the right version of **vmtune** is used for the version of AIX that you are running. Running the **vmtune** command from one release on a system with a different VMM release might result in AIX becoming inoperable or in inconsistent results.

This is less of a problem with AIX 5L. AIX 5L Version 5.1 supports both the 32-bit kernel and the 64-bit kernel. Therefore, in addition to the **vmtune** command that supports the 32-bit kernel, there is a **vmtune64** command which supports the 64-bit kernel. If the **vmtune** command is executed on a system running the 64-bit kernel, the system automatically forks/execs the **vmtune64** program.

It should be noted that the functionality of the **vmtune** command also varies between versions of the operating system. Later versions of AIX support new options that are unavailable on older versions. It is therefore highly advisable to review the tuning information provided before using the **vmtune** command to change system parameters.

The minfree value

Another issue that users need to consider is the **minfree** value. If the value of **minfree** (**-f**) is set too high, it can result in excessive paging. This is because premature stealing of pages is required to support the size of the memory free list. It is therefore advisable to ensure that the difference between the **maxfree** value and the **minfree** value is equal to (or greater than) the **maxpgahead** value. On SMP systems, the value of the **maxfree** and **minfree** as displayed by **vmtune** is the sum of the **maxfree** and **minfree** values for all of the memory pools. Users should use the **vmsta** command to determine the correct value for **minfree**.

The maxpin value

maxpin specifies the maximum percentage of real memory that can be pinned. The default value for this is 80. This value should not be

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

set very low because the kernel may need to pin memory at times. When changing the value of the **maxpin** value, ensure that there is always at least 4MB of memory available for the kernel.

Saving changes

Any changes made using **vmtune** will be lost after a reboot. Therefore, in order to ensure that the changed **vmtune** values are set at boot time, you should insert the appropriate **vmtune** command in the */etc/inittab* file. This could look something like this:

```
vmtune:2:once:/usr/samples/kernel/vmtune -P 3Ø -p 5 -c 4 -W 128 -R 16
```

The only exceptions to this are the **-C -m -v -g -L** flags. These require a **bosboot** and a reboot, and it is essential that these are not inserted into the */etc/inittab* file. Furthermore, the **-b** option should be executed before the filesystems are mounted.

-l LruBucke

-l LruBucke indicates the size (in 4KB pages) of the Least Recently Used (LRU) page-replacement bucket size. This is the number of page frames that will be examined for page replacement before starting over at the beginning of this set of page frames to examine the pages again. If insufficient pages are found that can be stolen, LRU proceeds to the next bucket of pages. The default value is 512MB, and the minimum is 256MB. Tuning this option is not recommended.

CONCLUSION

We have always found **vmtune** to be a useful component of AIX. However, users need to closely follow the precautions suggested in this article to prevent serious system errors.

Systems Programmer (UK) © Xephon 2002

Have you come across any undocumented features in AIX Version 5L? Why not share your discovery with others? Send your findings to us at any of the addresses shown on page 2.

 © 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

Implementing file mirroring using rsync

The RSYNC utility is a free, open source program for Unix and Windows that enables fast incremental file transfer. It is written and maintained by Andrew Tridgell and Paul Mackerras. This tool lets you copy files and directories between a local host and a remote host over slow or fast networks using a custom-written rsync algorithm/protocol and its own uniquely assigned 'port'. Rsync tries to transfer only the changed parts of the files. Thus if only 50 lines of a 100MB file have changed, rsync will try to transfer just the changed lines and not the entire contents of the file.

The following is a list of the main features provided by this utility:

- It can update whole directory trees and filesystems.
- The rsync server and the rsync client cooperate in order to minimize the time and the amount of information that has to be transferred to implement the mirroring.
- It is able to transfer only changed portions of the files.
- It supports the anonymous mode of mirroring.
- It supports various forms of CRC-based checks.
- It can be directed to delete obsolete files.
- It can use a wide range of security options, including working over Secure Shell (SSH), utilizing passwords and *host.allow* configuration files.
- The server portion has to be installed by superuser.
- The client portion can be used by any user.

