42

AIX update

April 1999

In this issue

3 Creating SSA pdisks and hdisks
9 Tuning AIX parameters (part 2)
36 DCE for AIX
43 Systems monitoring shell script
52 inetd – implementing new TCP/IP services
56 AIX news

© Xephon plc 1999

AIX Update

Published by

Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 550955
From USA: 01144 1635 33823
E-mail: HarryLewis@compuserve.com

North American office

Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75077-2150
USA
Telephone: 940 455 7050

Contributions

If you have anything original to say about AIX, or any interesting experience to recount, why not spend an hour or two putting it on paper? The article need not be very long – two or three paragraphs could be sufficient. Not only will you actively be helping the free exchange of information, which benefits all AIX users, but you will also gain professional recognition for your expertise and that of your colleagues, as well as being paid a publication fee – Xephon pays at the rate of £170 ($250) per 1000 words for original material published in AIX Update.

To find out more about contributing an article, see *Notes for contributors* on Xephon's Web site, where you can download *Notes for contributors* in either text form or as an Adobe Acrobat file.

Editor

Harold Lewis

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.

Subscriptions and back-issues

A year's subscription to *AIX Update*, comprising twelve monthly issues, costs £175.00 in the UK; $265.00 in the USA and Canada; £181.00 in Europe; £187.00 in Australasia and Japan; and £185.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the November 1995 issue, are available separately to subscribers for £15.00 ($22.50) each including postage.

AIX Update on-line

Code from *AIX Update* is available from Xephon's Web page at www.xephon.com (you'll need the user-id shown on your address label to access it).

© Xephon plc 1999. 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.

Creating SSA pdisks and hdisks

When SSA disks are configured in AIX, both logical (*hdisk*) and physical (*pdisk*) disk entries are defined in the ODM. The SSA disks also have a unique serial number that's used to create the SSA connection location (*connwhere_shad*). We therefore have a relationship between the *connwhere_shad* and *pdisk* and *hdisk* numbers.

When you add SSA disks to your system, by default, the *pdisk* and *hdisk* numbers are assigned serially. In large sites this can cause confusion, especially when you 'twin-tail' the SSA disks to different systems in a HACMP cluster!

Ideally, it would be really nice to be able to influence the order in which the *pdisk* and *hdisk* numbers are assigned and to create place holder *pdisks* and *hdisks* to represent 'blank' SSA slots in a drawer.

Well, now you can, using the script below. The script accepts the following flags and parameters:

1 You must specify either the **-4** or **-9** flag, which tells the script you are installing either 4.5 GB or 9.1 GB SSA disks. You cannot mix and match the size of the disks you want the script to install.

2 You may optionally use the **-s** flag and a number, which tells the script to start numbering *pdisks* and *hdisks* from this value.

3 You must supply the name of an input file that contains a list of the SSA serial numbers that appear in the order in which you want to number them. You may also add the word 'dummy' to the list to create a 'dummy' disk to represent that slot in the SSA drawer.

If you have the **maymap** utility, then the following two lines can be used to output a list of all SSA serial numbers in the order in which they are installed:

```
maymap -s -a -b1 | grep "\-\-\|" | sed 's/^.*--|//
s/|--.*$//'
```

Another way to list all SSA disk serial numbers in the order they are installed is to use the *Link Verification* option of the **umayf** command (*/usr/lpp/diagnostics/bin/umayf*).

Note that, as the script runs, it compares the list of SSA disk serial numbers supplied with those visible on the system; if the two match, it deletes the existing SSA *pdisk* and *hdisk* definitions so it can recreate them in a sequential order.

Let's suppose you have half an SSA drawer twin-tailed to two systems. You have only six SSA disks, so you put three of them and one blank panel in each bank of four slots in the drawer.

Your six SSA disks are installed, in the order shown by the serial numbers below, in drawer slots one to eight, with the SSA blank panels in slots four and eight.

```
AC7E0799
AC7E6BEF
AC7D0C1C
blank
AC9E2C8E
AC9E2CAD
29C826DD
blank
```

By default, *cfgmgr* would number your *pdisks* and *hdisks* as follows:

```
AC7E0799      pdisk2      hdisk4
AC7E6BEF      pdisk3      hdisk5
AC7D0C1C      pdisk1      hdisk3
AC9E2C8E      pdisk4      hdisk6
AC9E2CAD      pdisk5      hdisk7
29C826DD      pdisk0      hdisk2
```

However, if you use the script in this article and supply it with the following input file:

```
AC7E0799
AC7E6BEF
AC7D0C1C
dummy
AC9E2C8E
AC9E2CAD
29C826DD
dummy
```

you end up with the following *pdisk* and *hdisk* numbers:

```
AC7E0799      pdisk0      hdisk2
AC7E6BEF      pdisk1      hdisk3
AC7D0C1C      pdisk2      hdisk4
```

```
     dummy          pdisk3      hdisk5
     AC9E2C8E       pdisk4      hdisk6
     AC9E2CAD       pdisk5      hdisk7
     29C826DD       pdisk6      hdisk8
     dummy          pdisk7      hdisk9
```

Note that *pdisk3* and *pdisk7*, and *hdisk5* and *hdisk9*, are configured in a 'defined' state, as there is no actual disk associated with these entries.

If you now run the script on other systems, using the same input file, all systems will see the SSA disks as having the same *pdisk* and *hdisk* numbers, with each physical disk in slot order. In a twin-tailed HACMP environment this is extremely useful for keeping track of disks across the cluster.

If you later replace either of the blank SSA disks with a real disk, you first need to remove the 'dummy' definitions using the command below for both the *pdisk* and *hdisk* numbers.

```
     rmdev -dl xx
```

You can then either run **cfgmgr** to configure the new disk with the same *pdisk* and *hdisk* numbers as the definitions just deleted (but make sure you process only one disk at a time) or use the **mkdev** command to create the disks manually.

(Note the use of the continuation character ('➤') in the script below to indicate that one line of code maps to more than one line of print.)

THE SCRIPT

```
#!/usr/bin/ksh
#
# Written by Steve Diwell, Jedi Technology Ltd
#
# Function   : This script takes a list of SSA disk serial
#            : numbers in the input file and creates the disks
#            : in the order in which they appear in the file.
#            : The word 'dummy' is used to indicate that a
#            : dummy disk should be created at that location.

# Get and check the options the user has supplied
getoptions()
{
[[ -n ${DEBUG} ]] && set -x
set -- `getopt 49s:? $*`

while [[ $1 != -- ]]
do
   case $1 in

      "-4")   TYPE="4000mbC"
          ;;
      "-9")   TYPE="9100mbC"
          ;;
      "-s")   HD="-l hdisk"
         PD="-l pdisk"
         NUM=$2
         shift
          ;;
         *)   usage
          ;;
   esac

   shift
done

shift

FILE=$1

[[ -z ${TYPE} ]] && usage
[[ -f ${FILE} ]] || { echo "Unable to read input file ${FILE}" ;
➤  exit 1 ; }
}

# Does the disk already exist?
testdisk()
{
[[ -n ${DEBUG} ]] && set -x

lsparent -P -l $1 1>/dev/null 2>&1 && return 0

return 1
}

# Display the scripts usage
usage()
{
clear

[[ -n ${DEBUG} ]] && set -x

cat <<EOF
```

```
This script will allow you to influence the order in which SSA
pdisk and hdisk numbers are allocated.

You must specify either -4 or -9 to tell the script whether to
configure 4.5 Gig or 9.1 Gig SSA disks.  Only one of these flags is
allowed.

You may supply the -s flag and a number, which will tell the
script to start numbering the pdisks and hdisks from this value.

You need to supply the script with an input <disklistfile> which
is a list of the SSA disk serial numbers, in the order you would
like them numbered. You may add the word "dummy" to this list,
and the script will created a "Defined" SSA disk at this location
to represent SSA dummy disks.

usage: $0 [ -4 | -9 ] [ -s start ] disklistfile

Where:
        -4                Configure 4.5 Gig SSA Disks OR
        -9                Configure 9.1 Gig SSA Disks.
        -s <number>       The start number for pdisks and hdisks.
        disklistfile      The input file containing the list of SSA
                          disk serial numbers.

EOF

exit 0
}

# Start of main script
[[ -n ${DEBUG} ]] && set -x

# Set the working PATH for this script
export PATH=/usr/bin:/etc:/usr/sbin:/usr/ucb:/usr/bin/X11:/sbin
unset TYPE CONN_ADD PD HD NUM FILE

TMPFILE=/tmp/make_ssa_disks.$$
typeset -i NUM

getoptions $*

# Remove all Defined SSA disk.
lsdev -Cc '*disk*' | grep "Defined" | awk '/SSA/ {system
➤  ("rmdev -dl " $1 )}'

# Let's see if your list matches what the script can see ?
lsdev -Cc pdisk -F name | while read PDISK
do
   lsattr -El ${PDISK} -a connwhere_shad
```

```
done | sed 's/^.* 000.//
       s/00D.*$//' | sort > ${TMPFILE}

if ! sort ${FILE} | grep -v dummy | cmp - ${TMPFILE}
then
   echo "Mismatch of 8-digit serial numbers between list supplied
   ➤  and what is visible."
   rm ${TMPFILE}
   exit 2
fi

rm ${TMPFILE}

# Remove all existing SSA disks
lsdev -Cc '*disk*' | awk '/SSA/ {system ("rmdev -dl " $1 )}'

# Here is where we start to add the PDISKS and HDISKS
COUNT=${NUM:-1}

cat ${FILE} | while read LINE
do

echo "Working on the SSA disk with serial number" ${LINE}
sleep 1

[[ -n ${NUM} ]] && {

   testdisk pdisk${NUM} && { echo "pdisk${NUM} already exists!
   ➤  Script aborting!"; exit 3; }

   testdisk hdisk${NUM} && { echo "hdisk${NUM} already exists!
   ➤  Script aborting!"; exit 3; }
}

if [[ ${LINE} = "dummy" ]]
then

# Make a dummy HDISK and PDISK entry
   [[ ${COUNT} -lt 10 ]] && COUNT="00"${COUNT}
   [[ ${COUNT} -lt 100 ]] && COUNT="0"${COUNT}

   mkdev -d -c pdisk -t ossadisk -s ssar -p ssar -w "000000000"$
   ➤  {COUNT}"00D" ${PD}${NUM}
   mkdev -d -c disk -t hdisk -s ssar -p ssar -w     "000000000"$
   ➤  {COUNT}"00D" ${HD}${NUM}

else

# Get the connection address
   CONN_ADD=`for SSA in $(lsdev -CS1 -cadapter -tssa -Fname)
   do
      ssacand -a ${SSA} -P
   done | grep ${LINE}`

# Make the PDISK
   mkdev -c pdisk -t ${TYPE} -s ssar -p ssar -w ${CONN_ADD} ${PD}${NUM}

# Make the associated HDISK
   mkdev -c disk -t hdisk -s ssar -p ssar -w ${CONN_ADD} ${HD}${NUM}

fi

let COUNT=${COUNT}+1
[[ -n ${NUM} ]] && let NUM=${NUM}+1

done

exit 0
```

