115

AIX update

May 2005

In this issue

A itc PUBLICATION

Published by

Xephon Inc
PO Box 550547
Dallas, Texas 75355
USA

Phone: 214-340-5690
Fax: 214-341-7081

Editor

Trevor Eddolls
E-mail: trevore@xephon.com

Publisher

Colin Smith
E-mail: info@xephon.com

Subscriptions and back-issues

A year's subscription to *AIX Update*, comprising twelve monthly issues, costs $275.00 in the USA and Canada; £180.00 in the UK; £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 2000 issue, are available separately to subscribers for $24.00 (£16.00) each including postage.

AIX Update on-line

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

Contributions

When Xephon is given copyright, articles published in *AIX Update* are paid for at the rate of $160 (£100 outside North America) per 1000 words and $80 (£50) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of $32 (£20) per 100 lines. To find out more about contributing an article, without any obligation, please download a copy of our *Notes for Contributors* from www.xephon.com/nfc.

Printed in England.

Keeping the maximum number of back-ups of a file

INTRODUCTION

The script dirclean.sh is used to keep the maximum number of back-ups of a file.

Typical usage would be with log files, where a common prefix is used to rotate logs on a daily (or other) basis. Some examples are access and error logs of Web servers such as Apache or IBM HTTPD Server (IHS), or WebSphere Application Server.

An example listing of Apache access logs:

```
me@aixserv:/usr/HTTPServer/logs$ ls -l access_log.*
-rw-r—r—   1 root    system   11330325 Jan 14 18:59 access_log.1105660800
-rw-r—r—   1 root    system    4603307 Jan 15 18:59 access_log.1105747200
-rw-r—r—   1 root    system    3386311 Jan 16 18:59 access_log.1105833600
-rw-r—r—   1 root    system    8578147 Jan 17 18:59 access_log.1105920000
-rw-r—r—   1 root    system   13939217 Jan 18 18:59 access_log.1106006400
-rw-r—r—   1 root    system   14264170 Jan 19 18:59 access_log.1106092800
-rw-r—r—   1 root    system   14115202 Jan 20 18:59 access_log.1106179200
-rw-r—r—   1 root    system    9589473 Jan 21 11:40 access_log.1106265600
```

This script can be customized to list the files to process, and the maximum number of back-ups to keep.

It is recommended that you run this script from a privileged account such as root in a crontab on the required basis (hourly, daily, weekly, etc).

A log is kept listing the files that were deleted at the timestamp indicated.

Note: this script can be easily modified to work on any flavour of Unix by simply setting the appropriate PATH variable to match the system commands of your Unix flavour.

SCRIPT

```
#!/bin/sh
```

```
# Set the PATH for our commands in AIX
export PATH=/usr/bin

# List the prefix of the files we want to process
FILES="/usr/WebSphere/AppServer/logs/sas_client.log  \
       /usr/WebSphere/AppServer/logs/sas_server.log \
       /usr/HTTPServer/logs/access_log"

# Set the path to the logfile where we log everything we do
LOGFILE="/var/adm/ras/dirclean.log"

# Set the MAXFILES value at the number of files we want to keep
MAXFILES=5

# Main
for file in $FILES
do
      # Verify we do not have more than $MAXFILES of each file in $FILES
      FILECOUNT='ls $file* 2>/dev/null|wc -l|sed -e 's/ //g''

      # Ensure we have at least 1 file matching prefix
      if [[ $FILECOUNT -eq Ø ]];
      then
  echo "`/usr/bin/date` | Files with prefix $file not found" >> $LOGFILE
      fi

      if [[ $FILECOUNT -gt $MAXFILES ]];
      then
      HEADCOUNT='expr $FILECOUNT - $MAXFILES'
      DELETEFILES='ls -tr $file* | head -$HEADCOUNT'
      echo "`/usr/bin/date` | Deleting files:\n$DELETEFILES" >> $LOGFILE
          rm -f $DELETEFILES
      fi
done

exit Ø
```

Elvio Prattico
Consultant
PRATTICO Consulting (Canada)

Why not share your expertise and earn money at the same time? *AIX Update* is looking for shell scripts, program code, JavaScript, etc, that experienced users of AIX have written to make their life, or the lives of their users, easier. Articles can be of any length and should be e-mailed to the editor, Trevor Eddolls, at trevore@xephon.com.

Testing…testing…123

INTERVIEWS

I am often amazed by how few companies check their potential employees' competency during an interview. Anyone can put down anything on a CV, but proving your experience and knowledge is another matter. While IBM and other computer companies now provide certification tests, I have not come across one single employer who has ever asked to see the test certificates or even school exam certificates. However, I have worked with people who have lied on their CV, only got the job because of family ties, or were lucky enough to find an employer whose recruitment process was less than robust.

I personally have been tested during a job interview only twice. The last time, within the last year, I had a paper-based Unix test that was rather Sun Unix biased with five AIX questions at the end. The last question I had to answer was, 'Which is the earliest version of AIX to be year 2000 compliant?'. So, as you can see, there is plenty of scope for writing an AIX-specific set of questions. It was this experience that led me to write this article and come up with the questions below. But before that, I'll explain a little about the certification process for those who want a go at the 'real' tests.

CERTIFICATION

The certification process has come a long way since I first did mine. Yet, the computer-based exams are still the preferred method for testing. For those who have never had a go at this kind of torture then let me enlighten you.

The idea is to test your knowledge of a specific computer product or area of computing. IBM has a complete range of certification topics from PC to mainframe, software OSs to

application development. There must be hundreds of topics you can be certified in now, and with each new release of software, a new set of tests/recertification exams are born. The main AIX certification is IBM eServer Certified Advanced Technical Expert pSeries and AIX 5L – trips off the tongue doesn't it? This involved four separate tests, each multichoice, in various areas of expertise, eg performance and system tuning, communications, HACMP. There is a matrix as to which tests you must pass and which are optional.

The tests themselves are almost entirely multichoice. You get to sit in front of a PC in a room without any aids (ie phones, palmtops, books, etc). You will get a specified length of time (typically 90 minutes) to answer a number of questions (typically between 50 and 100) with a required pass mark (typically between 50% and 80%). Most questions have one answer, but some do have several correct answers and usually the question will helpfully state how many you need to choose. There is no set order for doing the questions, and you can always return to any question at any time.

Once you have finished, there is the '5 seconds of hell' as the computer either calculates your score and displays it on the screen or sends it directly to the printer. You are now the proud owner of yet another piece of paper, which breaks your overall percentage down into subtopics scores. Also the word 'pass' or 'fail' will appear at the top.

Two last points about what make this area attractive for the computer companies: first, each test costs money; second, each test or set of tests has accompanying education courses, which, er, cost money, what else?

IBM's eServer pSeries certification Web page can be found at www-03.ibm.com/certify/certs/ax_index.shtml.

There are test objectives, recommended educational resources, and a sample test, which is always fun (note: Sun changes its sample tests!).

A LITTLE TEST

I think the multiple choice questions are a little too easy and any good system administrator will be able to tell you the command(s) to use in most situations. So, I have set out below a few basic AIX-specific question that should help weed out the wheat from the chaff – which are you? I'm sure you can add more questions of your own.

Note: All the questions relate to situations I have faced in the last month in some form or other. And the answer to all the questions is *not* smit!

