60

AIX

October 2000

update

In this issue

Published by

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

North American office

Xephon/QNA
Post Office Box 350100, Westminster CO 80035-0100, USA
Telephone: (303) 410 9344
Fax: (303) 438 0290

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 £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 1995 issue, are available separately to subscribers for £16.00 ($23.00) each including postage.

AIX Update on-line

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

Printed in England.

IBM StorageWatch Serial Storage Expert

IBM StorageWatch Serial Storage Expert (StorX) is a PC-based utility for managing and monitoring SSA storage. SSA (Serial Storage Architecture) disks have been widely used within the RS/6000 community since the mid-1990s. These disks have many advantages over SCSI disks, including better performance, availability, and expandability. Despite fibre channel-based storage becoming more popular – verging on becoming the standard – SSA storage nevertheless remains widely deployed within IBM's customer base.

The management and monitoring of SSA storage is one major issue that has to be addressed by any SSA user. The ability to connect up to 96 disks in two separate loops to a single SSA I/O card, along with the ability to use multiple cards in a single computer and to connect disk loops to different computers, creates complex storage networking topologies. The standard menu-based tools supplied by IBM to manage SSA disks are unable to display the real topology of the storage network. IBM's StorageWatch Serial Storage Expert is a tool that was created to address just this need.

STORX SYSTEM REQUIREMENTS AND INSTALLATION

StorX is a client/server application, with clients running on either RS/6000s or NT systems connected to SSA devices, plus a PC-based graphical 'manager' tool. The manager communicates with clients using remote procedure calls (RPC) over a TCP/IP communication network. Consequently, RPC and TCP/IP must be enabled and active when StorX is used. Along with the manager software, which is called 'Live Viewer', a separate application – 'Planner' – is bundled with the product.

Planner is an unsophisticated graphical editor that can be used to plan SSA networks using boxes, text annotations, and icons that represent adapters, disks, and connections. Planner can read network maps created by Live Viewer, but the converse isn't true, so that maps created by the Planner cannot be read by Live Viewer.

The following are the minimum requirements to install StorX:

- An Intel-based PC with a 166 MHz or faster processor.
- At least 24 MB of RAM (64 MB recommended).
- At least 25 MB of available disk space.
- A VGA or better display (SVGA is recommended).
- Microsoft Windows 95, 98, or Windows NT 4.0.

The RS/60000 systems to be monitored must meet the following requirements:

1 At least one SSA card must be installed.

2 The OS level must be AIX Version 4.1.x (APAR 71759), AIX 4.2.*x* (APAR 71809), or AIX4.3.

3 Installed device drivers must be at their latest update level:

  – *devices.mca.8f97.com* (this applies to all AIX 4.1.*x* and AIX 4.2.*x* systems, but only MCA-based AIX 4.3 systems)

  – *devices.mca.8f97.diag*

  – *devices.ssa.disk.rte*

  – *devices.pci.1404500.diag* (PCI systems only)

  – *devices.pci.1404500.rte* (PCI systems only)

  – *devices.ssa.IBM_raid.rte*

  – *devices.ssa.disk.rte*

  – *devices.ssa.tm.rte* (for target-mode applications only).

4 SSA network agent (the client component of StorX):

  – *devices.ssa.network_agent.rte*.

Both StorX and the network agent software can be downloaded from the following Web site:

`http://www.storage.ibm.com/software/storwatch/storx.`

At time of writing, the links pointing to actual product locations were

empty. Should this still be the case when you try the site, you can download the latest version from:

```
ftp://index.storsys.ibm.com/sorwatch/storx/fixes.
```

Each ‘fixes’ file contains the complete version of the product.

You can evaluate the software for 60 days. After that, you must license the software by installing a special licence key that you can purchase from your local IBM representative. A licence is not required for the Planner component of the software.

WORKING WITH THE STORX

First let’s define some basic terms used by the software:

- *Host* is the computer connected to the SSA device in question and communicating with StorX using the SSA storage network agent.
- *Storage network* is the configuration of adapters, disks, and cables that enable a host computer or set of host computers to access data. The connections between devices determine the boundaries of an individual storage network. Two devices are considered to be in the same storage network if the devices are connected together.
- *Management set* is the description of all objects, such as hosts, disks, cables, adapters, and storage networks, known to StorX.

Figure 1 shows the display that appears when you start StorX.

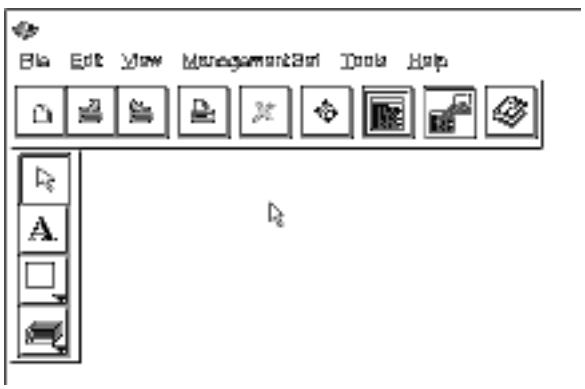

Figure 1: StorX’s GUI

The display conforms with conventions that are now commonplace among applications, and it contains the following features:

- A menu bar that lets you select all StorX's functions.
- A toolbar to access frequently used functions.
- A parts palette containing icons that let you add different elements to the management set, such as text, boxes, and enclosures.
- A canvas, where StorX displays the management set in graphical form.
- An information area where status information and on-line help information for various StorX functions are displayed.

In order to display your current storage network, you should perform a discover query on the specific host. This operation consists of the steps listed on page 7.

Icon	State	Meaning
None	Good	The device is available and fully operational.
Tick in a green circle	Changed	A device attribute has changed.
Green and yellow circle	Attention	The device is not fully operational.
Diagonal line across a connector	Off-line	The device is not available.
Question mark in circle	Suspect	The device was in a 'missing' or 'broken' state but is now in a 'good' state.
Jagged line in a red circle	Broken	The device has reported a failure.
Red X across a device's icon	Missing	The device was removed or the host attached to it was powered off.

Figure 2: State of SSA devices

1 Select the *New* icon in the toolbar (StorX opens the *Management Set Hosts* window).

2 Type the fully-qualified domain name or its TCP/IP address in the *Add New Host* box. (You can add more than one host.)

3 Click the *Add* button. StorX checks that the host exists and has a functional SSA network agent. If these conditions are met, the host name is added to list of hosts in the *Management Set* box.

4 When all hosts that you want to monitor have been added, click the *OK* button. StorX then 'discovers' the storage network components and displays them on the canvas.

StorX displays SSA controller cards and disks that are discovered. Functioning connections between devices in the management set are displayed as solid lines, while 'missing' connections are shown as broken lines. StorX marks icons representing the various management set elements according to their state. The table in Figure 2 summarizes these marks and the conditions they represent.

Figure 3: A management set

Figure 3 (page 7) shows part of a StorX canvas that contains a typical storage management set. The diagram shows a fully occupied 7133 storage enclosure with 16 disks connected to two SSA adapters in a single H70 server. Adapters are labeled with the IP address of the computer in which they're installed, while disks are labeled with their serial number, disk type, and size. The information displayed for each component can be customized using the *Management Set Properties* dialogue box, which can be accessed from the *File* menu.

One of the main problems with the uncustomized canvas is that, for large and complex configurations, it can be difficult to see all the devices easily. By adding the 7133 enclosure icon, which can be obtained from the parts palette, to the canvas using a simple drag and drop operation, and dragging the disks into the enclosure, the size of the canvas can be reduced. Figure 4 shows the customized StorX live view, using the enclosure icon for same configuration as in the previous example.

Figure 4: A view of a management set after customization

You can split the display of the enclosure into quadrants. This feature is useful when each quadrant is attached to a different controller. Figure 5 shows our sample configuration broken into quadrants.

Figure 5: A management set shown in four quadrants

It is possible to further customize the view by adding descriptive text to the canvas. You may add as much text as required to suit your needs. In addition, you can draw colored boxes to emphasize the various logical relationships between the physical disks displayed on the canvas.

EVENT MONITOR

When StorX performs a discover operation, the Event Monitor window opens to display a list of event log entries that occurred while StorX was running. StorX saves and restores the event log entries as part of the management set. To find a device on the canvas that is associated with a specific event in the Event Monitor window, click on the event in the Event Monitor window to highlight it, then click the *Actions* button in Event Monitor menu bar. A pull-down menu appears, from which you select *Find*. This causes the device associated with the event to be highlighted.

STORX EVENT FORWARDING

You can use StorX to notify your enterprise management software, such as Tivoli, Unicenter, or IT/O, automatically when various management set-related events occur. This function uses the Simple Network Management Protocol (SNMP), which is a widely used industry-standard management protocol. A pre-requisite is that you must have Microsoft Simple Network Management Protocol installed and active on the computer that is hosting the StorX manager.

To start or stop SNMP event forwarding, select *Tools* on the menu bar, then *Event Forwarding*. This is a toggle switch that enables or disables SNMP event forwarding.

USING STORX FOR DOCUMENTATION

You can use StorX to produce useful documentation describing your management sets. One convenient option is to produce an image file of the management set in GIF or BMP format. The following steps will produce such an image:

1 Select the *File* menu from menu bar.

2 Select the *Export* option.

3 Select the file format (BMP or GIF) for the exported image.

4 Type in the name of the exported image file.

Another way to obtain documentation is to print the contents of the canvas, which will include all storage networks in the displayed management set. If a management set is too big to fit on a single page, StorX formats it into page segments so it can be printed and reassembled. Each printed page contains the following information that aids the user in re-assembling the canvas:

- The management set name
- Page *x* of *y*
- Print date, time, and management set file location
- Row *x*, column *y*.

The row and column designations describe the relationship of each printed page to other printed pages in the management set.

You can preview the page layout by enabling the display of page boundaries before printing the canvas. This lets you to make adjustments to avoid page splits that intersect device icons when the management set is printed. To print the canvas, select the *File* menu, then *Print* and *Canvas*.

StorX aids you in documenting and labeling the connections in your storage network by providing the ability to print labels that help you to:

- Connect host adapters to enclosures so they match your storage network plan.
- Identify the source and target of a connection if there is a problem with the storage network.

For each connection, StorX prints two labels – one for the source and one for the target of the connection. The same information is printed on each half of the label, allowing you to attach the label to a cable by folding it in half over the cable. This lets you view the cabling information on each side of the folded label. StorX prints a unique identifier on each of a pair of labels, making it easy to identify the other half of a pair. The contents of the labels are:

- Enclosure, location, and port number (if the connection is from or to a device in an enclosure).
- Unique ID for the device, serial number, and port number (if the connection is from or to a device that is not in an enclosure).
- Host name, adapter name, slot number, and port number (if the connection is from or to an adapter).

