30

AIX

update

April 1998

In this issue

Published by

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

North American office

Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75067
USA
Telephone: 940 455 7050

Australian office

Xephon/RSM
GPO Box 6258
Halifax Street
Adelaide, SA 5000
Australia
Telephone: 08 223 1391

Contributions

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

Editor

Harold Lewis

Disclaimer

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

Subscriptions and back-issues

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

AIX Update on-line

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

Printed in England.

AIX 4.3's CacheFS feature

The Network File System (NFS) is one of AIX's most useful features. It provides a standard way of sharing files in a peer-to-peer network. NFS was developed by Sun Microsystems, and has been implemented on all major versions of Unix, as well as on a number of PC operating systems. The latest version, V3, was introduced by IBM in AIX 4.2.1 (see *AIX Version 4.2.1*, *AIX Update* Issue 22, page 47). CacheFS is the latest NFS feature to be implemented on AIX, being introduced in AIX 4.3.

CacheFS is a mechanism that improves the performance of read operations on remotely mounted filesystems and CD-ROMs. It's client-based, and so doesn't require any changes to NFS servers. The facility divides files and directories resident on NFS-mounted partitions into pages, each 4 KB long, and keeps them on the local filesystem. When attempts are made to access cached files, CacheFS tries to serve the client with locally cached data whenever possible. No caching is performed on write operations, so that all changes made to cached filesystems are immediately implemented on the system on which the filesystem resides.

One of the most important functions of CacheFS is to ensure the consistency of files and directories cached from remote systems. CacheFS performs periodical consistency checks, comparing the last modification time of locally cached files to that of the same files on remote systems; if the two differ, all data and attributes in the cache are purged and CacheFS immediately retrieves new data from the remote system. The same consistency check is performed whenever the user performs an operation on a cached file or directory.

CacheFS is not a general-purpose NFS accelerator – although it improves the read performance of clients that use it, its greatest performance benefits are in reducing the load on heavily loaded NFS servers and in reducing the load on busy networks.

THE LIFE-CYCLE OF A CACHED FILESYSTEM

To use CacheFS, first create a filesystem to be used for caching. I recommend creating the volume on a volume group other than *rootvg*, preferably in a separate logical volume. The following command creates the filesystem */test* in the volume group *datavg*:

```
# crfs -v jfs -g datavg -a size=20000 -m /test
```

Next mount the filesystem:

```
# mount /test
```

Use the command below to check the filesystem and ensure it's working:

```
# df -k /test

Filesystem  1K blocks  Free   %Used  Iused  %Iused  Mounted on
/dev/lv02    12288     11732    5%    17      1%    /test
```

Create the cache directory structure:

```
cfsadmin -c /test/cache
```

If you check the filesystem now, you should get the response below.

```
# df -k /test

Filesystem  1K blocks  Free   %Used  Iused  %Iused  Mounted on
/dev/lv02    12288     11640    6%    24      1%    /test

# ls -la /test/cache
total 184
d-----S---  3 root  sys    512 Jan 20 11:03 .
drwxr-sr-x  4 sys   sys    512 Jan 20 11:03 ..
-rw-------  1 root  sys     48 Jan 20 11:03 .cfs_label
-rw-------  1 root  sys     48 Jan 20 11:03 .cfs_label.dup
-rwx------  1 root  sys      0 Jan 20 11:03 .cfs_lock
drwx--S---  2 root  sys    512 Jan 20 11:03 .cfs_mnt_points
-rw-------  1 root  sys  69632 Jan 20 11:03 .cfs_resource
-rw-r--r--  1 root  sys     30 Jan 20 11:03 .nsr
#
```

The command **cfsadmin -c** creates a subdirectory and a number of hidden files in */test/cache*. We can now check the characteristics of the cache directory created:

```
# cfsadmin -l /test/cache

cfsadmin: list cache FS information
```

```
maxblocks      90%
minblocks       0%
threshblocks   85%
maxfiles       90%
minfiles        0%
threshfiles    85%
maxfilesize     3MB
```

The next thing to do is mount an exported filesystem from a suitable server (peer) using the newly created cache directory:

```
mount -V cachefs -o backfstype=nfs,cachedir=/test/cache server:/
usr/local /usr/local
```

Check that the filesystem has been mounted correctly:

```
# df
Filesystem     512-blocks  Free %Used  Iused %Iused Mounted on
/dev/hd4           73728   6424   92%   2066    12% /
/dev/hd2         1638400  56872   97%  33223    17% /usr
/dev/hd9var        24576  13648   45%    814    27% /var
/dev/hd3           49152  37456   24%    155     3% /tmp
/dev/hd1           40960  19016   54%   1212    24% /home
/dev/lv02          24576  22960    7%     43     2% /test
server:/usr/local 729088   8328   99%      -     - /test/cache/
                        .cfs_mnt_points/_usr_local
server:/usr/local 729088   8328   99%      -     -  /usr/local
```

As you can see, the directory *server:/usr/local* is mounted on two local filesystems – one copy is in the cache filesystem and the other is in the regular filesystem.

You can test that the filesystem is working by running a suitable command on it. In the example below I've used the command **time** on a 970 KB file.

```
# time sum pg.tar.gz
11576   954 pg.tar.gz
0.4u 0.1s 0:03 16% 5+71k 0+0io 5pf+0w

# time sum pg.tar.gz
11576   954 pg.tar.gz
0.2u 0.0s 0:00 100% 14+79k 0+0io 0pf+0w

# time sum pg.tar.gz
11576   954 pg.tar.gz
0.2u 0.0s 0:00 94% 6+80k 0+0io 0pf+0w

# time sum pg.tar.gz
11576   954 pg.tar.gz
```

```
0.3u 0.0s 0:00 92% 11+88k 0+0io 0pf+0w

# time sum pg.tar.gz
11576   954 pg.tar.gz
0.2u 0.1s 0:00 97% 14+77k 0+0io 0pf+0w
```

The command was issued a number of times to test the effect of caching. The output of the **time** command (the third line of each block) displays a summary of the CPU consumption of the current shell and its children (plus a few other statistics). The first two items show the number of seconds of CPU time accounted for by the user process and system. The third item shows the elapsed time (the sum of user, system, and I/O wait time) and the fourth item shows the total user and system CPU time as a percentage of elapsed time. As you can see, the user time (the first item) on the second and subsequent executions of the command is half that of the first execution. System and idle time are totally eliminated.

After executing some commands on the cached filesystem the status of the cache directory is:

```
# df -k /test

Filesystem  1K blocks  Free   %Used  Iused  %Iused  Mounted on
/dev/lv02    12288     3640     71%   582     15%   /test
```

The cache is now filled with pages from the mounted filesystem.

Let’s have a closer look at commands that manage CacheFS.

/USR/SBIN/CFSADMIN

The **cfsadmin** command performs the following operations:

- Creating a cached filesystem
- Deleting a cached filesystem
- Listing the contents and statistics of a cached filesystem
- Changing the parameters of a cached filesystem
- Reporting the status of a cached filesystem.

The syntax of the various forms of **cfsadmin** are shown below.

```
cfsadmin -c [-o parameter-list] cachedir
```

```
cfsadmin -d [cacheID|all] cachedir

cfsadmin -l cachedir

cfsadmin -u [-o parameter-list] cachedir

cfsadmin -s [mountpoint1 ... | all]
```

The available options are:

-c Creates a cache directory under *cachedir*.

-d Removes caching either for directory *cacheID* or for all cached directories under *cachedir*.

-l Lists filesystems backed up by *cachedir* and also provides cache resource parameters and details of current resource utilization.

-u Updates cache resource parameters. For the *cachcedir* that's currently utilized, parameters can only be increased; to decrease parameters, all directories should be unmounted before the change is applied.

-s Performs a consistency check either for all mount points or the ones specified.

The following list summarizes CacheFS's resource parameters.

- *maxblocks=n*
  The maximum percentage of blocks in the cache filesystem that can be utilized by CacheFS. The default is 90% (*maxblocks=90*).
- *minblocks=n*
  The minimum percentage of blocks in the cache filesystem that can be utilized by CacheFS. The default is 0% (*minblocks=0*).
- *threshblocks=n*
  The percentage of blocks in the cache filesystem beyond which CacheFS cannot claim resources once its block usage has reached the level specified by *minblocks*. The default value is 85% (*threshblocks=85*).
- *maxfiles=n*
  The maximum percentage of inodes in the cache filesystem that CacheFS can use. The default is 90% (*maxfiles=90*).

- *minfiles=n*
  The minimum percentage of inodes in the cache filesystem that CacheFS can use. The default is 0% (*minfiles=0*).
- *threshfiles=n*
  The percentage of inodes in the cache filesystem beyond which CacheFS cannot claim resources once its inode usage has reached the level specified by *minfiles*. The default value is 85% (*threshfiles=85*).
- *maxfilesize=n*
  The largest size of file, in megabytes, that CacheFS is allowed to cache. The default is 3 megabytes (*maxfilesize=3*).

/USR/SBIN/MOUNT

The **mount** command has special variants for mounting cached directories. Their format is:

```
/usr/sbin/mount -V cachefs -o cachefsoptions nfs_server:
remote_dir local_dir
```

The following list summarizes the CacheFS-specific options of the **mount** command:

- *backfstype=fstype*
  The file type of cached filesystem. The options currently available are *nfs* or *cdrfs*.
- *backpath=path*
  Specifies where the filesystem is currently mounted.
- *cachedir=dir*
  The name of the cache directory.
- *cacheID=id*
  id specifies an instance of a cache. If you do not specify a cache, CacheFS constructs one on your behalf.
- *write-around|non-shared*
  The option *write-around* (which is the default) causes all writes to be immediately performed on the remote system. The option

non-shared specifies that the mounted filesystem is used only by the system that performs the mount.