Software-based questions

1 What is the difference between *smit* and *smitty*?

2 What is the function of the ODM, and what are its three parts?

3 Which command was introduced in AIX V5.1 that shows the configuration of a system. Which command does it augment?

4 Where is the default console log written to and how do you interrogate it?

5 Which command do you use to check what would happen if you AIX system dumped?

6 Having started a system trace, which command halts it?

7 Which command will delete entries from the errpt error log?

8 Besides the ODM, which file contains filesystem definitions?

9 Which file contains the basic printer definitions?

10 Which command is used to see the status of all subsystems?

11 Which command is used to create an LVM software mirror of a single logical volume?

12 Which command would you use to view the setting for a device?

13 While installing a patch someone hits CTRL-C. Which command is used to tidy up afterwards?

14 Which command is used to create a bootable system back-up to tape?

15 Which command will create a table of contents file in a directory containing AIX patches?

Hardware-based questions

1 On the G and J class servers, which command stops the system doing a full system check at start-up?

2 Having been given secure shell access to an HMC, which command line command can be used to allocate the resources of that partitioned system (not ssh!)?

3 A newly-installed 10/100 Ethernet card seems to be running very slowly, what is the most likely cause of the problem?

4 Which command, run on an SP workstation, will tell you information about your SP frame and set-up?

5 After attaching a tape drive to an AIX server, which command is used to define the drive to the system?

6 Which command will change the LEDs on the front of an SSA cabinet?

These are just a few of thousands of questions you could come up with. I'll leave it to you to make up your own and to try the practice exams online. But one last question before the answers.

Additional question

1 An application is having problems running under NT. How would you go about correcting any problems?

ANSWERS

Software-based questions

1 *smit* will be graphical if graphics capabilities are available; smitty will always be character based.

2 The ODM is the AIX data manager for storing system-specific information. The three parts are user, root, and share.

3 **prtconf**. The previous command was **lscfg**.

4 */var/adm/ras/console*.

Command: **alog -f /var/adm/ras/console -o**

5 **sysdumpdev**

6 **trcstop**

7 **errclear**

8 */etc/filesystems*

9 */etc/qconfig*

10 **lssrc -a**

11 **mklvcopy**

12 **lsattr / lsdev**

13 **installp -C**

14 **mksysb (-f /dev/rmtx)**

15 **inutoc**.

Hardware-based questions

1 **mpcfg –cf 11 1**

2 **chhwres**

3 It is set to autodetect in the speed field; this usually causes problems.

4 **splstdat**

5 **cfgmgr (-v)**

6 **ssaencl (-l ssax -l id)**

Additional question

1 Get the application supplier to port it to Unix, of course!

Phil Pollard
Unix and Tivoli Administrator (UK)

Understanding the join command

The **join** command is a versatile one that allows you to combine data from multiple files into one file to create more meaningful output or more specific reports.

Suppose you had two text files, both containing several columns of data, and having one or more of them in common. The **join** command can inspect both files, seek the common information between them, and report data in a variety of useful output formats.

JOIN COMMAND BASICS

The basic syntax of the **join** command is shown below:

```
join [ -a FileNumber | -v FileNumber ] [ -e String ] [ -o List ]
[ -t Character ] [-1 Field ] [ -2 Field ] File1 File2
```

For example, suppose you had a file called dept.lst, each line containing an employee name and the department to which that person reports. You have another file called phone.lst containing the names in the first list, but instead of departments it shows telephone numbers. You can tell the **join** command to seek lines in each file containing the same employee name and report both the department and the phone number for that employee.

Suppose you had a third list called office1.lst with the names in the first along with the office number of the employee. Depending on the flags you use on the **join** command, you can even display data that shows all four columns – employee name, department, phone number, and office number.

Each list does not necessarily have to contain all the names in the first list. Any names in the first list found to have no corresponding entry in the second list will just be skipped in the output. However, there are flags that can make them appear anyway, depending on the needs of the output. See the examples below for details.

FLAGS FOR THE JOIN COMMAND

The following flags assist the usefulness of the **join** command:

- -a 1 or 2 – tells **join** to write an output line for common fields that do *not* match any line in the other file. Specifying **-a 2**, for example, tells **join** to write lines where the specified common field in infile2 matches no lines in infile1. Unlike the **-v** flag, output is written in addition to default output.
- -v 1 or 2 – tells **join** to write an output line for common fields that do *not* match a line in the other file. Specifying **-v 1**, for example, tells **join** to write lines where the specified common field in infile1 matches no lines in infile2. Unlike the **-a** flag, no default output is written.
- -e text – tells **join** to use the specified text where an empty output field would be shown. For example **-e No_Data**

would show the word ‘No_Data’ rather than a blank space if there was no data in the specified field. The **-e** flag is most often used with the **-o** flag.

- -o fn.fld – tells **join** to produce output using the specified pattern of file numbers and fields. For example **-o 2.3 1.1** tells **join** to write the third field in infile2 followed by the first field in infile1. If the zero is specified (**-o 0**), the common field is used.
- -t char – tells **join** to use the specified character as the field separator in both the input and the output files. To specify the tab character, press the tab key between single quotes.
- -1 num – tells **join** to use the specified field in infile1 as the common field. For example **-1 7** would use the seventh field of infile1 as the common field. The input file must be sorted on that field first.
- -2 num – tells **join** to use the specified field in infile2 as the common field.
- infile1 – the name of the first file to be joined.
- infile2 – the name of the second file to be joined.

EXAMPLES

These examples use the following files containing these lines. Note that the join field (usually the first field in these example files) is in sort order.

dept.lst

```
Baker_Alice             M19
Carlson_Edith           G67
Edwards_Peter           L49
Franco_Roberto          M19
Guillermo_Maria         L49
Jacobs_Abram            G76
Johannsen_Liv           Z05
Juarez_Emmanuel         L49
Liverpool_Jake          L49
```

```
Martin_Bryson           Z05
Miller_Scott            G67
Quigg_Pierre            M19
Samuels_Maggie          Z05
Wilson_Skip             Z05
```

office1.lst

```
Baker_Alice             B-114
Franco_Roberto          A-214
Guillermo_Maria         A-222
Jacobs_Abram            A-218
Johannsen_Liv           A-220
Juarez_Emmanuel         A-216
Miller_Scott            B-116
Quigg_Pierre            B-112
Samuels_Maggie          B-120
Wilson_Skip             B-118
```

office2.lst

```
Baker_Alice             A-214
Miller_Scott            A-222
Quigg_Pierre            A-218
Samuels_Maggie          A-220
Wilson_Skip             A-216
```

phone.lst

```
Carlson_Edith           222-1067
Jacobs_Abram            222-1104
Johannsen_Liv           375-9914
Martin_Bryson           375-0556
Miller_Scott            222-3945
Samuels_Maggie          375-2323
Wilson_Skip             375-9855
```

Note: for the purpose of these exercises, assume that the above examples use the tab character as the field delimiter in the actual files. Further, where **-t ' '** appears as a flag of the join command, you would press the tab key between the single quotes.

EXAMPLE 1

If you were to enter the command:

```
join -t '    ' dept.lst office1.lst > comb1.lst
```

the results would be:

```
Baker_Alice     M19     B-114
Franco_Roberto  M19     A-214
Guillermo_Maria L49     A-222
Jacobs_Abram    G76     A-218
Johannsen_Liv   Z05     A-220
Juarez_Emmanuel L49     A-216
Miller_Scott    G67     B-116
Quigg_Pierre    M19     B-112
Samuels_Maggie  Z05     B-120
Wilson_Skip     Z05     B-118
```

Note that because there are no entries for Edith Carlson, Peter Edwards, Jake Liverpool, nor Bryson Martin in the office1.lst, there are no corresponding entries in the results using this simplest form of the join command.