You should use labels with a format of two columns and 10 rows on each sheet. To print labels, select the *Print* option from the *File* menu, then select *Labels*.

You can print images of the enclosures in your management set. The enclosures you specify in the *Enclosure View Preferences* window determine the labels that are printed with the enclosure. StorX prints

one enclosure on each page. Empty slots in the enclosure are left blank, slots with blank devices are printed shaded, and slots hosting devices have the device icon printed in the slot with the device labels that you specified in the *Disk View Preferences* dialogue box. To print the enclosures in your management set, select the *Print* option from the *File* menu , then select *Enclosures.*

StorX allows you to export management set data for use in another application, such as a database or spreadsheet. Two kinds of report can be produced: a 'hosts' report and a management set topology report. Host reports provide general information describing the hosts contained in the management set and the devices that are connected to them, such as:

- List of hosts
- List of devices (adapters, physical disks, logical disks, and enclosures) connected to the host
- Mapping between logical and physical disks.

A topology report provides detailed information describing the devices that are contained in the management set, including the following:

- List of hosts
- List of enclosures
- Enclosure slot description – enclosure id, enclosure slot number, slot device id, device type (disk, blank, empty, etc)
- Storage network devices
- Storage network connections – from ID, from type (disk, adapter, enclosure), from port, to ID, to type (disk, adapter, enclosure), to port, connection attributes
- Host-to-adapter mappings.

To create a report, select the *File* menu, then *Report*, and the report type that you want (*Hosts* or *Topology*), and then enter the name of report file (StorX will supply default extension *.rpt*). The report is a text file with the pipe character ('|') as the field delimiter.

REFERENCES

1 *Understanding SSA Subsystems in Your Environment*, SG42-1599.

2 *Monitoring and Managing IBM SSA Disk Subsystems*, SG24-5251.

3 *IBM StorWatch Serial Storage Expert User's Guide*, SC26-7267-02

4 IBM Announcement Letters at *http://www.ibmlink.ibm.com*.

Alex Polak
System Engineer
APS (Israel)

A tape testing utility

Tape test (**tt.sh**) is a shell script that tests both a magnetic tape and a tape drive. The process of writing to magnetic tape and reading from it can fail for various reasons; this shell script prepares a small test file, writes the file to tape, and reads it back from tape, comparing the contents with the expected result to establish whether the tape and drive are functioning correctly.

TT.SH

```
#############################################################################
# Name     : tt.sh (tape test)
#
# Overview : This script tests both a tape device and tape by
#            writing a small test file to the tape and reading it
#            back from the tape.
#
# Notes    : 1. The script contains the following functions:
#               - main
#               - InitializeVariables
#               - PrepareTestFile
```

```
#               - PrepareListFile
#               - WriteToTape
#               - ViewLogFile
#               - ReadFromTape
#               - DisplayMessage
#               - LogMessage
#               - MoveCursor
#               - HandleInterrupt
#               - ProcessExit
################################################################################
################################################################################
# Name     : InitializeVariables
#
# Overview : The function initializes all variables.
################################################################################
InitializeVariables ()
{
#
# define temporary files
#
TEST_FILE="/tmp/tt_$$.dat"
OLD_TEST_FILE="/tmp/tt_$$.dat.old"
TEMP_FILE="/tmp/tt_$$.tmp"
LIST_FILE="/tmp/tt_$$.lst"
LOG_FILE="/tmp/tt_$$.log"
DATETIME=`date "+%d/%m/%Y at %H:%M:%S"`
#
# escape sequences
#
ESC = "\0033["
RVON = "\0033[7m"           # reverse video on
R = "\0033[7m"              # reverse video on
RVOFF = "\0033[27m"         # reverse video off
N = "\0033[27m"             # reverse video off
BOLDON = "\0033[1m"         # bold on
BOLDOFF = "\0033[22m"       # bold off
BON = "\0033[5m"            # blinking on
BOFF = "\0033[25m"          # blinking off
#
# define message prefixes
#
INFO = "tt.sh:INFO:"
ERROR = "tt.sh:ERROR:"
#
# define return codes
#
FALSE = 1
TRUE = 0
#
# define exit codes
```



```
#
SEC = 0
FEC = 1
#
# duration for sleep command
#
SLEEP_DURATION = 3
#
# define signals
#
SIGHUP = 1    ;  export SIGHUP     # when session disconnected
SIGINT = 2    ;  export SIGINT     # ctrl-c
SIGTERM = 15  ;  export SIGTERM
SIGTSTP = 18  ;  export SIGTERM    # ctrl-z command
#
#  message
#
WORKING = "Working"
INTERRUPT = "Program interrupted - quitting"
VIEW_LOG_FILE = "View log file"
REWINDING_TAPE = "Rewinding the tape"
REWIND_FAILED = "Failed to rewind the tape"
INIT_TEST_FILE = "Initialize test file"
WRITING_TO_TAPE = "Writing test file to tape"
READING_FROM_TAPE = "Reading from tape"
TAPE_WRITE_FAILED = "Failed to write test file to tape"
TAPE_READ_FAILED = "Failed to read test file from tape"
READING_FAILED = "Failed to read the test file from tape"
TAPE_TEST_OK = "Tape test succeeded"
TAPE_TEST_NOT_OK = "Tape test failed"
OSERROR = "\${ERR_MSG}"
INTERRUPT = "Program interrupted - quitting"
#
# initialize log file
#
echo "Log for tape test on ${DATETIME}" > ${LOG_FILE}
echo "================================" >> ${LOG_FILE}
}
##############################################################################
# Name     : HandleInterrupt
#
# Overview : The function calls ProcessExit.
##############################################################################
HandleInterrupt ()
{
DisplayMessage I "${INTERRUPT}"  N
LogMessage I "${INTERRUPT}"
ProcessExit $FEC
}
##############################################################################
```

```
# Name     : MoveCursor
#
# Input    : Y and X coordinates
#
# Returns  : None
#
# Overview : Moves the cursor to the required location (Y, X).
##############################################################################
MoveCursor  ()
{
YCOR = $1
XCOR = $2
echo "${ESC}${YCOR};${XCOR}H"
}
##############################################################################
# Name     : DisplayMessage
#
# Overview : The function displays the incoming message.
#
# Input    : 1. Message type (E = Error, I = Information)
#            2. Error Code
#            3. Message to be acknowledged flag  (Y = yes, N = no)
#
# Notes    : 1. If the message is to be acknowledged, the function
#               displays the message and waits for an input as
#               acknowledgement. If the message is not to be
#               acknowledged, the function appends three dots ('...')
#               to the end of the message to indicate that the
#               message does not need to be acknowledged.
##############################################################################
DisplayMessage ()
{
trap "HandleInterrupt " $SIGINT $SIGTERM $SIGHUP $SIGTSTP
MESSAGE_TYPE = $1
MESSAGE_TEXT = "`eval echo $2`"
ACKNOWLEDGE_FLAG = "$3"
#
# default the message acknowledge flag
#
if ["${ACKNOWLEDGE_FLAG}" = ""]
then
    ACKNOWLEDGE_FLAG = "Y"
fi
#
#
MoveCursor 24 1
if ["${MESSAGE_TYPE}" = "E"]
then
    if ["${ACKNOWLEDGE_FLAG}" = "N"]
    then
```

```
        echo "`eval echo ${RVON}${ERROR}`${MESSAGE_TEXT}...${RVOFF}\c"
    else
        echo "`eval echo ${RVON}${ERROR}`${MESSAGE_TEXT}${RVOFF}\c"
    fi
else
    if ["${ACKNOWLEDGE_FLAG}" = "N"]
    then
        echo "`eval echo ${RVON}${INFO}`${MESSAGE_TEXT}...${RVOFF}\c"
    else
        echo "`eval echo ${RVON}${INFO}`${MESSAGE_TEXT}${RVOFF}\c"
    fi
fi
#
# examine message acknowledge flag
#
if ["${ACKNOWLEDGE_FLAG}" = "Y"]
then
    read DUMMY
else
    sleep ${SLEEP_DURATION}
fi
return ${TRUE}
}
##############################################################################
# Name     : LogMessage
#
# Overview : The function writes the incoming message to a log file.
#
# Input    : 1. Message type (E = Error, I = Information)
#            2. Error Code
##############################################################################
LogMessage ()
{
trap "HandleInterrupt " $SIGINT $SIGTERM $SIGHUP $SIGTSTP
MESSAGE_TYPE = $1
MESSAGE_TEXT = "`eval echo $2`"
#
if ["${MESSAGE_TYPE}" = "E"]
then
    echo "${ERROR}${MESSAGE_TEXT}" >> ${LOG_FILE}
else
    echo "${INFO}${MESSAGE_TEXT}" >> ${LOG_FILE}
fi
}
##############################################################################
# Name     : PrepareTestFile
#
# Overview : The function initializes the test file that's to be
#            written to tape.
##############################################################################
```

```
PrepareTestFile ()
{
DisplayMessage I "${INIT_TEST_FILE}" N
echo "Test file contents" >> ${LOG_FILE}
echo "==================" >> ${LOG_FILE}
echo "Testing tape on ${DATETIME}." > ${TEST_FILE}
cat ${TEST_FILE} >> ${LOG_FILE}
echo "\n" >> ${LOG_FILE}
}
##############################################################################
# Name     : PrepareListFile
#
# Overview : The function initializes the list that includes the
#            name of the test file.
##############################################################################
PrepareListFile ()
{
echo "${TEST_FILE}" > ${LIST_FILE}
}
##############################################################################
# Name     : WriteToTape
#
# Overview : The function puts a copy of the test file on the tape.
#
# Returns  : $TRUE or
#            $FALSE
##############################################################################
WriteToTape ()
{
DisplayMessage I "${REWINDING_TAPE}" N
LogMessage I "${REWINDING_TAPE}"
(mt -f /dev/rmt0 rewind > ${TEMP_FILE} 2>&1)
RC = $?
if [$RC -ne 0]
then
    DisplayMessage I "${REWIND_FAILED}" N
    LogMessage E "${REWIND_FAILED}"
    ERR_MSG = `cat ${TEMP_FILE}`
    DisplayMessage I "${OSERROR}" N
    LogMessage E "${OSERROR}"
    return $FALSE
fi
DisplayMessage I "${WRITING_TO_TAPE}" N
LogMessage I "${WRITING_TO_TAPE}"
cat ${LIST_FILE} | cpio -ocBv 2>> ${TEMP_FILE} 1> /dev/rmt0
RC = $?
if [$RC -ne 0]
then
    DisplayMessage E "${TAPE_WRTTE_FAILED}" N
    LogMessage E "${TAPE_WRITE_FAILED}"
```

```
    ERR_MSG = `cat ${TEMP_FILE}`
    DisplayMessage E "${OSERROR}" N
    LogMessage E "${OSERROR}"
    return $FALSE
else
    return $TRUE
fi
}
################################################################################
# Name     : ReadFromTape
#
# Overview : The function reads the test file from tape.
################################################################################
ReadFromTape ()
{
DisplayMessage I "${READING_FROM_TAPE}" N
LogMessage I "${READING_FROM_TAPE}"
mv ${TEST_FILE} ${OLD_TEST_FILE}
cpio -iumcBv ${TEST_FILE} < /dev/rmt0 > ${TEMP_FILE} 2>&1
if [$? -ne 0]
then
    DisplayMessage E "${TAPE_READ_FAILED}" N
    LogMessage E "${TAPE_READ_FAILED}"
    ERR_MSG = `cat ${TEMP_FILE}`
    DisplayMessage E "${OSERROR}" N
    LogMessage E "${OSERROR}"
    return $FALSE
fi
echo "\nRestored test file contents" >> ${LOG_FILE}
echo "============================" >> ${LOG_FILE}
cat ${TEST_FILE} >> ${LOG_FILE}
}
################################################################################
# Name     : ViewLogFile
#
# Overview : The function allows the user to view the log file.
################################################################################
ViewLogFile ()
{
DisplayMessage I "${VIEW_LOG_FILE}" N
view ${LOG_FILE}
}
################################################################################
# Name     : ProcessExit
#
# Overview : The function implements a graceful exit.
#
# Input    : Exit code
################################################################################
ProcessExit ()
```

```
{
#
# assign parameter
#
P_EXIT_CODE = "$1"
#
rm -f ${TEST_FILE}
rm -f ${OLD_TEST_FILE}
rm -f ${LIST_FILE}
rm -f ${LOG_FILE}
rm -f ${TEMP_FILE}
exit ${P_EXIT_CODE}
}
################################################################################
# Name     : main
#
# Overview : The function implements the processing structure.
#
# Notes    : 1. The function calls the following functions:
#                - InitializeVariables
#                - HandleInterrupt
#                - DisplayMessage
#                - PrepareTestFile
#                - PrepareListFile
#                - WriteToTape
#                - ViewLogFile
#                - ProcessExit $FEC
#                - ReadFromTape
################################################################################
main ()
{
InitializeVariables
trap "HandleInterrupt " $SIGINT $SIGTERM $SIGHUP
DisplayMessage I "${WORKING}" N
PrepareTestFile
PrepareListFile
if !  WriteToTape
then
    ViewLogFile
    DisplayMessage E "${TAPE_TEST_NOT_OK}" N
    LogMessage E "${TAPE_TEST_NOT_OK}"
    ProcessExit $FEC
fi
if !  ReadFromTape
then
    DisplayMessage E "${TAPE_TEST_NOT_OK}" N
    LogMessage E "${TAPE_TEST_NOT_OK}"
    ViewLogFile
```

```
    ProcessExit $FEC
fi
#
# compare files
#
diff ${TEST_FILE} ${OLD_TEST_FILE} > /dev/null 2>&1
if [$? -eq 0]
then
    DisplayMessage I "${TAPE_TEST_OK}" N
    echo "\n" >> ${LOG_FILE}
    LogMessage I "${TAPE_TEST_OK}"
    ViewLogFile
else
    DisplayMessage E "${TAPE_TEST_NOT_OK}" N
    echo "\n" >> ${LOG_FILE}
    LogMessage E "${TAPE_TEST_NOT_OK}"
    ViewLogFile
fi
ProcessExit $SEC
}
#
# invoke main
#
main
```