Steve Diwell
Senior Consultant
Jedi Technology (UK)
© Steve Diwell/Jedi Technology 1999

Tuning AIX parameters (part 2)

This month's instalment concludes this article on tuning AIX parameters.

TAP (CONTINUED)

```
# Set the new value
COMMAND="${COMMAND} ${NEW_VALUE}"
/bin/sh "${COMMAND}"  > ${ERROR_FILE}   2>&1
if [ $? -ne 0 ]
then
    # Command failed
    ERM="ksh:`head -1 ${ERROR_FILE}`"
    DisplayMessage E  "${COMMAND_FAILED}"
    DisplayMessage B  "${BLANK}"
    DisplayMessage "SE"  "${SYSTEM_ERROR}"
    return   ${FALSE}
```

```
else
    DisplayMessage I  "${COMMAND_SUCCEEDED}"
    return ${TRUE}
fi
}

#######################################################################
#
#  Name     : TuneNetwork
#
#  Overview : This function sets the parameter passed.
#
#  Input    : Parameter name
#
#######################################################################
TuneNetwork  ()
{
trap "HandleInterrupt " $SIGINT  $SIGTERM $SIGHUP

TUNE_PARAM="$1"

# Prepare the file containing the parameter description
PrepareParameterFileForDisplay  "NET"  $TUNE_PARAM

# Append the current value obtained to the parameter
# description file
echo "\nCurrent Value : ${CUR_VALUE} \n"  >> ${SPECIFIC_PARAM_FILE}

# Let user view the file
clear
more  ${SPECIFIC_PARAM_FILE}

# Ask for new value
while true
do
   echo "Enter new value for ${TUNE_PARAM} (q to quit):\c"
   read NEW_VALUE
   case  ${NEW_VALUE}  in
     q|Q) return ${TRUE} ;;
      "") : ;;
      * ) # Check for numeric, positive number
          if  IsDigit  "${NEW_VALUE}"  -a ${NEW_VALUE} -ge 0
          then
                break ;
          else
               DisplayMessage E  "${INVALID_VALUE}" ;
               clear
               more  ${SPECIFIC_PARAM_FILE}
          fi ;;
   esac
```

```
done

# Set the new value
COMMAND="${COMMAND}${NEW_VALUE}"
/bin/sh "${COMMAND}"  > ${ERROR_FILE}  2>&1
if [ $? -ne 0 ]
then
    # Command failed
    ERM="ksh:`head -1 ${ERROR_FILE}`"
    DisplayMessage E  "${COMMAND_FAILED}"
    DisplayMessage B  "${BLANK}"
    DisplayMessage "SE"  "${SYSTEM_ERROR}"
    return   ${FALSE}
else
    DisplayMessage I  "${COMMAND_SUCCEEDED}"
    return ${TRUE}
fi
}

#######################################################################
#
#  Name     : TuneScheduler
#
#  Overview : This function sets the parameter passed.
#
#  Input    : Parameter name
#
#######################################################################
TuneScheduler ()
{
trap "HandleInterrupt " $SIGINT  $SIGTERM $SIGHUP

TUNE_PARAM="$1"

# Prepare file containing the parameter description
PrepareParameterFileForDisplay  "SC"  $TUNE_PARAM

# Append current value to the parameter description file
echo "Current Value : ${CUR_VALUE} \n"  >> ${SPECIFIC_PARAM_FILE}

# Let user view the file
clear
more  ${SPECIFIC_PARAM_FILE}

# Ask for the new value
while true
do
   if  [ "${TUNE_PARAM}" = "RESTORE-DEFAULTS"  ]
   then
      echo "Enter value for $TUNE_PARAM (1=Yes 0=No or q to quit):\c"
   else
      echo "Enter new value for $TUNE_PARAM(q to quit):\c"
   fi
   read NEW_VALUE
   case  ${NEW_VALUE}  in
     q|Q) return ${TRUE} ;;
      "") : ;;
      * ) # Check for numeric, positive value
          if  IsDigit  "${NEW_VALUE}"   -a ${NEW_VALUE} -ge 0
          then
               if  [ "${TUNE_PARAM}" = "RESTORE-DEFAULTS" ]
               then
                    if [ $NEW_VALUE -gt 1  ]
                    then
                        DisplayMessage E  "${INVALID_VALUE}" ;
                        clear ;
                        more  ${SPECIFIC_PARAM_FILE} ;
                    else
                        break;
                    fi ;
               else
                    break ;
               fi ;
          else
               DisplayMessage E  "${INVALID_VALUE}" ;
               clear ;
               more  ${SPECIFIC_PARAM_FILE} ;
          fi ;;
   esac
done

# Set the new value
if  [ "${TUNE_PARAM}" = "RESTORE-DEFAULTS" -a ${NEW_VALUE} -eq 0 ]
then
    return $TRUE
fi
if  [ "${TUNE_PARAM}" = "RESTORE-DEFAULTS" ]
then
    /bin/sh "${COMMAND}" > ${ERROR_FILE}  2>&1
else
    COMMAND="${COMMAND} ${NEW_VALUE}"
    /bin/sh "${COMMAND}" > ${ERROR_FILE}  2>&1
fi
if [ $? -ne 0 ]
then
    # Command failed
    ERM="ksh:`head -1 ${ERROR_FILE}`"
    DisplayMessage E  "${COMMAND_FAILED}"
    DisplayMessage B  "${BLANK}"
    DisplayMessage "SE"  "${SYSTEM_ERROR}"
```

© 1999. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
    return   ${FALSE}
else
    DisplayMessage I  "${COMMAND_SUCCEEDED}"
    return ${TRUE}
fi
}

######################################################################
#
#  Name     : PrepareParameterFileForDisplay
#
#  Overview : This function prepares a file that contains details
#             of the parameter in question.
#
#  Input    : 1. Parameter description file name
#             2. Parameter name.
#
######################################################################
PrepareParameterFileForDisplay ()
{
trap "HandleInterrupt " $SIGINT  $SIGTERM $SIGHUP

PARAM_FILE_MNEMONIC=$1
TUNE_PARAM=$2

# Specify split file names for the csplit command. These files
# have extensions such as 00 01 02
SPLIT_FILES="${TEMP_FILE}."

# Assign the file with .01 extension as relevant one
SPECIFIC_PARAM_FILE="${TEMP_FILE}.01"

# Call appropriate function to load all the parameters descriptions
# into the file ${PARAM_FILE}
if  [ "${PARAM_FILE_MNEMONIC}" = "VM" ]
then
     LoadVirtualMemoryTuneParameterFile

elif [ "${PARAM_FILE_MNEMONIC}" = "SC" ]
then
     LoadSchedularTuneParameterFile

elif [ "${PARAM_FILE_MNEMONIC}" = "NET" ]
then
     LoadNetworkTuneParameterFile
fi

# Extract relevant details from master file ${PARAM_FILE}
# into specific file ${SPECIFIC_PARAM_FILE}
#
```

```
csplit -s -f ${SPLIT_FILES} ${PARAM_FILE} '/START-'${TUNE_PARAM}'/+1'\
                '/END-'${TUNE_PARAM}'/-1'
}

#######################################################################
#
#  Name     : LoadVirtualMemoryTuneParameterFile
#
#  Overview : This function loads all the parameter descriptions
#             for virtual memory tuning into the file ${PARAM_FILE}.
#
#######################################################################
LoadVirtualMemoryTuneParameterFile ()
{
trap "HandleInterrupt " $SIGINT  $SIGTERM $SIGHUP

cat  <<! > ${PARAM_FILE}

START-MAXFREE

           Maximum free

Purpose  : To set the maximum size to which the VMM page frame free
           list may grow by page stealing.