EXAMPLE 2

Now let's assume you wanted to see entries for the missing personnel so you could see where data was needed. In that case, you would use the **-a** flag. If you were to enter the command:

```
join -t '    ' -a 1 dept.lst office1.lst > comb2.lst
```

the results would be:

```
Baker_Alice     M19     B-114
Carlson_Edith   G67
Edwards_Peter   L49
Franco_Roberto  M19     A-214
Guillermo_Maria L49     A-222
Jacobs_Abram    G76     A-218
Johannsen_Liv   Z05     A-220
Juarez_Emmanuel L49     A-216
Liverpool_Jake  L49
Martin_Bryson   Z05
Miller_Scott    G67     B-116
Quigg_Pierre    M19     B-112
Samuels_Maggie  Z05     B-120
Wilson_Skip     Z05     B-118
```

Note that the **-a** flag shows both the entries having matches

as well as those that have no matches.

EXAMPLE 3

If you wanted to show a specific string in the blank areas, either to indicate that you are aware that the data is missing or to ensure that counted fields are present, you can use the **-e** flag, as follows. If you were to enter the command:

```
join -t '   ' -a 1 -e unknown -o 1.1 1.2 2.2 dept.lst office1.lst >
comb3.lst
```

the results would be:

```
Baker_Alice     M19     B-114
Carlson_Edith   G67     unknown
Edwards_Peter   L49     unknown
Franco_Roberto  M19     A-214
Guillermo_Maria L49     A-222
Jacobs_Abram    G76     A-218
Johannsen_Liv   Z05     A-220
Juarez_Emmanuel L49     A-216
Liverpool_Jake  L49     unknown
Martin_Bryson   Z05     unknown
Miller_Scott    G67     B-116
Quigg_Pierre    M19     B-112
Samuels_Maggie  Z05     B-120
Wilson_Skip     Z05     B-118
```

This would put the string ‘unknown’ in place of the blanks. Here is how the output flag was interpreted for this example:

```
-o 1.1 1.2 2.2
```

Position the output as follows:

- 1.1 is the first field of the first file, the employee name.
- 1.2 is the second field of the first file, the department number.
- 2.2 is the second field of the second file, the office number.

EXAMPLE 4

Now let’s see what would happen if you wanted to display only

those entries that were missing data. In that case, you would use the **-v** flag. If you were to enter the command:

```
join -t '    ' -v 1 -e unknown -o 1.1 1.2 2.2 dept.lst office1.lst >
comb4.lst
```

the results would be:

```
Carlson_Edith    G67      unknown
Edwards_Peter    L49      unknown
Liverpool_Jake   L49      unknown
Martin_Bryson    Z05      unknown
```

Note that, unlike the **-a** flag used in Examples 2 and 3, the **-v** flag does not show matched entries.

EXAMPLE 5

Suppose you wanted to merge data from multiple invocations of the **join** command. Let's say you wanted to join the data displayed in Example 3 with the data in the phone.lst. If you were to enter the command:

```
join -t '    ' -a 1 -e unknown -o 1.1 1.2 1.3 2.2 comb3.lst phone.lst >
comb5.lst
```

the results would be:

```
Baker_Alice      M19      B-114    unknown
Carlson_Edith    G67      unknown  222-1067
Edwards_Peter    L49      unknown  unknown
Franco_Roberto   M19      A-214    unknown
Guillermo_Maria  L49      A-222    unknown
Jacobs_Abram     G76      A-218    222-1104
Johannsen_Liv    Z05      A-220    375-9914
Juarez_Emmanuel  L49      A-216    unknown
Liverpool_Jake   L49      unknown  unknown
Martin_Bryson    Z05      unknown  375-0556
Miller_Scott     G67      B-116    222-3945
Quigg_Pierre     M19      B-112    unknown
Samuels_Maggie   Z05      B-120    375-2323
Wilson_Skip      Z05      B-118    375-9855
```

This informative list would show all names, department numbers, office numbers, and phone numbers, as well as occurrences of missing office and phone numbers.

EXAMPLE 6

Now suppose you want to join using a specific field in one or both of the files. This could occur if some of the lines share conflicting data, and it is the conflict you want to eliminate. For example, suppose office1.lst contains current office numbers, and office2.lst shows that the people who formerly had office numbers beginning with B will now be sharing offices with those that began with A. You need a command that will compare the two lists using the office field (rather than the name field) to display a list containing the office numbers followed by the pair of names. As the join field must be sorted first, let us start by sorting the office2.lst by the office numbers. Please consult 'Understanding the sort command', *AIX Update,* issues 39 and 40, for information regarding sort keys.

If you were to enter the commands:

```
sort +1 -2 office2.lst > office2_sorted.lst

sort +1 -2 office1.lst | join -t '   ' -1 2  -2 2  -o 2.2 2.1 1.1  -
office2_sorted.lst > shared.lst
```

the results would be:

```
A-214   Baker_Alice     Franco_Roberto
A-216   Wilson_Skip     Juarez_Emmanuel
A-218   Quigg_Pierre    Jacobs_Abram
A-220   Samuels_Maggie  Johannsen_Liv
A-222   Miller_Scott    Guillermo_Maria
```

The file shared.lst shows who would now be officemates. Let us take a closer look at some of the flags used:

- -1 2 – use the second field in the first file (sorted office1.lst), which is the existing office number.
- -2 2 – use the second field in the second file (office2_sorted.lst), which is the new office number.

As you can see from these examples, the output from the **join** command can be used to merge data and display the results in a variety of useful ways.

David Chakmakian
Software Engineer (USA)

Restore rootvg with a different configuration

In a previous article in *AIX Update* (*Restore a volume group with a different configuration*, issue 114, April 2005) we showed how to recreate and restore a user-defined volume group (non-rootvg) using **savevg** and **restvg** with different definition or characteristics. In this article we will show a similar procedure that applies to rootvg. This is practically useful if you are restoring or cloning rootvg to a different system with different characteristics such as bigger disks. You can even use this procedure to restore mksysb on the same system where you want to change a rootvg definition that is not changeable after the initial installation.

Here are some rootvg definitions that you can change with this procedure:

- Restoring a mksysb with a larger PP size. This is helpful if you are cloning rootvg to bigger disks not supported by the current PP size, on a new or the same system.
- Restoring or cloning a mksysb without mirroring on a new system that does not have the same number of disks for rootvg.
- Restoring or cloning a mksysb on a new system that does not have the same number of disks for rootvg, in case rootvg was spread out on more than one disk.
- Restoring or cloning a mksysb and shrinking the filesystem sizes at the same time.
- Resizing one or more filesystems when restoring the volume group.

The following procedure will explain how to accomplish the above-mentioned tasks through examples. You can combine more than one change in one procedure if needed:

1 Create a new */image.data* file by running the following command:

```
mkszfile
```

2 Edit the file */image.data*:

```
vi  /image.data
```

3 The following examples show various possibilities of what can be changed in the rootvg definition. The changed lines are in italics.