SAMPLE LOG FILE

```
Log for tape test on 08/05/2000 at 18:33:47
============================================
Test file contents
==================
Testing tape on 08/05/2000 at 18:33:47.

tt.sh:INFO:Rewinding the tape
tt.sh:INFO:Writing test file to tape
tt.sh:INFO:Reading from tape

Restored test file contents
===============================
Testing tape on 08/05/2000 at 18:33:47.

tt.sh:INFO:Tape test succeeded
```

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

Integrating AIX's DNS and Windows 2000 domains

You need DNS to run your AIX environment. In most cases, this means running a BIND server, which comes as standard on just about all Unix platforms. So your existing network environment, including IBM AIX, probably already has operational BIND DNS servers.

Windows 2000 is another operating system that needs DNS. In contrast with its predecessor, Windows NT, Windows 2000 uses native TCP/IP, as Unix does, and needs a working DNS server in order to create domains and domain trees. The entire Active Directory relies on DNS namespace. If the DNS server doesn't work with Windows 2000 because it doesn't understand Windows 2000 requests, you won't even be able to install a domain, which is the basic entity of Windows 2000 networks.

What about mixed environments that have AIX as well as Windows 2000 systems? Do such environments require you to migrate or phase out DNS servers that have always worked perfectly? Of course not. It is neither necessary to replace these servers nor run DNS on more than one platform. It's even a good idea to leave your AIX DNS servers in place and integrate Windows 2000 with them.

That being said, you have to audit your DNS servers and you may need to upgrade their software to make them compatible with Windows 2000. Microsoft lists the following requirements for DNS servers on platforms other than Windows 2000 to interoperate with Windows 2000:

- *RFC 2052bis* – Service Location Resource Records (SRV RRs)
- *RFC 2136* – Dynamic Update Protocol
- *RFC 1995* – Incremental Zone Transfers.

Only Service Location Resource Records are a must – your server has to support them in order to work with Windows 2000. If your DNS server doesn't support SRV RRs, you need to upgrade it and, in some cases, you may even need another DNS server, which could be Microsoft DNS Server on Windows 2000 Server, as this is bundled with the operating system.

It is not necessary to support the Dynamic Update Protocol, though this eases administration by relieving you of the responsibility of keeping the DNS database up-to-date manually. This protocol is particularly helpful if you use DHCP.

If your DNS server does not support dynamic updates, you need to implement the following configuration:

- All clients use static IP addresses.
- IP addresses and hostnames are entered manually in the DNS server's database.

Security is a good argument for using static DNS servers, as only a manual configuration allows you to control which computers can be located on the network using DNS.

The least important requirement of the three listed above is *RFC 1995*, which concerns incremental zone transfers. You need at least two DNS servers on different computers to use this feature – the DNS servers are configured as *primary* and *secondary* servers for the same zone and exchange zone files automatically.

When DNS servers carry out a zone transfer, they usually copy the entire zone, so a considerable amount of data may flow over the network. The bigger your network, the more important it is to manage this data flow. Incremental zone transfers optimize your network as the servers copy only those parts of the database that have changed. Incremental zone transfers save time and resources.

If you have only one DNS server running under AIX, no zone transfers take place and you don't need support for *RFC 1995*. However, if you have more than one server, incremental zone transfers can reduce network traffic. If your server doesn't support this function, the entire zone is copied each time a transfer takes place.

Once you have this basic information about your environment, you may begin to configure the DNS network using the following steps:

1 If necessary, install a DNS server on an AIX system. The DNS server should support the RFCs listed by Microsoft, so BIND 8.2 or later is required.

2 Configure a DNS zone that is to become a Windows 2000 domain. The domain name has to exist in DNS before Windows 2000 adopts it and merges it into the Active Directory structure.

3 Activate dynamic updates within the zone.

4 Install Windows 2000 server. When configuring TCP/IP, provide the IP address of the AIX system that hosts the DNS service.

5 Start **dcpromo** from a Windows command prompt and define the new domain, tree, and forest. Use the same domain name that you defined as a zone or domain name on the DNS server.

6 Restart Windows 2000. After restart, the domain is active. There must be no error messages at start-up and logon.

7 You may test the environment and DNS name resolution using **ping** and **nslookup** from the Windows 2000 computer. Note, however, that it make no sense to **ping** your own machine, as names are first compared with the host's own name, so **ping**ing the host itself never requires DNS.

If your DNS server meets the stated requirements and you follow these steps, your network should work perfectly.

UNIX BIND

Use BIND version 8.1.2 or later, as this supports Server Location Resource Records and the Dynamic Update Protocol. This version also supports DNS Change Notification (*RFC 1996*), which is not supported by Windows 2000. Microsoft actually recommends BIND 8.2.1 for Windows 2000 integration, as it supports Negative Caching (*RFC 2308*) and DNS Security (*RFC 2535*). If you'd like more information on BIND Version 8, check the AIX on-line manuals (search for entries on **named8** and **nsupdate8**, as entries without the '8' suffix usually refer to BIND v4).

We tested BIND 8.2 on IBM AIX 4.3.3.0.02. If you overwrite an old version of BIND, you usually have to change the links as follows:

```
ln -fs /usr/sbin/named8 /usr/sbin/named
ln -fs /usr/sbin/named8-xfer /usr/sbin/named-xfer
ln -fs /usr/sbin/nsupdate8 /usr/sbin/nsupdate
```

BIND Version 8 can be started using **named**, though this could result in a mixed Version 4 and Version 8 environment if you haven't updated all links correctly.

If you have an old version installed, you may create a new configuration file, *named.conf*, from the existing *named.boot*. Use the following command to do this:

```
/usr/samples/tcpip/named-bootconf.pl named.boot > named.conf
```

You can open the configuration file with any editor, including **vi**. Entries for a fictitious *europe.com* domain may look like this:

```
// BIND Configuration File
options {
   directory "/usr/local/named";
   notify yes;
};
zone "europe.com" in {
   type master;
   file "db.europe";
   check-names ignore;
   allow-transfer { 10.1.1.100; };
   allow-update {any;};
};
zone "0.0.10.in-addr.arpa" in {
   type master;
   file "db.10.0.0";
   allow-transfer { 10.1.1.100; };
   allow-update {any;};
};
zone "0.0.127.in-addr.arpa" in {
   type master;
   file "db.127.0.0";
};
zone "." in {
   type hint;
   file "db.cache";
};
```

The entry 'notify yes' makes BIND notify all secondary servers if the zone file changes. If you don't activate this option, changed zones are transferred only at regular intervals. Another important entry is 'check-names ignore' – Active Directory adds numerous names to the DNS database that are not DNS names; these entries are required by Active Directory, so you should guarantee their registration using 'check-names ignore'.

It's a good idea to restrict dynamic updates to specific machines. You may, for example, allow your DHCP server to initiate dynamic updates but prevent clients from doing so. Note, however, that restricting dynamic updates does not improve the security of your network and data – a client's address and pointer entries may still be overwritten if a DHCP server carries out the update on behalf of a client. By contrast, Microsoft DNS Server on Windows 2000 decides whether a requested update is legitimate before carrying it out.