INSTALLATION AND CONFIGURATION

You can download the source code as well as precompiled binary packages from http://rsync.samba.org. At the time of writing this article, the latest version is 2.5.5 'Snowy River', released on 2 April 2002.

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

The following are typical compilation steps:

- Extract the files from the downloaded package: **gtar xzvf rsync—2.5.5tar.gz**.
- Change directory to the rsync-2.5.5 subdirectory: **cd ./rsync-2.5.5**.
- Run the **./configure** script to configure the package with definitions suitable for your system.
- Type **make** to compile the package.
- Type **make install** to copy the generated executable files to */usr/local/bin*.

SYSTEM SET-UP

For security reasons the rsync daemon uses a privileged TCP port. The following line should be added to your */etc/services* file in order to define it:

```
rsync    873/tcp
```

The daemon can be launched via **inetd** or as a stand-alone server from a start-up script or command line.

If you choose to launch the daemon from **inetd**, the following line should be added to your */etc/inetd.conf* file:

```
rsync stream tcp nowait root /usr/local/bin/rsync rsyncd  —daemon
```

Another possibility is to include the following lines in the */etc/rc.local* script:

```
# Starting rsync daemon
/usr/local/bin/rsync —daemon
```

After editing the */etc/inetd.conf* remember to send a HUP signal to the **inetd** process to tell it to re-read its configuration file.

To enable logging of the sessions in the */var/adm/syslog* file, add the following line to the file */etc/syslog.conf* .

```
local3.info     /var/adm/syslog
```


© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

RSYNC DAEMON SET-UP

When rsync is used in daemon mode, it uses a single configurations file, which is by default */etc/rsyncd.conf*, but this can be changed by the command line option *–config FILE*.

The format of the configuration file resembles that of SAMBA. The following is an example:

```
motd file = /etc/motd
max connections = 25
syslog facility = local3

[source]
     comment = mirrored data repository
     path = /data/mirror
     read only = yes
     list = yes
     uid = nobody
     gid = nobody
[tmp]
     comment = temporary data area
     path = /tmp
     read only = no
     list = yes
     hosts allow = 10.0.0.98 *.acme.com
     auth users = mark lea
     secrets file = /etc/rsyncd.secrets
```

The file starts with a global section followed by definitions of modules, which define filesystems used to hold the mirrored data. Any non-global option set in a global section changes the default value for that option. Each option is explained below:

- *motd file* – names a text file used to contain the message of the day to be displayed to clients just before the file transfer begins.
- *log file* – this option tells the rsync daemon to log messages to that file rather than using syslog. This is particularly useful on AIX where **syslog()** doesn't work for chrooted programs.
- *pid file* – directs the rsync daemon to write its process id into the specified file.
- *syslog facility* – the name of the syslog facility to use for logging data transfer statistics and connections. These values are normally defined in the syslog manual page.

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

- *socket options* – set socket communications options from the list available for the man page **setsockopt()** system call.
- *MODULE OPTIONS* – modules are exported by specifying a module name in square brackets, [module], followed by the options for that module.
- *comment* – specifies a description string to be displayed to the client that requests to see a list of exported modules.
- *path* – defines the directory to be exported by the module. You must specify this option for each module in */etc/rsyncd.conf*.
- *use chroot* – if the value of this option is true, the rsync server will chroot to the path before starting the file transfer with the client. This guards against possible implementation security holes, with side effects of requiring superuser privileges and of not being able to follow symbolic links outside of the new root path when reading. When *use chroot* is false, for security reasons symbolic links may only be relative paths pointing to other files within the root path, and leading slashes are removed from absolute paths. The default for *use chroot* is true.
- *max connections* – specifies the maximum number of simultaneous connections that will be supported for this module. Any clients connecting when the maximum has been reached will receive a message telling them to try later. The default is 0, which means no limit.
- *lock file* – specifies the file to be used to implement the *max connections* option. The rsync server uses record locking on this file to ensure that the max connections limit is not exceeded. The default is */var/run/rsyncd.lock*.
- *read only* – if set to true, enables clients to copy/alter files in the module. The default is for all modules to be read only.
- *list* – if set to true, enables the display of module names when the client asks for a listing of available modules. By setting this to false you can create hidden modules. The default value is true.
- *uid* – the user id that rsync will change to while performing file transfers. For anonymous access, it is often set to user *nobody*.