Values   : The default is configuration-dependent; the range is
           16 to 204,800 4 KB frames.

Display  : vmtune

Change   : vmtune -F new_value
           Changes take effect immediately and are effective till
           the next boot. Changes are made permanent by adding the
           vmtune command to /etc/inittab.

Diagnosis: Observe free list size changes using vmstat n.

Tuning   : If vmstat n shows free list size frequently falling below
           minfree due to demand, increase maxfree to reduce calls to
           replenish free list. Keep the difference between maxfree
           and minfree below 100.

END-MAXFREE

START-MAXPERM

           Maximum permanent

Purpose  : To set the percentage of memory page frames occupied by
           permanent pages, above which permanent pages will have
           their frames stolen.

Values   : The default is 80% x (momory_size - 4 MB); the range
           is 5 to 100.

Display  : vmtune

Change   : vmtune -P new_value
           Changes take effect immediately and are effective till
           the next boot. Changes are made permanent by adding the
           vmtune command to /etc/inittab.

Diagnosis: Monitor disk I/O with iostat n.

Tuning   : If some files are known to be read repetitively, and I/O
           rates do not decrease with time from start-up, maxperm
           may be too low.

END-MAXPERM

START-MAXPGAHEAD

           Maximum read page ahead

Purpose  : To set the upper limit on the number of pages the VMM
           reads ahead when processing a sequentially accessed file.

Values   : The default is 8; the range is 0 to 16.

Display  : vmtune

Change   : vmtune -R new_value
           Changes take effect immediately and are effective till
           the next boot. Changes are made permanent by adding the
           vmtune command to /etc/inittab.

Diagnosis: Observe the elapsed execution time of critical I/O-
           dependent applications with the command time.

Tuning   : If execution time decreases with higher maxpgahead, check
           other applications to ensure that their performance has not
           deteriorated.

END-MAXPGAHEAD

START-MAXPIN

           Maximum pinned

Purpose  : To set the maximum percentage of real memory that can be
```

```
           pinned.

Values   : The default is 80% of RAM; the range is from at least 4 MB
           pinnable to at least 4 MB unpinnable.

Display  : vmtune

Change   : vmtune -M new_value
           Changes take effect immediately and are effective till
           the next boot.

Diagnosis: N/A.

Tuning   : Only change for extreme situations, such as maximum load
           benchmarking.

END-MAXPIN

START-MINFREE

           Minimum free

Purpose  : Sets the VMM page frame free list size at which the VMM
           starts to steal pages to replenish the free list.

Values   : The default is configuration-dependent; the range is 1
           to any positive integer.

Display  : vmtune

Change   : vmtune -f new_value
           Changes take effect immediately and are effective till
           the next boot. Changes are made permanent by adding the
           vmtune command to /etc/inittab.

Diagnosis: vmstat n

Tuning   : If processes are being delayed by page stealing, increase
           minfree by an equal or greater amount.

END-MINFREE

START-MINPERM

           Minimum permanent

Purpose  : Sets the percentage of page frames occupied by permanent
           pages below which the VMM steals frames from both permanent
           and working pages without regard to repage rates.
```

```
Values    : The default is 20% x (memory size - 4 MB); the range is
            5 to 100.

Display   : vmtune

Change    : vmtune -P new_value
            Changes take effect immediately and are effective till
            the next boot. Changes are made permanent by adding the
            vmtune command to /etc/inittab.

Diagnosis: Monitor disk I/O with iostat n.

Tuning    : If some files are known to be read repetitively, and I/O
            rates do not decrease with time from start-up, minperm may
            be too low.

END-MINPERM

START-MINPGAHEAD

            Minimum read page ahead

Purpose   : To set the number of pages the VMM reads ahead when it
            first detects sequential access.

Values    : The default is 2; the range is 0 to 16.

Display   : vmtune

Change    : vmtune -r new_value
            Changes take effect immediately and are effective till
            the next boot. Changes are made permanent by adding the
            vmtune command to /etc/inittab.

Diagnosis: Observe the elapsed execution time of critical sequential
            I/O-dependent applications with time command.

Tuning    : If execution time decreases with higher minpgahead, check
            other applications to ensure that their performance has not
            deteriorated.

END-MINPGAHEAD

START-NPSKILL

            Number of free paging-space for process kill

Purpose   : Sets the threshold number of free paging space pages
            at which processes begin to be killed.
```

```
Values    : The default is 128; the range is 0 to the number of pages
            in real memory.

Display   : vmtune

Change    : vmtune -k new_value
            Changes take effect immediately and are effective till
            the next boot.

Diagnosis: N/A.

Tuning    : N/A.

END-NPSKILL

START-NPSWARN

            Number of free paging space pages before warnings

Purpose   : Sets the minimum number of free paging space pages at
            which processes begin to receive SIGDANGER.

Values    : The default is 512; the range is from at least npskill
            to the number of pages in real memory.

Display   : vmtune

Change    : vmtune -w new_value
            Changes take effect immediately and are effective till
            the next boot.

Diagnosis: N/A.

Tuning    : Increase if you experience processes being killed for
            low paging space.

END-NPSWARN

START-NUMCLUST

            Number of clusters

Purpose   : Sets the number of 16 KB clusters processed by write
            behind.

Values    : The default is 1; the range is 1 to any positive integer.

Display   : vmtune

Change    : vmtune -c new_value
```

```
           Changes take effect immediately and are effective till
           the next boot.

Diagnosis: N/A.

Tuning   : May be appropriate to increase this parameter if striped
           logical volumes or disk arrays are being used.

END-NUMCLUST

START-NUMFSBUF

           Number of file system buffers

Purpose  : To set the number of file system bufstructs.

Values   : The default is 64; the range is 64 to any positive integer.

Display  : vmtune

Change   : vmtune -b new_value
           Changes take effect immediately and are effective till
           the next boot.

Diagnosis: N/A.

Tuning   : May be appropriate to increase if striped logical volumes
           or disk arrays are being used.

END-NUMFSBUF

!
}

######################################################################
#
#  Name     : LoadSchedularTuneParameterFile
#
#  Overview : The function loads all the parameter descriptions
#             for scheduler tuning into the file, ${PARAM_FILE}.
#
######################################################################
LoadSchedularTuneParameterFile ()
{
trap "HandleInterrupt " $SIGINT  $SIGTERM $SIGHUP

cat <<!  >  ${PARAM_FILE}

START-MAX_FORK_RETRY_INTERVAL
```

© 1999. Reproduction prohibited. Please inform Xephon of any infringement.

```
           Maximum fork retry interval

Purpose  : Specifies the amount of time to wait before retrying a fork()
           that failed for lack of paging space.

Values   : The default is 10 10-millisecond cycles; the range is 10
           to n clock ticks

Display  : schedtune

Change   : schedtune -f new_value
           Changes take effect immediately and are effective till
           the next boot. Changes are made permanent by adding the
           schedtune command to /etc/initab.

Symptoms : If processes have been killed for lack of paging space,
           monitor the situation with the sigdanger() subroutune.

Tuning   : If low paging space is a result of brief, sporadic
           workload peaks, increasing the retry interval may allow
           processes to wait long enough for paging space to be
           released. Otherwise allocate more paging space.

END-MAX_FORK_RETRY_INTERVAL

START-HIGH-MEMORY-OVERCOMMITMENT-THRESHOLD

           Memory load control parameters

Purpose  : Customizes the VMM memory load control facility to maximize
           system use while avoiding thrashing. The most frequently
           used parameter is h, the high memory over commitment
           threshold.

Values   : The default value of h is 6; the range is 0 to any positive
           integer

Display  : schedtune

Change   : schedtune [-h New Value]
           Changes take effect immediately and are effective till
           the next boot. Changes are made permanent by adding the
           schedtune command to /etc/initab.

Symptoms : Heavy memory loads cause wide variations in response time.

Tuning   : schedtune -h 0 turns off memory load control.

END-HIGH-MEMORY-OVERCOMMITMENT-THRESHOLD
```

```
START-PROCESS-MEMORY-OVERCOMMITMENT-THRESHOLD

           Memory load control parameters

Purpose  : Customizes the VMM memory load control facility to maximize
           system use while avoiding thrashing. The most frequently
           used parameter is p, the process memory over commitment
           threshold.

Values   : The default value of p is 4; the range is from 0 to any
           positive integer.

Display  : schedtune

Change   : schedtune [-p new_value]
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the
           schedtune command to /etc/initab.

Symptoms : Heavy memory loads cause wide variations in response time.

Tuning   : schedtune -p 2 requires a higher level of repaging by a
           given process before it is a candidate for suspension
           by the memory load control.

END-PROCESS-MEMORY-OVERCOMMITMENT-THRESHOLD

START-MINIMUM-LEVEL-OF-MULTIPROGRAMMING

           Memory load control parameters

Purpose  : Customizes the VMM memory load control facility to maximize
           system use while avoiding thrashing. The most frequently
           used parameter is m, the minimum level of multiprogramming.

Values   : The default value of m is 2; the range is from 0 to any
           positive integer.

