# 111

# AIX update

*January 2005*

## In this issue





A xephon 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.*

# Determine CPU speed in AIX

You might say you don't need to read an article in order to know your CPU speed. You could get a simple print of the configuration, and *voila*! I could not agree more – unless your OS level is not AIX 5, and the system that you are looking at is more than a few years old and still running AIX 4. Here the article will become very useful.

In all versions of AIX 5L, determining CPU speed is pretty easy because the **prtconf** utility provides you with this information along with the type and number of CPUs, as shown in the following example:

```
# prtconf | grep Processor
Processor Type: PowerPC_POWER3
Number Of Processors: 2
Processor Clock Speed: 200 MHz
  Model Implementation: Multiple Processor, PCI bus
+ proc0            P1-C1        Processor
+ proc2            P1-C2        Processor
```

An alternative command is also supplied, beginning with AIX 5.1. The **pmcycles** command lists the CPU speed. This command is part of the *bos.pmapi.pmsvc* fileset, which is not installed by default. Therefore this command might not be available on some systems unless that particular fileset is installed.

In previous versions of AIX, starting with AIX 4.1, 4.2, and 4.3, determining the CPU speed is a different ball game. In AIX 4 there is no single command to provide the CPU speed, but there is a procedure for finding out the speed and number of CPUs. This procedure is nothing new, and was compiled from older IBM technical documents.

The best place to start is with the **uname** command:

```
# uname -m
     xxyyyyyymmss
```

The meanings of the placeholders are as follows:

| *Model ID* | *Machine type* | *Processor speed (MHz)* | *Architecture* |
|---|---|---|---|
| 02 | 7015-930 | 25 | Power |
| 10 | 7013-530 | 25 | Power |
| 10 | 7016-730 | 25 | Power |
| 11 | 7013-540 | 30 | Power |
| 14 | 7013-540 | 30 | Power |
| 18 | 7013-53H | 33 | Power |
| 1C | 7013-550 | 41.6 | Power |
| 20 | 7015-930 | 25 | Power |
| 2E | 7015-950 | 41 | Power |
| 30 | 7013-520 | 20 | Power |
| 31 | 7012-320 | 20 | Power |
| 34 | 7013-52H | 25 | Power |
| 35 | 7012-32H | 25 | Power |
| 37 | 7012-340 | 33 | Power |
| 38 | 7012-350 | 41 | Power |
| 41 | 7011-220 | 33 | RSC |
| 43 | 7008-M20 | 33 | Power |
| 43 | 7008-M2A | 33 | Power |
| 46 | 7011-250 | 66 | PowerPC |
| 47 | 7011-230 | 45 | RSC |
| 48 | 7009-C10 | 80 | PowerPC |
| 4C | | See Note 1 in text | |
| 57 | 7012-390 | 67 | Power2 |
| 57 | 7030-3BT | 67 | Power2 |
| 57 | 9076-SP2 Thin | 67 | Power2 |
| 58 | 7012-380 | 59 | Power2 |
| 58 | 7030-3AT | 59 | Power2 |
| 59 | 7012-39H | 67 | Power2 |
| 59 | 9076-SP2 Thin w/L2 | 67 | Power2 |
| 5C | 7013-560 | 50 | Power |
| 63 | 7015-970 | 50 | Power |
| 63 | 7015-97B | 50 | Power |
| 64 | 7015-980 | 62.5 | Power |
| 64 | 7015-98B | 62.5 | Power |
| 66 | 7013-580 | 62.5 | Power |
| 67 | 7013-570 | 50 | Power |
| 67 | 7015-R10 | 50 | Power |
| 70 | 7013-590 | 66 | Power2 |
| 70 | 9076-SP2 Wide | 66 | Power2 |

*Figure 1: Processor table*

| *Model ID* | *Machine type* | *Processor speed (MHz)* | *Architecture* |
|---|---|---|---|
| 71 | 7013-58H | 55 | Power2 |
| 72 | 7013-59H | 66 | Power2 |
| 72 | 7015-R20 | 66 | Power2 |
| 72 | 9076-SP2 Wide | 66 | Power2 |
| 75 | 7012-370 | 62 | Power |
| 75 | 7012-375 | 62 | Power |
| 75 | 9076-SP1 Thin | 62 | Power |
| 76 | 7012-360 | 50 | Power |
| 76 | 7012-365 | 50 | Power |
| 77 | 7012-350 | 41 | Power |
| 77 | 7012-355 | 41 | Power |
| 77 | 7013-55L | 41.6 | Power |
| 79 | 7013-591 | 77 | Power2 |
| 79 | 9076-SP2 Wide | 77 | Power2 |
| 80 | 7015-990 | 71.5 | Power2 |
| 81 | 7015-R24 | 71.5 | Power2 |
| 89 | 7013-595 | 135 | P2SC |
| 89 | 9076-SP2 Wide | 135 | P2SC |
| 94 | 7012-397 | 160 | P2SC |
| 94 | 9076-SP2 Thin | 160 | P2SC |
| A0 | 7013-J30 | 75 | PowerPC |
| A1 | 7013-J40 | 112 | PowerPC |
| A3 | 7015-R30 | See Note 2 in text | PowerPC |
| A4 | 7015-R40 | See Note 2 in text | PowerPC |
| A4 | 7015-R50 | See Note 2 in text | PowerPC |
| A4 | 9076-SP2 High | See Note 2 in text | PowerPC |
| A6 | 7012-G30 | See Note 2 in text | PowerPC |
| A7 | 7012-G40 | See Note 2 in text | PowerPC |
| C0 | 7024-E20 | See Note 3 in text | PowerPC |
| C0 | 7024-E30 | See Note 3 in text | PowerPC |
| C4 | 7025-F30 | See Note 3 in text | PowerPC |
| F0 | 7007-N40 | 50 | ThinkPad |

*Figure 1: Processor table (continued)*

- xx = 00
- yyyyyy = unique CPU ID
- mm = Model ID (these are the numbers to use to determine CPU speed).

- ss = 00 (submodel)

By cross-referencing the *mm* values from the **uname –m** output with the table in Figure 1, you can easily determine the processor speed.

The number of CPUs can be easily identified with the command:

```
# lsdev -Cc processor
proc0 Available 00-00 Processor
proc2 Available 00-02 Processor
```

Notes:

1 Systems where **uname –m** outputs a model ID of 4C.

In general, the only way to determine the processor speed of a machine with a model ID of 4C is to reboot into System Management Services and choose the system configuration options. However, in some cases the

| *Uname -M* | *Machine type* | *Processor speed (MHz)* | *Processor architecture* |
|---|---|---|---|
| IBM,7017-S70 | 7017-S70 | 125 | RS64 |
| IBM,7017-S7A | 7017-S7A | 262 | RD64-II |
| IBM,7017-S80 | 7017-S80 | 450 | RS-III |
| IBM,7025-F40 | 7025-F40 | 166/233 | PowerPC |
| IBM,7025-F50 | 7025-F50 | See Note 4 in text | PowerPC |
| IBM,7026-H10 | 7026-H10 | 166/233 | PowerPC |
| IBM,7026-H50 | 7026-H50 | See Note 4 in text | PowerPC |
| IBM,7026-H70 | 7026-H70 | 340 | RS64-II |
| IBM,Model 7042/7043(ED) | 7043-140 | 166/200/233/332 | PowerPC |
| IBM,Model 7042/7043(ED) | 7043-150 | 375 | PowerPC |
| IBM,Model 7042/7043(ED) | 7043-240 | 166/233 | PowerPC |
| IBM,7043-260 | 7043-260 | 200 | Power3 |
| IBM,7248 | 7248-100 | 100 | PowerPersonal |
| IBM,7248 | 7248-120 | 120 | PowerPersonal |
| IBM,7248 | 7248-132 | 132 | PowerPersonal |
| IBM,9076-270 | 9076-SP Silver Node | See Note 4 in text | PowerPC |

*Figure 2: uname -M output*

| *FRU number* | *Processor type* | *Processor speed (MHz)* |
|---|---|---|
| E1D | PowerPC 601 | 75 |
| C1D | PowerPC 601 | 75 |
| C4D | PowerPC 604 | 112 |
| E4D | PowerN5¿604 | 112 |
| X4D | PowerPC 604e | 200 |

*Figure 3: Processor speed and FRU number*

information gained from the **uname –M** command can be helpful – see Figure 2.