- Example 1:

  Changing the PP size of a volume group from 8 to 16:

```
vg_data:
    VGNAME=rootvg
    PPSIZE=8   => PPSIZE=16
    VG_SOURCE_DISK_LIST=hdisk0
    QUORUM=2
    CONC_CAPABLE=no
    CONC_AUTO=no

lv_data:
    VOLUME_GROUP= rootvg
    LV_SOURCE_DISK_LIST= hdisk0
    LV_IDENTIFIER= 0000000067997cd1.1
    LOGICAL_VOLUME= hd4
    VG_STAT= active/complete
    TYPE= jfs
    MAX_LPS= 512
    COPIES= 1
    LPs= 10  => LPs=5
    STALE_PPs= 0
    INTER_POLICY= minimum
    INTRA_POLICY= middle
    MOUNT_POINT= /apps/data
    MIRROR_WRITE_CONSISTENCY= on
    LV_SEPARATE_PV= yes
    PERMISSION= read/write
    LV_STATE= opened/syncd
    WRITE_VERIFY= off
    PP_SIZE= 8  =>  PP_SIZE=16
    SCHED_POLICY= parallel
    PP= 10  =>  PP=5
    BB_POLICY= relocatable
    RELOCATABLE= yes
    UPPER_BOUND=32
    LABEL=/apps
    MAPFILE=
    LV_MIN_LPS=7  =>  LV_MIN_LPS= 4
```

Note that in this example the PPSIZE in the vg_data stanza has been changed from 8 to 16. Also the PP_SIZE in the lv_data stanza changed from 8 to 16. The LPs and PP in lv_data were divided by 2 to reduce the number of LPs the logical volume has. Remember that you have to make the same changes to PP_SIZE, LPs, PP, and LV_MIN_LPS for each lv_data stanza for each logical volume in rootvg (hd4, hd5, hd2, hd3, hd6, hd1, hd9var, hd10opt, and any other paging spaces, dump devices, and user-defined logical volumes in rootvg).

The division of the LPs, PP, and LV_MIN_LPS is necessary so that the logical volume is created at the same size. Figure 1 is a division table to use as a reference.

If the result of the division is a decimal number, eg 4.5, round it up to the nearest whole number. You can also reverse this example if your intention is to make the PP size smaller instead of larger.

- Example 2:

  Changing the number of copies and remove mirroring:

```
vg_data:
```

Old PP size	*New PP size*	*Divide LPs, PP, and LV_MIN_LPS by*
4	8	2
4	16	4
4	32	8
8	16	2
8	32	4
8	64	8
16	32	2
16	64	4
32	64	2
32	128	4
...		

Figure 1: Reference division table

```
     VGNAME=rootvg
     PPSIZE=8
     VG_SOURCE_DISK_LIST=hdisk0
     QUORUM=1  =>  QUORUM=2
     CONC_CAPABLE=no
     CONC_AUTO=no

lv_data:
     VOLUME_GROUP=rootvg
     LV_SOURCE_DISK_LIST= hdisk0
     LV_IDENTIFIER= 00000000067997cd1.1
     LOGICAL_VOLUME= hd4
     VG_STAT= active/complete
     TYPE= jfs
     MAX_LPS= 512
     COPIES= 2  =>  COPIES=1
     LPs= 10
     STALE_PPs= 0
     INTER_POLICY= minimum
     INTRA_POLICY= middle
     MOUNT_POINT= /apps/data
     MIRROR_WRITE_CONSISTENCY= on
     LV_SEPARATE_PV= yes
     PERMISSION= read/write
     LV_STATE= opened/syncd
     WRITE_VERIFY= off
     PP_SIZE= 8
     SCHED_POLICY= parallel
     PP= 20  =>  PP=10
     BB_POLICY= relocatable
     RELOCATABLE= yes
     UPPER_BOUND=32
     LABEL=/apps
     MAPFILE=
     LV_MIN_LPS=7
```

Note that in this example the COPIES in the lv_data stanza have been changed from 2 to 1. Also the PP in lv_data was divided by 2 to reduce the number of PPs because the number of copies was reduced by half. Remember that you have to make the same changes to LPs, COPIES, and PP for each lv_data stanza for each logical volume in rootvg (hd4, hd5, hd2, hd3, hd6, hd1, hd9var, hd10opt, and any other paging spaces, dump devices, and user-defined logical volumes in rootvg).

Finally, the QUORUM in vg_data stanza has been changed from 1 to 2 in order to enable the quorum on the volume group after removing the mirror.

- Example 3:

  Shrinking filesystems when restoring:

```
logical_volume_policy:
        SHRINK= no  =>  SHRINK=yes
        EXACT_FIT= no
```

  Changing SHRINK in the logical_volume_policy stanza will cause the logical volume to be created at the minimum size possible to accommodate the filesystem. This size is specified by the value of the LV_MIN_LPS field of the lv_data stanza.

  Note: AIX 5L provides an option, when restoring the mksysb in service mode, that will override SHRINK recorded in the */image.data* file. Therefore you might not need this procedure to shrink the size of the filesystem.

- Example 4:

  Resizing a certain filesystem only when restoring:

```
lv_data:
    VOLUME_GROUP= rootvg
    LV_SOURCE_DISK_LIST= hdisk0
    LV_IDENTIFIER= 00000000067997cd1.1
    LOGICAL_VOLUME= hd4
    VG_STAT= active/complete
    TYPE= jfs
    MAX_LPS= 512
    COPIES= 1
    LPs= 10  =>  LPs= 8
    STALE_PPs= 0
    INTER_POLICY= minimum
    INTRA_POLICY= middle
    MOUNT_POINT= /apps/data
    MIRROR_WRITE_CONSISTENCY= on
    LV_SEPARATE_PV= yes
    PERMISSION= read/write
    LV_STATE= opened/syncd
    WRITE_VERIFY= off
```

```
        PP_SIZE= 8
        SCHED_POLICY= parallel
        PP= 10  =>  PP= 8
        BB_POLICY= relocatable
        RELOCATABLE= yes
        UPPER_BOUND=32
        LABEL=/apps
        MAPFILE=
        LV_MIN_LPS= 7

fs_data:
  FS_NAME= /
  FS_SIZE= 163840  =>  FS_SIZE= 131072
  FS_MIN_SIZE= 114688
  FS_LV= /dev/hd4
  FS_FS= 512
  FS_NBPI= 4096
  FS_COMPRESS= no
```

In this example we resized the / root filesystem from 80MB to 64MB by changing LPs and PP in the lv_data stanza, and changing FS_SIZE in the fs_data stanza for the / root filesystem. Note that the new values are greater than or equal to LV_MIN_LPS and FS_MIN_SIZE. Remember that you have to make the same changes for each lv_data and fs_data stanzas for each logical volume and filesystem in rootvg that you need to resize.

4 Run **mksysb** from the command line without the **-i** switch. It will then use your edited */image.data* file:

```
mksysb -X   /dev/rmt#
```

where *rmt#* is your tape drive.

If you run **mksysb** from smit, remember to change 'Generate new /image.data file?' from 'yes' to 'no' in order not to overwrite the modified */image.data* file.

5 On the target system (or on the same system if you want), boot from the base operating system CD or from the bootable mksysb tape, then go to the maintenance menu and start restoring the mksysb on the target disk. This will restore rootvg with the new modifications you applied to the *image.data* file.

If another mksysb cannot be run, the following procedure can be used to modify the *image.data* file on the existing mksysb and then restore. Please be extra careful with this procedure because any mistake could destroy your existing mksysb tape:

1 On the target system or any other AIX system, place the mksysb tape in the tape drive.

2 Create a temporary directory to restore *image.data* file:

```
mkdir /tmp/image
```

3 Restore the system files from second sector of the mksysb tape:

```
cd /tmp/image
restore -s2 -xqvdf /dev/rmt#.1
```

This will restore the file *image.data* along with other system files (used to re-create rootvg) into */tmp/image* directory. Note that the tape device used with the restore command is rmt#.1, which is the no-rewind tape device.

4 Follow the examples in step 3 of the preceding section to edit the */tmp/image/image.data* file and make the necessary changes. Once the changes have been made continue with step 5.