Display  : schedtune

Change   : schedtune [-m new_value]
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the
           schedtune command to /etc/initab.

Symptoms : Heavy memory loads cause wide variations in response time.

Tuning   : schedtune -m 10 requires that the memory load control
           facility always leaves at least 10 user processes running
           when it suspends processes.
```

```
END-MINIMUM-LEVEL-OF-MULTIPROGRAMMING

START-PROCESS_PRIORITY_CALCULATION

           Process priority calculation

Purpose  : Specifies the amount by which a process's priority value is
           increased as a result of recent CPU usage and the rate at
           which the recent CPU usage value decays. The relevant
           parameters are r and d.

Values   : The default value is 16; the range is 0 to 32. When
           calculating the values of r and d, the value is divided by
           32. Thus the effective range of the factor is from 0 to 1 in
           increments of 0.03125.

Display  : schedtune

Change   : schedtune -r or schedtune -d
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the
           schedtune command to /etc/initab.

Diagnosis: Use ps -al. If you find that the PRI column has priority
           values for foreground processes (those with NI values of 20)
           that are higher than the PRI values of some background
           processes (NI values > 20), try reducing the r value.

Tuning   : Decreasing r makes it easier for foreground processes to
           compete. Decreasing d stops foreground processes competing
           with background processes for longer. schedtune -r 2 ensures
           that new foreground process receive at least half a second
           of CPU time before having to compete with other processes
           that have NI >= 24.

END-PROCESS_PRIORITY_CALCULATION

START-TIME_SLICE_EXPANSION_AMOUNT

           Time slice expansion amount

Purpose  : To set the number of 10-millisecond clock ticks by which the
           default 10 millisecond time slice is to be increased.

Values   : The default is 0; the range is 0 to any positive integer.

Display  : schedtune

Change   : schedtune -t new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the
           schedtune command to /etc/inittab.

Symptoms : N/A.

Tuning   : In general, this parameter should not be changed. If the
           workload consists almost entirely of very long-running,
           CPU-intensive programs, increasing this parameter may have
           some positive effect.

END-TIME_SLICE_EXPANSION_AMOUNT

START-RESTORE-DEFAULTS

           Restore Defaults

Memory Load control(h) = 6
Process  to suspend(p) = 4
Minimum Multi-programming Level(m) = 2
Fork Retry Interval(f) = 10
Time Slice Expansion(t)= 0
Process Priority Calculation(r) = 16
Process Priority Calculation(d) = 16

END-RESTORE-DEFAULTS

!
}

#####################################################################
#
#  Name     : LoadNetworkTuneParameterFile
#
#  Overview : This function loads all the parameter descriptions
#             for network tuning into the file ${PARAM_FILE}.
#
#  Note     : The file contains a section for each parameter.
#
#####################################################################
LoadNetworkTuneParameterFile ()
{
trap "HandleInterrupt " $SIGINT  $SIGTERM $SIGHUP

cat <<!  >  ${PARAM_FILE}

START-ARPT_KILLC

           ARP entry deletion

Purpose  : Sets the time before an inactive, complete ARP entry is
           deleted.
```

```
Values   : The default is 20 minutes; the range is not available.

Display  : no -a or no -o arpt_killc

Change   : no -o arpt_killc=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : N/A.

Tuning   : To reduce ARP activity in a stable network, increase
           arpt_killc. The effect of this, however, isn't large.

END-ARPT_KILLC

START-IPFORWARDING

           IP packets forwarding

Purpose  : Specifies whether the kernel forwards IP packets.

Values   : The default is 0 (no); the allowed values are 0 and 1.

Display  : no -a or no -o ipforwarding

Change   : no -o ipforwarding=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : N/A.

Tuning   : This is a configuration decision and has performance
           consequences.

END-IPFORWARDING

START-IPFRAGTTL

           IP packet fragment wait

Purpose  : Sets the Time To Live for IP packet fragments.

Values   : The default is 60 seconds; the range is 60 to n.

Display  : no -a or no -o ipfragttl

Change   : no -o ipfragttl=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Diagnosis: netstat -s

Tuning   : This is a configuration decision with performance
           consequences.

END-IPFRAGTTL

START-IPQMAXLEN

           IP input queue entry

Purpose  : Specifies the maximum number of entries on the IP input
           queue.

Values   : The default is 50; the range is 50 to n.

Display  : no -a or no -o ipqmaxlen

Change   : no -o ipqmaxlen=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Diagnosis: Use crash to access the IP input queue overflow counter.

Tuning   : Increase size.

END-IPQMAXLEN

START-IPSENDREDIRECTS

           IP redirect signal

Purpose  : Specifies whether the kernel sends redirect signals.

Values   : The default is 1 ( yes); the allowed values are 0 and 1.

Display  : no -a or no -o ipsendredirects

Change   : no -o ipsendredirects=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : N/A.

Tuning   : N/A. This is a configuration decision with performance
           consequences.
```

```
END-IPSENDREDIRECTS

START-LOOP_CHECK_SUM

           Loopback checksums

Purpose  : Specifies whether checksums are built and used on a loopback
           interface in AIX version 3.2.5.

Values   : The default is 1 (yes); the allowed values are 0 and 1.

Display  : no -a or no -o loop_check_sum

Change   : no -o loop_check_sum=0
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : N/A.

Tuning   : Turning checksum verification off (loop_check_sum=0) is
           recommended.

END-LOOP_CHECK_SUM

START-LOWCLUST

           Low-water mark for mbuf cluster

Purpose  : Specifies the low water mark for the mbuf cluster pool in
           AIX version 3.2.5.

Values   : The default is configuration-dependent; the range is 5 to n.

Display  : no -a or no -o lowclust

Change   : no -o lowclust=New Value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Diagnosis: netstat -m

Tuning   : If "request for more memory denied" is not zero, increase
           lowclust.

END-LOWCLUST

START-LOWMBUF
```

```
Purpose  : Specifies the low water mark for the mbuf pool in AIX
           version 3.2.5

Values   : The default is configuration-dependent; the range is
           64 to n.

Display  : no -a or no -o lowmbuf

Change   : no -o lowmbuf=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Diagnosis: netstat -m

Tuning   : If "request for more memory denied" is not zero, increase
           lowmbuf.

END-LOWMBUF

START-MAXTTL

           Time for RIP

Purpose  : specifies the Time To Live for Routing Information
           Protocol(RIP) packets.

Values   : The default is 255; the range is not available.

Display  : no -a  or no -o maxttl

Change   : no -o maxttl=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : N/A.

Tuning   : N/A.

END-MAXTTL

START-MB_CL_HIWAT

           High water mark for mbuf cluster pool

Purpose  : Specifies the high water mark for mbuf cluster pool in AIX
           version 3.2.5

Values   : The default is configuration-dependent; the range is not
           available.
```

```
Display  : no -a  or  no -o mb_cl_hiwat

Change   : no -o mb_cl_hiwat=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Diagnosis: netstat -m

Tuning   : If the number of mbuf clusters (referred to as "mapped
           pages" by netstat) is regularly greater than mb_cl_hiwat,
           increase mb_cl_hiwat.

END-MB_CL_HIWAT

START-NONLOCSRCROUTE

           Strict source-routed IP packets

Purpose  : Indicates that strict source-routed IP packets can be
           addressed to hosts outside the local ring. Loose source
           routing is not affected.

Values   : The default is 1 (yes); the allowed values are 0 and 1.

Display  : no -a or no -o nonlocsrcroute

Change   : no -o nonlocsrcroute=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : N/A.

Tuning   : This is a configuration decision with minimal performance
           consequences.

END-NONLOCSRCROUTE

START-RFC1122ADDRCHK

           Performing address validation

Purpose  : Specifies whether address validation is performed between
           communication layers.

Values   : The default is 0 (no); the allowed values are 0 and 1.

Display  : no -a or -o rfc1122addrchk
```

```
Change   : no -o rfc1122addrchk=new_value
           Change takes effect immediately. Change is effective until
           next boot. Permanent change is made by adding no command to
           /etc/rc.net.

Symptoms : N/A.

Tuning   : This value should not be changed.

END-RFC1122ADDRCHK

START-RFC1323

           tcp_sendspace and tcp_recvspace

Purpose  : Indicates whether tcp_sendspace and tcp_recvspace can
           exceed 64 KB.

Values   : The default is 0 (no); the allowed values are 0 and 1.

Display  : no -a or no -o rfc1323

Change   : no -o rfc1323=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : None.

Tuning   : Change this before attempting to set tcp_sendspace and
           tcp_recvspace to more than 64 KB

END-RFC1323

START-SB_MAX

           Maximum socket buffer

Purpose  : Specifies an absolute upper bound on the size of TCP and
           UDP socket buffers.

Values   : The default is 65536, the range is not available.

Display  : no -a or no -o sb_max

Change   : no -o sb_max=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to/etc/rc.net.
```

```
Symptoms : None

Tuning   : Increase its size, preferably by a multiple of 4096. This
           parameter should be about twice the largest socket buffer
           limit.

END-SB_MAX

START-SUBNETSARELOCAL

           Subnet mask consideration

Purpose  : Specifies that all subnets that match the subnet mask are to
           be considered local for the purpose of establishing, for
           example, the maximum TCP segment size

Values   : The default is 1 (yes); the allowed values are 0 and 1.