2 J-Series, R-Series, and G-Series systems.

You can determine the processor speed in an MCA SMP system from the FRU number of the CPU card by using the following command:

```
# lscfg -vl cpucard0 | grep FRU
FRU Number.................C1D
```

The output is shown in Figure 3.

3 E-Series and F-30 systems.

For the E-series and F-30 systems, use the following process to determine CPU speed. Execute:

```
# lscfg -vp | more
```

Look for the *CPU Card* stanza:

```
Part Number.................093H5280
EC Level....................00E76527
Serial Number...............17700008
FRU Number..................093H2431
Displayable Message.........CPU Card
Device Specific.(PL)........
Device  Specific.(ZA)........PS=166,PB=066,PCI=033,NP=001,CL=02,PBH
                                Z=64467000,PM=2.5,L2=1024
Device Specific.(RM)........10031997 140951 VIC97276
ROS Level and ID............03071997 135048
```

In the section *Device Specific (ZA)*, the *PS*= value is the processor speed in MHz.

4 F-50 and H-50 systems and SP Silver Node.

The following commands can be used to determine the processor speed of an F-50 system. Execute:

```
# lscfg -vp | more
```

Look for the *Orca M5 CPU* stanza:

```
Part Number.................Ø8L1Ø1Ø
EC Level....................E784Ø5
Serial Number...............L2Ø9Ø34579
FRU Number..................93H8945
Manufacture ID..............IBM98Ø
Version.....................RS6K
Displayable Message.........OrcaM5 CPU DD1.3
Product Specific.(ZC).......PS=Ø013c9eb00,PB=Ø009e4f58Ø,SB=Ø004f27
                                ac0,NP=Ø2,PF=461,PV=Ø5,KV=Ø1,CL=1
```

In the line containing *Product Specific (ZC)*, the entry *PS*= is the processor speed in hexadecimal notation. To convert this to an actual speed, use the following conversions:

– 0009E4F580 = 166 MHz

– 0013C9EB00 = 332 MHz.

The value *PF*= indicates the processor configuration:

– 251 = 1-way 166 MHz

– 261 = 2-way 166 MHz

– 451 = 1-way 332 MHz

– 461 = 2-way 332 MHz.

For example:

```
# lscfg -vp | grep PS=
Product Specific.(ZC).......PS=Ø013D92D4Ø,LB=Ø009EC96A0,SB=Ø005ABC )

# bc
    ibase=16
    Ø013D92D4Ø
    333ØØØØØØ
```

In this example, it means that you have CPU speed of 333 MHz.

Since we are on the topic of the CPU, I thought it would be exciting to throw in a nice useful utility that will show some running environmental data about your CPU. Some platforms provide environmental sensors that can be probed using a diagnostic utility to collect environmental readings such as temperature, fan speed, and voltage. The utility is called **uesensor**. This utility is not available on all platforms and is part of the *devices.chrp.base.diag* fileset. So I will leave you with this example:

```
# /usr/lpp/diagnostics/bin/uesensor -l
  Sensor = thermal sensor
  Status = Normal
  Value = 24 Degrees Celsius
  Physical Location Code = P2

  Sensor = fan-speed
  Status = Normal
  Value = 183Ø RPM
  Physical Location Code = F1

  Sensor = fan-speed
  Status = Normal
  Value = 237Ø RPM
  Physical Location Code = F2

  Sensor = voltage
  Status = Normal
  Value = 4999 MV
  Physical Location Code = P2

  Sensor = voltage
  Status = Normal
  Value = 3278 MV
  Physical Location Code = P2

  Sensor = voltage
  Status = Normal
  Value = 4999 MV
  Physical Location Code = P2

  Sensor = voltage
  Status = Normal
  Value = 12Ø77 MV
```

```
  Physical Location Code = P2

  Sensor = voltage
  Status = Normal
  Value = -12364 MV
  Physical Location Code = P2
```

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


# Network mapping using nmap in an AIX environment

## INTRODUCTION

This article looks at some of the issues associated with using network mapping devices such as **nmap** with AIX. It is designed primarily for SMEs; we have tended to find that smaller organizations often do not have the level of resources to devote to security issues that are found in large enterprises. It is often the case that security is not fully devolved to separate staff or departments, but is run by the systems administrators or similar individuals. In these cases it is difficult for overworked systems administrators to do everything that needs to be done on the security and administration fronts.

## NETWORK MAPPING

One task that security staff and/or systems administrators should do on a semi-regular basis is network mapping. The basic premise behind this is that before you can secure a network, you have to know how it can be threatened. If you can detect possible weaknesses in your network's security, they

can be fixed before they can be exploited by intruders. A number of tools can be used to monitor ports and map networks. In this article we will consider the use of **nmap** (network mapper) because it is still the best-known and widely-used monitoring tool. It is also free and it is the tool that malicious hackers are likely to use – so it provides security staff and systems administrators with the opportunity to gain the perspective of the hacker. However, there are other monitoring tools that will work on AIX, which include:

- **SAINT** – Security Administrator's Integrated Network Tool (Version 5.6.2).

  **SAINT** is a security assessment tool originally based on **SATAN**. Unlike **nmap**, **SAINT** is not a free product. The cost of **SAINT** is based on the total number of servers, workstations, peripherals, and other nodes (hosts) that will be scanned. However, **SAINT** is a highly sophisticated package. It is regularly updated (using **SAINTexpress**, which provides automatic updates whenever a scan is run). It scans for the majority of remotely detectable vulnerabilities and recommends fixes (it uses **SAINTwriter** software to design and generate vulnerability assessment reports). It can scan through a firewall. It uses the CERT and CIAC bulletins to update its security checks. It can also be used to determine compliance with current data privacy regulations such as GLBA, HIPAA, and COPPA. It also runs on a variety of other Unix platforms.

- **PIKT** – Problem Informant/Killer Tool V1.17.0.

  **PIKT** is a free utility to monitor systems, report and fix problems, and manage system configurations. It is an embedded scripting language and accompanying script interpreter. **PIKT** is also a sophisticated script and system configuration file preprocessor for use with the Pikt scripting language (or any scripting language of your choice). Finally, **PIKT** is a cross-platform, centrally run, script scheduler (like **cron**), customizing installer (like **rdist**),

command shell enhancement, and total script and configuration file management facility.

## WHY SCAN A NETWORK?

Before an external malicious hacker can attempt to crack your system, they have to go through several preliminary stages first. The first of these is finding a target machine. Once target networks and machines have been identified, the hacker needs to determine what services are running on the target's host. It is these services that the hacker will attack. The first technique to determine what services are running before a system can be compromised is port scanning. This is essentially finding machines on a network and testing them to see what ports are listening .

Attackers often specialize in cracking specific operating systems or applications. Attackers can use **nmap** for TCP stack fingerprinting to determine the type of machine being scanned. Essentially it works by sending a series of non-standard packets to an operating system and then interpreting what is returned. This 'fingerprint' is matched to **nmap**'s database of OS 'fingerprints'.

If an attacker using **nmap** finds that your site runs a particular version of a service, and there are known exposures with this level of code, the attacker can then use known attack methods to penetrate the system (these are widely available on the Internet). This is why it is so important to keep systems patched and up-to-date.

In addition to this, frequent security auditing will provide:

- An inventory of systems and services. It is still amazing how many enterprises do not know how many servers they have running. Remember, you need to know what you have before you can manage it.
- Identification of unauthorized systems and services. These can include both intruders and unacceptable use, such as

illegal ftp servers running within your enterprise (this happens more than you think).

- The ability to respond rapidly to application-specific exploitations.

## NMAP AND AIX

**Nmap** is still the tool for port scanning. It is probably going to be one of the first tools that a potential attacker will use. This is why those involved in AIX security and administration should be familiar with its use. Certainly IBM recommends its use in *Additional AIX Security Tools* (Redbook SG245971), and really no IDS testing would be complete without using **nmap**. It is an extremely useful information gathering tool that yields much of the necessary information about a system and its potential weaknesses.