- *noconst*
  Disables consistency checking. Used only when it is known that the mounted filesystem will not be modified.
- *local_access*
  Specifies that access checks for cached files are to be performed on the local system.
- *purge*
  Specifies that all cached information is to be purged from the local system.
- *rw|ro*
  Specifies whether access is to be *read/write* (the default) or *read-only*.
- *suid|nosuid*
  Allow (default) or disallow the execution of binary files that have the *set-uid* bit set.
- *acregmin=n*
  Specifies the minimum number of seconds in which to test modifications to files before fetching them again from the remote system. The default is 30 seconds (*acregmin=30*).
- *acregmax=n*
  Specifies the maximum number of seconds to hold files in cache before fetching them again from the remote system. The default is 30 seconds (*acregmax=30*).
- *acdirmin=n*
  Specifies the minimum number of seconds in which to test modifications to directories before fetching them again from the remote system. The default is 30 seconds (*acdirmin=30*).
- *acdirmax=n*
  Specifies the maximum number of seconds to hold directories in cache before fetching them again from the remote system. The default is 30 seconds (*acdirmax=30*).

- *actimeo=n*
  Sets the parameters *acregmin*, *acregmax*, *acdirmin*, and *acdirmax* to *n*.

/USR/SBIN/FSCK_CACHEFS

The **fsck_cachefs** command checks the integrity of data cached with CacheFS. The format of the command is:

```
fsck_cachefs cachefs -[ -m ] [-o noclean] cachedir
```

The list below summarizes **fsck_cachefs**'s options.

- *-m*
  Check but do not repair *cachedir*.
- *-o noclean*
  Check the cache even if no error is suspected.

The following *fast path* arguments take you to the **smit**, **smitty**, and **wsm** screens that perform various CacheFS tasks.

- *cachefs_admin_create*
  Set up a cache
- *cachefs_mount*
  Mount filesystem using CacheFS
- *cachefs_admin_change*
  Modify cache characteristics
- *cachefs_admin_change*
  Display cache data
- *cachefs_admin_check*
  Check filesystem integrity
- *cachefs_admin_remove*
  Remove a cache.

A WORD OF CAUTION

Although I was able to test the basic functionality of CacheFS on my newly installed AIX 4.3 system, I couldn't verify that all the features

(such as performance logging and statistics) were implemented. Some of the commands supplied with the OS (**cachefswssize**, **cachefslog**, etc) were definitely not functioning, while others (such as **cachefsstat**) were not even available! However, I understand that fixes and updates will be forthcoming from IBM in the near future.

A Polak
System Engineer
APS (Israel)

Tracking system configuration

One of the most important responsibilities of any AIX system administrator is keeping track of system configuration. The information that has to be collected and maintained up-to-date includes 'netnames', ip addresses, the number, type, and size of disk devices, volume group and logical volume configuration and location, and network routing information, to name but a few. If the administrator is in charge of one or two Unix systems, or all the systems are identically configured, this is not too daunting a task. However, if your Unix environment is anything like mine – 15 systems from four different vendors (and rising), each of which is configured differently – this becomes a considerable task.

To keep track of my environment I've written a script that prepares a single report describing each system's configuration. I run this report once a month (via **cron**), also running it whenever I carry out maintenance that may affect configuration, when I upgrade software, add or remove logical volumes or volume groups, and add or remove hardware. I keep five back copies of this report to allow me to establish trends in filesystem use and to determine how changes in system configuration may affect performance. I've often referred to these reports to answer questions from vendors, users, and management, and I always consider whether collecting additional information

would help me in handling bugs or performance problems whenever I'm confronted with such problems.

This script (**configstatus**) collects system configuration data using standard administrator commands. In some instances the output is directed 'as-is' to the report, while in others it's reformatted in a more meaningful way before being placed on the report.

I've found the information in this report invaluable on a number of occasions. The information is particularly useful for determining when additional capacity, such as disk drives, is going to be required. This allows me to have additional capacity in place when it's required, and not when users report performance problems. The information has also assisted me in configuring new systems that are to have similar characteristics and parameters to existing ones.

Below is a full listing of the script configstatus, followed by an edited sample report .

CONFIGSTATUS

```
#!/bin/ksh
#
#  configstatus
#
#  Reports the characteristics and status of an AIX system.
#
#  jls  10/03/96
#
OUT=config.out

echo "            $(hostname) Logical Volume and System Status" > $OUT
echo "  Created by configstatus on $(date)" >> $OUT
echo "                   $(uname -a)" >> $OUT
IP=$(grep $(hostname) /etc/hosts|cut -f1)
echo "                          $IP" >> $OUT
echo >> $OUT

echo "************************** " >> $OUT
echo "Current Disk Usage (df -k): " >> $OUT
echo "************************** " >> $OUT
df -k >> $OUT
echo >> $OUT

# Display memory installed
```

```
M=0
for x in $(lscfg|grep '^. mem'|cut -d' ' -f2|tr '\012' ' ')
do
  s=$(lsattr -E -l $x|grep '^size'|tr -s ' '|cut -d' ' -f2)
  M=`expr $M + $s`
done
echo "Memory:" >> $OUT
echo "         $M MB Installed." >> $OUT
echo >> $OUT

# Display Volume Group Information
echo  >> $OUT
echo "* Space Allocation By Volume Group *" >> $OUT

for vg in $(lsvg)
do
  lsvg -p $vg|grep -v :|grep -v PV_NAME|awk '
  BEGIN {vg="'$vg'"; tot=0; alloc=0; free=0 }
        {tot=tot+$3; free=free+$4}
  END   {tot=tot*4; free=free*4; alloc=tot-free
     printf("%-15s %6d MB total, %6d MB allocated, %6d MB
free.\n",vg,tot,alloc,free) } ' >> $OUT
done

echo  >> $OUT
echo "********************" >> $OUT
echo "Volume Group Detail" >> $OUT
echo "********************" >> $OUT

for vg in $(lsvg)
do
  echo "----------" >> $OUT
  echo " $vg" >> $OUT
  echo "----------" >> $OUT

  echo "* Logical Volume Summary in Volume Group $vg *" >> $OUT
  echo  >> $OUT
  lsvg -l $vg|grep -v :  >> $OUT

  echo >> $OUT
  echo "* Logical Volume Detail in Volume Group $vg *"  >> $OUT
  for lv in $(lsvg -l $vg|grep -v LV|grep -v :|cut -f1 -d" ")
  do
    lslv $lv|awk '
    ($1=="LOGICAL")      {lv=$3; vg=$6;next}
    ($1=="LV")           {per=$5;next}
    ($1=="VG")           {ls=$6;next}
    ($1=="TYPE:")        {tp=$2;next}
    ($1=="MAX")          {psz=$6+0;next}
    ($1=="COPIES:")      {mr=$2-1;pol=$5;next}
```

```
    ($1=="LPs:")         {lp=$2+0;pp=$4+0;lsz=lp*psz;next}
    ($1=="STALE")        {st=$3+0;next}
    ($3=="RELOCATABLE:") {rel=$4;next}
    ($1=="MOUNT")        {mnt=$3;next}
    ($1=="MIRROR")       {mwc=$4;next}
    ($1=="EACH")         {al=$9
      printf("\n")
      printf("Logical Volume:     %-20s",lv)
      printf("Volume Group:       %-15s\n",vg)
      printf("LV State:           %-20s",ls)
      printf("Type:               %-15s\n",tp)
      printf("LV Size (Mbytes):   %-20s",lsz)
      printf("Permissions:        %-15s\n",per)
      printf("Schedule Policy:    %-20s",pol)
      printf("Mirrors:            %-15s\n",mr)
      printf("Allocated LP:       %-20s",lp)
      printf("Allocated PP:       %-15s\n",pp)
      printf("Relocatable:        %-20s",rel)
      printf("Stale PPs:          %-15s\n",st)
      printf("MWC:                %-20s",mwc)
      printf("Strict Allocation:  %-15s\n",al)
      printf("Mount Point:        %-20s\n",mnt)    }
    ' >> $OUT
  echo >> $OUT
  done

  echo "* Physical Volumes In Volume Group $vg *" >> $OUT
  echo >> $OUT
  for pv in $(lsvg -p $vg|grep -v :|grep -v PV_NAME|cut -f1 -d" ")
  do
    lsvg -p $vg|grep -v : >> $OUT
    echo "** Logical Volume Distribution **" >> $OUT
    lspv -l $pv|grep -v : >> $OUT
    echo >> $OUT
  done
done

#
#  Display current sna information
#

# sna global information
echo >> $OUT
echo "*************** Current sna Information ********************" >>
$OUT

# sna server profiles
echo "sna Server Profiles:" >> $OUT

# dlc profiles
```

```
echo "    (exportsna -t sna_dlc_token_ring -UT)" >> $OUT
exportsna -t sna_dlc_token_ring -UT >> $OUT
echo >> $OUT

# link station profiles
echo "    (exportsna -t link_station_token_ring -UT)" >> $OUT
exportsna -t link_station_token_ring -UT >> $OUT
echo >> $OUT

# Global sna info
echo "    (sna -d g)" >> $OUT
sna -d g >> $OUT

# sna link info
echo >> $OUT
echo "       (sna -d l -o 'long')" >> $OUT
sna -d l -o 'long' >> $OUT

# sna APPN Topology
echo >> $OUT
echo "       (sna -d t -o 'long')" >> $OUT
sna -d t -o 'long' >> $OUT

# Token Ring Adapter Characteristics
echo >> $OUT
echo "*****************************************************" >> $OUT
echo "Token Ring Adapter Characteristics (lsattr -E -l tok0)" >> $OUT
echo "*****************************************************" >> $OUT
lsattr -E -l tok0 >> $OUT

# List all static routes
echo >> $OUT
echo "***************************" >> $OUT
echo "Static Routes (netstat -rn)" >> $OUT
echo "***************************" >> $OUT
netstat -rn >> $OUT
echo >> $OUT

#
#  Display Misc. System Information
#
echo "*********************************************" >> $OUT
echo "Current System Parameters (lsattr -E -l sys0):" >> $OUT
echo "*********************************************" >> $OUT
lsattr -E -l sys0 >> $OUT
echo >> $OUT

echo "***********************************" >> $OUT
echo "Current System Devices (lsdev -CH):" >> $OUT
echo "***********************************" >> $OUT
```

```
lsdev -CH |grep -v "^name"|sort -k1 >> $OUT
echo >> $OUT

echo "***********************************" >> $OUT
echo "Current crontab file (crontab -l): " >> $OUT
echo "***********************************" >> $OUT
crontab -l >> $OUT
echo >> $OUT

echo "                ** $(hostname) configstatus completed **" >> $OUT
```