Display  : no -a or no -o subnetsarelocal

Change   : no -o subnetsarelocal=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : N/A.

Tuning   : This is a configuration decision with performance
           consequences. If subnets do not all have the same MTU,
           fragmentation at bridges may degrade performance. If
           subnets do have the same MTU, and subnetsarelocal is 0,
           TCP sessions may use an unnecessarily small MSS.

END-SUBNETSARELOCAL

START-TCP_KEEPIDLE

           Idle TCP connection

Purpose  : Specifies total time to keep an idle TCP connection open.

Values   : The default is 14,400 units of half a second (two hours);
           the range is any positive number.

Display  : no -a or no -o tcp_keepidle

Change   : no -o tcp_keepidle=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.
```

```
Symptoms : N/A.

Tuning   : This is a configuration decision with minimal performance
           consequences. No change is recommended.

END-TCP_KEEPIDLE

START-TCP_KEEPINTVL

           TCP connection validation interval

Purpose  : Specifies the interval between packets sent to validate the
           TCP connection.

Values   : The default value is 150 units of half a second (75
           seconds); the range is any positive number.

Display  : no -a or no -o tcp_keepintvl

Change   : no -o tcp_keepintvl=new_value

Symptoms : N/A.

Tuning   : This is a configuration decision with minimal performance
           consequences. No change is recommended. If the interval is
           significantly shortened, consumption of CPU cycles and
           bandwidth could increase.

END-TCP_KEEPINTVL

START-TCP_MSSDFLT

           TCP default segment size

Purpose  : Specifies the default maximum segment size used in
           communicating with remote networks.

Values   : The default is 512; the range is 512 to the MTU of local
           net minus 64.

Display  : no -a or no -o tcp_mssdflt

Change   : no -o tcp_mssdflt=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : N/A.

Tuning   : Increase, if practical.
```

```
END-TCP_MSSDFLT

START-TCP_RECVSPACE

           Default TCP receive buffer socket size

Purpose  : Sets the default size of the TCP socket receive buffer.

Values   : The default is 16,384; the range is as follows:
             0 to 64 KB if rfc11323 = 0
             0 to  4 GB if rfc11323 = 1.
           Must be less than or equal to sb_max, should be equal to
           tcp_sendspace, and should be the same on all frequently
           accessed AIX systems.

Display  : no -a or no -o tcp_recvspace

Change   : no -o tcp_recvspace=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : Poor throughput.

Tuning   : Increase size, preferably to a multiple of 4096.

END-TCP_RECVSPACE

START-TCP_SENDSPACE

           Default TCP send buffer socket size

Purpose  : Sets the default size of the TCP socket send buffer.

Values   : The default is 16,384; the range is as follows:
             0 to 64 KB if rfc1323 = 0
             0 to  4 GB if rfc1323 = 1.
           Must be less than or equal to sb_max, should be equal to
           tcp_recvspace, and should be the same on all frequently
           accessed AIX systems.

Display  : no -a or no -o tcp_sendspace

Change   : no -o tcp_sendspace=New Value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : Poor throughput.
```

```
Tuning   : Increase size, preferably to multiple of 4096

END-TCP_SENDSPACE

START-TCP_TTL

           TCP packets wait

Purpose  : Specifies the Time To Live for TCP packets.

Values   : The default is 60 units of 10 milliseconds; the range is
           any positive integer.

Display  : no -a no -o tcp_ttl

Change   : no -o tcp_ttl=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Diagnosis: netstat -s

Tuning   : If the system is experiencing TCP time-outs, increasing
           tcp_ttl may reduce re-transmissions.

END-TCP_TTL

START-THEWALL

           Real memory upper bound for communication

Purpose  : Provides an absolute upper bound to the amount of real
           memory that can be used by the communication subsystem.

Values   : The default is 25% of real memory; the range is 0 to 50%
           of real memory.

Display  : no -a or no -o thewall

Change   : no -o thewall=new_value
           new_value is in KB, not bytes. Changes take effect
           immediately and are effective till the next boot. Changes
           can be made permanent by adding the no command to
           /etc/rc.net.

Symptoms : None

Tuning   : Increase size, preferably to a multiple of 4 (KB)

END-THEWALL
```

```
START-UDP_RECVSPACE

           UDP socket receive buffer default size

Purpose  : Sets the default value of the size of the UDP socket buffer.

Values   : The default is 41,600; the range is not available.

Display  : no -a or no -o udp_recvspace

Change   : no -o udp_recvspace=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : A non-zero value of n in the netstat -s report of udp
           means n socket buffer overflows.

Tuning   : Increase size, preferably to a multiple of 4096.

END-UDP_RECVSPACE

START-UDP_SENDSPACE

           UDP socket send buffer default size

Purpose  : Sets the default UDP socket send buffer size.

Values   : The default is 9216; the range is 0 to 65,536. This value
           must be less than or equal to sb_max.

Display  : no -a  or no -o udp_sendspace

Change   : no -o udp_sendspace=new_value
           Changes take effect immediately and are effective till the
           next boot. Changes are made permanent by adding the no
           command to /etc/rc.net.

Symptoms : N/A.

Tuning   : Increase size, preferably to a multiple of 4096.

END-UDP_SENDSPACE

START-UDP_TTL

           UDP packets wait

Purpose  : Specifies the Time To Live of UDP packets.
```

```
Values    : The default is 30 units of 10 milliseconds; the range is
            any positive integer.

Display   : no -a or no -o udp_ttl

Change    : no -o udp_ttl=new_value
            Changes take effect immediately and are effective till the
            next boot. Changes are made permanent by adding the no
            command to /etc/rc.net.

Symptoms : N/A.

Tuning    : N/A.

END-UDP_TTL

START-BIOD_COUNT

            Biod count

Purpose   : Specifies the number of biod processes available to handle
            NFS requests on a client.

Values    : The default is 6; the range is 1 to any positive integer.

Display   : ps -eaf | grep biod

Change    : chnfs -b new_value
            Changes normally take effect immediately and are permanent.
            The -N flag causes an immediate, temporary change. The -I
            flag causes a change that takes effect at the next boot.

Diagnosis: use netstat -s to look for UDP socket buffer overflows.

Tuning    : Increase buffer size until socket buffer overflows ceases.

END-BIOD_COUNT
!
}

####################################################################
#
#  Name       : main
#
#  Overview   : This function invokes the following functions:
#                    DefineVariables
#                    DisplayROOTMenu
#
####################################################################
main ()
```