**Nmap** is free to download, under the terms of the GNU General Public License (GPL), so it comes with the full source code, which you may modify and redistribute if necessary. **Nmap** was developed by Fyodor, and can be downloaded from www.insecure.org/nmap or any of the usual sources of AIX freeware such as Bull's freeware site (www.bullfreeware.com) and the University of California's public domain software site (http://aixpdslib.seas.ucla.edu/categories/network.html). It comes as a tarred source as well as RPM format for AIX 5L Linux users.

## BEFORE YOU START

The aim of this article is to show AIX administrators and security officers some of the issues associated with the use of **nmap** in their environment. This will provide a 'cracker's-eye view' of their network, which will enable them to initiate steps to provide a suitable defence.

**Nmap** is a very powerful tool that has many uses for the security community. There are, however, several important

factors to consider before you begin scanning. The use of port scanning on an enterprise network is an ethical minefield. Some larger enterprises will simply not allow a scan to take place. If you work for a small to medium company it may be easier. If you work in an organization where the chain of command is amorphous or the user base is considerable, then there could be complications, as we will discuss later. The important point is that you get the permission of those above you in the chain of command, and inform those likely to be affected in the user base before undertaking a scan.

Most texts on port scanning observe that permission should be obtained before undertaking a scan. For example IBM notes, 'Some organizations have strict policies against network scanning by unauthorized personnel. Be sure to get the proper approval in writing before running tools, such as **nmap**, on a production network.' It sounds simple. But who do you consult? Your supervisor is a good start, if you have one; the chief security officer and similar individuals are also a good idea. But, you may also want to consider the end users. A general e-mail stating that a scan will be run in the next two weeks should be sufficient (yes, many people will delete the mail and forget it, but at least you are covered). This is important if:

- You work in a technologically-aware enterprise.
- A lot of personal firewalls are installed in your enterprise. Today there are a plethora of personal firewall products that are either integrated with personal operating systems or available as stand-alone software. There are also other intrusion detection and access monitoring systems. So it is possible that a number of individuals have such software installed.

These users may become aware of your scans, which could be interpreted as an attack. This could cause:

- An increased number of calls to the security department or Help Desk.

- More sophisticated users to attack the scanning box.

It is really worth ensuring that you inform the users who would notice a scan before you carry it out. Also, have your written explanation and permission ready so that you can e-mail any irate users immediately.

## USING NMAP

Fyodor has provided some exceptionally detailed documentation for use with **nmap**. The following section provides an indication of how to use some of **nmap**'s facilities, but reading the user manual is strongly recommended. The principal scan types are as follows:

- Open scan **-sT** – TCP connect scan.

  This method uses the **connect()** system call, allowing rapid identification of any open or closed ports. If the **connect()** call succeeds, the port is open; if not, the port is closed.

  Open scanning techniques are quite easy to detect and to filter. This is because the scan uses a three-way TCP/IP handshake to open a full connection to a remote host. Still, this is how your initial security scans will be done.

- Half-open scan **-sS** – TCP SYN scan.

  This method does not open a full TCP connection. The client terminates the connection before the three-way handshake is completed. A SYN packet is sent. If a SYN|ACK is received, it indicates that the port is listening. An RST reveals a non-listener. If the SYN|ACK is received, an RST is immediately sent to tear down the connection. This means that the scan will often go unlogged by many connection-based IDSs.

- Stealth scans **-sF -sX -sN**

  A number of methods come under the heading of stealth scans (Stealth FIN, Xmas Tree, or Null scan modes). The

different techniques are not important. As you become more proficient at scanning your network, you will begin to see the use of these various scans. Open scans can be used to determine the network topology. But, using the half open and stealth modes will be useful for ascertaining whether the firewalls or other IDS devices can identify their presence. Over time you will begin to get a feel for how your network responds to the scans. Once you get to this stage you can see how well different firewalls and IDSs protect against the various scan types.

## SOME CONSIDERATIONS

Before scanning, here are some things to bear in mind:

- Get buy-in from the highest management levels, then the job will be much more straightforward.
- It is probably wise to start with small-scale scans.
- It is possible that, on very rare occasions, scanning could crash some older machines, which could have a significant impact if they are running batch jobs, etc. This can affect the network application, or the entire system may require restarting. Some devices such as printers or routers may reset themselves.
- Be ready to restart systems if necessary.
- Scanning multiple targets through one network device can slow that subnet's performance.
- Create exception lists that include infrastructure devices to be tested only by arrangement, and devices that are likely to crash under pressure.
- It is best not to waste time scanning rarely-changed devices such as routers and printers, or free subnets and broadcast addresses.
- If you can use a number of computers to speed throughput, this will save time.

- Because of scalability issues, it is better to map one port on many computers simultaneously rather than many ports on one computer.
- Determine where the bottlenecks are, such as hubs, wireless, and firewalls (places where many devices funnel through the same network connection).
- Detect new and updated systems, and scan these thoroughly.
- Maintain an OS and network services inventory.

## ADVANTAGES

As an internal security analyst you will have a number of advantages over an attacker. These include:

- Permission to scan intensively .
- High-speed access.
- SNMP identification or other means of identifying various targets.
- As you undertake the scans more often you will create longer-term records.

Vulnerability scanning only tells you where your security exposures are, it does not tell you what to do when you've found one. In fact, you will probably find it comparatively easy to find security exposures, because there are so many. The difficult part is determining how to close them. Tools like **SAINT** are helpful in this regard, but security staff should be familiar with the basics of IBM patches.

## OBTAINING AIX PATCHES

Users are able to download fixes by APAR or PTF number from the following URL:

- For 4.3.3 APARs, http://techsupport.services.ibm.com/rs6k/fixdb.html.

- For 5.1.0 and 5.2.0 APARs, http://techsupport.services.ibm.com/server/aix.fdc.

There are also troubleshooting databases that can be useful for debugging problems. Users can also download recommended maintenance levels: currently AIX 5100-11 Recommended Maintenance Package is available. This package provides a collection of updates containing fixes for problems reported after AIX 5.1.0 was made available. Download and installation tips are included with this service.

It is also possible to download selective fixes. The latest operating system fixes can be downloaded by individual fileset or by groups of filesets. A new interface allows users to select all filesets within categories such as monitoring tools or performance tools, or alternatively it is possible to select each individual fileset you wish to download.

More information can be obtained about accessing fixes using the Internet at http://techsupport.services.ibm.com/rs6k/fixes.html.

Users can also visit IBM Server Support to obtain fixes electronically or on physical media at http://www.ibm.com/server/support.

Security fixes are periodically bundled into a cumulative APAR. For more information on these cumulative APARs, including last update and list of individual fixes, send an e-mail to aixserv@austin.ibm.com with the words 'subscribe Security_APARs' in the subject line.

Remember that fixes are no longer provided for AIX versions prior to 4.3 because IBM no longer supports these. IBM recommends that users running pre-4.3 releases upgrade to 4.3.3 at the latest maintenance level, or to 5.1.

## CONCLUSIONS

Scanning is a job that security officers, or equivalent staff, need to perform on a regular basis. However, care should be

taken not to adversely affect anyone and to obtain written authorization. In this manner, scanning can produce many benefits without having any adverse effects.

You should ask three fundamental questions from your data:

- What can an intruder see on the target system?
- How can an intruder exploit the information?
- How are the scans manifested in the firewall logs?

It is very difficult to manage network security properly. Adequate resources need to be made available to ensure that appropriate defences are in place and kept up-to-date. IT staff may need specialist training. If the installation is large or complex, it is no use expecting over-stretched support staff and systems administrators to manage the work in addition to their other routine tasks.

With long-term records you can start to create a business case for:

- Increased security budget.
- Increased security training for your enterprise.
- Better firewalls, or filters, if required.

While gaining an understanding of the mindset of the malicious hacker and how that person might set about exploiting vulnerabilities is not in itself enough to secure a network installation, it is a step in the right direction.

