

CS 470: Unix/Linux Sysadmin
Spring 2025 Lab 5b
the ghost of commercial Unix past: AIX

Things from the prior labs that are required to begin this lab:

- the DNS setup from lab 1 is extended here

Things in this lab that are on the critical path for upcoming labs:

- having this lab's VM up and running is a requirement for grading lab 5b
- aside from the grading dependency, this lab can be worked out of order

We should start off by talking about IBM. Founded in 1911 (yes, really) as the Computing-Tabulating-Recording Company ("CTR"), IBM (its name since 1924) was the original "big tech" company, a pioneer of information technology since electricity was first standardized and the most primitive forms of IT were born ... all the way to the mainframe, and then the personal computer.

IBM has continually re-invented itself for over 100 years, riding the bloody edge of emerging technologies like punch-card time keeping and tabulating systems, electronic scale and measurements, electric typewriters, electronical and mechanical calculators, mainframes, minicomputers, supercomputers, and personal computers. IBM invented hard disks (in 1956!), bar codes, magnetic stripe cards, and DRAM. It made its first stored-program computer in 1952 with vacuum tubes for memory, and also in 1956, also made one of if not the first, demonstrations of artificial intelligence with a program that played checkers and "learned" from its experience.

Every single Windows PC and Intel Mac out there today is a descendant of the original IBM PC from way back in 1981. To call the company a cornerstone in computing would be understatement, to say the least. IBM is a unique company. IBM used to have a [corporate song book](#). Like seemingly every other marquee technology company of the last 50-plus years, IBM has its own variant of Unix, called AIX. A huge thanks to IBM's PowerSkills for hosting us for several years now, so that we can show off their boutique genetic AT&T-ish Unix operating system, and get all our hands on it.

I was surprised, reading the Wikipedia article for IBM's Jeopardy! champion AI, Watson, named after one of the early company principals. Not only does it **not** run on AIX, it doesn't even run on IBM's wholly-owned Linux (Red Hat). Strangely, Watson runs on SUSE Linux, another "enterprise" Linux based out of Germany.

https://en.wikipedia.org/wiki/IBM_Watson

https://en.wikipedia.org/wiki/SUSE_Linux

Watson does, however, run on IBM's POWER line of CPUs, and lots of things still do run on AIX. AIX still has a reputation for having fairly high-quality code and comparatively less security problems, though it does get seen by a far more exclusive audience. AIX still has a big following in the financial and health care industries, amongst others.

IBM's Unix is kind of its own beast. Though AIX is derived directly from pure cane AT&T Unix, it's always incorporated a lot of BSD code and tendencies too, a lot of IBM's own stuff, and even some stuff from Linux. Try as I might, I couldn't find the quote, but some used to be fond of saying something along the lines of: IBM got its hands on Unix and "fixed" (or maybe just IBM-ized) it, and AIX was the output of that process.

Originally, AIX was designed for IBM's line of RT workstations. RT stood for "RISC technology" when RISC CPUs

(IBM's ROMP line of CPUs in this case) were still the new kid on the block. AIX has since run on several instruction sets and several more architectures than that, including as a "pro" option on early IBM PCs (Intel x86), and on the ROMP, POWER/PowerPC, and IBM mainframe architectures.

POWER/PowerPC is a throughput monster family of CPUs that has at one time powered Apple Macintosh-branded computers, Cisco routers and switches, Sony PlayStation, Nintendo, and Xbox game consoles, and the aforementioned Watson. Since 1990, AIX has only run on IBM-branded POWER or PowerPC hardware.

Thus far, you've mostly used modernized distributions with forward-thinking, open-source software, and modern shells with command line editing, and history-based substitutions. This isn't me bashing AIX; version 7.3 is far more modern and accommodating than 7.2, but both are currently supported and we're not after creature comforts with this lab. There will be none of that here at the beginning, and whatever creature comforts you have, you'll be bringing with you ... aside from getting familiar with another flavor of commercial Unix, getting `bash`, `sudo`, and a third-party software package manager running to provide some modern-ish tools is the primary mission of this lab. You're in for a treat.