SAMPLE OUTPUT

```
AIR2 Logical Volume and System Status
  Created by configstatus on Thu Oct 30 12:29:56 CST 1997
                  AIX AIR2 1 4 000536536700
                        199.162.1.100

**************************
Current Disk Usage (df -k):
**************************
Filesystem    1024-blocks      Free %Used    Iused %Iused Mounted on
/dev/hd4            28672     15288   47%     2134    27% /
/dev/hd2          1011712    139312   87%    28188    12% /usr
/dev/hd9var         40960     10064   76%      229     3% /var
/dev/hd3            24576     23488    5%       56     1% /tmp
/dev/homelv        204800     40400   81%     2295     5% /home

Memory:
         128 MB Installed.


* Space Allocation By Volume Group *
rootvg            1920 MB total,   1220 MB allocated,    700 MB free.
uservg            2148 MB total,    588 MB allocated,   1560 MB free.

*******************
Volume Group Detail
*******************
----------
 rootvg
----------
* Logical Volume Summary in Volume Group rootvg *

LV NAME        TYPE       LPs   PPs   PVs  LV STATE      MOUNT POINT
hd6            paging     32    32    1    open/syncd    N/A
hd5            boot       2     2     1    closed/syncd  /blv
hd8            jfslog     1     1     1    open/syncd    N/A
hd4            jfs        7     7     1    open/syncd    /
```

```
hd2             jfs         247   247   1    open/syncd    /usr
hd3             jfs         6     6     1    open/syncd    /tmp
hd9var          jfs         10    10    1    open/syncd    /var

* Logical Volume Detail in Volume Group rootvg *

Logical Volume:     hd6                 Volume Group:       rootvg
LV State:           opened/syncd        Type:               paging
LV Size (Mbytes):   128                 Permissions:        read/write
Schedule Policy:    parallel            Mirrors:            0
Allocated LP:       32                  Allocated PP:       32
Relocatable:        yes                 Stale PPs:          0
MWC:                off                 Strict Allocation:  yes
Mount Point:        N/A


Logical Volume:     hd5                 Volume Group:       rootvg
LV State:           closed/syncd        Type:               boot
LV Size (Mbytes):   8                   Permissions:        read/write
Schedule Policy:    parallel            Mirrors:            0
Allocated LP:       2                   Allocated PP:       2
Relocatable:        no                  Stale PPs:          0
MWC:                on                  Strict Allocation:  yes
Mount Point:        /blv
```

Similar reports are produced for other logical volumes.

```
* Physical Volumes In Volume Group rootvg *

PV_NAME      PV STATE    TOTAL PPs   FREE PPs    FREE DISTRIBUTION
hdisk0       active      480         175         52..64..00..14..45
** Logical Volume Distribution **
LV NAME          LPs   PPs   DISTRIBUTION          MOUNT POINT
hd5              2     2     02..00..00..00..00    /blv
hd9var           10    10    03..00..00..07..00    /var
hd2              247   247   39..00..88..70..50    /usr
hd6              32    32    00..32..00..00..00    N/A
hd8              1     1     00..00..01..00..00    N/A
hd4              7     7     00..00..07..00..00    /
hd3              6     6     00..00..00..05..01    /tmp
```

Similar reports are produced for other volume groups

```
************** Current sna Information ********************
sna Server Profiles:
    (exportsna -t sna_dlc_token_ring -UT)
```

```
sna_dlc_token_ring:
    prof_name                                   = "TLINKDEFAULT"
    datalink_device_name                        = "tok0"
    force_timeout                               = 120
    user_defined_max_i_field                    = no
    max_i_field_length                          = 30729
    max_active_link_stations                    = 100
    num_reserved_inbound_activation             = 0
    num_reserved_outbound_activation            = 0
    transmit_window_count                       = 16
    dynamic_window_increment                    = 1
    retransmit_count                            = 8
    receive_window_count                        = 8
    priority                                    = 0
    inact_timeout                               = 48
    response_timeout                            = 4
    acknowledgement_timeout                     = 1
    link_name                                   = ""
    local_sap                                   = 0x04
    retry_interval                              = 60
    retry_limit                                 = 20
    dynamic_link_station_supported              = yes
    trace_base_listen_link_station              = no
    trace_base_listen_link_station_format       = long
    dynamic_lnk_solicit_sscp_sessions           = yes
    dynamic_lnk_cp_cp_sessions_supported        = yes
    dynamic_lnk_cp_cp_session_support_required  = no
    dynamic_lnk_TG_effective_capacity           = 4300800
    dynamic_lnk_TG_connect_cost_per_time        = 0
    dynamic_lnk_TG_cost_per_byte                = 0
    dynamic_lnk_TG_security                     = nonsecure
    dynamic_lnk_TG_propagation_delay            = lan
    dynamic_lnk_TG_user_defined_1               = 128
    dynamic_lnk_TG_user_defined_2               = 128
    dynamic_lnk_TG_user_defined_3               = 128
    dynamic_lnk_hpr_support                     = yes
    comments                                    = ""
```

Similar reports are produced for other SNA servers.

```
    (exportsna -t link_station_token_ring -UT)

link_station_token_ring:
    prof_name                                   = "TDEFAULT"
    use_control_pt_xid                          = yes
    xid_node_id                                 = "*"
    sna_dlc_profile_name                        = ""
    stop_on_inactivity                          = no
    time_out_value                              = 0
```

```
    LU_registration_supported                    = no
    LU_registration_profile_name                 = ""
    link_tracing                                 = no
    trace_format                                 = long
    hpr_support                                  = yes
    access_routing_type                          = link_address
    remote_link_name                             = ""
    remote_link_address                          = ""
    mac_addr_format                              = canonical
    remote_sap                                   = 0x04
    call_out_on_activation                       = yes
    verify_adjacent_node                         = no
    net_id_of_adjacent_node                      = ""
    cp_name_of_adjacent_node                     = ""
    xid_node_id_of_adjacent_node                 = "*"
    node_type_of_adjacent_node                   = learn
    solicit_sscp_sessions                        = yes
    activate_link_during_system_init             = no
    activate_link_on_demand                      = no
    cp_cp_sessions_supported                     = yes
    cp_cp_session_support_required               = no
    adjacent_node_is_preferred_server            = no
    initial_tg_number                            = 0
    restart_on_normal_deactivation               = no
    restart_on_abnormal_deactivation             = no
    restart_on_activation                        = no
    TG_effective_capacity                        = 4300800
    TG_connect_cost_per_time                     = 0
    TG_cost_per_byte                             = 0
    TG_security                                  = nonsecure
    TG_propagation_delay                         = lan
    TG_user_defined_1                            = 128
    TG_user_defined_2                            = 128
    TG_user_defined_3                            = 128
    comments                                     = ""
```

Similar reports are produced for other links.

```
    (sna -d g)
************************************
       SNA Global Information
************************************
Status                                Active
Control point (CP) name               PGLNET.COMMPU
CP alias                              COMMPU
Node ID (for XID)                     X'00000000'
Node type                             End node (EN)
Max. number of cached routing trees   500
Max. number of nodes in the TDB       500
```

```
Route additional resistance             128
Number of licensed sessions             1000
Maximum number of conversations         1000
Implicit partner LU support?            Yes
NMVT action when no NMVT process        Reject
Control Point (CP) profile comment
Product version                         3.1.2.1
Local hostname (TCP/IP)                 AIR2
Time of last verified configuration     Tue Jul  1 14:26:52 1997

       (sna -d l -o 'long')
************************************
      Active Links Information
************************************
Number of active data link devices      3

1>Data link device name                 /dev/dlctoken/tok0
  DLC profile name                      tok0.00011
  Device type                           Token ring
  Line type                             Switched
  Local link name
  Maximum number of link stations       100
  Minimum number of inbound links       0
  Minimum number of outbound links      0
  Dynamic link stations supported?      No
  Number of active links on this device 2
  Total bytes transmitted               163548045
  Total bytes received                  106686810
  Total frames transmitted              205994
  Total frames received                 805476
  Max. transmits ever queued            30
  Bad frames received from adapter      0
  Number of frames not transmitted      0

1.1>Link station profile name           @tok0.8 (dynamic)
    Destination DLC address
    Remote link name                    (not applicable)
    Link tracing active?                No
    Verify adjacent node?               No
    Adjacent node CP name
    Adjacent node type
    Adjacent node ID (for XID)          X'00000000'
    CP-CP sessions supported?           No
    Solicit SSCP-PU session             No
    Host SSCP ID                        (not applicable)
    ANR Label                           X'8004'
    HPR Capability                      No HPR support
    Link activated                      Pending remote contact
    Link state                          Base listening
    Local active and activating session 0
```

```
    Number of ISR sessions              0
    Link in use by Generic SNA          No
    Deactivating link                   No
    Link station role                   Secondary
    Max frame data (BTU) size           99
    Transmission group number           255
    Effective capacity                  4.30 megabits per second
    Cost per connect time               0
    Cost per byte                       0
    Security                            Nonsecure
    Propagation delay                   384.00 microseconds (lan)
    User defined parameter 1            128
    User defined parameter 2            128
    User defined parameter 3            128
    Test commands sent                  0
    Test command failures               0
    Test commands received              0
    Data frames transmitted             0
    Data frames retransmitted           0
    Max. contiguous frames retrans.     0
    Data frames received                0
    Invalid frames received             0
    Adapter detected receive errors     0
    Adapter detected transmit errors    0
    Received inactivity timeouts        0
    Primary polls sent                  0
    Primary repolls sent                0
    Max. primary contiguous repolls     0
```