## USEFUL ADDRESSES

The following security-related sites will provide a useful introduction to some of the general issues of IT security:

- www.cert.org – the Computer Emergency Response Team at Carnegie Mellon University.
- www.ciac.org – aite of the Computer Incident Advisory Capability, a useful information source.

- http://project.honeynet.org – the HoneyNet project, a useful site for vulnerability research and information.
- www.insecure.org – home site of Fyodor, author of the **nmap** security scanner. The ../tools.html page has an authoritative list of what Fyodor identifies as the top 50 security tools.
- www.iss.net – International Security Systems.
- www.securityfocus.com – hosts Bugtraq among other security matters.
- www.whitehats.com – a security information and reporting site.

*John Edwards*
*Systems Administrator (UK)*


## Business-driven capacity planning for AIX: techniques and approaches

This is the second article in a three-part series on business-driven AIX capacity planning. The first article addressed foundation principles and concepts. This article addresses the 'how to' for business-driven capacity planning. We will expand on the key steps from the previous article in more detail and dive into how to apply the techniques.

Our primary goal is to keep the business-driven approach simple. Many AIX installations have tried to quantify the relationship between business drivers and IT metrics for their forecasts and have been unsuccessful. The reasons for that are numerous:

- Choosing the wrong candidate business drivers (usually because their executive management believes that they

are drivers). For example, assuming revenue would be a good business driver is always wrong unless prices never change.

Note: simply put, revenue is the sum of price times quantity sold for each product. While product quantity sold is a potential business driver, price is not (unless you can accurately predict the elasticity of demand curve for each product – which is where people get into trouble: it isn't predictable). Combining a business driver with a non-business driver will preclude any correlation with IT metrics unless the price doesn't change for any of the products sold.

- Getting carried away with elaborate statistics like multi-variate regression and multi-variate correlation, or trying to develop a precise polynomial formula for curve fitting, since true business drivers usually result in relatively simple formulae of the form $y = mx$, where $x$ is the business driver and $y$ is the IT metric.
- Not using a hierarchical approach to business driver correlation. If a good business driver correlation doesn't exist at the AIX system level, then you need to go to the application level, and then to the 'transaction' (logical grouping of processes and threads) level. Correlations between business drivers and 'transactions' will always exist – but that level is too granular to be practical in all cases.
- Requiring that correlations be above 0.9. I've not found many business driver correlations that are above 0.9 unless I'm working with correlations at the 'transaction' level. A reasonable acceptance level is a correlation coefficient of 0.65 or above.
- Using too few sample points. You should use at least 25 volume-specific measurement points.
- Confusing time series measurements with volume-specific measurements and trying to do correlations involving

growth over time versus volume changes within a short period of time. The correlation effort is focused on volume changes over a short period of time such as the past two weeks so that application and system changes don't bias the correlation work.

Our approach in this article is to describe the techniques within the context of the eight steps described in the previous article. A few guiding principles for doing these techniques quickly and successfully are:

- Keep the formulae simple. A true business driver will need only a simple linear formula like $y = mx$. Complex formulae mean that you may be forcing a coincidental statistical relationship to fit the data and you are missing the causality requirement.
- Stop the analysis at the highest level of the system hierarchy. That is, if a correlation coefficient is above 0.65 at the system or dedicated server level with a single application, work with that. If that doesn't work, try the application level and then the transaction level, in that order. This will avoid spending too much time experimenting with the work at unfruitful levels.

## TWO KEY PRINCIPLES FOR BUSINESS-DRIVEN AIX CAPACITY PLANNING

There are two key principles that are important to understand in any business-driven IT forecasting technique and especially in the following eight steps:

- The time horizon used for business-driven capacity forecasts cannot be any longer than the time horizon used for business planning. Essentially, if the capacity forecasting time horizon is longer, then the business is requiring a capacity forecast beyond the time frame that the business is comfortable using for its own forecast. The capacity planners aren't business forecasters and a business-driven capacity plan must be synchronized with the business time horizon.

- Statistical correlation doesn't imply causality and if A precedes B that doesn't mean that A causes B. These are two basic statistical principles, but they are particularly critical for business-driven capacity planning. For example, if sales go up proportionately with IT spending, that doesn't mean that sales caused the increase in IT spending, nor does it mean that IT spending caused sales to go up. These are cases of statistical coincidence and require further exploration to determine causality.

## EIGHT STEPS OF AIX BUSINESS-DRIVEN CAPACITY PLANNING

The description of these steps assumes that you are very familiar with your own environment from an application and system perspective. The data collection process and these eight steps were described in the article *Business-driven capacity planning for AIX: concepts and principles* in November 2004's issue of *AIX Update*. However, the order of these steps has been changed from that in the original article to reflect the chronological sequence in which they would be performed.

### 1. Identify applications of interest

Most large organizations have hundreds of applications that each application owner believes to be critical. However, for capacity planning purposes (politics aside), the ones that matter are those that handle the biggest volume of work, consume the most CPU cycles, or are 'mission critical' to the business. In most environments, this comes down to the 'Top 10' applications. For business-driven capacity planning, you may want to start with the 'Top 5'.

### 2. Identify the candidate business metrics used in business planning and operational measurements and map to selected business transactions and applications

These Top 5 should have some intuitive business drivers. For example, a sales ordering system is assumed to have number of sales orders as a business driver. Likewise, bill of materials

processing is assumed to have either a number of parts or depth of the bill of materials as the business driver. These intuitive (but not necessarily correct) business drivers are the first set of candidate business drivers you can use for your analysis.

For the candidate business drivers, you need to find out whether those drivers are measured by the business on a daily, weekly, or monthly basis. Most business operational metrics are measured daily and sometimes hourly. Next, you need to determine whether business forecasts for these operational metrics exist – they usually do, it's just difficult to find the person who owns these numbers.

These candidate business drivers get mapped to the applications and, potentially, to the 'transactions' that they might drive. For our purposes, an application is a group of transactions. A transaction is defined as a set of processes, or threads, running on an AIX system that performs the same unit of work on behalf of a user. This unit of work is considered to be delimited by the time taken from when a message is received by the AIX system and the results (not just an acknowledgement) returned to the user.

Clarity and consistency in the definition of an application and a transaction are critical to the mapping.

### 3. Understand current usage characteristics and patterns over time

Capacity planning starts long before the forecast is developed. On-going analysis of a system's work needs to be done to characterize its behaviour over time. For applications that run on dedicated systems, this is simple. The workload is characterized in somewhat generalized system-independent terms by the system activity by hour, by week, or by month:

- System utilization:
  - system time.

| | Monday | Tuesday | Wednesday | Thursday | Friday |
|---|---|---|---|---|---|
| **Business driver candidates** | | | | | |
| • Sales orders | 24,000 | 20,000 | 22,000 | 18,000 | 17,000 |
| • Line items on sales orders | 95,000 | 75,000 | 80,000 | 70,000 | 72,000 |
| **IT Metrics** | | | | | |
| • Average CPU utilization | 78.00% | 75.00% | 65.00% | 65.00% | 70.00% |
| • Average I/O rate per minute | 820 | 700 | 720 | 800 | 630 |
| • Daily CPU seconds — Order Entry application | 410 | 370 | 390 | 330 | 360 |

*Figure 1: Typical data view for weekly patterns*

| | Friday | Thursday | Tuesday | Wednesday | Monday |
|---|---|---|---|---|---|
| **Business driver candidates** | | | | | |
| • Number of sales orders | 17,000 | 18,000 | 20,000 | 22,000 | 24,000 |
| • Number of line items | 72,000 | 70,000 | 75,000 | 80,000 | 95,000 |
| **IT metrics** | | | | | |
| • Average CPU utilization | 70.00% | 65.00% | 75.00% | 65.00% | 78.00% |
| • Average I/O rate per minute | 630 | 800 | 700 | 720 | 820 |
| • Daily CPU seconds — Order Entry application | 360 | 330 | 370 | 390 | 410 |

*Figure 2: Volume-ordered data by candidate business driver*

  - user time.
- I/O rate (program).
- Paging rate (usually not a characterization of a workload since it is configuration dependent).
- Message rate (in/out).
- Bytes in/out (not absolutely necessary).
- I/O content (a derived measure representing the I/O or CPU intensity of the workload).