INSTALLATION AND ACTIVATION

As part of our test, we installed BIND 8.2 on AIX. We had to define the Windows 2000 domain name as a zone (domain) manually on the DNS server – Windows 2000 is not capable of registering its own domain name with a DNS server running on AIX. If you try, you get an error message stating that Windows 2000 cannot reach the DNS server that is authoritative for the zone, and the domain cannot be established under these conditions. The configuration comprises the zone file's file name and an authorization for entering dynamic updates. Note that there's no point in either including comments in the file or formatting it, as Windows 2000 automatically removes comments and reformats the file when it writes dynamic updates.

We then entered the DNS server's IP address in Windows 2000's TCP/IP configuration and started **dcpromo**, the Active Directory Wizard. Here we defined the Windows 2000 Server as a domain controller for the new domain and created a new tree and forest, which are the structural components of Active Directory. The existing DNS domain name was chosen as Windows 2000's domain name. Windows 2000 accepted this and installed the Active Directory. In the meantime we could observe that Windows 2000 had registered its hostname on AIX. The hostname and IP address were first written to cache and then to the configuration text file.

Windows 2000 behaved exactly as it would if the DNS server were just another Windows 2000 system. After restarting, Windows 2000 was fully functional, used DNS name resolution, and we could see the new name registrations on the AIX server.

The installation of additional domain controllers and member servers

in the Window 2000 domain is also unchanged. The servers need the DNS server's IP address, just as the first domain controller did. Of course, you may define the DNS server via DHCP, in which case you have to define the server's IP address only once using the appropriate option on the DHCP server.

NEW SUBDOMAINS

After the first test, we demoted one of the Windows 2000 domain controllers to a member server and started **dcpromo** once again. We reinstalled Active Directory, creating a new subdomain, which belongs to the same tree and forest as the first domain.

This time we didn't need to create the new domain name manually on the AIX DNS server. Windows 2000 registered the new DNS subdomain and its new domain controller automatically with **named8**. No problems occurred and computers in both domains used DNS name resolution.

Therefore, only the initial Windows 2000 domain has to be created manually using a BIND zone file – subsequent subdomains are created automatically.

Klaus Ebner MCSE + PSE
Advisory Education Specialist (Austria)

AIX printing on MVS AFP printers (part 2)

This month's instalment concludes this article on printing from AIX systems on AFP printers.

MVS PROGRAM DISPATCH

```
         TITLE ' UNIXSRV WORK SCHEDULER '
         ENTRY DISPATCH
DISPATCH RMODE 24
DISPATCH AMODE 31
DISPATCH CSECT
```

```
         BAKR  R14,0
         LR    R10,R15
         USING DISPATCH,R10
         LR    R11,R1                  Save parameter list
*
         GETMAIN RU,LV=TCBPARML
         LR    R12,R1
         USING TCBPARM,R12
*
         LA    R9,CONVRPL              Establish addressability
         USING IFGRPL,R9                  to RPL
         LA    R8,CONVRPL6             Establish addressability
         USING ISTRPL6X,R8                to RPL Extension
         L     R2,0(R11)
         MVC   CONVRPL(LEN_RPL),0(R2)
         L     R2,4(R11)
         MVC   CONVRPL6(LEN_RPL6),0(R2)
         L     R2,8(R11)
         MVC   REQCOUNT(2),0(R2)
*
         APPCCMD RPL=CONVRPL,AAREA=CONVRPL6,                           X
               CONTROL=RCVFMH5,AREA=CONVFMH5,AREALEN=L'CONVFMH5,       X
               CONMODE=CS,                                             X
               OPTCD=SYN
         LTR   R0,R0
         BNZ   PROCEXIT
         LTR   R15,R15
         BNZ   PROCEXIT
         LA    R5,CONVFMH5       * GET FMH-5 ADDRESS
         USING ISTFM5,R5         * ESTABLISH ADDRESSABILITY
         TM    FM5FLAG2,FM5PIPPR
         BNO   NOPIP
         APPCCMD RPL=CONVRPL,AAREA=CONVRPL6,                           X
               CONTROL=RECEIVE,FILL=LL,                                X
               QUALIFY=SPEC,                                           X
               AREA=CONVAREA,AREALEN=L'CONVAREA,                       X
               OPTCD=SYN
         LTR   R0,R0
         BNZ   RECVERR
         LTR   R15,R15
         BNZ   RECVERR
         LA    R2,CONVAREA        * SET ADDRESS OF START OF PIP DATA
         USING FM5PIPFM,R2        * ESTABLISH ADDRESSABILITY
         LA    R3,4(,R2)          * SET ADDRESS OF PIP DATA SUBFIELD
         USING FM5PIPSM,R3        * ESTABLISH ADDRESSABILITY
*
         MVC   PARM(80),FM5PIPSD
NOPIP    DS    0H
         SR    R2,R2
         ICM   R2,B'1',FM5LNTPN
```

```
         BCTR  R2,0
         LA    R1,FM5TPNAM
         MVC   PGM,=CL8' '
         EX    R2,MVTPNAME
         WTO   'UNIXSRV- SCHEDULING WORK'
         XC    TCBECB,TCBECB
         MVC   TRANSACT(L'TRANSACT),PGM
         LR    R1,R12
         ATTACHX ESTAI=(RECOVERY,(R12)),SF=(E,ATTACHX)
         SR    R15,R15
PROCEXIT PR
*
RECVERR  WTO   'UNIXSRV- ERROR IN RECEIVE'
         APPCCMD RPL=CONVRPL,AAREA=CONVRPL6,                          X
               CONTROL=DEALLOC,                                       X
               QUALIFY=ABNDPROG,                                      X
               RECLEN=0,                                              X
               OPTCD=SYN
         LR    R1,R12
         FREEMAIN RU,LV=TCBPARML,A=(1)
         PR
*
MVTPNAME MVC   PGM(0),0(R1)
*
RECOVERY DS    0H
         DROP ,
         BAKR  R14,0
         LR    R9,R15
         USING RECOVERY,R9
         LR    R8,R1
*
         C     0,TESTNOSDWA        test for sdwa present
         BE    NOSDWA              jump to nosdwa if not present
HAVESDWA DS    0H
         USING SDWA,R8
         USING TCBPARM,R12
         L     R12,SDWAPARM                get parm area address
         B     MAIN_RECV
NOSDWA   PR                        * No SDWA present
         LA    R15,0               * Request percolation
         PR                        * bye bye
*
* MAIN RECOVERY ROUTINE
*
MAIN_RECV DS   0H
         ST    R13,SAVERET13
         LA    R13,SAVEAREA
*
         MVC   RECV_WTO(MODLWTOL),MODLWTO
*
```

```
         MVC   REC_MSG_LEN(2),=AL2(L'RECV_MSG1+8)
         MVC   REC_MSG_TXT(L'RECV_MSG1),RECV_MSG1
         MVC   REC_MSG_TXT+L'RECV_MSG1(8),TRANSACT
         WTO   TEXT=REC_MSG,MF=(E,RECV_WTO)
*
         APPCCMD RPL=CONVRPL,AAREA=CONVRPL6,                           X
               CONTROL=DEALLOC,                                        X
               QUALIFY=ABNDPROG,                                       X
               RECLEN=0,                                               X
               OPTCD=SYN
*
         WTO   TEXT=RECV_MSG2,MF=(E,RECV_WTO)
*
         LTR   R8,R8
         BZ    NOSET
*
         LR    R1,R8
         SETRP RC=16
*
NOSET    DS    0H
         LA    R13,SAVERET13
         LR    R1,R12
         FREEMAIN RU,LV=TCBPARML,A=(1)
         LA    R15,16                  * no retry, no percolate
         PR
*
*********************************************************************
*                                                                   *
*        Partner LU information                                     *
*                                                                   *
*********************************************************************
*
         LTORG
MODLWTO  WTO   TEXT=,MF=L
MODLWTOL  EQU   *-MODLWTO
RECV_MSG1 DC    C'UNIXSRV- ABEND DETECTED FOR TRANSACTION: '
RECV_MSG2 DS  0H
          DC  AL2(RECV_MSG2_LEN-2)
          DC  C'UNIXSRV- LU 6.2 VERB DEALLOCATE TYPE=ABEND_PROG'
          DC  C' ISSUED'
RECV_MSG2_LEN EQU *-RECV_MSG2
*
PGM      DC    CL8' '
TCBECB   DC    F'0'
TESTNOSDWA DC  A(12)
*
ATTACHX  ATTACHX EP=TCBINIT,SF=L
*********************************************************************
* Variable area                                                     *
*********************************************************************
```

```
CONVAREA DS    0XL200        * APPCCMD input buffer area
CONVLL   DS    XL2
CONVTXT  DS    XL200
CONVFMH5 DS    XL255
*************************************************************************
*  Mappings                                                             *
*************************************************************************
         IFGRPL AM=VTAM
         ISTFM5
         ISTSLCNS
SLCLEN   EQU    SLCEND-ISTSLCNS
         ISTUSFBC
         IFGACB AM=VTAM
         IHASDWA                     SYSTEM DIAGNOSTIC WORK AREA
*
         TCBPARM
*************************************************************************
*    Register EQUATES                                                   *
*************************************************************************
         REGS
         END
```

MVS PROGRAM TCBINIT

```
TCBINIT  AMODE 31
TCBINIT  RMODE 24
TCBINIT  CSECT
MAIN     BAKR  R14,0
         LR    R12,R15
         USING TCBINIT,R12
         LR    R10,R1              *  Establish addressability
         USING TCBPARM,R10         *    to the TCB input parameter
*
         STORAGE OBTAIN,LENGTH=VARAREA_LEN,COND=YES,LOC=BELOW
         LR      R11,R1
         USING VARAREA,R11
         MVC   ACRON(7),=CL7'TCBINIT'
*
         LA    R13,SAVEAREA
         MVC   4(4,R13),=C'F1SA'
*************************************************************************
* Variable initialization                                               *
*************************************************************************
         WTO    'UNIXSRV- (TCBINIT) TP SCHEDULED'
         LR    R1,R10
         LA    R2,TRANSACT
         LINK  EPLOC=(R2)
*************************************************************************
* Program exit routine                                                  *
```

```
*********************************************************************
RETURN   DS    OH
         WTO   'UNIXSRV- (TCBINIT) TRANSACTION PROGRAM ENDED'
* Release input parmarea storage
         STORAGE RELEASE,LENGTH=TCBPARML,ADDR=(R12),COND=YES
*
         CLC   0(7,R11),=CL7'TCBINIT'   *Validate private data area
         BNE   MEMERROR
* Release private storage
         STORAGE RELEASE,LENGTH=VARAREA_LEN,ADDR=(R11),COND=YES
         SR    R15,R15
         PR
*
MEMERROR DS    OH
* Isseu message if damaged private area
         WTO   'UNIXSRV- (TCBINIT) PRIVATE DATA AREA DAMAGED,',        C
               ' UNABLE TO FREE MEMORY'
         SR    R15,R15
         PR
*
TCBMEMER DS    OH
* Isseu message if damaged private area
         WTO   'UNIXSRV- (TCBINIT) TCB INPUT PARAMETER AREA DAMAGED',  C
               ', UNABLE TO FREE MEMORY'
         SR    R15,R15
         PR
*
INVPARM  DS    OH
* Isseu message if damaged private area
         WTO   'UNIXSRV- (TCBINIT) INVALIB INPUT PARAMETER AREA'
         SR    R15,R15
         PR
*****************************************************************
* Constants                                                     *
*****************************************************************
         LTORG
*
*********************************************************************
* Dummy sections                                                    *
*********************************************************************
VARAREA  DSECT
ACRON    DS    CL6        * Data area identifier
         DS    0D        * Double word work area for mathematics
WORKAREA DS    CL8
VARAREA_LEN EQU *-VARAREA
*********************************************************************
* Program DSECTS                                                    *
*********************************************************************
         TCBPARM
         IFGRPL AM=VTAM
```

```
         ISTFM5
         ISTSLCNS
SLCLEN   EQU    SLCEND-ISTSLCNS
         ISTUSFBC
         IFGACB AM=VTAM
         REGS
         END
```