5 Rewind the tape and fast-forward back to the beginning of the second sector of the tape:

```
tctl   -f /dev/rmt# rewind
tctl   -f /dev/rmt#.1 fsf 1
```

6 Back-up all system files in the */tmp/image* directory back to the second sector of the mksysb tape. Note that the tape device used with the back-up command is rmt#.1, which is the no-rewind tape device:

```
cd /tmp/image
find . -print | backup -iqvf /dev/rmt#.1
```

7 Verify that the files were backed up to the second sector of the mksysb tape, and the file can be read:

```
tctl   -f /dev/rmt# rewind
restore -s2 -Tqvf /dev/rmt#.1
```

8 Verify that your mksysb tape is still OK and you did not overwrite the data on it in the previous step by reading the contents of the fourth sector of the mksysb tape, which contains the data from the rootvg filesystems:

```
tctl   -f /dev/rmt# rewind
restore -s4 -Tqvf /dev/rmt#.1
```

9 If it all looks great, boot from the base operating system CD or from the bootable mksysb tape on the target system, then go to the maintenance menu and start restoring the mksysb on the target disk. This will restore rootvg with the new modifications you applied to the *image.data* file.

Basim Chafik
Senior Systems Analyst
IBM Certified Advanced Technical Expert (CATE)
Plexus (Division of BancTec) (Canada)

Successful business continuity: naming structures

This is the second in a series of articles discussing how to implement AIX in an environment dedicated to business continuity. The topic for this article is the assignment of machine names, host names, adapter names, aliases, etc. It is important that the naming scheme provides enterprise-wide unique values for each network location for normal operations as well as for disaster recovery and business continuity. The techniques discussed here will illustrate the need for a unique naming scheme for each network location and what constitutes a network location.

The purpose of this series of articles is to provide a foundation

for business continuity. In support of that aim, each topic discussed in this article is divided into the following:

- Policies
- Guidelines
- Standards
- Procedures.

Each organization should define its own set of policies, guidelines, standards, and procedures to define their enterprise-wide rules of design and implementation. These rules ensure the ability of an organization to operate on a day-to-day basis as well as in a disaster recovery effort.

Definition: enterprise wide unique – refers to a parameter that has one distinct value across any or all platforms throughout the entire enterprise.

NAMING SCHEME

In order to achieve maximum flexibility during normal maintenance, disaster recovery, and business continuity efforts, it is important to provide a naming standard for business functions that can be translated easily into machine names and/or resource groups. The purpose of using hostnames instead of IP addresses is that they are easier to remember and use. Hostnames are not necessary, but usually desirable.

In many organizations, it is policy that hostnames should not reflect the supported business function. The recommendations here will use business functions to derive hostname examples because it makes the principles discussed easier to understand. Translation of the logic behind the naming structure, to an environment where using business functions to derive hostnames is against policy, is an exercise that will be left to the reader.

Designing backwards from the business continuity perspective, we will want to provide names for business functions, assign those business functions to applications, identify the resources supporting those applications, and provide naming structures for supporting those resources. This means that to define hostnames, we will begin by identifying business functions.

First define the business functions associated with your organization, such as:

- Warehousing
- Finance
- Transportation.

Divide these business functions into resource groups; warehousing is used here as an example:

- Seattle warehouse
- Phoenix warehouse
- Dallas warehouse
- St Louis warehouse
- Atlanta warehouse
- Baltimore warehouse
- Boston warehouse
- Cleveland warehouse
- Chicago warehouse
- Fargo warehouse.

Each of these defines a group of resources that support the business function. These resources include applications, computer hardware and software, networking hardware and software, SAN disks, NAS disks, etc. Each resource group should be given a name that is used to identify it. This name will be used by users to access the various resources

associated with this resource group. In order to achieve maximum flexibility during maintenance, disaster recovery, and business continuity, the name assigned to this resource group will be used as a DNS alias to the networking resources supporting this resource group. A DNS alias is normally referred to as a 'CNAME' record. Users requiring access to the business functions provided by any resource group should use only the alias name for the resource group specified by the DNS CNAME record. Users should never access business functions using machine names or network adapter names.

System administrators may need to access the systems in a variety of ways such as for maintenance, support, troubleshooting, failover, back-ups, etc. In order to provide a consistent approach for these needs, a naming structure should be defined. This naming structure will identify machine names, host names, network adapter names, back-up interface names, system management names, and any names required for system administration. Each of the administration names assigned should also have an associated IP address. Each of these names should be enterprise-wide unique DNS 'A' records. The machine name should be identified separately from the host name because a single AIX machine may house several hosts. So for configuration management purposes, the machine should have a name different from the host, even on machines with only one host (single standard).

POLICIES: NAMING SCHEME

All business functions will be identified by resource group names.

Users may only access business functions using resource group names.

Each resource group will have an enterprise-wide unique DNS name defined in the DNS as a CNAME record (different from the network adapter, host, or machine names).

Each network adapter will have an enterprise-wide unique

DNS name defined in the DNS as an A record (different from the host or machine names).

Each AIX host will have an enterprise-wide unique DNS name defined in the DNS as an A record (different from any network adapter name).

Each AIX host will have an enterprise-wide unique system management name defined in the DNS as an A record (different from any network adapter name).

Each AIX host will have an enterprise-wide unique back-up name defined in the DNS as an A record (different from any network adapter name).

Each AIX machine will have an enterprise-wide unique name to identify it separately from the hosts contained in the enclosure (for configuration management purposes).

GUIDELINES: NAMING SCHEME

Only the service aliases are meant to be accessed by regular users; all other names are reserved for use by system administration personnel.

A service alias may be a name that is easily remembered by regular users and may reflect some specific business function or activity.

Each name should be 14 characters or less, aliases should be 8 characters or less.

STANDARDS: NAMING SCHEME

Begin by identifying the business functions and derive a naming standard from that perspective.

Business function naming

Assign a three-letter identifier for each business function. For example:

```
Administration and Financial (adm)
  acp = Accounts payable
  acr = Accounts receivable
  acc = Accounting Fixed assets
  bud = budgeting
  ord = order entry
  per = personnel
  ach = checks
  abr = bank reconciliation
  pro = profit sharing
  tra = travel expenses
  prp = profit planning
  tre = treasury
  pur = purchasing
  pay = payroll

Marketing (mar)
  quo = quotations
  reb = customer rebates
  sal = customer sales
  mon = monthly sales
  sta = sales statistics

Distribution (dis)
  pic = warehouse picking
  dtr = transportation
  con = contract management
  war = warehouse management

General Services (gen)
  mai = electronic mail
  off = office systems

Manufacturing (man)
  bom = Bill of materials/labor
  cos = job costing
  for = forecasting
  sch = master schedule
  rou = routing/capacity planning
  qua = quality control
  saf = safety
```

Resource group naming

Assign a name for each resource group using the business function name followed by a two-digit number. This will uniquely identify multiple instances of a resource group in separate environments or simultaneously in a single environment. For example:

```
Distribution
  pic01 = warehouse picking (Phoenix)
  pic02 = warehouse picking (Dallas)
  pic03 = warehouse picking (Atlanta)
  pic04 = warehouse picking (Boston)
  pic05 = warehouse picking (Chicago)
  pic06 = warehouse picking (Fargo)

  dtr01 = transportation (Phoenix)
  dtr02 = transportation (Dallas)
  dtr03 = transportation (Atlanta)
  dtr04 = transportation (Boston)
  dtr05 = transportation (Chicago)
  dtr06 = transportation (Fargo)

  con01 = contract management (Phoenix)
  con02 = contract management (Dallas)
  con03 = contract management (Atlanta)
  con04 = contract management (Boston)
  con05 = contract management (Chicago)
  con06 = contract management (Fargo)

  war01 = warehouse management (Phoenix)
  war02 = warehouse management (Dallas)
  war03 = warehouse management (Atlanta)
  war04 = warehouse management (Boston)
  war05 = warehouse management (Chicago)
  war06 = warehouse management (Fargo)
```