Similar reports are produced for other active links.

```
       (sna -d t -o 'long')
************************************
        Topology Information
************************************
Number of nodes                       2

1>Node name                           .
  Node type                           End node (EN)
  Node quiescing?                     No
  Route additional resistance         255
  Congested?                          No
  Intermediate sessions (IS) supported? No
  IS resources depleted?              Yes
  End point resources depleted?       Yes
  APPN gateway supported?             No
  Central directory supported?        No
  Number of TGs                       3
```

```
1.1>TG partner CP name                  PGLNET.COMMPU
    Transmission group number           21
    TG partner node type                Real
    Operational?                        Yes
    Activate on demand?                 No
    TG quiescing?                       No
    CP-CP sessions supported?           No
    TG type                             End point
    Effective capacity                  4.30 megabits per second
    Cost per connect time               0
    Cost per byte                       0
    Security                            Nonsecure
    Propagation delay                   384.00 microseconds (lan)
    User defined parameter 1            128
    User defined parameter 2            128
    User defined parameter 3            128
    DLC data                            (not applicable)
```

Similar reports are produced for other nodes and partners.

```
******************************************************
Token Ring Adapter Characteristics (lsattr -E -l tok0)
******************************************************
bus_intr_lvl  4              Bus interrupt level                False
intr_priority 3              Interrupt priority                 False
xmt_que_size  160            TRANSMIT queue size                True
bus_io_addr   0x86a0         Bus I/O address                    False
dma_lvl       5              DMA arbitration level              False
dma_bus_mem   0x504000       Address of bus memory used for DMA  False
ring_speed    16             RING speed                         True
attn_mac      no             Receive ATTENTION MAC frame        True
beacon_mac    no             Receive BEACON MAC frame           True
use_alt_addr  yes            Enable ALTERNATE TOKEN RING address True
alt_addr      0x400000005006 ALTERNATE TOKEN RING address       True

**************************
Static Routes (netstat -rn)
**************************
Routing tables
Destination      Gateway           Flags     Refs     Use  Interface
Netmasks:
255.0.0
255.255.255

Route Tree for Protocol Family 2:
default          201.126.1.1       UG         10   140037  tr0
127              127.0.0.1         U           6    16183  lo0
201.126.1        201.126.1.103     U          11    25660  tr0
```

```
******************************************
Current System Parameters (lsattr -E -l sys0):
******************************************
keylock     normal State of system keylock at boot time
False
maxbuf      20     Max number of pages in block I/O BUFFER CACHE True
maxmbuf     32768  Max Kbytes of real memory allowed for MBUFS   True
maxuproc    1000   Max number of PROCESSES allowed per user      True
autorestart true   Automatically REBOOT system after a crash     True
iostat      true   Continuously maintain DISK I/O history        True
realmem     131072 Amount of usable physical memory in Kbytes    False
conslogin   enable System Console Login                          False
maxpout     32     HIGH mark for pending write I/Os per file     True
minpout     22     LOW mark for pending write I/Os per file      True
memscrub    false  Enable memory SCRUBBING                       True
fullcore    false  Enable full CORE dump                        True

**********************************
Current System Devices (lsdev -CH):
**********************************

aio0       Defined                Asynchronous I/O
bus0       Available 00-00        Microchannel Bus
cd0        Available 00-00-0S-2,0 SCSI Multimedia CD-ROM Drive
dlctoken   Available              Token-Ring Data Link Control
fd0        Available 00-00-0D-00  Diskette Drive
fda0       Available 00-00-0D     Standard I/O Diskette Adapter
gsna       Available              N/A
hd2        Defined                Logical volume
hd3        Defined                Logical volume
hd4        Defined                Logical volume
hd5        Defined                Logical volume
hd6        Defined                Logical volume
hd8        Defined                Logical volume
hd9var     Defined                Logical volume
hdisk0     Available 00-00-0S-0,0 2.0 GB SCSI Disk Drive
hdisk1     Available 00-00-0S-1,0 2.2 GB SCSI Disk Drive
homelv     Defined                Logical volume
inet0      Available              Internet Network Extension
ioplanar0  Available 00-00        I/O Planar
lo0        Available              Loopback Network Interface
loglv00    Defined                Logical volume
lvdd       Available              LVM Device Driver
mem0       Available 00-0D        64 MB Memory Card
mem1       Available 00-0H        64 MB Memory Card
paging00   Defined                Logical volume
ppa0       Available 00-00-0P     Standard I/O Parallel Port Adapter
proc0      Available 00-00        Processor
pty0       Available              Asynchronous Pseudo-Terminal
rmt0       Available 00-08-00-3,0 5.0 GB 8mm Tape Drive
```

```
rootvg     Defined                 Volume group
sa0        Available 00-00-S1      Standard I/O Serial Port 1
sa1        Available 00-00-S2      Standard I/O Serial Port 2
sa2        Available 00-05         8-Port Asynchronous Adapter EIA-232
scsi0      Available 00-00-0S      Standard SCSI I/O Controller
scsi1      Available 00-08         SCSI I/O Controller
sio0       Available 00-00         Standard I/O Planar
siokta0    Available 00-00-0K      Keyboard/Tablet Adapter
sioma0     Available 00-00-0M      Mouse Adapter
sna        Available               N/A
snam       Available               N/A
snap       Available               N/A
sys0       Available 00-00         System Object
sysplanar0 Available 00-00         CPU Planar
sysunit0   Available 00-00         System Unit
tok0       Available 00-01         Token-Ring HPerf Adapter (8fc8)
tr0        Available               Token Ring Network Interface
tty0       Available 00-00-S1-00   Asynchronous Terminal
tty1       Available 00-05-01-00   Asynchronous Terminal
tty2       Available 00-05-01-01   Asynchronous Terminal
uservg     Defined                 Volume group
vpack01    Available 00-06-01      Digital Trunk Processor
vpack02    Available 00-06-02      Digital Trunk Processor
vsca0      Available 00-06         Digital Trunk Dual Adapter
vsca1      Available 00-07         Digital Trunk Dual Adapter

**********************************
Current crontab file (crontab -l):
**********************************
# @(#)08    1.15.1.3  src/bos/usr/sbin/cron/root, cmdcntl, bos411,
9428A410j 2/11/94 17:19:47
#####
# added local scripts. KEPT commented scripts
#####
#
# COMPONENT_NAME: (CMDCNTL) commands needed for basic system needs
#
# FUNCTIONS:
#
# ORIGINS: 27
#
# (C) COPYRIGHT International Business Machines Corp. 1989,1994
# All Rights Reserved
# Licensed Materials - Property of IBM
#
# US Government Users Restricted Rights - Use, duplication or
# disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
#
#0 3 * * * /usr/sbin/skulker
#45 2 * * 0 /usr/lib/spell/compress
```

```
#45 23 * * * ulimit 5000; /usr/lib/smdemon.cleanu > /dev/null
0 11 * * * /usr/bin/errclear -d S,O 30
0 12 * * * /usr/bin/errclear -d H 90
#
# Local scripts follow:
#
30 00 1 * * /root/config.sh             # Run configuration reporter
00 01 * * * /root/daily.sh                   # Run this script daily
00 21 * * 0 /root/reboot.sh >> /sh.log 2>&1  # Reboot on Sunday at 9pm
00 03 * * 6 /root/trim.sh                    # Trim every Sat at 3 AM

               ** AIR2 configstatus completed **
```

Joel Shank
AIX Administrator
People's Energy Corporation (USA)

AIX large file support

In versions of AIX prior to AIX 4.1, the size of both a single file and an entire filesystem was limited to one byte short of two gigabytes. While AIX version 4.1 raised the limit on the size of filesystems, the limit on the size of files remains the same. This limit comes from the fact that the *off_t* data type is a signed 32-bit integer, and so the largest positive value that can be stored in a variable of this type is 2^{31} or 2,147,483,648 (2 GB). *off_t* is the data type of the offset file pointer, and it's defined in */usr/include/sys/types.h*. AIX 4.2's large file support, which allows files larger than 2 GB, is based on a 64-bit pointers. Support for this type is available in AIX 4.2's C compiler.