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

- *gid* – the user id that rsync will change to while performing file transfers. For anonymous access, it is often set to user *nobody*.
- *exclude* – specifies a comma-separated list of file name patterns to exclude from transfers. See the section of exclude patterns in the rsync man page for information on the syntax of this option.
- *exclude from* – specifies the name of the file containing a list of file name patterns to exclude from transfers.
- *include* – specifies a comma-separated list of file name patterns to include in transfers. See the section of exclude patterns in the rsync man page for information on the syntax of this option.
- *include from* – specifies the name of the file containing a list of file name patterns to include in transfers.
- *auth users* – specifies a comma and space separated list of usernames that will be allowed to connect to this module. If *auth users* is set, the client will be challenged to supply a username and password to connect to the module. A challenge-response authentication protocol is used for this exchange. The plain text usernames and passwords are stored in the file specified by the *secrets file* option. The default is for all users to be able to connect without a password (this is called anonymous rsync).
- *secrets file* – this option specifies the name of a file that contains the username:password pairs used for authenticating this module. This file is consulted only if the *auth users* option is specified. The file is line-based and contains username:password pairs separated by a single colon.

  There is no default for the *secrets file* option; you must choose a name (such as */etc/rsyncd.secrets*). The file must normally not be readable by others – see *strict modes*.
- *strict modes* – if set to true, the *secrets file* must not be readable by any user id other than the one that the rsync daemon is running under. If *strict modes* is false, the check is not performed. The default is true.
- *hosts allow* – this option allows you to specify a list of patterns that

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

are matched against a connecting client's hostname and IP address. If none of the patterns match, the connection is rejected.

You can also combine *hosts allow* with a separate *hosts deny* option. If both options are specified, the *hosts allow* option is checked first and a match results in the client being able to connect. The *hosts deny* option is then checked and a match means that the host is rejected. If the host matches neither the *hosts allow* nor the *hosts deny* patterns it is allowed to connect.

The default is no *hosts allow* option, which means all hosts can connect.

- *hosts deny* – this option allows you to specify a list of patterns that are matched against a connecting clients hostname and IP address. If the pattern matches, then the connection is rejected

  The default is no *hosts deny* option, which means all hosts can connect.

- *ignore errors* – directs rsync daemon to ignore I/O errors on the server when deciding whether to run the delete phase of the transfer. Normally rsync does not perform deletion of files on the destination if any I/O errors have occurred, in order to prevent disastrous deletion because of a temporary resource shortage or other I/O error. In some cases this test is counter-productive, so you can use this option to turn off this behaviour.

- *ignore nonreadable* – directs rsync server to completely ignore files that are not readable by the user.

- *transfer logging* – if set to true, enables file-based transfer statistic logging.

- *log format* – specifies the format used for the logging file. The format is a text string containing embedded single character escape sequences prefixed with a per cent (%) character.

  A perl script, called rsyncstats, to summarize this format is included in the rsync source code distribution.

- *timeout* – directs the rsync daemon to use a certain value for I/O

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

timeouts for this module. The timeout is specified in seconds. A value of zero means no timeout and is the default.

- *refuse options* – specifies a space-separated list of rsync command line options that will be refused by your rsync daemon.
- *dont compress* – directs the rsync daemon to select filenames based on wildcard patterns that should not be compressed during transfer.