```
{
DefineVariables
DisplayROOTMenu
}

# invoke main ()
#
main
```

BIBLIOGRAPHY

1 *AIX Performance Tuning* by Frank Waters

Arif Zaman
DBA/System Administrator
High-Tech Software (UK)

© Xephon 1999

DCE for AIX

DCE CONCEPTS

The Distributed Computing Environment from the Open Software Foundation is a standard for client/server computing. It is supported on numerous platforms and by various vendors, and can thus be used to create and administer heterogeneous environments. DCE was first released by OSF in January 1992.

OSF DCE is a complete architecture, offering a set of services and APIs that can be used to build distributed applications. Several management tools are included with DCE software to help administer the distributed environment. DCE is often referred to as middleware, as most network operating system and DCE services are implemented on the operating system in a way that is partially hidden from users. Users see only the distributed client/server application, and it doesn't matter to them whether the application is local or remote, or what the underlying operating system is on which it runs. While its classification as middleware may suggest to some that DCE software cannot exist as a separate product but must be bundled with a vendor’s operating system, this is not the case.

COMPONENT OVERVIEW

DCE implementations comprise various components that work closely together. The components are:

- Cell Directory Service (CDS) and Global Directory Services (GDS), often referred to collectively as just ‘directory services’.
- The Remote Procedure Call (RPC) service.
- The DEC security service.
- DCE threads.
- Distributed File Service (DFS).
- Distributed Time Service (DTS).

These components are described below.

While threads, RPC, DCS, security, and DTS comprise the ‘secure core’ and are required for any DCE installation, DFS is an optional component.

Of course, we need software tools in order to administer these features. Such tools are delivered with the DCE implementation you are purchasing, such as IBM Directory and Security Server for AIX.

ADVANTAGES AND IMPLICATIONS

There are two benefits of DCE: it provides services that are not included with operating systems and it enhances the OS’s standard services. An excellent example is security services, which provides a reliable way of determining whether a user of a distributed system should be allowed to perform a certain action.

DCE integrates different services in a manner that makes them more valuable as a whole than they are separately.

Portability and interoperability are both supported by providing the developer with the ability to hide differences between the various hardware, software, and networking elements. In this way the application can deal with a large network.

The fact that the user doesn't have to know where an application resides and under which operating system or platform it's running is one of the most significant benefits of DCE environments. This allows you to handle heterogeneous networks easily, allowing the user to concentrate on his or her application.

DCE also offers a great deal to software developers, who can now write platform-independent applications. What they need are the common operating system and communication components that are included with the DCE software. DCE threads can be mapped directly to operating system threads or, if the underlying operating system does not support threads, DCE threads are handled in user mode. Another example is DCE Remote Procedure Calls – while RPC exists in several operating systems, DCE extends it to a platform-independent mechanism.

DCE THREADS

Threads are small chunks of code that can be scheduled for execution by the CPU. In fact, threads are the smallest pieces of code that can be scheduled for processing. One process (one program) has at least one thread (if it has only one thread, it's a single-threaded program), but many modern operating systems allow it to contain multiple threads. Threads allow the creation, management, and (of necessity) the synchronization of multiple concurrent execution paths within a single program. The DCE threads API is based on the POSIX 1003.4 Draft 4. As mentioned before, POSIX threads can be mapped directly to operating system threads, where available (Windows for Workgroups, for instance, does not support threads).

DCE core services and all applications that depend on it use threads. The use of multiple threads of control is enabled by RPC. Each client request can be handled by a separate thread. While threads are invisible to the user, programmers must be aware that they use threads through DCE RPC runtime services.

© 1999. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

DCE REMOTE PROCEDURE CALLS

This service allows programmers to call procedures on remote systems as if they were local. Based on the client/server model, it allows programmers to extend the local procedure call to a distributed environment. This allows applications to be ported to diverse architectures and operating systems without needing to be rewritten.

The DCE implementation of Remote Procedure Calls allows them to be used in multi-platform environments. While each operating system uses its own RPC mechanism, DCE extends this to heterogeneous environments. DCE RPC standards have been adopted by X/Open, COSE, and ISIO.

DCE DIRECTORY SERVICE

Directory services is certainly the best-known DCE service. It facilitates system and network management, and its reputation goes beyond the existing installed base of DCE systems.

Large, distributed systems comprise users, processors (PCs, servers, etc), printers, data sources (sometimes containing vast amounts of data), and other resources, which may be geographically dispersed. Directory services provides a transparent and easy-to-use means of addressing information to network resources. This means that users can identify any resource using the resource's name without having to know where the resource is located. Users can continue referring to the resource by the same name, even when the resource's network address changes.

The DCE Directory Service is a distributed and replicated service. The database information is stored in different locations (different servers). The service can replicate the information to make it more readily available, and this has the added benefit that redundant data enhances data security and, hence, the reliability of the entire system.

We have to deal with two different directory services. The first, Cell Directory Service (CDS), manages resources within a cell. The second, Global Directory Service (GDS), provides user access to servers in outside cells. GDS is based on the CCITT X.500/ISO 9594 international standard, and it's positioned so it can participate in a worldwide X.500 directory service. GDS works closely with the Domain Name Service (DNS).

So what is a cell? A cell is the administration unit used by directory services. As it groups machines, users, and resources, a cell can be compared to a domain. However, bear in mind that a cell can contain several IP domains.

DISTRIBUTED FILE SERVICE (DFS)

DFS should be mentioned alongside directory services. The Distributed File Service is responsible for enabling access to resources located within the network environment. Users do not need to know where specific information is physically stored, they need to know only the name of the resource.

Files are part of a single ‘name space’, and therefore each file can be retrieved using its unique name.

DCE DFS includes a physical file system, the DCE Local File System (LFS), that provides special features useful in a distributed environment. The LFS replicates data, logs file system data, and simplifies administration by dividing the file system into smaller units called ‘filesets’. In order to control user access, the LFS supports Access Control Lists (ACLs).

With DCE for AIX, the DFS may be used to export an AIX CD-ROM filesystem from a DFS File Server. This filesystem can be mounted in the DFS file space and then accessed from DFS client machines.

DCE DISTRIBUTED TIME SERVICE

The DTS provides precise, fault-tolerant clock synchronization on computers participating in a distributed environment over LANs and WANs.

Within the confines of a single system, one clock, the system clock, provides time information to all applications. However, a distributed system contains numerous nodes, and each node has its own clock. It is extremely difficult, if not impossible, to set all these clocks to

© 1999. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

exactly the same time (even if you succeeded in doing so, the clocks would drift away from this consistent time at different rates). Inconsistent measurements of time may be a problem for applications that need to order or synchronize events.

Synchronized clocks enable DCE applications to determine event sequencing, duration, and scheduling. DTS synchronizes a DCE host's time with Universal Coordinated Time (UTC), which is an international time standard.

Another component is a complete set of DCE DTS APIs, which are offered as the Time Provider Interface (TPI). These interfaces allow a time-provider process to pass its UTC time values to a DTS server. UTC is communicated by radio, telephone, and satellite.

DCE SECURITY SERVICE

Security is one of the most important concerns of electronic data processing. IBM's AIX has numerous security features; the use of DCE enhances these security features and adds a reliable method of identifying and certifying objects, thus allowing the identification of users, clients, systems, and servers. The DCE Security Service offers integrity and privacy of communications and enables controlled access to resources.

Different security breaches may come from the components of a distributed system. These include: eavesdropping (reading data as it is transmitted over the network), unauthorized modification of data, masquerading to benefit from another user's access rights, and denial of service (legitimate users are denied access to services by an unauthorized authority).

DCE adds security by implementing tried and tested Kerberos technology. You have to install a security server that maintains trusted key information. Clients and servers do not store this information. The security server is responsible for authentication and provides information about the privileges associated with each object.

Kerberos itself is available from MIT for free. However, DCE is not only an authentication framework, like Kerberos, but also a complete security framework. Its architecture enforces a discretionary security policy, for which the Access Control Lists (ACLs) mentioned above play a substantial role.

DCE embeds privacy and authentication into the RPC communication facility and provides a strong security framework to developers of DCE applications. When using Kerberos without DCE, the programmer must take care of client/server communication and the encryption, and also has to decide whether authentication is required only once or for each event.

The security service can be replicated to replication partners that hold a complete registry database. Only the master security server is updated; in turn the update will filter through to all so-called slave servers.

When a user is authenticated he or she gets a ticket that contains the user's security profile. DCE cells function only when the security service is available to obtain tickets. The maintenance of a large number of replication servers increases the availability of the cell security service and provides a kind of load balancing as well. On the other hand, many security servers can lead to a reduction in the system's throughput and create some network overhead. Running many security servers also results in a higher risk of your security system being cracked.

ORGANIZING DCE NETWORKS

Within the Distributed Computing Environment, all resources are administered together. These resources are systems, services, users, printers, nodes, etc.

The Cell is an administrative unit that comprises a collection of resources that use a common naming and security policy. A cell must include at least DCE RPC, threads, and at least one instance of each core service, such as the Cell Directory Server, the Security Server, and Distributed Time Server (Time Servers are actually optional, though recommended).

The current OSF DCE reference implementation runs over TCP/IP.

© 1999. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

Apart from IP, which is used by all services, both TCP and UDP can be used as transport layer protocols.

DCE IMPLEMENTATION ON AIX

The IBM RISC System/6000 running AIX versions 4.1.5 and 4.2 is often referred to as a reference platform for DCE. The DCE product name for AIX is IBM Directory and Security Server for AIX (DSS for AIX).

The current DCE implementation for AIX is based on OSF DCE 1.1 and provides complete client and server support for all DCE services.

The second part of this article will describe the installation and configuration of DCE for AIX.

Klaus Ebner
Curriculum Manager
IBM Austria E+T (Austria)

© Xephon 1999

Systems monitoring shell script

Whether you have a single AIX system to monitor or wish to monitor hundreds of systems, the need to monitor systems on a real-time basis is vital. In order to accommodate the need for real-time systems monitoring, I developed the shell script presented in this article.

It was our desire to actively monitor our AIX servers and be notified of any errors or discrepancies that were considered noteworthy. The following shell script monitors many aspects of the server and sends error messages to user IDs defined in the *Mail()* function. Note that the user IDs specified can be changed to send alerts to alphanumeric pagers as long as the pager can display pages that originate from Internet e-mail.

The functions in the shell script are each preceded by a comment section that explains the purpose of each function. Further explanation is provided at the end of the article.

CHECKITALL

```
#!/bin/ksh
######################################################################
#                                                                    #
# Script  : /usr/local/commands/checkitall                           #
#                                                                    #
# Created : 07/01/98                                                 #
#                                                                    #
# Author  : Jarrod Brown                                             #
#                                                                    #
# Purpose : Automated script called by crontab on a periodic basis. #
#           The script monitors system thresholds and sends an      #
#           alphanumeric page and e-mail in the event a pre-defined #
#           threshold has been reached.                              #
#                                                                    #
######################################################################
LOGFILE=/tmp/checkitall.log
HOST=`hostname`

######################################################################
#
# Thresholds for alert management
#
#   These values are used to establish thresholds for notifying
#   the system administrator; for example, when paging reaches
#   75%, if the root filesystem reaches 90% utilization, if there
#   are fewer than two active SNA sessions, etc.
#
######################################################################
PAGE_MAX=75
ROOT_MAX=90
VAR_MAX=90
TMP_MAX=90
HOME_MAX=90
U01_MAX=90
CPU_MAX=5
UPTIME_MAX=35
ACTIVESNA_MIN=2