IMPORTANT: do not reboot or shut down your AIX instance. It's in IBM's cloud, and we don't have access to the power controls for each instance. If you `reboot` and it doesn't come back up, we have to mail IBM to turn it back on. If you shut it down, it will not come back up. Take a look after you log in: I've run `chmod 0400 /usr/sbin/shutdown` and `/usr/sbin/reboot` to try to save you from shooting yourself in the foot.

part one: get it

We don't really have to *get* it this time ... AIX only runs on Power/PowerPC, and this has already been done for us. IBM's educational cloud has given us each a VM (a logical partition, or "LPAR" in IBM's parlance) running AIX 7.2, and we just have to log in and explore.

These systems are meant as crash test dummies; don't try to mess up your VM, but don't be afraid of it, either. When I first logged into my AIX system to develop this lab document in 2019, my AIX LPAR told me ...

```
472525 unsuccessful login attempts since last login.
```

... **this** is why we typically don't expose SSH services to the internet. This was the result of automated software attacking this instance, trying to brute-force guess the root password. We typically hide SSH ports behind firewalls and VPN services requiring multi-factor authentication. This year, IBM has gotten its educational cloud with the times, and now requires us to connect to a VPN service to get around their firewall before we connect to our AIX instances.

Go to, or fetch the file, at the URL <http://slagheap.net/media/cs470/aix/YourRedID.txt> ... replace "YourRedID" with your numeric RedID number. Inside the file is the IP address and root password of your AIX instance for this exercise, and your login for IBM's remote access VPN service to its educational cloud.

You have two options for how you can connect to IBM's cloud. You can either use the commercial Cisco VPN client, or you can use `openconnect`, a compatible open-source VPN client, so we're going to install them both. If anything ever doesn't work with `openconnect`, fall back on the Cisco-branded client.

To install the graphical openconnect VPN client on Windows, just go to the [project's website](#) and download and install the software.

To install and use the command line openconnect VPN client on a Mac, first install `homebrew`, a way of adding third-party packages to your Mac.

If you need to install `homebrew`, run this on a command line on your Mac, not inside a VM:

```
$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

Then, to install `openconnect` ...

```
$ brew update
```

```
$ brew install openconnect
```

Verify the installation with `brew list`.

To install the Cisco VPN client software, go to <https://vpnisv.isv.ihost.com> in a web browser then select the group Anyconnect and log in with the VPN credentials from the file you just downloaded. **If you have an issue with this URL and are using a Chromium-based browser (Chrome/Edge/Brave/Opera/SRware Iron), use Safari instead if you're on a Mac, or grab Firefox if you're on Windows.**

If you already have the Cisco AnyConnect VPN or the Cisco Secure VPN client, it just should fire up and establish a VPN connection once you successfully log in. Most of you don't have it, though, I'm sure ... and the next page will welcome you to IBM, presumably IBM's internal name for its new educational cloud. After you click "continue" it'll give you a link to download the VPN client for your host operating system. Click the link, and when the download is done, run the installer.

!! NOTE: SDSU's campus VPN (Palo Alto Global Protect) and IBM's VPN (AnyConnect) will almost certainly not work well together. VPN clients typically don't work nicely together, as they all typically expect to be in charge of the DNS resolver client and a system's routing table. You should not expect for both VPN connections to be up at once ... but fortunately, you don't need both VPN connections to be up at once, either. When working on this lab, disconnect from the campus VPN.

After the first time connecting to download the Cisco VPN client, you'll use this process every following time to connect with the Cisco VPN client:

- Run the Cisco AnyConnect VPN client. Newer versions may call themselves "Cisco Secure Client." Don't worry, it's the same software. Those of you on Macs, look in the Applications folder. Those of you on Windows, can find it by hitting the start menu button and typing "Cisco" to see it. In the text entry box under "ready to connect," specify the hostname `vpnisv.isv.ihost.com` ...
- ... as before, select the group "Anyconnect" if you're offered a choice of groups, and the VPN username and password provided in the file you fetched with your RedID. Click "accept" when welcomed to IBM.
- Once a connection is successfully established, the Cisco VPN client may minimize to the system tray on Windows, or the menu bar on the Mac. You can find it there, if you seem unable to talk to your AIX

instance.

- While connected, the VPN might erroneously set your host's DNS server to 192.168.25.4, which is unreachable from your host. You should know how to change it to something sensible, if you find you can't resolve internet sites after connecting.

Connecting with the Windows-based `openconnect` GUI is very similar to the Cisco client.

To connect on your host computer's command line with `openconnect`:

```
$ sudo openconnect --user=psa-sdsu-vpn-user-X --authgroup=Anyconnect vpnisv.isv.ihost.com
```

`openconnect` will stay active in your host's terminal window, and will not return a new prompt. If you connect to IBM's cloud this way, start a new terminal tab or window to work in ... leave `openconnect` running in that terminal window or tab, and end it in that terminal window or tab when you're done with a `control-c`.

Regardless of how you connected to the VPN, now you're able to use SSH to log in to your AIX instance. It should ask you to reset your root password and log you out after you do. If it doesn't prompt you to change your password, change it now. You should know how to by now. Log back in with your new password afterwards. If you can't get in, please let me know ASAP.

Also, I've posted IBM's course materials for their exhaustive training on AIX at http://slagheap.net/media/cs470/aix/AN10_5.zip for your perusal.

part two: configure it

Since this AIX instance is out across the internet on IBM's educational cloud, a lot of the configuration we ordinarily do in each lab is infeasible on this one. It'll be kind of difficult, and of limited value, to plug it into the services we set up in the rest of the labs ... our DNS, LDAP, e-mail, and NFS are all hidden behind your laptop, acting as a network address translating (NATing) firewall.

Even if we could point our little `cs470.internal` domain from somewhere online, we'd need to pull a few firewall tricks, maybe stand up another VPN, and use dynamic DNS to help this AIX instance find our VMware private network to enable communication between them. We're going to do a little bit of this in lab 7, when we play with public cloud, but not today. This system is just going to be a one-off, so we can explore some outlier operating systems.

1. Log back into your AIX instance if you've logged out of it. Since you're logging in with a password somebody else gave you, as root no less, please reset your password.
2. The `uname` tool is a very useful one; we introduced it way back in lab 1. In the 1970s, at the dawn of Unix porting, they realized a single tool was needed on each system to be able to field answers about the operating system, kernel, CPU architecture, and other things. On certain operating systems, `uname` will show you build numbers for the kernel. On others, it will print machine hardware identifiers, like the hex number in the `uname` output just below, from my AIX LPAR.

```
# uname -a
AIX sdsu-aix72-36 2 7 0001F99A7A00
```

On virtually all Unix-derived OSs, the `-a` switch will show you everything (all) that `uname` is able to provide. In earlier releases of AIX, including earlier releases of AIX 7.2, it also used to do so ... now it no longer shows the hardware architecture. You have to use the `-p` switch to get `uname` to cough it up now.

```
# uname -p
powerpc
```

This is neither an Intel nor an ARM CPU here anymore, Toto ... it's not Kansas, either.

If you haven't already, try `uname` out on your other VMs.

3. A lot of you may noticed already, but we don't have command line editing here, or history.

```
# echo $SHELL
/usr/bin/ksh
```

How about a more modern, friendly shell?

```
# which bash
no bash in /usr/bin /etc /usr/sbin /usr/ucb /usr/bin/X11 /sbin
/usr/java8_64/jre/bin /usr/java8_64/bin /opt/freeware/bin
# which tcsh
no tcsh in /usr/bin /etc /usr/sbin /usr/ucb /usr/bin/X11 /sbin
/usr/java8_64/jre/bin /usr/java8_64/bin /opt/freeware/bin
```

Sooooo unhelpful. Let's see what's registered with the operating system.

```
# cat /etc/shells
/bin/csh
/bin/ksh
/bin/psh
/bin/tsh
/bin/bsh
/usr/bin/csh
/usr/bin/ksh
/usr/bin/psh
/usr/bin/tsh
/usr/bin/bsh
```

`bsh` is apparently the original Bourne shell, from the manual, not the Bourne-again shell (`bash`), `tsh` is the "trusted shell," a security-enhanced version of the Korn shell (`ksh`), and `psh` doesn't have a man page. Nope, no modern shell! Also no `sudo` ...