Resource group alias names

Create DNS alias names for each resource group. Use a naming structure that would make sense to the user, such as pre-pending a three-letter city code to the business function code. For example:

```
Distribution (dis)

  phopic = Alias name for Phoenix warehouse picking resource group
  dalpic = Alias name for Dallas warehouse picking resource group
  atlpic = Alias name for Atlanta warehouse picking resource group
  bospic = Alias name for Boston warehouse picking resource group
  chipic = Alias name for Chicago warehouse picking resource group
  farpic = Alias name for Fargo warehouse picking resource group

  phodtr = Alias name for Phoenix transportation resource group
  daldtr = Alias name for Dallas transportation resource group
  atldtr = Alias name for Atlanta transportation resource group
```

```
bosdtr = Alias name for Boston transportation resource group
chidtr = Alias name for Chicago transportation resource group
fardtr = Alias name for Fargo transportation resource group

phocon = Alias name for Phoenix contract management resource group
dalcon = Alias name for Dallas contract management resource group
atlcon = Alias name for Atlanta contract management resource group
boscon = Alias name for Boston contract management resource group
chicon = Alias name for Chicago contract management resource group
farcon = Alias name for Fargo contract management resource group

phowar = Alias name for Phoenix warehouse management resource group
dalwar = Alias name for Dallas warehouse management resource group
atlwar = Alias name for Atlanta warehouse management resource group
boswar = Alias name for Boston warehouse management resource group
chiwar = Alias name for Chicago warehouse management resource group
farwar = Alias name for Fargo warehouse management resource group
```

Resource group service names

Create DNS A record names for each resource group. Using the resource group names, pre-pend the three-letter identifier for the general category to the beginning. For example:

```
Distribution (dis)

dispic01 = Service name for Phoenix warehouse picking resource group
dispic02 = Service name for Dallas warehouse picking resource group
dispic03 = Service name for Atlanta warehouse picking resource group
dispic04 = Service name for Boston warehouse picking resource group
dispic05 = Service name for Chicago warehouse picking resource group
dispic06 = Service name for Fargo warehouse picking resource group

disdtr01 = Service name for Phoenix transportation resource group
disdtr02 = Service name for Dallas transportation resource group
disdtr03 = Service name for Atlanta transportation resource group
disdtr04 = Service name for Boston transportation resource group
disdtr05 = Service name for Chicago transportation resource group
disdtr06 = Service name for Fargo transportation resource group

discon01 = Service name for Phoenix contract management resource group
discon02 = Service name for Dallas cntract management resource group
discon03 = Service name for Atlanta contract management resource group
discon04 = Service name for Boston contract management resource group
discon05 = Service name for Chicago contract management resource group
discon06 = Service name for Fargo contract management resource group

diswar01 = Service name for Phoenix warehouse management resource group
```

```
diswar02 = Service name for Dallas warehouse management resource group
diswar03 = Service name for Atlanta warehouse management resource group
diswar04 = Service name for Boston warehouse management resource group
diswar05 = Service name for Chicago warehouse management resource group
diswar06 = Service name for Fargo warehouse management resource group
```

Machine names

For the purpose of providing enterprise-wide unique machine names, it is recommended that the serial numbers be used to identify the AIX machines. This is a useful practice in configuration management to identify machines supporting multiple LPARs, with multiple hosts. It allows the machine enclosure to be identified separately from each of the hosts it supports. For RS/6000 architecture machines, the serial number should be pre-pended with an ‘r’, and for pSeries architecture machines, the serial number should be pre-pended with a ‘p’. See Figure 1 for examples.

Architecture type	*Serial number*	*Machine name*
RS/6000	10-12345	r1012345
RS/6000	10-23456	r1023456
pSeries	12-34567	p1234567
pSeries	12-45678	p1245678
pSeries	12-56789	p1256789

Figure 1: Example machine names

Host names

The use of serial numbers as part of the machine name works well; however, attempting to use the serial number as part of the host name is quite confusing in practice. A more administrator-friendly method of identifying hosts should be implemented. Host names should allow a host to be identified uniquely, quickly, and exactly without confusion. Host names will be used only by administrators; users should never access a host by its host name.

Since a machine may contain more than one host, each host in a machine must be uniquely identified. To provide an enterprise-wide unique identity, a two-digit number base 36 number is assigned for each host within a machine. A base 36 number would be in the range of 0-Z, ie 0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ.

A double-digit base 36 number provides 1,296 possible combinations, which translates to 1,296 possible host names per machine. For most organizations, this can probably be reduced to a single digit providing 36 hostnames per machine. For a machine architecture that is capable of supporting only a single host, it is recommended that a host number of ‘00’ or ‘0’ be assigned.

One usual assumption of business continuity is that there is an alternative site where disaster recovery procedures are performed. Using this assumption to derive a host naming structure, a three-letter identifier for the data centre where the host exists may be used as part of the host name. Examples of three-letter data centre identifiers are:

```
ddc = Dallas Data Center
bhq = Boston Headquarters
awh = Atlanta warehouse
pdc = Phoenix Data Center
```

Assuming a multi-platform environment, it will also be useful to identify the platform using a three-letter identifier in the host name:

```
aix = AIX
as4 = AS/400
mvs = MVS
```

Combine the elements of platform type, location, double-digit base 36 machine identifier, and a single-digit base 36 host identifier into a host name. As an example, to derive the host name for an AIX machine that exists in the Dallas data centre, assigning it an enterprise-wide unique, three-digit base 36 identifier of 001, and assigning it a partition identifier of A: the derived host name becomes aixddc001A.

The three-digit base 36 identifier should be an enterprise-wide unique value for each machine; this number should not be duplicated within a data centre or between data centres. Ensuring the uniqueness of this value allows the host names to be manipulated in any way that makes sense to an organization.

Machine name	*Long host name*	*Short host name*	*Partition*
r1012345	aixddc0010	a0010	No partition
r1023456	aixddc0020	a0020	No partition
p1234567	aixawhe01A	ae01A	First LPAR
p1234567	aixawhe01B	ae01B	Second LPAR
p1234567	aixawhe01C	ae01C	Third LPAR
p1245678	aixpdcn01A	an01A	First LPAR
p1245678	aixpdcn01B	an01B	Second LPAR
p1256789	aixbhq903A	a903A	First LPAR
P1256789	aixbhq903b	a903B	Second LPAR

Figure 2: Examples of long and short host names

It may be useful to assign a range of values from the enterprise-wide unique base 36 identifier for each data centre, thus allowing the host name to be sorted using that identifier, while maintaining the ability to group hosts by location. For instance, assigning the range of 000–8ZZ to the Dallas data centre, 900–DZZ to Boston headquarters, E00–MZZ to the Atlanta warehouse, and N00–ZZZ to the Phoenix warehouse would permit unique identification of each host while at the same time identifying its location. The host name could be shortened from aixddc001A to a001A. The leading ‘a’ identifies the platform type as AIX.

This method would limit the total number of hosts allowed in the numbering scheme by data centre, but it may make sense, depending on the goals of an organization. Examples of long and short host names are shown in Figure 2.