These system metrics are used to identify patterns of behaviour during a day such as peak hour, peak day during the week, and peak week during the month, etc. I/O content is a measure of the mix of the workload and should be derived hourly to determine whether the workload is consistent or shifting throughout the day or month.

Note: I/O content is the ratio between CPU busy seconds for the application and the number of logical I/Os per second for that application. The physical I/O rate, exclusive of the page rate, can be used as a surrogate if needed.

Once this data is collected, then a baseline is established to represent a point in time that is representative of the work that you would like to forecast. The baseline would contain all of the system's metrics indicated above plus the corresponding business metrics for that same time period and volume of consistent business activity. A typical data view for weekly patterns is shown in Figure 1.

In Figure 1, the data is organized by time period. While this view is useful in understanding the workload's behaviour during the week over the production period, it is not the most important view for doing business driver analysis.

Volume-ordered data by candidate business driver is shown in Figure 2.

In Figure 2, the table is organized by one of the candidate

business drivers, *Sales orders*. This helps to visualize whether there is a possible correlation between the business volumes of sales orders and one or more of the IT metrics.

In business-driven capacity planning, a volumetric view is used. This view is taken from current data over a short period of time such as the last two to three weeks. In this view, some candidate business volume or business driver is used to organize the daily or hourly IT metrics in ascending order by business volume rather than chronologically.

Note: for our examples, we are using a small subset of data and over a very small time period in order to get the concepts across. Typically, you would like to have more than 20 volume-oriented data points to work with so that outliers can be excluded.

### 4. Determine correlation between business and IT metrics

This is the heart of the study. The relationship of interest is the sensitivity of IT metrics to the volume of business activity. For example, does application usage on the AIX server track consistently with sales order volumes (ie does a 30% increase in the number of sales orders result in a 30% increase in application utilization or CPU seconds for the same time period?) If not, then sales order volume will not be a good predictor of future application or CPU utilization.

If we take the same data from Figure 2 and put it in a Microsoft Excel spreadsheet, we can do some simple statistics to determine which business variables correlate with particular IT metrics for the same measurement period. The CORREL function within MS Excel does that very nicely and it is done for us in Figures 3 and 4 with *Number of sales orders* as the independent variable in Figure 3 and the IT metrics as the y variable (dependent variable) and the resulting correlation coefficient, r, in the last column. In our studies, not many correlations are above 0.9 and so we accept correlations as low as 0.65 for our initial effort to identify a candidate business driver for further analysis.

| | Friday | Thursday | Tuesday | Wednesday | Monday | 'Sales Order' correlation coefficient |
|---|---|---|---|---|---|---|
| **Business driver candidates (independent variables)** | | | | | | |
| • Number of sales orders | 17,000 | 18,000 | 20,000 | 22,000 | 24,000 | |
| • Number of line items | 72,000 | 70,000 | 75,000 | 80,000 | 95,000 | |
| **IT Metrics (dependent variables)** | | | | | | |
| • Average CPU utilization | 70.00% | 65.00% | 75.00% | 65.00% | 78.00% | 0.4681 |
| • Average I/O rate per minute | 630 | 800 | 700 | 720 | 820 | 0.5600 |
| • Daily CPU seconds — Order Entry application | 360 330 370 | 390 410 | 0.8865 | | | |

*Figure 3: Correlation coefficient for number of sales orders (data columns are in ascending order by number of sales orders)*

| | Friday | Thursday | Tuesday | Wednesday | Monday | **'Line Item' correlation coefficient** |
|---|---|---|---|---|---|---|
| **Business driver candidates (independent variables)** | | | | | | |
| • Number of sales orders | 17,000 | 18,000 | 20,000 | 22,000 | 24,000 | |
| • Number of line items | 72,000 | 70,000 | 75,000 | 80,000 | 95,000 | |
| **IT Metrics (dependent variables)** | | | | | | |
| • Average CPU utilization | 70.00% | 65.00% | 75.00% | 65.00% | 78.00% | 0.6385 |
| • Average I/O rate per minute | 630 | 800 | 700 | 720 | 820 | 0.5268 |
| • Daily CPU seconds — Order Entry application | 360 | 330 | 370 | 390 | 410 | 0.9020 |

*Figure 4: Correlation coefficient for number of line items (data columns are in ascending order by number of sales orders)*

Figure 3 shows the correlation coefficient for *Number of sales orders* (the data columns are in ascending order by number of sales orders).

Figure 4 shows the correlation coefficient for *Number of line items* (data columns are in ascending order by number of sales orders).

From Figure 3 we see that *Number of sales orders* correlates well with *Daily CPU seconds – order entry application*, and from Figure 4 we see that *Number of line items* within a sales order correlates well with *Daily CPU seconds – order entry application*. While *Sales orders* seemed like the most intuitive business driver, it turns out that the total *Number of line items* (there are multiple line items within a sales order) had a higher correlation with *Daily CPU seconds – order entry application*.

This was the conclusion from our more detailed analysis of their sales order application that was performed at a major pharmaceutical company running SAP. *Number of line items* proved to be a much better predictor than *Number of sales orders*, which had been used for years as a predictor of CPU usage.

Also, what we have been describing is how to perform a statistical correlation. The question of whether there was a 'cause-and-effect' relationship always needs to be answered. As we saw from this example, after exploring the logic of the application, it became apparent that a cause-and-effect relationship existed.

### 5. Develop a regression curve for projections

Once a reasonable statistical correlation and causality relationship has been established, a formula needs to be developed to forecast the IT metric based on a business driver. Software packages such as MICS and SAS provide statistical packages and routines to go against large amounts of data stored in their performance database (PDB) to automate this process. However, those packages aren't required for a quick spreadsheet analysis.

| Formula | 2x growth | 3x growth | 4x growth | 5x growth | 6x growth |
|---|---|---|---|---|---|
| **Business driver (using 75,000 line items/day as a base)** | | | | | |
| • # of line items = x (independent variable) | 150,000 | 225,000 | 300,000 | 375,000 | 450,000 |
| **Forecasted daily CPU seconds** | | | | | |
| • Forecasted Daily CPU Seconds = y-value (dependent variable) | 750 | 1125 | 1500 | 1875 | 2250 |

*Figure 5: Forecasting daily CPU seconds from a base of 75,000 line items/day using Excel*

| Formula | Friday | Thursday | Tuesday | Wednesday | Monday |
|---|---|---|---|---|---|
| **Business driver** | | | | | |
| • # of line items = x (independent variable) | 95,000 | 80,000 | 75,000 | 72,000 | 70,000 |
| **IT metric measured** | | | | | |
| • Daily CPU Seconds = y-value (dependent variable) | 360 | 330 | 370 | 390 | 410 |
| **Computed result for validation of closeness to fit** | | | | | |
| • Validation of y = mx, where m = .005 | 475 | 400 | 375 | 360 | 350 |

*Figure 6: Cross-checking your formula (data columns are in ascending order by number of sales orders)*

My experience has been to keep the approach as simple as possible and use the quickest method possible to go through the initial analysis as fast as possible to eliminate as many candidate business drivers as possible.

Simple linear curve fitting and extrapolations for a forecast can be done with either the TREND or FORECAST functions of Microsoft Excel. For exponential curve fitting, the LINEST or GROWTH functions can be used within MS Excel.

In Figure 5, the MS Excel TREND or Forecast formula can be used:

TREND(*known_y's*, *known_x's*, *new x's*, *const*)

where the *known-y's* were *Daily CPU seconds – order entry application* and the *known_x's* were the candidate business drivers (independent variable), *total # line items*. The *new_x's* are the values that you would like the TREND formula to use in deriving new *y* values. The const value was left blank. The FORECAST formula has a slightly different format and doesn't allow for a constant to be included:

FORECAST(*new x*, *known_y's*, *known_x's*)

For business-driven capacity planning using the TREND and GROWTH formulas, you are using a simple formula for extrapolating or interpolating your IT metrics (dependent variables) based on the business driver estimates (independent variables) that your business planners provide you with.