```
# which sudo
no sudo in /usr/bin /etc /usr/sbin /usr/ucb /usr/bin/X11 /sbin
/usr/java8_64/jre/bin /usr/java8_64/bin /opt/freeware/bin
```

... `sudo` is so accepted, that even the canonical Solaris operating system from Oracle now includes it. We'll come back to this later, but like I said, we're not in Kansas anymore or any parts familiar either.

We're in an older, classic Unix that lacks a lot of the familiar creature comforts of modern Unix and Linux.

4. Next, let's create a regular user account so we're not always running around as root, and let's all use the same username as our LDAP user here. We will not be using LDAP here, as there's no way to have this AIX instance phone home to our LDAP server, so we will not be creating a failsafe user account.

There's an `/etc/passwd` file here on AIX ...

```
# ls -l /etc/*passwd*
-rw-r--r--  1 root      security      666 May 05 2022  /etc/passwd
```

... but not an `/etc/shadow` ... I web searched and found [this article](#) ... so rather than manually editing new and unfamiliar files, I tried to find native commands for creating users under AIX but ran into an older Unix problem right away, one that we haven't had to deal with on the newer operating systems we've used thus far in these labs ...

```
# man -k user
man: file /usr/dt/man/whatis not found
      Create the whatis database using the catman -w command.
```

... the `whatis` database is the database of one-line synopses of all the installed `man` pages that we search with `man -k`, and most OSs include `whatis` databases out of the box nowadays, or just search them on the fly ... not AIX. Run `catman -w` to build the database. Running `man -k user` again produced *waaaay* too much output, so I piped it through `grep ^user` ...

```
# man -k user | grep ^user
user(4) - Contains extendeduser attributes.
user(4) - Contains thelist of roles for each user.
useradd(1)      - Creates a new user account.
userdel(1)      - Removes a user account.
usermod(1)      - Changes user attributes.
users(1)        - Displays acompact list of the users currently logged on to the system.
```

... reading the man page for `useradd`, it says all group IDs used must be valid, so I decided to first make my regular user account its own group (editing `/etc/group` with `vi`, of course) ...

```
peter::1001:
```

... and then created the user account ...

```
# useradd -d /home/peter -g 1001 -u 1001 peter
```

... and then set the new account's password.

```
# passwd peter
```

Inquiring minds will use `ls`, like I did, and notice that though I asked `useradd` to create a home directory with the `-d` option above, AIX's `useradd` didn't do it ...

```
# ls -l /home
total 0
drwxr-xr-x  2 guest  usr      256 May 04 2022  guest
```

```
drwx-----  2 root      system      256 May 05 2022  lost+found
drwxr-xr-x   2 203      system      256 May 05 2022  srvproxy
```

... so I made it myself manually ...

```
# mkdir /home/peter
# chown peter:peter /home/peter
# ls -ld /home/peter
drwxr-xr-x   2 peter    peter      256 Nov 04 13:04 /home/peter
```

... and now we have a full, regular user account, but unfortunately, however, we're going to need to stay root for most of this lab because we don't have `sudo`. Not yet, at least ... so **be careful!**

5. Network time ... first, I decided to look if it was running, and noted that AIX is an AT&T-derived Unix, so `ps` historically prefers the AT&T-ish switches (`-eaf`) like Solaris does, but testing showed that AIX's `ps` command also works nicely with the BSD-descended switches (`-aux`) ... and that `ntpd` wasn't running.

```
# ps -eaf | grep ntpd
```

Not quite sure how I found it initially in searching, else I'd share, but search I had to, and AIX uses the token `src` to represent services and subsystems. Nice one, IBM!

As we saw in lab 1, virtually every other Unix OS uses `src` as an abbreviation for "source," as in "source code." Linux and BSD both use `/usr/src` as the home of kernel and operating system source. From the output of `man -k src` this appears to stand for "System Resource Controller" ... sweet, another new acronym. (In case I've failed to spell out my opinion here, acronyms are poor branding, and poor branding is a primary problem in selling our field.)

Near the bottom of the output of `man -k src` I spotted this ...