LARGE FILE-ENABLED FILESYSTEM

In order to support large files, a new filesystem structure was developed for the new AIX Journalled File System (AIX JFS 2). Only this filesystem and later versions support files larger than two gigabytes. Filesystems enabled for large files cannot be migrated to earlier

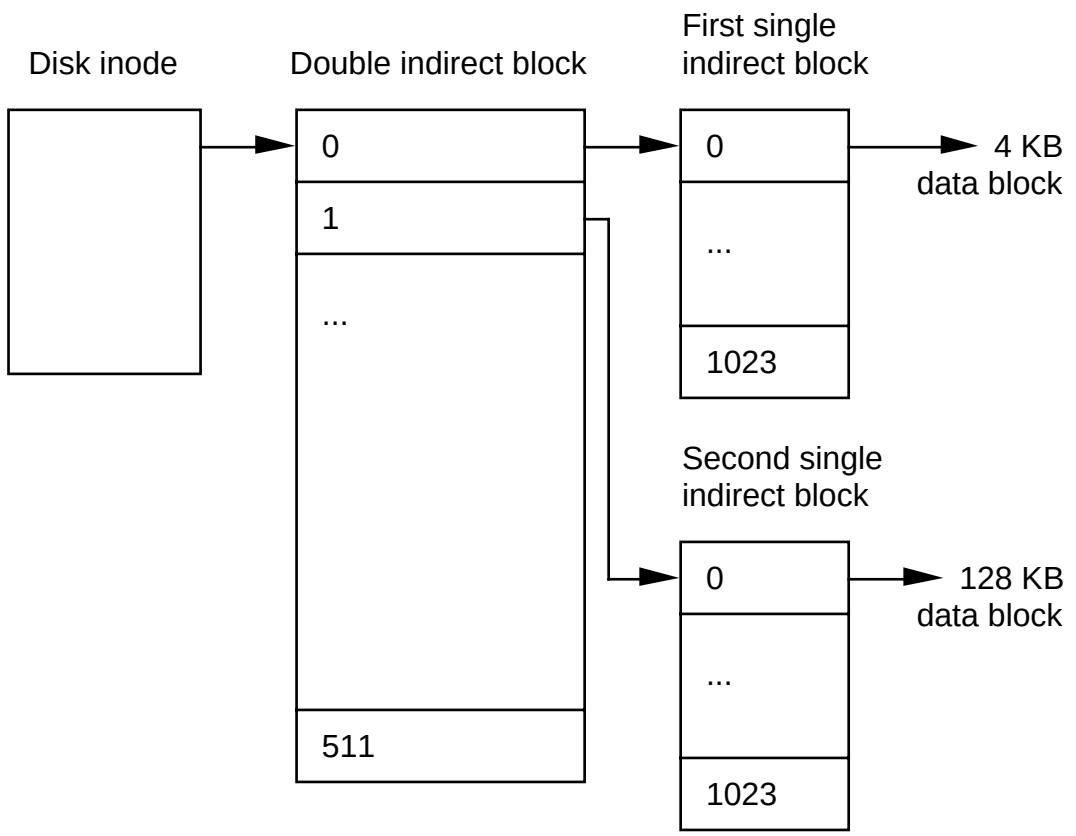

Figure 1: AIX's large file memory model

releases of AIX, which don't support the new 'superblock' structure. However, filesystems from AIX version 3 and 4.1 can be migrated forward.

The command to check the filesystem's version number is:

```
dumpfs <file system> | grep "filesystem version"
```

In filesystems enabled for large files, all data up to the first four-megabyte file offset is allocated in blocks of 4096 bytes (4 KB). Thereafter data is allocated in blocks of 128 KB.

Figure 1 above shows the new AIX file model. In this model the disk inode points to a 'double indirect block' – an array of 512 pointers. The first pointer in this array (pointer 0) points to the first 'single indirect block'. This is an array of 1024 pointers to 4 KB blocks of memory.

The next 511 pointers in the double indirect block (pointers 1 to 511) point to the second single indirect block onwards. Each of these is an array of 1024 pointers to 128 KB blocks of memory. The maximum file size in this set-up is therefore (1 x 1024 x 4 KB) + (511 x 1024 x 128 KB) or 68,589,453,312 bytes (about 64 GB). While this means that there is still a limit to the size of files, this limit is no longer a 'paltry' two gigabytes but a whopping 64 GB. Can anyone still have file size problems?

When adding a journalled filesystem that includes large files to AIX, use the new **-o** option **bf=true** in conjunction with the commands **crfs** and **mkfs**. If you use **smit** or **smitty**, make sure you set the option *Add a Large File Enabled Journalled File System*. I intend to discuss AIX 4.2's Journalled File Systems (JFS) in greater detail in a future article in *AIX Update*.

FILE SIZE LIMITS

The default largest file that a particular user's process can create or extend is set in the file */etc/security/limits* by the line that contains the variables *fsize_hard* and *fsize*. By default, *fsize_hard* is set to *fsize*, which in turn is set to 2,097,151 blocks when AIX is first installed. To be able to create or expand files larger than 2 GB, *fsize* must be changed. Note that *fsize* must be less than or equal to *fsize_hard*.

The **ulimit** command may also be used to change *fsize* up to the limit defined by *fsize_hard*. Setting *fsize* to a value of *-1* implies that it is 'unlimited'.

LARGE FILE PROGRAMMING INTERFACE

AIX's filesystem programming interfaces revolve around the definition of *off_t*. As stated earlier, in previous AIX releases this was a signed 32-bit integer, which resulted in a maximum file size one byte short of two gigabytes. New programming interfaces were defined in AIX 4.2 to allow applications to use large files. While these allow existing applications execute exactly as before, applications will not be able to access large files. In fact, even if they don't access large files, they may fail if they access files in filesystems that contain large files.

AIX 4.2 provides two different methods for applications to access large files. The first is to define *_LARGE_FILES*, which redefines all the relevant data types, structures, and subroutine names to their large file-enabled counterparts. The second is to re-code the application to call the large-enabled file subroutines explicitly.

Using *_LARGE_FILES* has the advantage of maximizing application portability, as the application still conforms to the normal POSIX and X/Open standards. This approach results in *off_t* being redefined as a signed 64-bit integer. In addition, all subroutines dealing with file size and offset are redefined in terms of their large-file enabled counterparts.

PROGRAMMING PITFALLS

The use of large file-enabled filesystems quickly exposes poor 32-bit coding practice. Consider the two code fragments below.

This is wrong:

```
int             file_size;
struct stat     s;
file_size = s.st_size;
```

This is right:

```
off_t           file_size;
struct stat     s;
file_size = s.st_size;
```

If an application attempts to store a file size or file offset in an *int* variable, the resulting value is truncated and loses significance.

The data type used by the *fseek()* and *ftell()* subroutines is *long* and cannot be re-cast to the appropriate 64-bit data type in the *_LARGE_FILES* environment. Application programs that access large files and use *fseek()* and/or *ftell()* need to be converted. The *fseeko()* and *ftello()* subroutines are functionally equivalent to *fseek()* and *ftell()* with the important exception that the offset is of type *off_t* rather than *long*. Make sure that you convert all variables that use offsets to the appropriate type. The code fragments below show how this should be handled.

Wrong:

```
long        offset;
fseek(fp, offset, SEEK_SET);
```

Right, but not ideal:

```
off_t       offset;
fseeko(fp, offset, SEEK_SET);
```

Ideal solution:

```
#ifndef        _LARGE_FILES
   long        offset;
   fseek(fp, offset, SEEK_SET);
#else
   off_t       offset;
   fseeko(fp, offset, SEEK_SET);
#endif
```

LARGE FILE COMMAND SUPPORT

Many – but not all – AIX 4.2 commands support large files. In addition some commands offer a limited degree of large file support.

Some commands, such as **date** and **kill**, don't need to access files and are therefore not affected. Some system files, such as */etc/passwd* and */etc/inittab*, are restricted to being smaller than two gigabytes, so that their associated commands (**mkuser** and **su**) don't need large file support. File headers whose format is restricted by industry standards may also be unable to use large files. For example:

- *XCOFF* defines the format of object files and executable files. Consequently commands such as **ld**, **as**, **strip**, etc, do not need to support large files.
- Archiving utilities, such as **cpio**, **tar**, and **pax**, are restricted in format and don't support the archiving of large files. AIX 4.2's **backup** and **restore** commands must be used to archive these files. If you've ever wondered why AIX 4.2's **mksysb** no longer uses **tar**, now you know!
- **pack**'s header format doesn't have enough characters to store files of over 2 GB.

The default AIX printer back-end **piobe** command doesn't support files larger than 2 GB. If you think about it, this isn't such a problem: if an average page contains 2000 characters and your printer has a print speed of 10 pages per minute, it'll take 1800 hours to complete a 2 GB print job. Increasing the print speed to 100 pages per minute and squeezing 100 minutes into every hour and 25 hours into every day still leaves you with over four days of printing!

Although a directory may contain large file entries, the directory itself may not be larger than 2 GB. Hence, commands such as **rmdir** don't support large directories.

The **awk** and **nawk** commands provide a good example of limited large file support. While both these commands are able to handle data files larger 2 GB, their scripts are restricted by the 2 GB limit. Another example of limited support is the **find** command, which can process files larger than 2 GB, but doesn't support the **-size** *number* flag where *number* is larger than 2 GB.

ADDITIONAL INFORMATION

Large file support is one of the key enhancements to AIX 4.2, and an entire chapter – *Large File Support* – is devoted to the subject in the RedBook *AIX 4.2 Differences Guide*, SG-244807.

Werner Klauser
Klauser Informatik (Switzerland)

Sending faxes from applications

Much time, effort, and expense is taken up in IS departments in printing and despatching statements, reminders, price lists, and other paperwork that isn't actually necessary and doesn't fulfil a legal requirement (that is, the paperwork isn't necessary – transmitting the

information itself may be vital to your business). In particular, paperwork requires you to buy specialized forms, envelopes, and printer supplies, to pay for postage, to load forms into printers, and to devote valuable staff to such ‘demanding’ tasks as stuffing envelopes and sorting mail by destination.