The formula that you are striving for with TREND and GROWTH is:

$y = mx + b$

where:

- $y$ = IT metric (like CPU seconds)
- $x$ = business driver candidate (like number of line items)
- $m$ =slope of line
- $b$ = constant.

Ideally, you would like *b* to be 0 so that there is a direct relationship between *y* and *x* since that means there is a high probability that *x* causes *y*. If *b* is not 0, then that really becomes your statistical fudge factor to narrow the gap. Figure 6 shows the use of this formula to compute the forecast number, *y*, the IT metric called *Daily CPU seconds*.

Figure 6 shows cross-checking your formula (data columns are in ascending order by number of sales orders).

In Figure 6 we can see that the value of *m* does fit well for all values, except for the outlier for Friday. This was because of the small sample set of 5, which was used for illustrative purposes. Also, because of the very small sample set, we didn't get rid of the outliers.

A good sample would include a range of 25 or more business volume metrics, by either hour or day, that approximate the following multiples of the average business volume: 0.25*x*, 0.5*x*, 0.75*x*, 1.0*x*, 1.25*x*, and 1.5*x* – where *x* is equal to the average business volume. Any values below 0.25*x* and above 1.5*x* would be discarded – an even tighter range for less volatile workloads might be used.

Note: the focus on measurements for the correlation and regression analysis is on organizing the data by volumes, not by time. Time-organized data is useful for trends, seasonality, and other behavioural patterns. Volumes are useful for business driver analysis. This means that your data collection efforts could be done within a few days or in less than two weeks if you can get the recommended volume differentials that I've suggested within that time period.

### 6. Identify key business events and activities that may affect application volumes or changes in the mix

Once you've quantified the relationship between your business drivers and your business metrics, you can gather information on key business events (eg sales incentives like a US SuperBowl ad), introduction of a new government regulation,

and seasonal activity (eg holiday shopping, which would drive more sales and more credit card authorizations).

For your business-driven forecast, an important question to ask of the business planner is, “Are there any known events that will cause a significant short-term increase in business volumes?”. I asked this question when I was forecasting the load for a PeopleSoft labour time accounting system on an AIX server for a state transportation department. Someone reluctantly mentioned that during snow emergencies everyone (approximately 13,000 workers) needed to file and process their labour time every day rather than twice a month in order to receive federal funds for the snow emergency. This was new information to everyone in the meeting, including long-time IT staff.

So, rather than sizing for the normal trickle-in of time cards, we then had to plan for a daily peak of 13,000 time card submissions and processing. The cost of missing that target was the loss of federal funds for their labour costs.

The lesson learned was to not assume that any application doesn’t have critical unusual business events that may have a significant impact on your capacity forecast.

### 7. Translate business forecast and business events into volume changes over time

Translating business forecasts and business events into volume changes over time is fairly straightforward. The formula that you developed in the *Develop a regression curve for projections* step can be applied here. With the business driver metrics from the business planner for the next 18 months or so, you can use the formula directly or use the TREND or GROWTH formulae within MS Excel, or their counterparts within MICS or SAS, to generate the appropriate IT load.

### 8. Identify configuration to meet service targets at each major projected volume change

For each milestone in your forecast, the estimate of your IT

metric (like CPU utilization or CPU seconds) needs to be translated into load on a particular configuration. This is usually done on a trial-and-error basis or with the use of a modelling tool.

If you are forecasting multiple applications on an AIX server, your individual application forecasts can be added together to get the overall server load. The appropriate server can be chosen to meet your service requirements and service objectives (eg CPU utilization at peak period is not to exceed 75%) during your projected peak period. Then, add a buffer or reserve to accommodate future growth.

SUMMARY

In this article, we've demonstrated a simple approach to business-driven capacity planning using MS Excel formulae with spreadsheet data. In the next article, we'll provide additional case studies to solidify the concepts.

*Russ Egeland*
*Managing Consultant*
*IBM Global Services (USA)*


## Oracle user satisfaction

Although AIX is an IBM product, users don't have to use DB2 UDB as their database. Because AIX is a Unix variant, any database that runs on Unix runs on AIX. That makes mySQL a popular choice for many sites. Other sites, and usually ones that have other Unix platforms, are quite often to be found running Oracle – both Oracle the well-known database product and Oracle applications.

The UK Oracle User Group has recently, and uniquely, surveyed its members to find out their opinion of Oracle, and the results are very interesting and worth passing on.

The good news for Oracle was that the satisfaction rating of the software recorded by the 500+ users has gone up this year. The results showed a 12% increase in satisfaction when compared with the 2003 survey findings.

The bad news is that users still had concerns about costs. Many felt that products such as Microsoft's SQL Server were by comparison much cheaper – and, of course, mySQL is free.

Other areas for Oracle to address were the users' view of Oracle's consulting division. Nearly a quarter (22%) were unhappy, with value for money being highlighted as the biggest concern. There was also concern over support and escalation of queries. About a quarter of respondents were happy (125 out of 448), but over a third (37%) of the 197 people to use it were unhappy with the response. In a keynote address at the conference, Jean Reiczyk, who is the vice-president of the support service division, addressed this issue. He said that Oracle's own figures showed that 65% of problems brought to the attention of his division were a result of the users' own configuration problems rather than related to the Oracle software being used.

An area that Oracle is addressing, but which is still not quite right yet (according to survey respondents), is the installation of products such as 9*i*AS and AS10*g* – its application server software.



## Recover a deleted file in AIX

While Windows provides, in addition to a recycling bin, many third-party and shareware utilities to recover wrongly deleted files, AIX does not come with a built-in utility to recover a deleted file. In fact no Unix operating system provides such a

utility by default. So what can you do to recover a file or script that you removed to free up some space in the filesystem, but later realized you needed and you have no back-up of it? This article will provide you with a procedure for recovering the deleted file that works on all AIX versions.

In AIX, when a file is deleted or removed only the link to the file is lost. The actual data in the file is still on the hard drive blocks (inaccessible) until another file comes and uses the same blocks, overwriting whatever is in them. Therefore recovery is possible to some extent.

I will use an example to explain the procedure. In this example, we will recover a file called *myfile*, in a JFS file system called */test*. The full path to the file is */test/mydir/myfile*, and the logical volume for */test* file system is */dev/lv07*. Please note that you will be using this procedure at your own risk and it is always recommended to do a back-up of your current filesystem for restore purposes in case of a mistake.

Unmount the filesystem as soon as possible in order to prevent any major updates or changes to the filesystem meta data and to prevent the deleted file blocks from being overwritten:

```
# umount  /test
```

Run the **fsdb** command on the logical volume */dev/lv07*. The command **fsdb** is a filesystem debug utility and is part of the *bos.rte.filesystems* fileset. The **fsdb** command has a different interface for a JFS filesystem and a JFS2 filesystem. This example shows the **fsdb** subcommands for a JFS filesystem only. For more information about the **fsdb** command please consult the man pages.

```
# fsdb /dev/lv07


File System:                            /dev/lv07

File System Size:                         16384  (512 byte blocks)
Disk Map Size:                                2  (4K blocks)
Inode Map Size:                               2  (4K blocks)
```

```
Fragment Size:                                  4096  (bytes)
Allocation Group Size:                          2048  (fragments)
Inodes per Allocation Group:                    2048
Total Inodes:                                   2048
Total Fragments:                                2048
```

The root inode for any filesystem is always 2. Switch to that inode in **fsdb** by running the **2i** subcommand:

```
2i
i#:      2  md: d-g-rwxr-xr-x  ln:    4  uid:    3  gid:    3
szh:        0  szl:      512  (actual size:      512)
a0: 0x09         a1: 0x00         a2: 0x00         a3: 0x00
a4: 0x00         a5: 0x00         a6: 0x00         a7: 0x00
at: Mon Nov 08 11:56:11 2004
mt: Mon Nov 08 11:56:17 2004
ct: Mon Nov 08 11:56:17 2004
```