```
startsrc(1)      - Starts a subsystem,a group of subsystems, or a subserver.
```

... and a short `man` page reading later (DO IT), this is the command I was looking for to start NTP ...

```
# startsrc -s xntpd
0513-059 The xntpd Subsystem has been started. Subsystem PID is 9175444.
```

... what you're seeing here with `0513-059` is a very classic IBM-ish insistence on message numbers here, which used to turn up much more often than it does now. This is the command we use to list services ...

```
# lssrc -a | grep ntp
xntpd          tcpip          9175444        active
```

... finally, edit the file `/etc/rc.tcpip` and uncomment the line that starts `ntpd` at boot time.

6. We're out across the internet on another private network here, and in order to send e-mail, we really need to have a publicly available DNS name. If our system's hostname doesn't look up on the internet, most e-mail servers online will discard e-mail coming from it.

For those of you not familiar with it, I recommend that you stand up a free account at FreeDNS (<http://freedns.afraid.org>) and grab a hostname for your AIX instance. You can use FreeDNS later for your home network installation, to get mail out of your VM network, or whatever you like.

To find out what our public IP address is, we're going to use the service ipify.org. If you hit their main page and click on "API usage," you'll see the API URL you can grab to get back a text response in a terminal. Fortunately, AIX typically installs a bunch of freeware under `/opt/freeware`, including `curl`, a versatile client for various network protocols, including HTTP/HTTPS, of course. Conveniently, there's a symlink to `/opt/freeware/bin/curl` in `/usr/bin`. Use `curl` and `ipify.org` to get the public-facing IP address of your AIX instance's network locale and register it with FreeDNS.

I grabbed myself the name `aches.mo00.com` as you can see below.

- Now let's set up internal DNS. We're going to set up a `hosts` file entry and two records in internal DNS, one to help us access our AIX instance over the VPN, and the other to help us find the public lookup we just stood up at FreeDNS.

First, set up an entry for `aix.cs470.internal` to your `hosts` file on your host operating system, using the **private** IP address of your AIX instance on IBM's cloud ... the IP address you use to SSH into it, NOT the public IP. This way you can SSH into your AIX instance from your host computer by name.

Next, so that we can SSH into our AIX instances by name from our VM networks, add a DNS address (A) record for `aix.cs470.internal` in the zone file for `cs470.internal` with the private address supplied by IBM for your AIX instance.

Also set up a DNS alias (a CNAME record) for `aix-public.cs470.internal` in the zone file for `cs470.internal` on your OpenBSD VM, pointing to the public DNS name you used at FreeDNS. What a symbolic link is to the file namespace, a CNAME record is to DNS. When a DNS client (resolver) does a lookup on the name `aix-public.cs470.internal` inside your VM network, the CNAME tells the client that it has to look up the DNS name you set up in FreeDNS, and it goes there for the answer.

Even if we wanted to, we can't use a CNAME in a `hosts` file. When you get the setup correct, it should

look like this ...

```
$ host aix.cs470.internal
aix.cs470.internal has address 172.20.14.233
$ host aix-public.cs470.internal
aix-public.cs470.internal is an alias for aches.mooo.com.
aches.mooo.com has address 131.239.211.201
```

I know we haven't done a CNAME before ... go find out how! It's not rocket science, it's just a web search.

8. Now we're ready for e-mail. If you look in `/etc/mail` you'll see that AIX comes with `sendmail`.

First, edit `/etc/mail/sendmail.cf` and uncomment the line with `Dj` and add in your hostname, so that it looks like this ...

```
Djaches.mooo.com
```

... with your hostname from FreeDNS or another DNS service, of course, instead of mine. AIX doesn't have a `pkill` or a `killall` command for sending signals to processes based upon the name of the command they were started with ... we use this command to ask `sendmail` to re-check its configuration:

```
# refresh -s sendmail
0513-095 The request for subsystem refresh was completed successfully.
```

Add a line for `root` in `/etc/mail/aliases` – and note that you want to use a real, publicly-routable e-mail address here. Use your SDSUid address, **not** your `@cs470.internal` e-mail – and run `newaliases`.