Network adapter names

In an HACMP environment, IP addresses associated with network adapters may float from adapter to adapter and between hosts; therefore it is recommended that the network adapter names not be associated with any particular machine name, host name, or network adapter. In order to support normal maintenance, disaster recovery, and business continuity efforts, network adapter names will not be tied to any particular resource group, because under certain conditions, a network adapter may be required to support multiple resource groups.

In an HACMP environment, three different adapter functions need to be considered for the naming structure. These functions are:

- Service
- Boot
- Standby.

In non-HACMP environments, the adapter functions would be limited to ‘boot’ and the naming structure should reflect that. This means that the network address assigned at boot time will probably not change, therefore the naming structure derived to support the HACMP environment should work in the non-HACMP environment as well.

Depending on the HACMP configuration, a machine may be configured with two or more of the adapter functions. In configurations using IPAT via IP address takeover, a boot IP address will need to be assigned to at least one network adapter. Also at least one standby IP address will be assigned to a network adapter. In this configuration the boot IP address is switched with the service IP address when a resource group is brought online. In the event of a failure of the service network adapter, the standby address on another adapter is switched with the service address on the failed adapter.

In an HACMP configuration using IPAT via IP aliases, all

Platform	*Location*	*Machine*	*Network*	*Long host name*	*Short host name*
Aix	Dallas	0010	0	Aixddc00100	A00100
Aix	Dallas	0010	1	Aixddc00101	A00101
Aix	Dallas	0010	Management	Aixddc0010m	A0010m
Aix	Dallas	0010	Backup	Aixddc0010b	A0010b
Aix	Dallas	0020	0	Aixddc00200	A00200
Aix	Dallas	0020	1	Aixddc00201	A00201
Aix	Dallas	0020	Management	Aixddc0020m	A0020m
Aix	Dallas	0020	Backup	Aixddc0020b	A0020b
Aix	Dallas	003a	0	Aixddc003a0	A003a0
Aix	Dallas	003a	1	Aixddc003a1	A003a1
Aix	Dallas	003a	Management	Aixddc003am	A003am
Aix	Dallas	003a	Backup	Aixddc003ab	A003ab
Aix	Boston	903a	0	Aixbhq903a0	A903a0
Aix	Boston	903a	1	Aixbhq903a1	A903a1
Aix	Boston	903a	Management	Aixbhq903am	A903am
Aix	Boston	903a	Backup	Aixbhq903ab	A903ab
Aix	Boston	903b	0	Aixbhq903b0	A903b0
Aix	Boston	903b	1	Aixbhq903b1	A903b1
Aix	Boston	903b	Management	Aixbhq903bm	A903bm
Aix	Boston	903b	Backup	Aixbhq903bb	A903bb

Figure 3: Boot name examples

adapters will have boot addresses, may also have a heartbeat address, and the service addresses are dynamically aliased onto the network adapter as needed. Therefore, each network adapter will require a boot name associated with the boot address. A heartbeat name is not required.

Therefore, in order to support both HACMP configuration options, it is recommended that a boot name be assigned to each adapter, regardless of whether or not it is used. This will provide the greatest flexibility for future configuration changes, disaster recovery, and business continuity efforts.

In Figure 3, the network adapter boot names reflect the name of the data centre and host name in which they exist. The boot name will not be associated with the resource group it supports.

The recommended list of names to be defined for each machine, host, and adapter is:

- A three-letter business function identifier.
- Resource group name.
- Resource group alias (for user access).
- Service name(s) associated with each resource group.
- Machine name.
- A three-character, base 36, enterprise-wide, unique machine ID.
- A three-letter location identifier.
- A three-letter platform identifier.
- Boot name of each network adapter (using the short host name).
- Alias to boot name of each network adapter (using the long host name).
- System management name.
- Back-up/restore name.

This list of names to be defined for each machine, host, and adapter translates into the following DNS requirements:

- DNS A records:
  - boot name of each network adapter (using the short host name)
  - service name(s) associated with each resource group
  - system management name
  - back-up/restore name.
- DNS CNAME records:
  - resource group (service) alias (for user access)

Machine	*Host*	*Mgmt*	*Back-up*	*Boot*	*Boot alias*	*Service*	*Service alias*
R1012345	A0010	A0010m	A0010b	A00100	Aixddc00100	Dispic02	Dalpic
				A00101	Aixddc00101		
R1023456	A0020	A0020m	A0020b	A00200	Aixddc00100	Diswar02	Dalwar
				A00201	Aixddc00101		
P1234567	Ae01a	Ae01am	Ae01ab	Ae01a0	Aixawhe01a0	Diswar03	Atlwar
				Ae01a1	Aixawhe01a1		
P1234567	Ae01b	Ae01bm	Ae01bb	Ae01b0	Aixawhe01b0	Disdtr03	Atldtr
				Ae01b1	Aixawhe01b1		
P1256789	A903a	A903am	A903ab	A903a0	Aixbhq903a0	Admacp04	Bhqacp
				A903a1	Aixbhq903a1		
P1256789	A903b	A903bm	A903bb	A903b0	Aixbhq903b0	Admacr04	Bhqacr
				A903b1	Aixbhq903b1		

Figure 4: Example of the names used in this article

– alias to boot name of each network adapter (using the long host name).

Figure 4 provides several examples of the various names described here. In the table of examples, the term ‘service alias’ is used interchangeably with ‘resource group alias’.

The naming scheme described here will probably seem to be a radical departure from currently implemented structures. However, the purpose of this structure is to provide flexibility for maintenance, disaster recovery, and business continuity. The use of aliases for user access ensures that the users can be directed easily and quickly to the resources configured to service their requests, even if those resources are subject to frequent change. In the past, service names have probably been configured to reflect the primary host name on which a resource group was configured. This new structure recognizes that a resource group may float freely between hosts, machines, and/or data centres.

Distinguishing between machine name and host name is important for asset and configuration management. It provides a granular separation of resources devoted to specific tasks and can be used for a variety of accounting practices, including chargeback.

The overall theme to the naming structure is enterprise-wide unique values. This concept will be reinforced repeatedly in this series of articles because it is vitally important to disaster recovery and business continuity. Unique values allow the referenced resources to be uniquely identified, regardless of the location or condition of the resources. Unique values also eliminate problems caused by duplicate value conflicts during a time-critical activity, such as disaster recovery.

PROCEDURES: NAMING SCHEME

To define a new machine, host, resource group, hostname alias, etc for use in an organization, the following set of procedures should be followed:

1 Assign the new machine in support of new or existing business function(s).

2 Assign new or existing resource group(s) associated with the business functions to be supported on the new machine.

3 Define service name(s) and service aliases for each resource group to be supported on the new machine. Also assign TCP/IP addresses to service name(s).

4 Define a hostname, management name, and back-up name to be assigned on the new machine, and also assign TCP/IP addresses to each name.

5 Define boot name(s) and boot aliases for each network adapter to be assigned on the new machine, and assign TCP/IP addresses to each adapter.

6 Define a machine name using the machine's serial number.

7 Perform DNS changes.

8 Perform name and TCP/IP address assignments on the new machine.

CONCLUSION

In addition to the main purpose of business continuity, the rules described in this article are also meant to provide a single set of policies, guidelines, standards, and procedures that can be implemented enterprise wide for clustered as well as non-clustered AIX machines.

The next article in this series will examine naming structures for volume groups, logical volumes, JFS log logical volume names, and mount points, for use in a business continuity environment.

Dana French
President
Mt Xia (tel: 615-556-0456) (USA)

More AIX–Solaris differences

This month we have more examples in our two-way guide for system administrators working in multi-platform environments with AIX and Solaris.