Get a directory listing by running the **fd** subcommand:

```
fd
d0 (slot=0):    2  . (d_reclen/d_namlen = 12/1)
d1 (slot=12):    2  .. (d_reclen/d_namlen = 12/2)
d2 (slot=24):   16  lost+found (d_reclen/d_namlen = 20/10)
d3 (slot=44):   17  mydir (d_reclen/d_namlen = 468/5)
d4 (slot=512):    0   (d_reclen/d_namlen = 512/0)
d5 (slot=1024):    0   (d_reclen/d_namlen = 512/0)
d6 (slot=1536):    0   (d_reclen/d_namlen = 512/0)
d7 (slot=2048):    0   (d_reclen/d_namlen = 512/0)
d8 (slot=2560):    0   (d_reclen/d_namlen = 512/0)
d9 (slot=3072):    0   (d_reclen/d_namlen = 512/0)
d10 (slot=3584):    0   (d_reclen/d_namlen = 512/0)
```

The *mydir* subdirectory is at inode 17. Switch to inode 17 by running the **17i** subcommand:

```
17i
i#:     17  md: d-g-rwxr-xr-x  ln:    2  uid:    0  gid:    3
szh:        0  szl:      512  (actual size:      512)
a0: 0x0a         a1: 0x00         a2: 0x00         a3: 0x00
a4: 0x00         a5: 0x00         a6: 0x00         a7: 0x00
at: Mon Nov 08 11:56:39 2004
mt: Mon Nov 08 11:57:13 2004
ct: Mon Nov 08 11:57:13 2004
```

In this directory *myfile* can be found. We need to find out at what offset of the directory this file name occurs. The **fc** subcommand can be used for this purpose:

```
fc

0x000000a000:   \0 \0 \0 \? \0 \? \0 \? .  \0 \0 \0 \0 \0 \0 \?
0x000000a010:   \? (  \0 \? .  .  \0 \0 \0 \0 \0 \? \? F  \0 \?
0x000000a020:   m  y  f  i  l  e  \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0x000000a030:   \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0x000000a040:   \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0x000000a050:   \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0x000000a060:   \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0x000000a070:   \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0x000000a080:   \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0x000000a090:   \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0x000000a0a0:   \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0x000000a0b0:   \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0x000000a0c0:   \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0x000000a0d0:   \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
…
```

The previous output shows that the string *myfile* starts at 0x000000a020 in this particular inode.

Please note that the previous output can be extremely large. Therefore, you can use the **grep** command to get the required information (note the double spaces between the file name letters):

```
# echo 17i fc | fsdb /dev/lv07 | grep "m  y  f  i  l  e"
0x000000a020:   m  y  f  i  l  e  \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
```

At this point we can get a hex dump of the inode with the **fx** subcommand:

```
fx

0x000000a000:    0000  0011  000C  0001  2E00  0000  0000  0002
0x000000a010:    01F4  0002  2E2E  0000  0000  0012  01E8  0006
0x000000a020:    6D79  6669  6C65  0000  0000  0000  0000  0000
0x000000a030:    0000  0000  0000  0000  0000  0000  0000  0000
0x000000a040:    0000  0000  0000  0000  0000  0000  0000  0000
0x000000a050:    0000  0000  0000  0000  0000  0000  0000  0000
0x000000a060:    0000  0000  0000  0000  0000  0000  0000  0000
0x000000a070:    0000  0000  0000  0000  0000  0000  0000  0000
0x000000a080:    0000  0000  0000  0000  0000  0000  0000  0000
0x000000a090:    0000  0000  0000  0000  0000  0000  0000  0000
0x000000a0a0:    0000  0000  0000  0000  0000  0000  0000  0000
0x000000a0b0:    0000  0000  0000  0000  0000  0000  0000  0000
0x000000a0c0:    0000  0000  0000  0000  0000  0000  0000  0000
…
```

The name of the file is displayed in hex at offset 0x000000a020 as we found before. This is 6D7966696C65. The previous two bytes (0006, at the offset 0x000000a01e, on the previous line) refer to the number of bytes in the name, and the two bytes previous to those (01E8, at offset 0x000000a01c) are the pointer to the next reference. You can ignore those bytes, but look at the previous four bytes (0000 0012, at offset 0x000000a018), which are the inode number of this file, in hex. Hex 12 in decimal is 18.

Also note that the previous output can be extremely large. Therefore, you can use the **grep** command to get the required information:

```
# echo 17i fx | fsdb /dev/lv07 | grep "0x000000a0[1,2]"
0x000000a010:    01F4  0002  2E2E  0000  0000  0012  01E8  0006
0x000000a020:    6D79  6669  6C65  0000  0000  0000  0000  0000
```

Switch to the inode that we discovered in the previous step, which is 18, by entering the **18i** subcommand in **fsdb**:

```
18i
i#:      18  md: f---rw-r--r--  ln:    0  uid:    0  gid:    3
szh:         0  szl:    24169  (actual size:    24169)
a0: 0x0b         a1: 0x0c         a2: 0x0d         a3: 0x0e
a4: 0x0f         a5: 0x10         a6: 0x00         a7: 0x00
at: Mon Nov 08 11:56:36 2004
mt: Mon Nov 08 11:56:37 2004
ct: Mon Nov 08 11:57:13 2004
```

We can see the original permissions (-rw-r—r--), the owner (root uid 0), the group (sys gid 3), and the size of the file (24,169 bytes). Notice the ‘ln:  0’. This is the link count, and it is zero. We’ll recover this file by setting the link count to 1:

```
ln=1
0x0000020908  :  0x00000001 (1)
```

Running the **fsck** command on */dev/lv07* should recover the file into the *lost+found* directory:

```
# fsck /dev/lv07


** Checking /dev/rlv07 (/test)
** Phase 1 - Check Blocks and Sizes
```

```
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
Unreferenced file  I=18  owner=root mode=100644
size=24169 mtime=Nov 08 11:56 2004 ; RECONNECT? y
** Phase 5 - Check Inode Map
Bad Inode Map; SALVAGE? y
** Phase 5b - Salvage Inode Map
** Phase 6 - Check Block Map
Bad Block Map; SALVAGE? y
** Phase 6b - Salvage Block Map
10 files 656 blocks 15728 free
***** Filesystem was modified *****
```

In the previous **fsck**, you needed to answer 'Y' when prompted to reconnect and salvage in order to recover the file. Alternatively, you can use the **-y** switch with the **fsck** command to answer yes automatically.

Mount the */test* filesystem (*/dev/lv07*), and the file *myfile* is recovered into a file in */test/lost+found* called by the name of the inode, which in our case is 18:

```
# ls -l /test/lost+found
total 48
-rw-r--r--   1 root     sys           24169 Nov 08 11:56 18
```

Finally, the file can now be renamed into its original name */test/mydir/myfile*:

```
# mv /test/lost+found/18  /test/mydir/myfile
```

If you deleted more than one file, you will have to repeat this procedure for each file you need to recover.

For JFS2 file systems the procedure will differ slightly, especially regarding the **fsdb** subcommands that are used to recover the file. I will discuss this in a future article in *AIX Update*.

---

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


---

# Understanding the split command

Have you ever had the need to move a very large file from one workstation to another without having a network available? Or perhaps you need to store a tar archive file on CDs or diskettes. You might even need to send a large file via e-mail, but your e-mail server has a size limit for each attachment.

How can you accomplish these tasks? The AIX **split** command can help.

The **split** command takes as input a file name, either a text file or binary, and splits it into smaller, more manageable files that can be stored on other media or sent via e-mail as attachments in multiple mailings.

You or other users can then use the AIX **cat** command to concatenate the smaller files back into a new copy of the source file.

In addition to the above tasks, you could also use **split** to make text reviews easier, or even split large files and separate the output files as part of a security strategy.

## SPLIT COMMAND BASICS

The default behaviour of **split** is to break the input file into 1000-line output files. However, there are flags to tell the **split** command to specify a different line count, or for binary files a byte count.

The default output file names contain three characters, the first being *xaa*, and continuing with *xab*, *xac*, etc, up to *xzz*. The *x* is considered the prefix and the *aa*, *ab*, etc, is the suffix. The **split** command has flags that can change the prefix to any name you like, and change the suffix to use a greater number of characters to accommodate a larger number of output files. The syntax for the **split** command is described later.