However, all this trouble and expense can be avoided by sending the output directly to the recipient by fax from the application that generated it. The best way to do this is to use a fax product that has an Application Programming Interface (API).

We ourselves installed one such product from a Swiss company, Advanced Computer Software (ACS). You can find out more about the company and its products from their Web site at *www.acs-software.ch*.

TELE COMMUNICATION SYSTEM (TCS)

The product is called TCS, and was developed for sending messages by fax, telex, e-mail, Comtext-fax, Comtext-telex, and Comtext mail.

The product is menu-driven and can run on AIX ASCII terminals as well as on Windows 95 using a client/server interface.

The API is fully integrated into the product, so you can, for instance, check the status of faxes from your application. You can also re-send faxes, print extra copies, and perform many other functions that are required to control message interchange.

Thus, instead of piping print to a print spooler, you pipe it to the fax spooler (*/usr/bin/faxprint*), closing and opening the pipe after each customer. You can also include a simple header that tells the fax system what to do with the output.

The fax spooler creates an intermediate file for each fax in the fax directory *(/tmp/tcs*). A fax daemon (*/usr/bin/faxsend.inittab*) scans this directory every two minutes or so, handing messages found to the TCS fax system, which then sends the fax at the requested time and date.

Installation

You can buy the TCS package through the ACS Web site. After installing the package, create the intermediate directory */tmp/tcs*. If you choose a different directory name, change the code examples accordingly.

Customization

Change the application to CLOSE and OPEN the printer after each customer. Also include code to print a header before the first line of each customer's printout. The header should include the following fields, which must be separated by at least one space:

- *Network*

  Fax Ordinary fax

  Tlx Ordinary telex

  Cfx Fax via Comtext service

  Ctx Telex via Comtext service

  Cml Mail via Comtext service.

- *Identification*

  A text identifier for the customer (eg customer number or name). No spaces are allowed in this field.

- *Fax number*

  The fax number, telex number, or e-mail address of the recipient.

- *User-id*

  The user-id (which may be omitted, as indicated by a dash '-') of the user responsible for the transmission. Sent faxes will appear on the specified user's transmission screen.

- *Logo number*

  The system can add your letterhead (graphics) to simulate printing

on custom stationery. Specify the logo number.

- *Date*

  If you want the transmission to take place on a specific date, you specify it using *DDMMYY* format. Use a dash (-) to stand for 'now'.

- *Time*

  If you want the transmission to take place at a specific time, specify it using *HHMM* format. A dash (-) stands for 'now'.

Of course, to use this solution you must have the customer's fax number, telex number, or e-mail address.

If the output is designed to be printed on a pre-printed form with frames and text, you probably have to include some or all of this text when you send it to the fax-spooler, as it will otherwise be printed on plain or letter-head paper on the receiver's fax, telex, or e-mail.

The output that you want sent to your customer should be printed using one file per customer (each with a unique filename) in the */tmp/tcs* directory, or you can pipe the output to the **/usr/bin/faxsend** command (described below), which does the same thing.

If you use Micro Focus COBOL you can print to 'FAX' by setting (exporting) the variable:

```
dd_FAX=>/usr/bin/faxsend
```

For other programming languages, similar mechanisms can be used.

Installing the faxsend command

Copy this command to one of your AIX command libraries (for instance */usr/bin*) and make sure it has execution privileges. See the example below for the content of the command file.

Installing the faxsend.inittab command

Copy this command to one of your AIX command libraries (as before, */usr/bin* is a suitable choice) making sure that it has execution privileges. An example of this command file appears at the end of the article.

Insert the following line in the */etc/inittab*:

```
faxsend:2:respawn:/usr/bin/faxsend.inittab
```

Getting the system up and running

You are now ready to send faxes from your application. Note that you can place any file in the */tmp/tcs* directory from any environment as long as it has the format described above and each line of the file is separated by CR/LF as it would be in a normal print file. The data format used is that of an IBM Proprinter data stream, and you can therefore use special print attributes, such as bold, 17-cpi, double height/width, and so on.

If you use AIX Connections or Network File System, you can also dump files from PCs and elsewhere on the network to this directory, thus allowing PCs and other systems to send faxes simply by writing them to this directory.

When your faxes are handed over to the TCS fax system, you can track their status with the user interface.

THE CODE

/USR/BIN/FAXSEND

```
#
# /usr/bin/faxsend
#
File=/tmp/tcs/FAX.$LOGNAME.`date +%y%m%d`.`date +%H%M`.$$
cat > $File
```

/USR/BIN/FAXSEND.INITTAB

```
#
# /usr/bin/faxsend.inittab
#
#
#  This script scans the /tmp/tcs directory and transfers all files to
#  the TCS communications system, provided the first line of each file
#  has the following format:
#
#  +---------------------------------------- {Fax|Tlx|CFX|CTL|CML}
#  |     +---------------------------------- <Identification>
```

```
#  |     |    +------------------------------ <Fax.no.>
#  |     |    |     +------------------------- {<userid>|-}
#  |     |    |     |      +------------------ {<logo no.>|-}
#  |     |    |     |      |      +----------- {<date>|-}
#  |     |    |     |      |      |        +-- {<time>|-}
#  |     |    |     |      |      |        |
# <Net> <ID> <fax> <user> <logo> <DDMMYY> <HHMM>
#
#
cd /tmp/tcs
if [ -n "`ls 2>/dev/null`" ] ; then
for x in `ls` ; do
#
#   Is this file still busy, then skip it
#
    if [ -z "`2>&1 fuser $x | awk '{print $2}'`" ] ; then
#
#      Get header
#
       head -1 $x | tr -s ' ' | tr '\015' ' ' > /tmp/faxprint.$$
#
#      Get individual fields from header and file name
#
       Net="`cat /tmp/faxprint.$$ | cut -d' ' -f1`"
       Fax="/tmp/`cat /tmp/faxprint.$$ | cut -d' ' -f2 | \
         tr '/' '_' | tr -d'+' `.$$"
       FaxNo="`cat /tmp/faxprint.$$ | cut -d' ' -f3|tr -d '\015'`"
       UserFileName="`echo $x | cut -d'.' -f2`"
       UserHeader="`cat /tmp/faxprint.$$ | cut -d' ' -f4`"
       if [ "$UserHeader" = "-" -o -z "$UserHeader" ] ; then
          User="$UserFileName"
       else
          User="$UserHeader"
       fi
       LogoNo="`cat /tmp/faxprint.$$ | cut -d' ' -f5`"
       if [ "$LogoNo" = "-" ] ; then
          LogoNo=1
       fi
       SendDate="`cat /tmp/faxprint.$$ | cut -d' ' -f6`"
       SendTime="`cat /tmp/faxprint.$$ | cut -d' ' -f7`"
#
       if [ "$SendDate" = "-" -o -z "$SendDate" ] ; then
          SendDate=`date +"%d%m%y"`
       fi
       if [ "$SendTime" = "-" -o -z "$SendTime" ] ; then
          SendTime=`date +"%H%M"`
       fi
       Time="`echo $SendTime|cut -c1-2`:`echo $SendTime|cut -c3-4`- \
          `echo $SendDate|cut -c1-2`/`echo $SendDate|cut -c3-4`/ \
          `echo $SendDate|cut -c5-6`"
```

```
#
        tail +2 $x > $Fax
#
#       Create command to execute under a different user ID
#
        cat > /tmp/faxsend.exe.$$ <<EOF
        TCSWP=PRO;export TCSWP
        tcs snd $Fax $Time $FaxNo Fax a ${LogoNo:-1}
EOF
#
#       Make command executable and execute it
#
        chmod 777 /tmp/faxsend.exe.$$
        su $User /tmp/faxsend.exe.$$
        sleep 5
#
#       Remove work files
#
        rm -f /tmp/faxsend.exe.$$ $Fax /tmp/faxprint.$$ $x
    fi
done
fi
#
# Wait two minutes before scanning the directory again
#
sleep 120
```


Login performance in AIX 4.3

The time it takes to logon to AIX systems can deteriorate dramatically if the system supports a large number of users – this is the case regardless of whether the users are actually logged on or not. For example, I recently came across a busy system at a university department where about 30% of system resources were being used for logons. The reason logins were consuming so many resources was that, for every logon, the system had to perform a sequential search of the following three files:

- */etc/passwd* – the original Unix password file.
- */etc/security/passwd* – IBM's encrypted password file.
- */etc/security/lastlog* – the file that holds the date and time of the last logon.

IBM has provided a simple solution to the problem of logon performance in AIX 4.3. The **mkpasswd** command, which is also available via a **smit** menu, creates the following indexes.

- */etc/passwd.nm.idx*
- */etc/passwd.id.idx*
- */etc/security/passwd.idx*
- */etc/security/lastlog.idx*.

The two indexes created for the */etc/passwd* file index it by user name and user-id. The original text files remain unaltered. One index is created for the */etc/security/passwd* file and one also for the */etc/security/lastlog* file.

USING INDEXES

The indexing system is optional, and can be enabled using the command **mkpasswd -c**. As the original text files can be modified manually (a practice that is not recommended), the system checks the date stamp on */etc/passwd* and */etc/security/passwd* to establish if either is newer than the index, in which case AIX reverts to the sequential method. To rebuild the indexes, use the command **mkpasswd –f**. You can also disable indexing at any time using the **mkpasswd** command with the **-d** flag.

Running password-related AIX commands, such as **mkuser**, automatically updates the indexes.

Understanding the grep command

As an AIX administrator you may find that you need to locate files that contain certain strings of characters. For example, you may be required to search all your users' login profiles to find ones that contain statements with a particular path that no longer exists. This facility is also useful to end-users, who may need, for instance, to locate all development files that contain a specific field name that needs to be changed.