######################################################################
#
# Cycle Times
#
#   The following variables determine how long to wait before
#   paging again for the same error. This script is called every
#   five minutes from the root crontab. The following variables
#   are compared to a series of counters. Each time the error
#   is encountered, the counter is incremented. When the variable
#   reaches the LIMIT threshold specified below, the system sends
#   a page. A value of four indicates that the system will not
#   page more than every 25 minutes for any error, as four times
#   five plus five is twenty five.
#
#   Value       Interval
#     11         1 Hour        (((11*5)+5)/60)= 1 hour
#     71         6 Hours       (((71*5)+5)/60)= 6 hours
#    287        24 Hours       (((287*5)+5)/60)= 24 hours
#    863         3 Days        (((863*5)+5)/60)= 72 hours = 3 Days
#
######################################################################
STALELIMIT=11
PAGELIMIT=5
ROOTLIMIT=71
VARLIMIT=71
TMPLIMIT=71
HOMELIMIT=71
U01LIMIT=71
CPULIMIT=5
DB2LIMIT=5
SNALIMIT=5
OFFDISKLIMIT=287
PV2VGLIMIT=287
PROCLIMIT=5
QDAEMONLIMIT=5
UPTIMELIMIT=287
ERRORLOGLIMIT=287
ACTIVESNALIMIT=5
QUORUMLIMIT=5
MIRRORSAMEPVLIMIT=863
MIRRORCNTPVLIMIT=863
MIRRORPVOFFLIMIT=863
DEFAULTCHECKLIMIT=47

######################################################################
#
# Log_Time()
#
#   Log_Time initializes the log file specified by the LOGFILE
#   variable each time the script is called from crontab.
#
######################################################################
LogTime()
{
  echo "***********************************************************"
  ➤  >> $LOGFILE
  date  >> $LOGFILE
  echo "Beginning check of system..."  >> $LOGFILE
  GOOD=0
}

######################################################################
#
# Read_Counters()
#
#   Read_Counters references counters in /tmp/Counters used for
#   alert management. This helps prevent mail being sent each
#   time the script is called from crontab. If no error is
#   encountered, the specific counter is set to zero. If an error
#   is encountered, the counter is incremented until it reaches
#   LIMIT, at which point it is reset to 0. The statement:
#
#     if [ -r /tmp/Counters ]
#
#   checks for the existence of the file /tmp/Counters. If the
#   file does not exist, all counters are set to zero.
#
######################################################################
Read_Counters()
{
  if [ -r /tmp/Counters ]
    then
      STALECNT=`cat /tmp/Counters | grep STALECNT | awk '{print $2}'`
      PAGECNT=`cat /tmp/Counters | grep PAGECNT | awk '{print $2}'`
      ROOTCNT=`cat /tmp/Counters | grep ROOTCNT | awk '{print $2}'`
      VARCNT=`cat /tmp/Counters | grep VARCNT | awk '{print $2}'`
      TMPCNT=`cat /tmp/Counters | grep TMPCNT | awk '{print $2}'`
      HOMECNT=`cat /tmp/Counters | grep HOMECNT | awk '{print $2}'`
      U01CNT=`cat /tmp/Counters | grep U01CNT | awk '{print $2}'`
      CPUCNT=`cat /tmp/Counters | grep CPUCNT | awk '{print $2}'`
      DB2CNT=`cat /tmp/Counters | grep DB2CNT | awk '{print $2}'`
      SNACNT=`cat /tmp/Counters | grep SNACNT | awk '{print $2}'`
      OFFDISKCNT=`cat /tmp/Counters | grep OFFDISKCNT | awk
      ➤  '{print $2}'`
      PV2VGCNT=`cat /tmp/Counters | grep PV2VGCNT | awk '{print $2}'`
      PROCCNT=`cat /tmp/Counters | grep PROCCNT | awk '{print $2}'`
      QDAEMONCNT=`cat /tmp/Counters | grep QDAEMONCNT | awk
      ➤  '{print $2}'`
      UPTIMECNT=`cat /tmp/Counters | grep UPTIMECNT | awk '{print $2}'`
      ERRORLOGCNT=`cat /tmp/Counters | grep ERRORLOGCNT | awk
      ➤  '{print $2}'`
      ACTIVESNA_CNT=`cat /tmp/Counters | grep ACTIVESNA_CNT | awk
      ➤  '{print $2}'`
      QUORUMCNT=`cat /tmp/Counters | grep QUORUMCNT | awk '{print $2}'`
      MIRRORSAMEPVCNT=`cat /tmp/Counters|grep MIRRORSAMEPVCNT|awk
      ➤  '{print $2}'`
      MIRRORCNTPVCNT=`cat /tmp/Counters|grep MIRRORCNTPVCNT|awk
      ➤  '{print $2}'`
      MIRRORPVOFFCNT=`cat /tmp/Counters|grep MIRRORPVOFFCNT|awk
      ➤  '{print $2}'`
      DEFAULTCHECKCNT=`cat /tmp/Counters|grep DEFAULTCHECKCNT|awk
      ➤  '{print $2}'`
    else
      STALECNT=0
      PAGECNT=0
      ROOTCNT=0
      VARCNT=0
      TMPCNT=0
      HOMECNT=0
      U01CNT=0
      CPUCNT=0
      DB2CNT=0
      SNACNT=0
      OFFDISKCNT=0
      PV2VGCNT=0
      PROCCNT=0
      QDAEMONCNT=0
      UPTIMECNT=0
      ERRORLOGCNT=0
      ACTIVESNA_CNT=0
      QUORUMCNT=0
      MIRRORSAMEPVCNT=0
      MIRRORCNTPVCNT=0
      MIRRORPVOFFCNT=0
      DEFAULTCHECKCNT=0
  fi
}

######################################################################
#
# Stale()
#
#   Stale() checks all physical volumes defined to the system
#   for stale partitions. If stale partitions are found and the
#   STALECNT variable read from /tmp/Counters is zero, an alert
#   is sent.
#
######################################################################
Stale()
{
  typeset -i STALECNT=$STALECNT
  for s in `lspv | awk '{print $1}'`
  do
    BADPPs=`lspv $s | grep STALE | awk '{print $3}'`
    if [ $BADPPs != 0 ]
      then
```

```
        echo "There are $BADPPs stale partitions on $s."  >> $LOGFILE
        GOOD=1
        if [ $STALECNT = 0 ]
          then
            echo "Mailing system administration..."  >> $LOGFILE
            STALECNT=$STALECNT+1
            Mail A $BADPPs $s
          else
            if [ $STALECNT -ge $STALELIMIT ]
              then
                STALECNT=0
              else
                STALECNT=$STALECNT+1
            fi
        fi
      else
        STALECNT=0
    fi
  done
  echo "STALECNT $STALECNT"  > /tmp/Counters
}

#####################################################################
#
# Paging()
#
#   Paging() checks paging space in use. If the amount of paging
#   space exceeds PAGE_MAX, and PAGECNT (read from /tmp/Counters)
#   is zero, an alert is sent.
#
#####################################################################
Paging()
{
  typeset -i PAGECNT=$PAGECNT
  PAGE=`lsps -s | tail -1 | awk '{print $2}' | awk -F% '{print $1}'`
  if [ $PAGE -ge $PAGE_MAX ]
    then
      echo "Paging space has reaced $PAGE% utilization."  >> $LOGFILE
      GOOD=1
      if [ $PAGECNT = 0 ]
        then
          echo "Mailing system administration..."  >> $LOGFILE
          PAGECNT=$PAGECNT+1
          Mail B $PAGE
        else
          if [ $PAGECNT -ge $PAGELIMIT ]
            then
              PAGECNT=0
            else
              PAGECNT=$PAGECNT+1
          fi
        fi
      else
        PAGECNT=0
    fi
    echo "PAGECNT $PAGECNT"  >> /tmp/Counters
}

######################################################################
#
# Root_Check()
#
#   Root_Check() checks the amount of space used by the root
#   filesystem. If this exceeds MAX, and the counter is set to
#   zero, an alert is sent.
#
######################################################################
Root_Check()
{
    typeset -i ROOTCNT=$ROOTCNT
    d="/dev/hd4 (root)"
    USED=`df /dev/hd4|grep -v Filesystem|awk '{print $4}'|awk -F%
    ➤  '{print $1}'`
    if [ $USED -ge $ROOT_MAX ]
      then
        echo "$d has reached $USED% utilization."  >> $LOGFILE
        GOOD=1
        if [ $ROOTCNT = 0 ]
          then
            echo "Mailing system administration..."  >> $LOGFILE
            ROOTCNT=$ROOTCNT+1
            Mail C $d $USED
          else
            if [ $ROOTCNT -ge $ROOTLIMIT ]
              then
                ROOTCNT=0
              else
                ROOTCNT=$ROOTCNT+1
            fi
        fi
      else
        ROOTCNT=0
    fi
    echo "ROOTCNT $ROOTCNT"  >> /tmp/Counters
}

######################################################################
#
# Var_Check()
#
```

```
#   Var_Check() checks the amount of space used by the var
#   filesystem. If this exceeds MAX, and the counter is set to
#   zero, an alert is sent.
#
#########################################################################
Var_Check()
{
  typeset -i VARCNT=$VARCNT
  d="/dev/hd9var (var)"
  USED=`df /dev/hd9var|grep -v Filesystem|awk '{print $4}'|awk -F%
  ➤  '{print $1}'`
  if [ $USED -ge $VAR_MAX ]
    then
      echo "$d has reached $USED% utilization."  >> $LOGFILE
      GOOD=1
      if [ $VARCNT = 0 ]
        then
          echo "Mailing system administration..."  >> $LOGFILE
          VARCNT=$VARCNT+1
          Mail C $d $USED
        else
          if [ $VARCNT -ge $VARLIMIT ]
            then
              VARCNT=0
            else
              VARCNT=$VARCNT+1
          fi
      fi
    else
      VARCNT=0
  fi
  echo "VARCNT $VARCNT"  >> /tmp/Counters
}