The suffix specification was designed for easy reassembly using the **cat** command because of the alphabetically ordered arrangement of the file names. When experimenting with the **split** command, you may use the AIX **diff** command between the original and the restored versions to ensure that the **split** and **cat** commands worked as desired.

## SYNTAX

Typical syntax for the **split** command is:

```
split   -b Num [ k | m ]  [ -a Suff_Lgth ]  [ File [ Prefix ] ]
split   [ -l LineCnt ]    [ -a Suff_Lgth ]  [ File [ Prefix ] ]
```

where:

- -b Num tells **split** the number of bytes each output file is to contain. Adding the k or m after the number tells **split** to break the input file into output files, each containing that number of kilobytes or megabytes.
- -l LineCnt tells **split** to break the input file into output files, each containing that number of lines.
- -a Suff_Lgth tells **split** how many characters to use in the suffix portion of the output file names. The more characters in the suffix, the more output files may be generated. If your output is likely to contain more than 676 files (26 x 26) you could use the **-a 3** flag, which would tell **split** to use up to 17,576 (26 x 26 x 26) output files.
- File tells **split** the input file name.
- Prefix tells **split** the desired fixed prefix to add to each suffix in the output file names.

## EXAMPLES

### E-mail

Many e-mail servers have limits placed on the size of attachments you can send or receive, such as 10MB. This is

to help prevent e-mails containing large attachments from congesting the server. But what if you had a binary file containing 12MB that you needed to send to a colleague? Entering **split** with the **-b** flag would take care of this.

Suppose file *reference.pdf* was 12MB in size. The command:

```
split -b 6m reference.pdf refer_
```

would create two files called *refer_aa* and *refer_ab*, each of which would be about 6MB. You do not have to split the file evenly. The command:

```
split -b 10m reference.pdf refer_
```

would create a 10MB file called *refer_aa* and a file called *refer_ab* of about 2MB, each of which could then be e-mailed without exceeding the server-specified limit. The recipient would then put both files together on the same drive and enter:

```
cat refer_* > reference.pdf
```

to restore the original file.

Note: **split** creates files with sizes on typical AIX byte boundaries. For the example above, telling **split** to create a 10MB file would actually create a file of 10,485,760 bytes. If the e-mail program actually restricts files to 10,000,000 bytes, you would need to specify a lower value.

### Diskettes

Let's say you had a large file that you needed to copy from one machine to another and you did not have a network connecting them or you needed to send media containing the file elsewhere. Using the file from the previous example, entering:

```
split -1m reference.pdf
```

would yield 12 files – xaa, xab, xac, xad, xae, xaf, xag, xah, xai, xaj, xak, and xal – each of which would be about 1MB in size and could easily fit onto a diskette. Copy each file to a diskette and you can move the smaller output files to another machine

or elsewhere to be reconstructed into the input file using the **cat** command:

```
cat x?? > reference.pdf
```

### CDs

A similar example would be for CDs. Suppose you had a 3GB tar archive file named backup.tar.Z that you needed to store on CDs. Entering:

```
split -b 650m backup.tar.Z back_
```

would yield five files named back_aa, back_ab, back_ac, back_ad, and back_ae. The first four would be about 650MB in size, and the latter would be about 400MB. Writing the files to CDs and then later copying the files to another drive would enable you to recreate the tar file by entering:

```
cat back_* > backup.tar.Z
```

### Reviews

Suppose you had a large multi-page document called *experiment_raw_data* formatted to print 66 lines per page that you need reviewed by multiple reviewers. If the document had 3600 lines and you were to enter:

```
split -l 660 experiment_raw_data exper_
```

you would get the following six files:

- exper_aa
- exper_ab
- exper_ac
- exper_ad
- exper_ae
- exper_af.

Each file would contain 660 lines (10 pages) except for the last, which would contain 300 lines (about 4.5 pages). Six

reviewers could then review a reasonable amount of data and make their corrections. When the files are returned, the command:

```
cat exper_* > experiment_raw_data
```

would reassemble the final report.

## Security

Here is an interesting way to secure a large file. Suppose you had a 9MB executable file called beta_driver. If you were to enter:

```
split -b 10k -a 3 beta_driver
```

you would get 880 files named xaaa, xaab, xaac, etc, up to xbhv. Suppose you moved all the files ending with the letter 'f' to a diskette. In other words, files xaaf, xabf, xacf, xadf, etc would effectively be removed and stored off-line. Then, anyone who wanted to recreate your executable would need that diskette with the 34 extracted files. Without those secured files they would just have hundreds of segments with every 26th one removed.

To restore the executable after the 34 files are put back, enter:

```
cat x??? > beta_driver
```

## SUMMARY

Working with very large files on AIX does not have to be problematic if you use the **split** command to break them up into more manageable pieces, and then use the **cat** command to restore the original files.

*David Chakmakian*
*Software Engineer (USA)*

# Creating load on a machine to measure performance

Creating load on a machine to measure performance is a difficult task; it often also means creating load elsewhere, for instance on the filesystem or I/O of the machine.

The following one-liner creates mostly CPU load by calculating prime numbers:

```
u=$(date); i=1; while [ $i -lt 10000 ]; do  if [ $( factor $i|wc -w) -lt
3 ]; then echo $i; fi;  ((i+=1)); done ; echo $u; date
```

It also shows when the script started and when it finished. Usually I just run three or four of these scripts in several X-Windows on one machine or fork them to background.

I tested the script on the following OSs:

- AIX 4.3.2.0
- AIX 5.1.0 ML 03
- AIX 5.2.0 ML 03
- Suse Linux Kernel 2.4.

*Robert Kaiser*
*Systems Analyst*
*Bayerischer Rundfunk (Germany)*



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. We are also looking for explanatory articles, and hints and tips, from experienced users.

We will publish your article (after vetting by our expert panel) and send you a cheque, as payment, and two copies of the issue containing the article. Articles can be of any length and should be e-mailed to the editor, Trevor Eddolls, at trevore@xephon.com.

# AIX news

Cendura has announced Version 3.0 of Cohesion, which delivers proactive, policy-based, change, configuration, and compliance management across complex distributed enterprise infrastructures. The product provides service managers with an integrated way to automatically discover and track applications, create policies, compare and audit actual systems against reference systems, and manage change and configurations for hundreds of applications and services across the enterprise.

As well as AIX, the product runs on Windows, Linux, Solaris, and HP/UX.

For further information contact:
URL: www.cendura.com/news/091704.html.

* * *

DataCenter Technologies has released DC-Protect XA, its archiving and back-up software.

Its new Web interface allows users to protect reference data from tens-of-thousands AIX 4.3.3 (other Unix), PC, and Linux servers and clients.

The product is focused on archiving and back-up of file data from any client, anywhere on the network, to any type of disk-based storage pool. The solution uses a file fingerprint system to reduce back-up time, network traffic, and back-up storage requirements.

For further information contact:
URL: www.datacentertechnologies.com/index.php?option=content&task=view&id=358&Itemid=221.

* * *

Argus Systems Group has announced the availability of its PitBull trusted platform security products for AIX 5L Version 5.2. The PitBull product line provides a trusted platform that is designed to protect servers from malicious attacks that attempt to obtain or alter data and Web pages. Trusted operating systems protect servers containing or having access to classified or confidential information or monetary assets.

PitBull technology running on AIX allows AIX administrators to isolate system resources, applications, and administrative functions so that security holes in applications and CGI scripts cannot be exploited to gain system-wide access.

For further information contact:
URL: www.argus-systems.com./feature/currel/AIX52FD.shtml.

* * *

Midrange Performance Group has launched Power Navigator for capacity planning for AIX and Linux.

Initially, Power Navigator will help business partners and IBMers do capacity planning for AIX and Linux running on the pSeries or in an iSeries partition. Over time, however, MPG plans to incorporate all the features of Performance Navigator, eventually enabling users to do capacity planning, data modelling, and performance management for all the operating systems capable of running on the iSeries or pSeries.

For further information contact:
URL: www.mpginc.com

* * *