Operating system crash dump

- AIX:

  Your dump device is a logical volume:

  – create:

```
# mklv -y <dump-logical-volume> -t sysdump <volume-group> <size> hdiskN
```

  – activate:

```
# sysdumpdev -P -s /dev/<dump-logical-volume>
```

- Solaris:

  Crash dumps are written to swap partitions by default. During the next reboot, this file is read back from swap, compressed, and written to */var/crash/<hostname>*, to allow for later analysis.

  – reconfiguring the dump device to a dedicated dump device:

```
# dumpadm -d <device-name>
```

  for example:

```
# dumpadm -d /dev/dsk/c0t2d0s2
```

Physical memory

- AIX:

```
# bootinfo -r
```

- Solaris:

```
# /usr/sbin/prtconf | grep Memory
```

Non-rewinding tape drive

- AIX:

```
/dev/rmt0.1
```

- Solaris:

```
/dev/rmt/0n
```

Diagnostic

- AIX:

```
# /usr/sbin/diag
```

- Solaris:

```
# /usr/platform/'uname -m'/ sbin/prtdiag
```

and:

```
ok test-all
```

and:

```
# /opt/SUNWvts/bin/sunvts
```

List terminal

- AIX:

```
# lsdev -Cc tty
```

- Solaris:

```
# pmadm -l
```

CPU

- AIX:

```
# lsdev -Cc processor
```

- Solaris:

```
# psrinfo -v
```

Device drivers

- AIX:

```
# lscfg
```

- Solaris:

```
# prtconf -D
```

Remove a device

- AIX:

```
# rmdev -l <device> -d
```

- Solaris:

```
# rem_drv <device-driver>
```

and:

```
# cfgadm -x remove_device <Ap_id>
```

(get Ap_Id from **cfgadm –al**.)

NTP daemon

- AIX:

Configuration file:

```
/etc/ntp.conf
```

Start daemon:

```
# startsrc -s xntpd
```

- Solaris:

Configuration file:

```
/etc/inet/ntp.conf
```

Start daemon:

```
# /etc/init.d/xntpd
```

Timezone management

- AIX:

```
/etc/environment
```

- Solaris:

```
/etc/Timezone
```

or:

```
/etc/default/init
```

List kernel modules

- AIX:

```
# genkex
```

- Solaris:

```
# modinfo
```

Kernel

- AIX:

```
/usr/lib/boot/unix_up
```

- Solaris:

```
/kernel/genunix
```

Reconfigure the kernel

- AIX:

```
# chdev -l sysØ -a
```

- Solaris:

  Make changes in:

```
/etc/system
```

  Then:

  reboot

Start scripts in

- AIX:

```
/etc/rc
```

- Solaris:

```
/etc/init.d
```

Device listing

- AIX:

```
# lsdev -C
```

- Solaris:

```
# sysdef
```

Information about disk partitioning

- AIX:

```
# lspv -l hdiskN
```

- Solaris:

```
# prtvtoc /dev/rdsk/c#t#d#s2
```

Show network parameters

- AIX:

```
# no -a
```

- Solaris:

```
# ndd /dev/[tcp|ip] ?
```

Printers

- AIX:

  Printer queues:

```
/etc/qconfig
```

  Stop/start LP:

```
# stopsrc -s lpd
# startsrc -s lpd
```

  Submit print jobs:

```
# enq
# lp
# lpr
```

```
# qprt
```

LP statistics:

```
# enq -A
# lpq
# lpstat
# qchk
```

Remove print jobs:

```
# cancel
# lprm
# qcan
# enq -x
```

Add printer queue:

```
# smit mkpq
```

Remove printer queue:

```
# smit rmpq
```

Make default printer:

```
# export LPDEST="pq"
```

- Solaris:

  Printer queues:

  ```
  /etc/lp/interfaces/*
  ```

  Stop/start LP:

  ```
  # /usr/lib/lp/lpshut
  # /usr/lib/lp/lpsched
  ```

  Submit print jobs:

  ```
  # lp
  # lpr
  ```

  LP statistics:

  ```
  # lpstat
  ```

  Remove print jobs:

  ```
  # cancel
  # lprm
  ```

Add printer queue:

```
# lpadmin -p pq
```

Remove printer queue:

```
# lpadmin -x pq
```

Make default printer:

```
# lpadmin -d pq
```

Recover root password

- AIX:
  1. Boot from CD/tape
  2. Installation/maintenance start limited shell
  3. **getrootfs hdisk0**
  4. Delete 13-character encrypted root passwd in */etc/security/passwd*
  5. Reboot.
- Solaris:
  1. Go to ok prompt and boot from CDROM:

     ```
     ok boot cdrom -s
     ```

  2. Mount ***/a*** from CDROM as your root-disk:

     ```
     # mount  /dev/dsk/c0t0d0s0 /a
     ```

  3. Delete 13-character encrypted root passwd in *vi /a/etc/shadow*

     ```
     root:<13 char>: ...
     ```

  4. Reboot.

To view the boot log

- AIX:

  ```
  # alog -o -t boot
  ```

- Solaris:

```
# dmesg
```

Checking filesystem consistency

- AIX:

  First umount filesystem:

```
# fsck -y /dev/<lv-name>
```

  fsck command will check all the file systems with the attribute check=TRUE while booting in */etc/filesystems*.

- Solaris:

  First umount filesystem:

```
# fsck /dev/rdsk/c#t#d#s#
```

  All systems in the */etc/vfstab* file with entries in the *fsck pass* field greater than zero are checked while booting.

Editor's note: this article will be concluded next month.

Adnan Akbas
Senior System Administrator
TURKCELL (Germany)

In addition to *AIX Update*, Xephon also publishes *CICS Update*, *DB2 Update*, *MQ Update*, *MVS Update*, *RACF Update,* and *TCP/SNA Update*. Details of all of these can be found on the Xephon Web site at www.xephon.com.

AIX news

Engenio Information Technologies has announced Replication Express software for DB2. This new software improves back-up and recovery speed by automating the use of SANtricity Storage Management Software replication features such as Snapshot and Volume Copy.

Replication Express software for DB2 is specifically tested and designed for DB2 database environments, using Tivoli Storage Manager on AIX operating systems.

For further information contact:
URL: www.engenio.com/default.aspx?pageID=713.

* * *

NetManage has announced Version 7.2 of OnWeb, its middleware tier, which now includes support for .NET and Java-based connectors for enterprise applications such as SAP, PeopleSoft, Oracle, Siebel, and JD Edwards.

OnWeb enables companies to convert host-based business processes into reusable components, such as Web services, .NET Assemblies, Java Beans, EJBs, and portlets, which can be integrated with other enterprise applications or be used as components in a new application.

OnWeb 7.2 also includes support for iSeries single sign-on through Secure Sockets Layer (SSL) and Secure Shell (SSH), and Kerberos security standards. It interfaces with Microsoft BizTalk 2004 and has enhanced monitoring and reporting capabilities. There is improved support for Linux and AIX systems.

For further information contact:
URL: www.ftp.com/products/onweb/index.asp.

* * *

IBM, SSA Global, and Intermec have announced a RFID solution consisting of hardware, software, middleware, and deployment services.

SSA RFID is a component of SSA Global's supply chain management portfolio, which is deployable on the IBM eServer iSeries platform. IBM WebSphere middleware provides integration to Intermec's scanning, printing, and labelling equipment.

Customers can integrate business applications across i5/OS, AIX, Linux, and Windows.

For further information contact your local IBM representative.

* * *