The command that accomplishes all this – and much more – is **grep**. This article explains how to use this command to implement solutions to a series of problems that administrators and users frequently have to solve.

GREP COMMAND BASICS

In Unix terminology (and that of a number of other platforms) the **grep** command is a *filter* – a program that reads input, transforms it, then writes the results to the screen, a specified file, or another named device.

In its simplest form, **grep** lets you search a file or files for strings of characters, displaying the results on the standard output device (normally the screen). It also lets you redirect the output to a file. This is useful when you need to extract data from one set of files and use the extracted data as the input to another operation that carries out further processing.

Another common way that **grep** is used is by piping the output of one **grep** command into either another **grep** command or another filter (more on this later).

SYNTAX OF THE GREP COMMAND

Below is the basic syntax of the **grep** command.

```
grep options searchspec filespec
```

- *options*

An optional flag or flags used to narrow or otherwise to modify the search.

- *searchspec*
  The pattern (or a file containing a pattern list) used to specify the character strings for which to search.
- *filespec*
  The file or files to search.

The following points should be noted:

1 If *searchspec* contains blanks, enclose the entire search pattern in quotation marks (" ").

2 If *filespec* is not specified, **grep** uses the standard input device as the data to search for *searchspec*.

3 If *filespec* is not found or is not readable, **grep** issues the warning 'cannot open file'.

By default, if only one input source is specified in *filespec*, **grep** displays the full contents of each line that matches *searchspec*.

For example, suppose you need to find all instances of the string 'INCLUDE' in the file *myprog1.c*. Do this with the command:

```
grep INCLUDE myprog1.c
```

A sample output of this command is shown below:

```
#INCLUDE <stdio.h>
#INCLUDE <stdlib.h>
```

If you specify more than one name in *filespec*, **grep** displays the name of the file containing the match, followed by a colon (:) and the contents of the line that matches the search pattern. This is repeated for each match found.

For example, suppose you need to find every line that contains the string 'INCLUDE' in all files with names like *myprog*.c*. The command to do this is:

```
grep INCLUDE myprog*.c
```

Below is a sample output of this command:

```
myprog1.c:#INCLUDE <stdio.h>
myprog1.c:#INCLUDE <stdlib.h>
myprog2.c:#INCLUDE <stdio.h>
myprog2.c:#INCLUDE <stdlib.h>
myprog3.c:#INCLUDE <stdio.h>
myprog3.c:#INCLUDE <stdlib.h>
```

A BRIEF HISTORY OF GREP

The **grep** command got its name from the form of a type of argument used for conducting searches with the **ed** editor. The **ed** search argument *g/re/p* stands for '*global/regular expression/print*', and it tells the processor to seek all (*global*) occurrences of the desired information (*regular expression*) and to write (*print*) the results to the standard output device.

REGULAR EXPRESSIONS

grep's search facilities extend beyond ordinary strings of text. **grep** utilizes a subset of the **ed** editor's regular expressions to specify the desired search pattern.

A regular expression is a string of text that can be enhanced using special characters. These characters allow you to specify sophisticated search criteria.

Below are a few of the most common special characters used in regular expressions along with their meanings.

^	Match from the beginning of the line.
$	Match from the end of the line.
.	Match any single character.
[]	Match one character from this set.
[^]	Match one character not in this set.
*	Match zero or more instances of the preceding character.
\	'Escape' (disregard the special meaning of) the next character.

Now let's take a closer look at each definition.

MATCH FROM THE BEGINNING OF THE LINE

The ‘caret’ (^) tells **grep** to search for lines that begin with the text following the caret.

For example:

```
grep ^From: notebook.file
```

The above command displays all lines in *notebook.file* that begin with the string ‘From:’.

A sample output of this command is shown below.

```
From:  Ken Smith

From:  Tom Walker

From:  Nancy Smith

From:  Thomas Baker
```

Note that, if any lines in the file contain leading spaces (spaces before the desired text), this version of the **grep** command would not find them. The string must be at the beginning of the line otherwise it’s not found. Thus, to find the same text, but with three leading spaces, use the command:

```
grep ^'   From:' notebook.file
```

Later you’ll see that the caret is also used as a negation indicator. With practice, you’ll learn when the caret is used to indicate the beginning of line and when it means ‘not’.

MATCH FROM THE END OF THE LINE

The dollar sign ($) tells **grep** to look for lines that end with the text that precedes the dollar.

For example:

```
grep 'break;'$ myprog.c
```

The command above displays all lines in *myprog.c* that end with the string ‘*break;*’. A sample output of this command is shown below.

```
            break;

default:  break;
```

```
        break;
```

Unlike the caret, there is no need to worry about leading spaces before the search pattern when using the dollar sign. In these cases, all leading spaces are displayed along with the string ‘break;’ at the end of the line. (Trailing spaces are, however, not ignored.)

MATCH A SINGLE CHARACTER

The ‘dot’ or ‘period’ (.) represents any single character in a line of text. If the dot is used as the entire search pattern, all non-blank lines are selected. For this reason, the dot is usually combined with other text.

For example:

```
grep 866-.... myphone.book
```

The above command displays all telephone numbers in the file *myphone.book* that start with the exchange ‘866’. A sample output of this command is shown below.

```
Baker, Thomas    866-0942

Smith, Ken       866-1440

Smith, Nancy     866-2350

Walker, Tom      866-2377
```

In effect, the search pattern says “display all entries that include 866 followed by a hyphen followed by at least four characters.”

MATCH ONE FROM THIS SET

The square brackets (‘[’ and ‘]’) tell **grep** to match one character from the set in the brackets.

For example:

```
grep [Ss]pecific proc.file
```

Searches for lines in *proc.file* that include either of the strings ‘Specific’ or ‘specific’. A sample output of this command is shown below.

```
to locate specific text within your document.  Once the text is
```

```
Specific text can also be found using the search facility by

unless you specifically state the requirements in your procedure.
```

In effect, the search pattern says “display all entries that begin with either a lower or upper case ‘s’ followed by the string ‘pecific’.”

You can also specify ranges of characters using square brackets. Below are some examples of using square brackets to specify characters in specific ranges.

[a-z]9	Matches any lowercase letter (‘a’ to ‘z’) followed by the numeral ‘9’. For instance: ‘b9’, ‘m9’, and ‘r9xy’.
[a-zA-Z]..	Matches any lowercase or uppercase letter followed by at least two characters of any type. For instance: ‘rtp’, ‘d97’, ‘Q+r’, ‘M()’, and ‘Active’.
Data[0-9]	Matches the string ‘Data’ followed by a digit. For instance: ‘Data3’, ‘Data7’, and ‘myData0’.
[^A-Z]data	Matches any character other than an uppercase letter followed by the string ‘data’. (See below for an explanation of this use of the caret.) For instance: ‘8data’, ‘mdata’, and ‘(data22)’.
ETC[^0-9]	Matches the string ‘ETC’ followed by any character other than a digit. For instance: ‘ETCA’, ‘ETC%’, and ‘98ETCm’.

MATCHING A CHARACTER NOT IN A SET

When a caret is used in the context of a square bracket (‘[^]’), it tells **grep** to match any character other than those in the set in the brackets. This is the opposite of ‘match one from this set’. Note that the caret goes inside the square brackets.

For example:

```
grep ^[^aeiou] fruit.list
```

The above command displays all lines in *fruit.list* that don’t begin with a lower case vowel. A sample output of this command is shown below.

```
bananas
```

```
grapes

lemons

pears

plums
```

Note this example uses the caret in two different contexts to mean two different things. The first caret means 'lines that begin with', while the second means 'characters other than *a*, *e*, *i*, *o*, and *u*'. Note that you can use ranges of characters with the caret.

MATCHING ZERO OR MORE INSTANCES OF A CHARACTER

The asterisk (*) tells **grep** to search for all lines that contain zero or more instances of the character preceding the asterisk.

For example:

```
grep first.*last test.file
```

The above command matches all lines in *test.file* that contain the string 'first' followed by zero or more characters (.*) followed by the string 'last'.

Below are some lines that match the pattern in this command:

```
firstlast

first last

first and then last

That is the first place I looked last night.
```

The character preceding the asterisk can be designated by a set of characters in square brackets. For example:

```
grep [Xx]*data test.file
```

The above command matches all lines that contain zero or more occurrences of either an uppercase or lowercase 'x' followed by the string 'data'.

Some lines containing strings that match this pattern are shown below.

```
data
```

```
xdata

Xdata

XXXdata

indexdata
```

Note that a common mistake is use the asterisk in **grep** in the same way as it is used in AIX shells – as zero or more instances of any character (as opposed to zero or more instances of the preceding character). What's worse, if you do make this mistake you'll find that the results usually 'look' right, as you'll not be aware that potential matches have been left out. It's important to understand the distinction between how **grep** and Unix shells interpret special characters.

ESCAPE THE FOLLOWING CHARACTER

The backslash (\) tells **grep** to disregard the special meaning (if any) of the next character.

For example:

```
grep '\.h' prog.c
```

Displays all lines containing the string '.h' in *prog.c*. A sample output of this command is shown below.

```
#include <stdio.h>

#include <stdlib.h>

/*  TMP_DIR is defined in stdio.h  */
```

In the above the backslash tells the **grep** to disregard the fact that the period usually stands for any character and instead to display all lines that contain a dot followed by an 'h'.

If you forget the backslash, you get every line in the file that contains an 'h' that's not the first character in the line.