MVS PROGRAM WRT2SPL

```
WRT2SPL  AMODE 31
WRT2SPL  RMODE 24
WRT2SPL  CSECT
MAIN     BAKR  R14,0
         LR    R12,R15
         USING WRT2SPL,R12
         LR    R10,R1                  Establish addressability
         USING TCBPARM,R10                 TO MAIN ROUTINE RPL
*
         STORAGE OBTAIN,LENGTH=VARAREA_LEN,COND=YES,LOC=BELOW
         LR      R11,R1
         USING VARAREA,R11
         MVC   ACRON(7),=C'WRT2SPL'
*
         LA    R13,SAVEAREA
         MVC   4(4,R13),=C'F1SA'
*********************************************************************
* Variable initialization                                           *
*********************************************************************
*
         XC    STATUS,STATUS                Clear error flags
*
         MVC   DCBSYSOUT(DCB_LEN),DCB
         MVC   ADDPARML(OUTADD_LEN),OUTADD
         MVC   DELPARML(OUTDEL_LEN),OUTDEL
*
         MVC   SYSOUT(2),=AL2(DALSYSOU)
         MVC   SYSOUT+2(2),=AL2(1)
         MVC   SYSOUT+4(2),=AL2(1)
         MVC   OUTPUT(2),=AL2(DALOUTPT)
         MVC   OUTPUT+2(2),=AL2(1)
         MVC   OUTPUT+4(2),=AL2(8)
         MVC   ODNAME(3),=CL3'OUT'
         MVC   DDNAME(2),=AL2(DALRTDDN)
         MVC   DDNAME+2(2),=AL2(1)

         MVC   DDNAME+4(2),=AL2(8)
         MVC   DISPTXT(2),=AL2(DUNOVDSP)
         MVC   DISPTXT+2(2),=AL2(1)
```

```
         MVC   DISPTXT+4(2),=AL2(1)
         MVC   DISP(1),=X'04'              * purge disp
         MVC   DESTDEF(2),=AL2(DODEST)
         MVC   DESTDEF+2(2),=AL2(1)
         MVC   DESTDEF+4(2),=AL2(8)
         MVC   FORMSDEF(2),=AL2(DOFORMS)
         MVC   FORMSDEF+2(2),=AL2(1)
         MVC   FORMSDEF+4(2),=AL2(8)
         MVC   FORMDEF(2),=AL2(DOFORMD)
         MVC   FORMDEF+2(2),=AL2(1)
         MVC   FORMDEF+4(2),=AL2(6)
         MVC   PAGEDEF,=AL2(DOPAGEDE)
         MVC   PAGEDEF+2(2),=AL2(1)
         MVC   PAGEDEF+4(2),=AL2(6)
         MVC   CHARS(2),=AL2(DOCHARS)
         MVC   CHARS+2(2),=AL2(1)
         MVC   CHARS+4(2),=AL2(4)
*
         MVC   SPOOLTXT(MSG1_LEN),MSG1
         MVC   LINE#TXT(MSG2_LEN),MSG2
*
         WTO    'UNIXSRV- (WRT2SPL) TP SCHEDULED'
*
         LH    R4,REQCOUNT
         CVD   R4,WORKAREA
         UNPK  WORKAREA(5),WORKAREA+4(4)
         OI    WORKAREA+4,X'F0'
         LA    R1,WORKAREA
         MVC   REQNUM(5),WORKAREA
*********************************************************************
* Move input data into the text units
*********************************************************************
         BAL   R14,RECV_DATA        * Receive sysout parameters
         LTR   R15,R15
         BNZ   FAILURE
         TM    WHATRCV,RPL6WDAC
         BC    8,NO_PARMS           * Error if no parms
*
         TM    WHATRCV,RPL6WCFM     * Test if confirm requested
         BC    8,DEALTEST
         BAL   R14,CONFIRM          * Issue confirm
         LTR   R15,R15
         BNZ   FAILURE              * Error on confirm failure
*
DEALTEST DS    0H
         TM    WHATRCV,RPL6WDAL     * Test if dealloc received
         BC    8,BUILD
         WTO   'UNIXSRV- DEALLOC RECEIVED FROM PARTHER LU'
         B     FAILURE
*********************************************************************
```

```
* Build text units based on received parameters                      *
**********************************************************************
BUILD    DS    0H
         LA    R5,CONVTXT+2               * Establish addressability
         USING JESPARMS,R5                * to the received parameter
                                          * list
*
* Convert input parameters to EBCDIC and to caps high
*
         TR    USERDIN(L'USERDIN),CONVTBL     * Convert userid
         OC    USERDIN(L'USERDIN),=CL80' '
         TR    CLASSDIN(L'CLASSDIN),CONVTBL   * Convert class
         OC    CLASSDIN(L'CLASSDIN),=CL80' '
         TR    DESTDIN(L'DESTDIN),CONVTBL     * Convert dest
         OC    DESTDIN(L'DESTDIN),=CL80' '
         TR    FORMSIN(L'FORMSIN),CONVTBL     * Convert forms
         OC    FORMSIN(L'FORMSIN),=CL80' '
         TR    FORMDIN(L'FORMDIN),CONVTBL     * Convert formdef
         OC    FORMDIN(L'FORMDIN),=CL80' '
         TR    PAGEDIN(L'PAGEDIN),CONVTBL     * Convert pagedef
         OC    PAGEDIN(L'PAGEDIN),=CL80' '
         TR    CHARSDIN(L'CHARSDIN),CONVTBL   * Convert chars
         OC    CHARSDIN(L'CHARSDIN),=CL80' '
*
         LH    R1,N_LINES          * Get total number of lines
         ST    R1,TOT_LINE#        * and save it
         MVC   MSGUSERID,USERDIN
         MVC   SPLUSERID,USERDIN
**********************************************************************
* Change TCB userid                                                  *
**********************************************************************
         LA    R1,USERDIN
         CALL  CHUSER
         LTR   R15,R15
         BZ    CHANGEOK
         WTO   'UNIXSRV- (WRT2SPL) UNABLE TO CHANGE TCB USERID'
         B     FAILURE
**********************************************************************
* Alloc spool dataset                                                *
**********************************************************************
CHANGEOK DS    0H
         BAL   R14,ALLOC_SPOOL
         LTR   R15,R15
         BNZ   FAILURE
**********************************************************************
* Receive and write                                                  *
**********************************************************************
         L     R8,MAX#SPOOL               * Load spool line limit
         SR    R1,R1
         ST    R1,LINE#
```

```
LOOP     DS    0H
         BAL   R14,RECV_DATA              * Receive data
         LTR   R15,R15
         BNZ   FAILURE
*********************************************************************
         TM    WHATRCV,RPL6WDAC           * Test for data received
         BC    8,TEST_END
         BAL   R14,WRITE                  * Write into sysout dataset
         LTR   R15,R15
         BNZ   FAILURE
*
         L     R1,LINE#                   * Increment number lines
         LA    R1,1(R1)
         ST    R1,LINE#
         L     R2,TOT_LINE#
         CR    R1,R2                      * Test for overflow
         BH    COUNTERR
*********************************************************************
TEST_END DS    0H
         TM    WHATRCV,RPL6WCFM+RPL6WDAL  * Test dealloc+confirm
         BC    1,RECV_END
*********************************************************************
TESTCONF DS    0H
         TM    WHATRCV,RPL6WCFM     * Test if confirm requested
         BC    8,TESTDEAL
         BAL   R14,CONFIRM          * Send confirm to the partner LU
         LTR   R15,R15
         BNZ   FAILURE
*********************************************************************
TESTDEAL DS    0H
         TM    WHATRCV,RPL6WDAL     * Test for dealloc
         BC    8,TESTMAX
         WTO   'UNIXSRV- (WRT2SPL) DEALLOC RECEIVED FROM PARTNER LU'
         B     FAILURE
*********************************************************************
TESTMAX  DS    0H
         BCT   R8,LOOP
         WTO   'UNIXSRV- (WRT2SPL) MAXIMUM ALLOWED NUMBER OF LINES',   C
               ' WRITEN TO SPOOL REACHED'
         B     FAILURE
*********************************************************************
* Exit routine                                                      *
*********************************************************************
RECV_END DS     0H
         WTO   'UNIXSRV- (WRT2SPL) END DETECTED'
         L     R1,LINE#
         L     R2,TOT_LINE#
         CR    R1,R2          * Validate number of written lines
         BNE   COUNTERR
         B     SUCCESS
```

```
*********************************************************************
* Success routine                                                   *
*********************************************************************
SUCCESS  DS    0H                     * Normal program return
         WTO   'UNIXSRV- (WRT2SPL) FILE WRITTEN WITH SUCCESS'
*
         LA    R2,DCBSYSOUT           * Close sysout file
         CLOSE ((R2))
         NI    STATUS,255-OPEN_OK
*
         OUTDEL NAME=ODNAME,MF=(E,DELPARML)  * Delete output DESC.
         NI    STATUS,255-DESC_OK
*
         MVC   DISP(1),=X'08'          * Set DISP to keep
         BAL   R14,DEALLOC             * Dealloc sysout file
         NI    STATUS,255-ALLOC_OK
*
         WTO   'UNIXSRV- (WRT2SPL) ABOUT TO ISSEU LAST CONFIRM'
         BAL   R14,CONFIRM          * Send final confirm
         LTR   R15,R15
         BNZ   FAILURE
*
         L     R4,LINE#             * Build statistics message
         CVD   R4,WORKAREA
         UNPK  WORKAREA(5),WORKAREA+4(4)
         OI    WORKAREA+4,X'F0'
         LA    R1,WORKAREA
         MVC   NUMLINES(5),WORKAREA
         MVC   MSGCLASS,CLASS
         MVC   MSGDEST,DESTDPRM
         WTO   TEXT=LINE#TXT        * Issue statistics message
         B     RETURN
*********************************************************************
* Program exit routine                                              *
*********************************************************************
RETURN   DS    0H
         TM    STATUS,OPEN_OK       * Test if sysout file if output
         BC    8,NOT_OPEN           * file was opened
         LA    R2,DCBSYSOUT         * If not, close it
         CLOSE ((R2))
NOT_OPEN DS    0H
         TM    STATUS,DESC_OK       * Test if output descriptor
         BC    8,NO_DESC            *    was created
         OUTDEL NAME=ODNAME,MF=(E,DELPARML)  * If not, delete it
*
NO_DESC  DS    0H
         TM    STATUS,ALLOC_OK      * Test if sysout file was
         BC    8,NO_ALLOC           * deallocated
         BAL   R14,DEALLOC          * If not, dealloc
*
```

```
NO_ALLOC DS    0H
         WTO   'UNIXSRV- (WRT2SPL) TRANSACTION PROGRAM ENDED'
*
         CLC   0(7,R11),=CL7'WRT2SPL'   *Validate private data area
         BNE   MEMERROR
* Release private storage
         STORAGE RELEASE,LENGTH=VARAREA_LEN,ADDR=(R11),COND=YES
         SR    R15,R15                RETURN TO CALLER
         PR
*
MEMERROR DS    0H
* Issue message if damaged private area
         WTO   'UNIXSRV- (WRT2SPL) PRIVATE DATA AREA DAMAGED,',        C
               ' UNABLE FREE MEMORY'
         SR    R15,R15
         PR
*********************************************************************
* Write to the sysout dataset the received data                     *
*********************************************************************
WRITE    DS    0H
         BAKR  R14,0
         LA    R1,CONVTXT+2
         TR    0(255,R1),CONVTBL
         LA    R1,255(R1)
         TR    0(255,R1),CONVTBL
         PUT   DCBSYSOUT,CONVTXT+2
         SR    R15,R15
         PR
*
EODAD    DS   0H
         WTO  'UNIXSRV- (WRT2SPL) EXIT EODAD ACTIVADA'
         LA    R15,12
         PR
*********************************************************************
* Error routines                                                    *
*********************************************************************
COUNTERR DS    0H
         WTO   'UNIXSRV- (WRT2SPL) INVALID NUMBER OF LINES RECEIVED !'
         B     FAILURE              * Unsuccessful execution
*
NO_PARMS DS   0H
         WTO  'UNIXSRV- (WRT2SPL) NO PARAMETERS RECEIVED'
         B    FAILURE
*
FAILURE  DS   0H

         WTO  'UNIXSRV- (WRT2SPL) DEALLOC ABEND USER ISSUED'
         APPCCMD RPL=CONVRPL,AAREA=CONVRPL6,                          X
               CONTROL=REJECT,                                        X
               QUALIFY=CONV,                                          X
```

```
               SENSE=X'084C0000',                                      X
               OPTCD=SYN
          B    RETURN
*************************************************************************
* Confirm request                                                       *
*************************************************************************
CONFIRM  DS   OH
         BAKR  R14,0
         APPCCMD RPL=CONVRPL,AAREA=CONVRPL6,                           X
               CONTROL=SEND,                                           X
               QUALIFY=CONFRMD,                                        X
               OPTCD=SYN
         LTR   R15,R15
         BNZ   CONF_ERR
         LTR   R0,R0
         BNZ   CONF_ERR
         SR    R15,R15
         PR
CONF_ERR DS    OH
         WTO   'UNIXSRV- (WRT2SPL) ERROR RECEIVED',                    C
               'DURING A CONFIRM REQUEST'
         LA    R15,12
         PR
*************************************************************************
* Recive data from the parthner lu                                      *
*************************************************************************
RECV_DATA DS  OH
         BAKR  R14,0
         LA    R1,CONVAREA
         XC    CONVAREA(255),CONVAREA
         LA    R1,CONVAREA+255
         XC    0(255,R1),0(R1)
         APPCCMD RPL=CONVRPL,AAREA=CONVRPL6,                           X
               CONTROL=RECEIVE,FILL=LL,                                X
               QUALIFY=SPEC,                                           X
               AREA=CONVAREA,AREALEN=CONVALEN,                         X
               OPTCD=SYN
         LTR   R15,R15
         BNZ   RECV_ERR