The default setting is:

```
*.gz *.tgz *.zip *.z *.rpm *.deb *.iso *.bz2 *.tbz
```

RSYNC CLIENT SIDE USAGE

Suppose you have a directory called source, and you want to back it up into the directory destination on a remote host. To accomplish this you'd use:

```
rsync -av source/ username@remoteserver:/destination/
```

The -v (verbose) flag directs the rsync client to issue detailed diagnostic messages. This command is equivalent to:

```
rcp -a source/* remoteserver:/destination/
```

One clarification about usage of trailing slashes in rsync parameters lists – most Unix commands ignore trailing slashes in file names. For example, if *src* and *dst* are two directories, then **cp -a src dst** is equivalent to **cp -a src/ dst/**. However, rsync *does* care about the trailing slash, but only on the source argument. For example, let src and dst be two directories, with the file data initially inside directory a. Then this command:

```
rsync -a src dst
```

produces dst/src/data, whereas this command:

```
rsync -a src/ dst
```

produces dst/data. The presence or absence of a trailing slash on the destination argument (dst, in this case) has no effect.

If a file was originally in both source/ and destination/ (from an earlier

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

rsync, for example), and you delete it from source/, you probably want it to be deleted from destination/ on the next rsync. However, the default behaviour is to leave the copy at destination/ in place. Assuming you want rsync to delete any file from destination/ that is not in source/, you'll need to use the **-delete** flag:

```
rcp -a -delete source/* remoteserver:/destination/
```

To run the rsync-with-backup command from the previous section every morning at 4:20am, for example, edit the root cron table (as root):

```
crontab -e
```

Then add the following line:

```
20 4 * * * rsync -a —delete source/* remoteserver:destination/
```

Finally, save the file and exit. The back-up will happen every morning at precisely 4:20am, and root will receive the output by e-mail. Don't copy that example verbatim, though; you should use full path names (such as */usr/bin/rsync* and */home/source/*) to remove any ambiguity.

The following list summarizes the most important command line options of the rsync client command:

-v **verbose** – increase verbosity.

-q **quiet** – decrease verbosity.

-c **checksum** – always checksum.

-a **archive** – archive mode.

-r **recursive** – recurse into directories.

-R **relative** – use relative path names.

-b **backup** – make back-ups (default ~ suffix).

backup-dir – make back-ups into this directory.

suffix=SUFFIX – override back-up suffix.

-u **update** – update only (don't overwrite newer files).

-l **links** – preserve soft links.

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

-L **copy-links** – treat soft links like regular files.

copy-unsafe-links – copy links outside the source tree.

safe-links – ignore links outside the destination tree.

-H **hard-links** – preserve hard links.

-p **perms** – preserve permissions.

-o **owner** – preserve owner (root only).

-g **group** – preserve group.

-D **devices** – preserve devices (root only).

-t **times** – preserve times.

-S **sparse** – handle sparse files efficiently.

-n **dry-run** – show what would have been transferred.

-W **whole-file** – copy whole files, no incremental checks.

-x **one-file-system** – don't cross filesystem boundaries.

-B **block-size=SIZE** – checksum blocking size (default 700).

-e **rsh=COMMAND** – specify rsh replacement.

rsync-path=PATH – specify path to rsync on the remote machine.

-C **cvs-exclude** – auto ignore files in the same way CVS does.

existing – only update files that already exist.

delete – delete files that don't exist on the sending side.

delete-excluded – also delete excluded files on the receiving side.

delete-after – delete after transferring, not before.

ignore-errors – delete even if there are IO errors.

max-delete=NUM – don't delete more than NUM files.

partial – keep partially transferred files.

force – force deletion of directories even if not empty.

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

numeric-ids – don't map uid/gid values by user/group name.

timeout=TIME – set I/O timeout in seconds.

-I **ignore-times** – don't exclude files that match length and time.

size-only – only use file size when determining whether a file should be transferred.

modify-window=NUM – timestamp window (secs) for file match (default=0).

-T **temp-dir=DIR** – create temporary files in directory DIR.

compare-dest=DIR – also compare destination files relative to DIR.

-P – equivalent to partial progress.