SUPPRESSING METACHARACTERS IN SEARCH PATTERNS

Let's take a closer look at how the backslash or 'escape character' is used. Many characters have a special meaning in the context of a regular expression, and others have special meanings in the context of

AIX shells. As seen above, there are even characters that have different meanings depending on whether they're used in a regular expression or in an AIX shell, an example being the asterisk (*). Whenever you use the **grep** command from inside an AIX shell, you have to consider both special meanings.

These special characters are known as *metacharacters*, the term being general and not restricted to discussion of either **grep** or regular expressions. There are times when you need to search for text that contains special characters ignoring their special meaning. For instance, how would you search for the string '[like]'? In other words, how would you prevent **grep** from searching for any line containing at least one from the character set 'l', 'i', 'k', and 'e'?

There are several ways in which you can tell the **grep** to ignore the special meanings of characters; I'll explain the use of the four methods below in this article.

- Quoted
- Escaped
- Quoted escaped
- Escaped escape.

Just to make life a little more interesting, not all the above methods work with all metacharacters.

Quoted

The first and most common method is simply to enclose the character in single quotes.

Escaped

In some cases, you may escape the character by preceding it with a backslash.

Quoted escaped

In other cases, you may need both to escape the character with a backslash and surround it in single quotes.

Escaped escape

The last method is to escape the escape character itself. This tells the shell to pass a single backslash followed by the metacharacter to the **grep** command.

Let's use this final method to solve our '[like]' problem above.

```
grep \\[like] test.file
```

This tells the shell to pass the following sequence to the **grep** command:

1 A single backslash (the 'escaped' backslash)

2 A left square bracket

3 The string 'like'

4 A right square bracket.

The **grep** command then sees the escaped square bracket as a request to search for the character itself, rather than as a character set indicator.

METACHARACTER SUPPRESSION METHODS

Figure 1 on page 48 lists twelve metacharacters that have special meaning in either an AIX shell or in a regular expression (or both). Each character is followed by an indication of whether its metacharacter nature can be negated by each of the four methods described above. Where the method is supported, an example of the search pattern is displayed. If the method is not supported, a note is provided explaining how an attempt to use the method would be interpreted.

While the table in Figure 1 may be used for reference, it's intended to illustrate how you may have to experiment with different methods in order to create search patterns when they contain metacharacters. It's worth emphasizing that, while the table shows how search patterns involving metacharacters are interpreted, the results may differ according to the context of the search pattern. If other characters in your search pattern lead to a result different from that in Figure 1, a different method may be appropriate.

Metacharacter		Quoted	Escaped	Quoted escaped	Escaped escape
^	Caret	Displays all lines [1]	Displays all lines [1]	'\^'	\\^
$	Dollar sign	Displays all lines [1]	Displays all lines [1]	'\$'	\\$
.	Period	Displays all non-blank lines [2]	Displays all non-blank lines [2]	'\.'	\\.
*	Asterisk	"*"	*	'*'	*
[	Left square bracket	Finds unmatched '[' and ']' or '(' and ')' [3]	Finds unmatched '[' and ']' or '(' and ')' [3]	'\['	\\[
\	Backslash	Specifies that last character is '\' [4]	Specifies that last character is '\' [4]	'\\'	Specifies that STDIN used for file specification [5]
(	Open bracket	'('	\(	Finds unmatched '[' and ']' or '(' and ')' [3]	Searches for an 'unexpected' '(' or ')' [6]
)	Close bracket	')'	\)	Finds unmatched '[' and ']' or '(' and ')' [3]	Searches for an 'unexpected' '(' or ')' [6]
>	Greater than	'>'	\>	'\>'	Specifies the last character is '\' and overlays the input file [4, 8]
<	Less than	'<'	\<	'\<'	Specifies the last character is '\' [4]
|	Pipe	"|"	|	'|'	Specifies the last character is '\' and cannot execute [4, 7]
;	Semicolon	';' ;	\;	'\;'	Specifies the last character is '\' and cannot execute [4, 7]

Figure 1: Methods of supressing metacharacters

The notes below provide further explanation of the results shown in Figure 1.

1 An unescaped caret or dollar sign tells **grep** to display all lines in the file. In effect, the **grep** command is "list all lines that have either a beginning or an end."

2 An unescaped period tells **grep** to display all lines that contain any character. Thus **grep** displays all non-blank lines.

3, 6 An unescaped square bracket or parenthesis tells the shell that a character set or command list is being passed. If the matching bracket is not present, the system responds with a message warning of either an unmatched bracket or an unexpected character.

4, 5 Depending on the context, an unescaped backslash is interpreted by the shell either as a line continuation character or as an escape sequence. One possible outcome of this is that you get a message warning that the "last character is \." Another possible outcome is that the system displays a blank prompt, indicating that **grep** is waiting for input from the standard input device rather than from a file. Use CONTROL+C to restore the cursor should this occur.

7 An unescaped pipe or semicolon tells the shell that it's reached the end of a clause. The shell then mistakenly tries to execute the token that follows the 'end of clause' character, which is usually the input file for the **grep** command, and is thus probably not executable.

8 This is the most destructive of all the errors discussed here. An unescaped greater than sign is misinterpreted as a redirection symbol. This results in the shell redirecting the output of a malformed **grep** command onto the file that it's supposed to be searching, effectively destroying the file.

FLAGS FOR THE **GREP** COMMAND

The **grep** command comes with a number of flags that enrich its functionality. The sections that follow explain their use.

Ignoring case in search patterns (-i)

The **-i** flag tells **grep** to ignore differences in case (UPPERCASE and lowercase) between the search pattern in the command and text in files being searched. This flag is very commonly used, to the extent that some users specify it every time they use **grep** to ensure they find all relevant data regardless of case.

Displaying lines that don't match the search pattern (-v)

Another flag that's frequently used is **-v**, which tells **grep** to display all lines that do not match the specified pattern. This can be very useful when the output of one **grep** command is piped into another. For instance, consider the following command:

```
grep administ test.file | grep -v administrative
```

The first part of the command selects all lines from *test.file* that contain the string 'administ', while the second one filters out lines that contain the string 'administrative'. The net result is to select all lines that contain the word 'administ' but not 'administrative'.

However, what if the words 'administrator' and 'administrative' occur in the same line? Unfortunately the above **grep** command filters out this line. If you need to find all lines that contain a particular string, regardless of whether they also contain another similar string in which you're not interested, use the **-f** flag (described later) instead of **-v**.

Another way to use the **-v** flag is to filter out comments in scripts. Many AIX programs and scripts use the asterisk (*) as a comment indicator. Some programs include dozens of comments per a line of code. To display only uncommented lines in a file called *source*, use the command:

```
grep -v ^\* source
```

The above command displays all lines that do not begin with an asterisk in *source*. Note that the asterisk has to be escaped with a backslash as it has a special meaning in the context of a shell.

This article concludes in next month's issue of AIX Update.

David Chakmakian (USA)

Contributing to *AIX Update*

AIX Update is primarily written by practising AIX specialists in user organizations – not journalists, or consultants, or marketing people. In our view, the information and advice provided by such people – people like you and your colleagues – are far more valuable to their fellow professionals than the alternatives available from other sources.

We don't expect you to write an original article from scratch – just send us listings or specifications of any relevant programs, utilities, scripts, user modifications, or other code that might be of use to other installations, with a short explanation of why it was developed and what it does. And most IS departments produce a great many internal technical reports and other documents, many of which can easily be adapted for publication. Xephon's editorial staff will transform even the most informal listing or document into a polished article fit for publication. So you don't have to spend any time on reformatting or rewriting your contribution – just send it as it is and we'll do the rest.

Contributors aren't just helping their fellow professionals – they also receive a significant material reward themselves. We pay good rates for the articles we publish: $250 (£170) per 1000 words if we get copyright, and $140 (£90) per 100 lines of code.

A copy of *Notes for contributors* can be downloaded from Xephon's Web site at *www.xephon.com*. Articles for submission to *AIX Update* can be sent to the editor, Harry Lewis, at *HarryLewis@compuserve.com*.

AIX news

IBM has announced the second release of its San Francisco Java business process components framework, with new modules for building order processing and warehouse/inventory management applications.

The new release provides national language support, more sample JavaBeans, performance enhancements to the Foundation layer, a framework for building GUIs to San Francisco-built applications, and support for additional deployment platforms and data formats.

The new components cost US$470 per seat and, in addition to AIX, also run on OS/400 and NT.

Also new is Net.Commerce Pro, which is aimed at large corporates, and includes catalogue tools and back-end integration tools for accessing software such as CICS, MQSeries, IMS, SAP R/3, and EDI. It runs on AIX, and also on OS/390, NT, and Solaris.

For further information contact your local IBM representative.

* * *

One way of running 32-bit Windows applications from AIX systems is to configure AIX as a client of an NT server. To enable this, Citrix has announced AIX workstation clients based on its Independent Computing Architecture (ICA) technology for 'thin-client/server' computing. This allows AIX users to run applications remotely on the NT system. Other Unix clients supported are Solaris, HP-UX, Digital Unix, and IRIX workstations.

For further information contact:
Citrix Systems, 6400 Northwest 6th Way, Fort Lauderdale, FL 33309, USA
Tel: +1 954 267 3000
Fax: +1 954 267 9319
Web: http://www.citrix.com

Citrix Systems UK Ltd, Eagle House, The Ring, Bracknell, Berks RG12 1HB, UK
Tel: +44 1344 382100
Fax: +44 1344 382136

* * *

Candle has announced a version of its Command Center for R/3, geared to increasing performance and availability of R/3 applications, with promises to reduce downtime and increase application response times. The product is available for AIX, HP-UX, Solaris, and Digital Unix, with support for Oracle and Informix.

Out now, prices start at US$20,000.

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

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