         LTR   R0,R0
         BNZ   RECV_ERR
*
         MVC   WHATRCV,(RPL6RCV1-ISTRPL6X)+CONVRPL6
* Test for dealloc, data complete and confirm
         TM    WHATRCV,RPL6WDAL+RPL6WDAC+RPL6WCFM
         BC    8,RECV_ERR
         SR    R15,R15
         PR
RECV_ERR DS    OH
         WTO   'UNIXSRV- (WRT2SPL) UNEXPECTED INDICATOR RECEIVED'
```

```
         LA    R15,16
         PR
*
*********************************************************************
* Dealloc the spool dataset                                         *
*********************************************************************
DEALLOC  DS OH
         BAKR  R14,0
         WTO   'UNIXSRV- (WRT2SPL) ABOUT TO DEALLOC SPOOL DATASET'
*
* INVOKE DYNALLOC TO DEALLOCATE THE SYSOUT DATA SET. CHANGE
* THE DISPOSITION TO KEEP.
*
         LA    R6,SVC99PRM          PTR TO SVC 99 RB PTR
         LA    R7,SVC99PRM+4        PTR TO SVC 99 RB
         USING S99RBP,R6            RB PTR
         USING S99RB,R7             RB
*
* INITIALIZE THE SVC 99 REQUEST BLOCK
*
         MVI   S99VERB,S99VRBUN
         LA    R1,TXTPTRS3         ADR SVC 99 TEXT PTRS
         ST    R1,S99TXTPP         STORED IN RB
*                                  INDICATES LAST TEXT
         LA    R1,DDNAME
         ST    R1,TXTPTRS3
         MVC   DDKEY(2),=AL2(DUNDDNAM)
         LA    R1,DISPTXT
         ST    R1,TXTPTRS3+4
         OI    TXTPTRS3+4,S99TUPLN   HIGH ORDER BIT ON
*                                    INDICATES LAST TEXT
         LR    R1,R6                 UNIT POINTER
*                                    ADR OF RB POINTER
*
* ISSUE DYNALLOC TO DEALLOCATE THE SYSOUT DATA SET
*
         DYNALLOC                   INVOKES SVC99
         LTR   R15,R15
         BNZ   DEAL_ERR
         NI    STATUS,255-ALLOC_OK
*
         SR    R15,R15
         PR
*
DEAL_ERR DS OH
         WTO   'UNIXSRV- ERROR IN DYNAMIC DEALLOCATION'
         LA    R15,16
         PR
*
*********************************************************************
```

```
* Alloc the spool dataset                                              *
************************************************************************
ALLOC_SPOOL DS 0H
         BAKR  R14,0
         LA    R5,CONVTXT+2        * Establish input parameter
         USING JESPARMS,R5         * addressability
*
         LH    R1,REGLEN           * Set spool register length
         STH   R1,SPOOLLEN         * from input parameters
*
         WTO   TEXT=SPOOLTXT       * Issue spool message
*
         LA    R1,GROUPID          * Set sysout group text unit ptr
         ST    R1,TXTPTRS1
         LA    R2,TXTPTRS1
*
         MVI   CLASS,C'X'          * Set default class to 'X'
         CLI   CLASSDIN,C' '       * Test if class specified
         BE    TDEST
         MVC   CLASS,CLASSDIN      * Set sysout class
*
TDEST    MVC   DESTDPRM,=CL18'LOCAL' * Set default dest
         CLI   DESTDIN,C' '          *Test if dest specified
         BE    TFORMS
         MVC   DESTDPRM(18),DESTDIN
*
         LA    R2,4(R2)
         LA    R1,DESTDEF
         ST    R1,0(R2)              * Add dest text unit
*
TFORMS   CLI   FORMSIN,C' '          * Test if form present
         BE    TFORMDEF
         MVC   FORMSPRM,FORMSIN
         LA    R2,4(R2)
         LA    R1,FORMSDEF           * Add forms text unit ptr
         ST    R1,0(R2)
*
TFORMDEF CLI   FORMDIN,C' '          * Test if formdef present
         BE    TPAGEDEF
         MVC   FORMDPRM,FORMDIN
         LA    R2,4(R2)
         LA    R1,FORMDEF            * Add formdef text unit
         ST    R1,0(R2)
*
TPAGEDEF CLI   PAGEDIN,C' '          * Test if pagedef present
         BE    TCHARSDF
         MVC   PAGEDPRM,PAGEDIN
         LA    R2,4(R2)
         LA    R1,PAGEDEF
         ST    R1,0(R2)              * Add pagedef text ptr
```

```
*
TCHARSDF CLI   CHARSDIN,C' '          * Test if chars present
         BE    SETLAST
         MVC   CHARSVAL,CHARSDIN
         LA    R2,4(R2)
         LA    R1,CHARS
         ST    R1,0(R2)               * Add chars text ptr
*
SETLAST  OI    0(R2),X'80'            * Set last text ptr
*
* Issue outadd to create an output descriptor
*
         OUTADD NAME=ODNAME,TEXTPTR=TXTPTRS1,                          C
               MF=(E,ADDPARML)
         LTR   15,15                  * Test return code
         BZ    OUTADDOK
* If error, delete the descriptor and add it again
         OUTDEL NAME=ODNAME,MF=(E,DELPARML)
         OUTADD NAME=ODNAME,TEXTPTR=TXTPTRS1,                          C
               MF=(E,ADDPARML)
         LTR   15,15                  * Test return code
         BNZ   DESC_ERR
OUTADDOK DS    0H
         OI    STATUS,DESC_OK         * Set status flag
*
* Invoke dynalloc to allocate a sysout data set.  specify
* sysout class a and reference the output descriptor
* 'OUT%%%%%'.  set up addressability to the dynamic allocation
* (SVC 99) request block.
*
DYNAL    DS    0H
         LA    R6,SVC99PRM
         LA    R7,SVC99PRM+4
         USING S99RBP,R6
         USING S99RB,R7
         ST    R7,S99RBPTR
         OI    S99RBPTR,S99RBPND
*
* INITIALIZE THE SVC 99 REQUEST BLOCK
*
         XC    S99RB(RBLEN),S99RB  ZERO THE RB
         MVI   S99RBLN,RBLEN       RB LENGTH
         MVI   S99VERB,S99VRBAL    RB VERB CODE=ALLOC
         LA    R1,SPIN
         ST    R1,TXTPTRS2        * Set spin text ptr
         LA    R1,DDNAME
         ST    R1,TXTPTRS2+4      * Set ddname text ptr
         LA    R1,SYSOUT
         ST    R1,TXTPTRS2+8      * Set sysout text ptr
         LA    R1,OUTPUT
```

```
         ST    R1,TXTPTRS2+12     * Set output text ptr
*
         LA    R1,TXTPTRS2        * Addr SVC 99 text ptrs
         ST    R1,S99TXTPP        * store it in the RB
         OI    POINTERC,S99TUPLN  * Set last entrie
*
* Issue dynalloc to allocate a sysout data set that
* references the output descriptor, OUT%%%%%.
*
         LR    R1,R6              * Address RB
         DYNALLOC                 * Invoke SVC99
         LTR   R15,R15
         BNZ   ALLOC_ERROR
         OI    STATUS,ALLOC_OK    * Set status flag
*
         LA    R4,DCBSYSOUT
         USING IHADCB,R4
         MVC   DCBDDNAM,RETDDNAM  * Set sysout DDNAME
         LH    R1,SPOOLLEN
         STH   R1,DCBLRECL        * Set sysout LRECL
         LA    R2,DCBSYSOUT
         OPEN  ((R2),OUTPUT)      * Open sysout file
         LTR   R15,R15
         BNZ   OPEN_ERR
         OI    STATUS,OPEN_OK     * Set status flags
         LA    R15,0
         PR
*
OPEN_ERR DS    OH
         WTO   'UNIXSRV- ERROR OPENING SPOOL DATASET'
         LA    R15,8
         PR
*
DESC_ERR DS    OH
         WTO   'UNIXSRV- ERROR CREATING SYSOUT DESCRIPTOR'
         LA    R15,12
         PR
*
ALLOC_ERROR DS OH
         WTO   'UNIXSRV- ERROR IN DYNAMIC ALLOCATION'
         LA    R15,16
         PR
*********************************************************************
* Program constants                                                 *
*********************************************************************
MSG1      DC    AL2(MSG1_LEN-2)
          DC    C'UNIXSRV- SPOOL ACCESS REQUEST RECEIVED FROM USER '
          DS    CL8
MSG1_LEN  EQU   *-MSG1
*
```

```
MSG2      DC    AL2(MSG2_LEN-2)
          DC    C'UNIXSRV- '
          DS    CL5
          DC    C' LINES WRITTEN TO SPOOL, TO CLASS '
          DS    C
          DC    C' DESTID '
          DS    CL8
          DC    C' FOR USER '
          DS    CL8
MSG2_LEN  EQU   *-MSG2
*****************************************************************
* Constants                                                     *
*****************************************************************
* Constants for dynamic output invocation
*
GROUPID  DS 0H               FORMS DEFINITION TU
         DC AL2(DOGROUPI)    KEY=FORMDEF
         DC XL2'0001'        HOW MANY VAL FIELDS
         DC XL2'0004'        LEN OF VALUE FIELD
         DC CL4'UNIX'        VALUE FIELD (TO BE ALTERED)
*
* Parameter lists for OUTADD and OUTDEL macros
*
OUTADD   OUTADD MF=L
OUTADD_LEN EQU *-OUTADD
OUTDEL   OUTDEL MF=L
OUTDEL_LEN EQU *-OUTDEL
*
* Sysout file DCB area
*
DCB       DCB   DDNAME=*,RECFM=FBA,                                C
               MACRF=(PM),DSORG=PS,EODAD=EODAD
DCB_LEN  EQU   *-DCB
*
* Static text units
*
SPIN     DS 0H
         DC AL2(DALSPIN)
         DC XL2'0001'
         DC XL2'0001'
         DC XL1'80'
*
          DS    0A
MAX#SPOOL DC   AL4(200000)       * Max. allowed number of sysout lines
*
         LTORG
*
* Table used to convert ASCII to EBCDIC (Portuguese code page)
*
```

```
CONVTBL  DC    CL16'                '     HEX 0X
         DC    CL16'                '     HEX 1X
         DC    CL16' !'#$%&&''()*+,-./'   HEX 2X
         DC    CL16'0123456789:;<=>?'     HEX 3X
         DC    CL16'@ABCDEFGHIJKLMNO'     HEX 4X
         DC    CL16'PQRSTUVWXYZ:\#O_'     HEX 5X
         DC    CL16'`abcdefghijklmno'     HEX 6X
         DC    CL16'pqrstuvwxyz{]}~ '     HEX 7X
         DC    CL16'                '     HEX 8X
         DC    CL16'                '     HEX 9X
         DC    CL16'A|[1^2|5,4][^[?ô'     HEX AX
         DC    CL16'&!|:Ô-63)]$.789,'     HEX BX
         DC    CL16'ÀÁÃTX;Ç4ÉÊ3ÌÍ67'     HEX CX
         DC    CL16'%Z_Ó,Õ%' ,Ú#@_>R'     HEX DX
         DC    CL16'àáâãCG*çMéêLìíOP'     HEX EX
         DC    CL16'<I(ó.õ</`)ú$*(+'      HEX FX
*************************************************************************
* Dummy sections                                                        *
*************************************************************************
VARAREA  DSECT
ACRON    DS    CL7              * Data area identifier
*
SPOOLTXT DC    AL2(SPLTEXTL)
         DC    C'UNIXSRV- SPOOL ACCESS REQUEST RECEIVED FROM USER '
SPLUSERID DS   CL8
SPLTEXTL EQU   *-SPOOLTXT-2
*
         DS    0D
LINE#    DS    AL4
TOT_LINE# DS   AL4
*
LINE#TXT DC    AL2(LINE#TXTL)
         DC    C'UNIXSRV- '
NUMLINES DS    CL5
         DC    C' LINES WRITTEN TO SPOOL, TO CLASS '
MSGCLASS DS    C
         DC    C' DESTID '
MSGDEST  DS    CL8
         DC    C' FOR USER '
MSGUSERID DS   CL8
LINE#TXTL EQU  *-LINE#TXT-2
*
* Pointer table used to create sysout descriptor
*
TXTPTRS1 DS 0F              TEXT UNIT POINTERS
         DC AL4(GROUPID)    ADR TU FOR DEST
         DS AL4             * supported text units: DESTID
         DS AL4             *                       FORMS
         DS AL4             *                       FORMDEF
```

```
         DS AL4              *                       PAGEDEF
         DS AL4              *                       CHARS
*
* Pointer table for sysout dynamic allocation
*
TXTPTRS2 DS OF
         DC AL4(SPIN)        * Pointer to SPIN text unit
         DC AL4(DDNAME)      * Pointer to DDNAME text unit
         DC AL4(SYSOUT)      * Pointer to SYSOUT text unit
POINTERC DC AL4(OUTPUT)      * Pointer to OUTPUT text unit
*
* Pointer table for sysout dynamic deallocation
*
TXTPTRS3 DS OF
         DC AL4(DDNAME)      * Pointer to DDNAME text unit
         DC AL4(DISPTXT)     * Pointer to DISPTXT text unit
*
* Text units
*
DDNAME   DS OH
DDKEY    DC AL2(DALRTDDN)
         DC XL2'0001'
         DC XL2'0008'
RETDDNAM DS CL8'      '
*
SYSOUT   DC AL2(DALSYSOU),XL2'0001',XL2'0001'
CLASS    DS CL1
*
DISPTXT  DC AL2(DUNOVDSP),XL2'0001',XL2'0001'
DISP     DC X'08'
*
OUTPUT   DC AL2(DALOUTPT),XL2'0001',XL2'0008'
ODNAME   DS CL3'OUT'
REQNUM   DS CL5
*
DESTDEF  DC AL2(DODEST),XL2'0001',XL2'0012'
DESTDPRM DS CL18
*
FORMSDEF DC AL2(DOFORMS),XL2'0001',XL2'0008'
FORMSPRM DS CL8
*
FORMDEF  DC AL2(DOFORMD),XL2'0001',XL2'0006'
FORMDPRM DS CL6
*
PAGEDEF  DC AL2(DOPAGEDE),XL2'0001',XL2'0006'
PAGEDPRM DS CL6
*
CHARS    DC AL2(DOCHARS),XL2'0001',XL2'0004'
CHARSVAL DS CL4
```

```
*
WHATRCV  DS    C         * What received flags
         DS    0D        * Double word work area for mathematics
WORKAREA DS    CL8
SPOOLLEN DS    AL2       * Sysout register length
*
* Declare space for the RB, which is mapped by IEFZB4D0
*
         DS 0F
SVC99PRM DS    CL(RBLEN+4)       *  RB ptr & RB storage
RBLEN    EQU   (S99RBEND-S99RB)  *  Length of the RB
*
STATUS   DS    X       * Status flags
OPEN_OK  EQU   X'80'   * Sysout file opened
ALLOC_OK EQU   X'40'   * Sysout file allocated
DESC_OK  EQU   X'20'   * Sysout descriptor created
*
DCBSYSOUT DCB   DDNAME=*,RECFM=FBA,                                    C
               MACRF=(PM),DSORG=PS,EODAD=EODAD
ADDPARML OUTADD MF=L
DELPARML OUTDEL MF=L
*
CONVAREA DS    0H          * APPCCMD macro reveive buffer area
CONVLL   DS    XL2
CONVTXT  DS    XL514
CONVALEN EQU   *-CONVAREA
*
VARAREA_LEN EQU *-VARAREA
*********************************************************************
* Program DSECTS                                                    *
*********************************************************************
         DS    0F
         TCBPARM
         JESPARMS
*
         IFGRPL AM=VTAM
         ISTFM5
         ISTSLCNS
SLCLEN   EQU    SLCEND-ISTSLCNS
         ISTUSFBC
         IFGACB AM=VTAM
         IEFZB4D0                  MAP SVC 99 REQUEST BLOCK
         IEFZB4D2                  MAP SVC 99 KEY TABLE
         IEFDOKEY                  DEFINES DOBURST, ETC
         DCBD  DSORG=PO,DEVD=DA    DCB symbolic definitions
         REGS
         END
```