-z **compress** – compress file data.

exclude=PATTERN – exclude files matching PATTERN.

exclude-from=FILE – exclude patterns listed in FILE.

include=PATTERN – don't exclude files matching PATTERN.

include-from=FILE – don't exclude patterns listed in FILE.

version – print version number.

daemon – run as a rsync daemon.

address – bind to the specified address.

config=FILE – specify alternate rsyncd.conf file.

port=PORT – specify alternate rsyncd port number.

blocking-io – use blocking I/O for the remote shell.

stats – give some file transfer stats.

progress – show progress during transfer.

log-format=FORMAT – log file transfers using specified format.

password-file=FILE – get password from FILE.

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.

bwlimit=KBPS – limit I/O bandwidth, KB per second.

-h help – show help screen.

RESOURCES

1 http://rsync.samba.org – main Rsync Internet site.

2 http://www.mikerubel.org/computers/rsync_snapshots/ – tutorial-*Easy Automated Snapshot-Style Backups with Linux and Rsync*.

3 http://sunsite.dk/info/guides/rsync/rsync-mirroring.html – Rsync mirroring howto and FAQ.

4 http://perso.club-internet.fr/ffaure/rsync.html – Rsync tutorial.

5 http://lists.samba.org/pipermail/rsync/1999-December/001598.html – another Rsync tutorial.

Alex Polak
System Engineer
APS (Israel)

© Xephon 2002

Call for papers

Why not share your expertise and earn money at the same time? *AIX Update* is looking for technical articles and hints and tips that experienced AIX users have written to make their life, or the lives of their users, easier. We would also be interested in articles about performance and tuning of AIX.

We will publish it (after vetting by our expert panel) and send you a cheque when the article is published. Articles can be of any length and can be sent or e-mailed to Trevor Eddolls at any of the addresses shown on page 2. You can download a free copy of our *Notes for Contributors* from our Web site. Point your browser at www.xephon.com/nfc.

© 2002. Reproduction prohibited. Please inform Xephon of any infringement.

AIX news

Trend Micro has launched its ScanMail for Lotus Notes, providing antivirus security for sites running Lotus Domino on AIX, OS/390, OS/400, Solaris, and Windows NT and 2000.

The software prevents viruses from entering, leaving, and spreading through e-mail attachments shared through internal databases and customers and partners outside the organization. It provides scanning tools that utilize a multi-threaded memory scanning architecture, which is said to enable faster performance and greater scalability across the enterprise with minimal overhead on the Domino server.

File Name blocking allows administrators to block known malicious files from propagating throughout the network.

E-mail and bandwidth management allows administrators to manage their e-mail policy and bandwidth by setting up e-mail filter rules, which can be used to block, delay, prioritize, and notify users. Multiple rules can be set with individual notification messages. ScanMail is integrated with the Domino R5 console and fully supports any Web browser.

For further information contact:
Trend Micro, 10101 N De Anza Blvd, 2nd Floor, Cupertino, CA 95014, USA.
Tel: (408) 257 1500.
URL: http://www.trendmicro.com.

* * *

IBM has announced its WebSphere Business Connection Offering, which promises cross-enterprise process and data sharing. It enables communication between all partners to be accomplished through a standards-based programmable interface and can be integrated with existing systems.

The specific versions include Business Connection Enterprise Edition, Business Connection, and Business Connection Express Edition, and all run with WebSphere Application Server with a DB2 repository and Windows 2000 Server.

It's available on AIX, as well as on Solaris and Windows 2000.

For further information contact your local IBM representative.
URL: http://www.ibm.com/software.

* * *

IBM has announced Tivoli Privacy Manager for e-business V1.1, which helps publish the defined policy on the corporate Web site. The product is used to help simplify policy management and implement privacy practices for business and IT infrastructures. It also helps monitor and enforce access to personally identifiable information (PII) by applications based on the privacy policy, end-user consent to the policy, and end-users' choices allowed by the policy.

It runs on AIX, Solaris, and Windows 2000, and ships electronically.

It maintains an audit trail of access to PII and the conformance of the data accesses to the privacy policy, and it can generate reports based on the audit logs.

For further information contact your local IBM representative.
URL: http://www.tivoli.com/products