#########################################################################
#
# Tmp_Check()
#
#   Tmp_Check() checks the amount of space used by the tmp
#   filesystem. If this exceeds MAX, and the counter is set to
#   zero, an alert is sent.
#
#########################################################################
Tmp_Check()
{
  typeset -i TMPCNT=$TMPCNT
  d="/dev/hd3 (tmp)"
  USED=`df /dev/hd3|grep -v Filesystem|awk '{print $4}'|awk -F%
  ➤  '{print $1}'`
  if [ $USED -ge $TMP_MAX ]
```

```
    then
      echo "$d has reached $USED% utilization."  >> $LOGFILE
      GOOD=1
      if [ $TMPCNT = 0 ]
        then
          echo "Mailing system administration..."  >> $LOGFILE
          TMPCNT=$TMPCNT+1
          Mail C $d $USED
        else
          if [ $TMPCNT -ge $TMPLIMIT ]
            then
              TMPCNT=0
            else
              TMPCNT=$TMPCNT+1
          fi
      fi
    else
      TMPCNT=0
  fi
  echo "TMPCNT $TMPCNT"  >> /tmp/Counters
}

#####################################################################
#
# Home_Check()
#
#   Home_Check() checks the amount of space used by the home
#   filesystem. If this exceeds MAX, and the counter is set to
#   zero, an alert is sent.
#
#####################################################################
```

This article concludes in next month's issue of AIX Update, with the remainder of the code for ***checkitall****.*

Jarrod Brown
AIX Programmer/Administrator (USA) © Xephon 1999

Xephon has just launched four weekly e-mail news services covering the following subject areas: data centre, distributed systems, networks, and software. To subscribe to one or more of these news services, or review recent articles, point your browser to http://www.xephon.com/newz.html.

inetd – implementing new TCP/IP services

AIX’s **inetd** daemon handles most common TCP/IP services, such as **telnet**, **ftp**, and **finger**. Roughly speaking, the daemon’s process listens to the proper TCP or UDP port for each of these services and starts the executable file that implements the requested service when a client tries to connect.

The **inetd** demon is, however, more interesting than this, as it is possible with only a small amount of work to make it handle user-implemented TCP or UDP services as well. In this article I’ll show you what needs to be done in order to do just this.

THE CODE FOR THE SERVICE

The only code you need to write is that needed by the core service. In other words, you don’t need to handle such complementary tasks as socket creation or management. The only thing you have to know is how to make your code communicate with the client through the standard input and standard output file descriptors.

As an example, let’s write the C code for a sample (and simple) TCP service. I’ll name it **genstr**: when a client connects to this service, it receives a fixed number of lines in return filled with a variable number of instances of the character ‘a’.

GENSTR.C

```
/*
**
** genstr.c
**
** 'genstr' service
**
*/

/*
 *-------------------------------------------------------------------
 * include section
 *-------------------------------------------------------------------
 */
```

© 1999. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
#include <stdlib.h>
#include <string.h>

/*
 *---------------------------------------------------------------------
 * define section
 *---------------------------------------------------------------------
 */
#define EXIT_OK    0           /* correct exit code              */
#define EXIT_KO    1           /* incorrect exit code            */

/*
 *---------------------------------------------------------------------
 * main - Code for 'genstr' service
 *---------------------------------------------------------------------
 */
int
main (int argc, char* argv[])
{
  char        msg[80];                    /* message buffer         */
  int         genlines;                   /* generated lines        */
  int         i, j;                       /* auxiliary variables    */

  switch ( argc )
  {
    case 1:
      genlines = 100;
      break;
    case 2:
      genlines=atoi(argv[1]);
      break;
    default:
      exit(EXIT_KO);
      break;
  }

  for ( i=0; i<genlines; i++ )
  {
    (void) memset(msg, 0x0, 80);
    j = (int)(rand()/500);
    (void) memset(msg, 'a', j);
    j = strlen(msg);
    sprintf(msg+j, " [%d,%d]\n\r", i, j);
    j = strlen(msg);
    (void) write (0, msg, j);
  }

  exit(EXIT_OK);

} /* main */
```

HOW TO CONFIGURE THE INETD SERVER

When you've finished writing the code for your new service and you have produced the executable file from this code, you must configure your system to allow the **inetd** server to handle the new service.

First of all, you must edit the */etc/services* file so it contains at least one entry with the name of the new service. If your service is called 'genstr', and uses TCP port '12000/TCP', your */etc/services* file must contain the following line:

```
genstr          12000/TCP
```

Next you must configure the **inetd** server. **inetd** configuration is possible either by editing a text file (the default is */etc/inetd.conf*) or when starting the server itself. Each line in */etc/inetd.conf* has the following format (the two lines below are just one line in the file):

```
<service name> <socket type> <protocol> <wait/nowait> <user>
<server program> <arguments>
```

Note that the only optional field is the last one and that fields must be separated by at least a blank or tab. Let's now look at each of these fields in more detail:

- *service name* is the name of the new service (eg **telnet** or **ftp**). It must be the same as the name declared in */etc/services*.
- *socket type* is the socket type you want to be used for the new service. The most common allowed values are *stream* and *dgram*. If you specify the first of them, communication is by means of a sequenced two-way byte stream; otherwise it is by means of datagrams, which are connectionless messages of a fixed maximum length. You should specify *stream* if you're using the TCP protocol and *dgram* if you're using UDP.
- *protocol* is the transport protocol you're using. The two allowed values are *TCP* and *UDP*. (The values are fairly self-explanatory.) Remember that UDP isn't a reliable protocol, so you're not guaranteed that a packet you've sent is received by the destination address.
- *wait/nowait* is the 'wait status' of the **inetd** server. If the value *wait* is specified, **inetd** won't accept any new connection requests

© 1999. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

to the new service until the last incoming request is completely satisfied. Otherwise, if the value *nowait* is specified, the server will accept new incoming requests (in this case the service implemented is said to be ‘concurrent’).

- *user* is the login ID under which the service program should be run.
- *server program* is the full path name of the executable file that implements the service.
- *arguments* is the list of arguments to be passed to the server program when **inetd** executes it. Note that the first argument must be the executable file name without the complete path and that you cannot use more than five arguments.

In our case, the line we must add in the configuration file for **inetd** is the following:

```
genstr stream tcp nowait root /home/bin/genstr.exe genstr.exe
9000.
```

When you’ve finished editing the configuration file */etc/inetd.conf* you must apply the change by sending a *SIGHUP* signal to the **inetd** server using, for example, the **kill** shell command.

HOW TO TEST THE NEW SERVICE?

To test the new service you can use any suitable **telnet** client. For example, from an NT workstation you could submit the following command line:

```
telnet myhost 12000
```

Where *myhost* is the host name of the AIX server that is running the **inetd** daemon just configured.

Marco Pirini
System Administrator (Italy)

© Xephon 1999

AIX news

IBM has announced MQSeries Release 5.1 for AIX and other platforms. New features manage servers to maximize resource utilization and provide automatic recovery. New publish/subscribe facilities allow MQSeries to act as an information distribution service, allowing users to share information across the enterprise. Available now, prices start at US$3,000.

IBM has also announced its next-generation SP POWER3 node, promising double the processing power of existing nodes at the same price. The new nodes are aimed at high-end data mining business applications, as well as traditional technical applications.

For further information contact your local IBM representative.

* * *

Candle has announced that its Roma suite of application integration products now supports RS/6000 clusters. The suite includes an integration broker, an IDE, and system management software. Candle also announced the Candle Command Center (CCC) for High-Availability Systems. This is to be marketed as part of a bundle that includes IBM RS/6000 hardware, AIX 4.3.2, and third-party ERP software.

For further information contact:
Candle Corp, 2425 Olympic Blvd, Santa Monica, CA 90404, USA
Tel: +1 310 829 5800
Fax: +1 310 582 4287
Web: http://www.candle.com

Candle, 1 Archipelago, Lyon Way, Frimley, Camberley, Surrey GU16 5ER, UK
Tel: +44 1276 414700
Fax: +44 1276 414777

* * *

Inprise has announced Entera Version 4.2, an RPC-based middleware product that offers CORBA support via the company's own DCE-CORBA bridge, also featuring a new Internet firewall and improved support for Java. It's out now on AIX, Solaris, HP-UX, and NT. Prices weren't announced.

For further information contact:
Inprise, 100 Enterprise Way, Scotts Valley, CA 95066, USA
Tel: +1 831 431 1000
Web: http://www.inprise.com

Inprise, 8 Pavilions, Ruscombe Business Park, Twyford, Berks RG10 9NN, UK
Tel: +44 1734 320022
Fax: +44 1734 320017

* * *

IBM's 64-bit AIX 4.3.1 has been certified at the C2 security level. C2-certified systems conform with a range of security requirements, such as enforcing a discretionary access control policy. Note that C2 certification merely guarantees that, if suitably configured, the system is capable of providing secure discretionary access, etc – it doesn't assure the system's ability to resist hacking attacks, as NT has amply proven since gaining C2 certification eighteen months ago.