MVS PROGRAM CHUSER

```
CHUSER   AMODE 24
CHUSER   RMODE 24
CHUSER   CSECT
MAIN     BAKR  R14,0
         LR    R10,R15
         USING CHUSER,R10
         LR    R12,R1                  Establish addressability
         USING INPARM,R12                 to MAIN routine RPL
*
         LA      R0,VARAREA_LEN
         STORAGE OBTAIN,                                           X
               LENGTH=(0),COND=YES,LOC=BELOW
         LR      R11,R1
         USING VARAREA,R11
*                           * Initialize dynamic variable area
         LA    R13,SAVEAREA
         MVC   4(4,R13),=C'F1SA'
*
         MVC   USERTXT(MSG3_LEN),MSG3
         MVC  ACEE_DEL(M_ACEE_DEL_L),M_ACEE_DEL
         MVC  ACEE_CRE(M_ACEE_CRE_L),M_ACEE_CRE
*
         MVC   SPLUSERID,USERDIN
         WTO   TEXT=USERTXT
         WTO   'UNIXSRV- DELETING ACEE'
*        AUTH ON
         MODESET KEY=ZERO
         LA     R2,RACSAFE
        RACROUTE REQUEST=VERIFY,                                   C
               ENVIR=DELETE,                                       C
               WORKA=(R2),                                         C
               MF=(E,ACEE_DEL),RELEASE=1.9.2
         USING SAFP,R9
         LA    R9,ACEE_DEL
*
         LTR   R15,R15
         BNZ   CHUSER_FAIL
*
         L     R15,SAFPRRET
         LTR   R15,R15
         BNZ   CHUSER_FAIL
*
         L     R15,SAFPRREA
         LTR   R15,R15
         BNZ   CHUSER_FAIL
*
         LA    R1,8
```

```
         LA    R2,USERDIN+7
LEN      CLI   0(R2),C' '
         BNE   SET_LEN
         BCTR  R2,0
         BCT   R1,LEN
SET_LEN  DS    0H
         STC   R1,USERID
         MVC   USERNAME,USERDIN
*
         MVC   SPLUSERID,USERNAME
         WTO   TEXT=USERTXT
         WTO   'UNIXSRV- CREATING ACEE'
         LA    R2,USERID
         LA    R3,RACSAFE
         RACROUTE REQUEST=VERIFY,                                  C
               ENVIR=CREATE,                                       C
               USERID=(R2),                                        C
               WORKA=(R3),                                         C
               MF=(E,ACEE_CRE)
*
         USING SAFP,R9
         LA    R9,ACEE_CRE
*
         LTR   R15,R15
         BNZ   CHUSER_FAIL
*
         L     R15,SAFPRRET
         LTR   R15,R15
         BNZ   CHUSER_FAIL
*
         L     R15,SAFPRREA
         LTR   R15,R15
         BNZ   CHUSER_FAIL
*
         MODESET KEY=NZERO
*        AUTH OFF
         WTO   'UNIXSRV- USER ID CHANGED'
         LA   R15,0
         PR
*
CHUSER_FAIL DS 0H
         LA    R2,RACSAFE
         RACROUTE REQUEST=VERIFY,                                  C
               ENVIR=CREATE,                                       C
               USERID=DFTUSER,                                     C
               WORKA=(R2),                                         C
               MF=(E,ACEE_CRE)
         USING SAFP,R9
         LA    R9,ACEE_CRE
```

```
*
         LTR   R15,R15
         BNZ   DEFAULT_ERR
*
         L     R15,SAFPRRET
         LTR   R15,R15
         BNZ   DEFAULT_ERR
*
         L     R15,SAFPRREA
         LTR   R15,R15
         BNZ   DEFAULT_ERR
         WTO   'UNIXSRV- UNDEFINED USER ID'
*
         MODESET KEY=NZERO
*        AUTH OFF
         LA   R15,16
         PR
*
DEFAULT_ERR DS  0H
         WTO   'UNIXSRV- UNABLE TO SET UP DEFAULT USER'
*
         MODESET KEY=NZERO
*        AUTH OFF
         LA   R15,16
         PR
*
* ************************************************************
*                     CONSTANTS                               *
* ************************************************************
*
M_ACEE_DEL RACROUTE REQUEST=VERIFY,                                    C
               ENVIR=DELETE,                                           C
               RELEASE=1.9.2,                                          C
               MF=L
M_ACEE_DEL_L   EQU  *-M_ACEE_DEL
*
M_ACEE_CRE RACROUTE REQUEST=VERIFY,                                    C
               ENVIR=CREATE,                                           C
               PASSCHK=NO,                                             C
               RELEASE=1.9.2,                                          C
               MF=L
M_ACEE_CRE_L   EQU  *-M_ACEE_CRE
*
MSG3      DC    AL2(MSG3_LEN-2)
          DC    C'UNIXSRV- SUBMIT REQUEST RECEIVED FROM USER '
          DS    CL8
MSG3_LEN  EQU   *-MSG3
*
DFTUSER  DC    X'8'
```

```
         DC    CL8'ROOT'
         LTORG
*
* ********************************************************************
*                    DUMMY SECTION                                   *
* ********************************************************************
*
VARAREA  DSECT
*
ACEE_DEL RACROUTE REQUEST=VERIFY,                                     C
               ENVIR=DELETE,                                          C
               RELEASE=1.9.2,                                         C
               MF=L
*
ACEE_CRE RACROUTE REQUEST=VERIFY,                                     C
               ENVIR=CREATE,                                          C
               PASSCHK=NO,                                            C
               RELEASE=1.9.2,                                         C
               MF=L
*
USERTXT   DC    AL2(USERTEXTL)
          DC    C'UNIXSRV- SUBMIT REQUEST RECEIVED FROM USER '
SPLUSERID DS    CL8
USERTEXTL EQU   *-USERTXT-2
*
USERID   DS    X
USERNAME DS    CL8
*
SAVEAREA  DS    18F
REGS      DS    20A
*
         DS    0F
RACSAFE  DS    128F
*
VARAREA_LEN EQU *-VARAREA
*
INPARM   DSECT
USERDIN  DS     CL8
*
         ICHSAFP
         REGS
         END
```

MVS PROGRAM TCBPARM

```
           MACRO
.*****************************************************************
.*** MACRO DOCUMENTATION INFORMATION                          ***
```

```
.*** NAME:     TCPPARM                                              ***
.*** PURPOSE:  TP INPUT PARAMETER LIST MAPPING                      ***
.*********************************************************************
           TCBPARM
TCBPARM    DSECT
CONVRPL    RPL   AM=VTAM        * APPCCMD convrpl area
LEN_RPL    EQU   *-CONVRPL
CONVRPL6   ISTRPL6              * APPCCMD convrpl6 area
LEN_RPL6   EQU   *-CONVRPL6
PARM       DS    CL80           * TP input parm list
REQCOUNT   DS    AL2            * TCB serial number
*
* Recovery routine variables
*
SAVEAREA   DS    18F
SAVERET13  DS    F
TRANSACT   DS    CL8
*
RECV_WTO   WTO   TEXT=,MF=L
REC_MSG    DS    0H
REC_MSG_LEN DS  AL2
REC_MSG_TXT DS CL80
*
TCBPARML   EQU   *-TCBPARM
           MEND
```

MVS PROGRAM JESPARMS

```
           MACRO
.*********************************************************************
.*** MACRO DOCUMENTATION INFORMATION                                 ***
.*** NAME:     JESPARMS                                             ***
.*** PURPOSE:  TP LU 6.2 INPUT PARAMETER CONTAINING JES2            ***
.***           ALLOCATION PARAMETERS                                ***
.*********************************************************************
         JESPARMS
JESPARMS DSECT
USERDIN  DS     CL8
CLASSDIN DS     C
DESTDIN  DS     CL18
FORMSIN  DS     CL8
FORMDIN  DS     CL6
PAGEDIN  DS     CL6
CHARSDIN DS     CL4
         DS     C
REGLEN   DS     XL2
N_LINES  DS     XL2
JESPARMSL EQU *-JESPARMS
           MEND
```

MVS PROGRAM REGS

```
         MACRO
         REGS
*
*********************************************************************
*                                                                   *
*    Register EQUATES                                               *
*                                                                   *
*********************************************************************
*
R0       EQU   0
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R10      EQU   10
R11      EQU   11
R12      EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         END
         MEND
```

Fernando Manuel Carvalho Nunes
System Programmer
Companhia de Seguros Bonanca (Portugal)

November 1995 – October 2000 index

Overleaf is an index of all topics covered in *AIX Update* since Issue 1, November 1995. The numbers in **bold** are issue numbers, and the numbers in brackets are page numbers. If you'd like to order back-issues of *AIX Update*, please contact Xephon. You'll find contact information listed on page 2 – back-issues are available from Issue 1, and can be obtained from any of Xephon's offices.

Computer Associates has announced eTrust Policy Compliance Version 7.2, an enhanced version of Policy Compliance Manager, which periodically scans systems and applications for conformance with the latest security advice from industry groups and operating system vendors. Databases and Web servers can be scanned and interfaces are supported to enable integration with system management and trouble-ticketing systems. The product encompasses risk assessment, attack detection, and loss prevention and works with the Unicenter management framework.

eTrust Policy Compliance supports for AIX, Solaris, HP-UX, Linux, NT, and Windows 2000. Details on pricing are available on request from the vendor.

For further information contact:
Computer Associates International, 1 Computer Associates Plaza, Islandia, NY 11788, USA
Tel: +1 516 342 5224
Fax: +1 516 342 5734
Web: http://www.cai.com

Computer Associates, Ditton Park, Riding Court Road, Datchet, Slough, Berkshire SL3 9LL, UK
Tel: +44 1753 577733
Fax: +44 1753 825464

* * *

Internet Security Systems has announced RealSecure 5.0 and Internet Scanner 6.1. The former is an intrusion detection system and the latter a vulnerability assessment product, and both now benefit from tighter integration with the SAFEsuite platform.

RealSecure now provides security administration of both AIX and HP-UX servers and manages applications protection centrally, while RealSecure 5.0 now features ActiveAlert, which allows security issues, such as the discovery of viruses, to be reported for immediate attention through the SAFEsuite platform.

Both RealSecure 5.0 and Internet Scanner 6.1 are available for download from the ISS Web.

For further information contact:
ISS Group Inc, 6600 Peachtree-Dinwitty Road, 300 Embassy, Atlanta, GA 30328, USA
Tel: +1 770 395 0150
Fax: +1 770 395 1972
Web: http://www.iss.net

ISS UK, 14 Brewery Court, High Street, Theale, Berkshire RG7 5AJ, UK
Tel: +44 118 930 4224
Fax: +44 118 930 5802

* * *

IBM has launched Host Access Client Package Version 2.0, a WebSphere Host Integration Solution component that provides Web-based access to host applications. The software combines Personal Communications with the WebSphere Host On-Demand and Screen Customizer, combining traditional emulation, thin client emulation, and screen rejuvenation in one package.

Out now, it runs on AIX, Linux, OS/390, OS/400, OS/2, Solaris, Windows 2000, NT, and HP-UX. It costs US$300 per user.