```
# newaliases
WARNING: Group writable directory /var/spool
WARNING: Group writable directory /var/spool/mqueue
/etc/mail/aliases: 26 aliases, longest 17 bytes, 300 bytes total
```

Finally, let's change the hostname of the system to match the name you configured in FreeDNS ...

```
# smitty hostname
```

... and here, you'll get your first introduction to `smit`, the single most distinctive tool in IBM's Unix and the biggest result of IBM "fixing" what they licensed from AT&T. Anybody who has used Hewlett Packard's HP/UX users will notice a similarity to the `sam` tool from that operating system. At the menu, choose to set the hostname, and set it to the name you registered above in FreeDNS, and hit `return` to perform the action. It should say `inet0 changed`. Then hit `F10` to exit `smit`. Going out to the command line, and running the command `hostname` will confirm that it worked.

```
# hostname
aches.mooo.com
```

Time to test mail; of course, replace my SDSUid e-mail address with yours ...

```
# echo "test" | mail -s test pbartoli@sdsu.edu
```

Check your spam bin before you conclude that it didn't arrive.

To make sure you get logs from the mail subsystem, you can add this line in `/etc/syslog.conf` ...

```
mail.debug /var/log/mail/maillog rotate size 200k files 4
```

And:

```
# mkdir /var/log/mail
# touch /var/log/mail/maillog
# refresh -s syslogd
```

part three: disk management

In this section of the labs, we're going to be playing with AIX's volume management capabilities, the very same sort of technology I asked you to steer clear of (LVM) in both the Linux labs. Volume management abstracts filesystems away from disks, so that filesystems can span multiple physical disks, among other storage strategies.

Like I said back then, on standalone VMs on a desktop hypervisor, volume management is an unnecessary layer of abstraction. In a VM in a cloud, it may well not be unnecessary ... it might be useful for making a filesystem span multiple backing stores for failover purposes. In this part of the lab, we'll explore AIX's volume management capabilities by shrinking and growing filesystems, and mirroring our root filesystem and volume group across multiple disk devices.

9. Disk management on AIX is a little different. `df` on AIX doesn't have a `-h` switch; I find it gets the most human-readable output with the `-m` switch ...

```
# df -m
Filesystem      MB blocks      Free %Used      Iused %Iused Mounted on
/dev/hd4         9312.00      9263.09    1%       2962     1% /
/dev/hd2         2336.00       553.53   77%      37215    22% /usr
/dev/hd9var       1216.00      1162.61    5%        989     1% /var
/dev/hd3          1024.00      1020.77    1%         35     1% /tmp
/dev/hd1         21504.00     21500.36    1%          8     1% /home
/dev/hd11admin    128.00       127.63    1%          5     1% /admin
/proc            -             -          -          -      - /proc
/dev/hd10opt       5120.00      4783.96    7%      11312     2% /opt
/dev/livedump       256.00       255.64    1%          4     1% /var/adm/ras/livedump
/dev/repo00        7904.00        85.99   99%       2151    10% /usr/sys/inst.images
/ahafs            -             -          -          35     1% /aha
```

... as you can see, `m` is for megabytes.

These AIX systems are VMs (or LPARs), and often, not all available disk space is allocated during installation. If you look at the file `/var/adm/ras/bosinst.data` (I'm not sure what `ras` is, but `bos` is short for "base operating system," because acronyms), you'll find a line `SIZE_MB` that'll tell you the size of the disk in your "LPAR."

```
# grep -i size /var/adm/ras/bosinst.data
SIZE_MB = 20480
```

So this LPAR had about 20 GB (or is it GiB?) of space at some point in time, but as we saw in the output of `df`, it's using much more space than 20 GB. Our LPARs were probably cloned outside the regular installation process.

Filesystems on AIX are provisioned out of space in “volume groups.” These volume groups are like the LVM technology we avoided as an unnecessary abstraction on Linux. AIX, however, requires a few different partitions by default, so it is not an unnecessary abstraction here, and we're going to briefly explore what can be done with volume groups.

You can list a system's volume groups with the `lsvg` utility.

```
# lsvg
rootvg
```

The answer is the default for AIX systems; the root volume group typically contains the bootable operating system. On most systems, there's only one volume group.

To get more information about the root volume group, supply it as an argument to `lsvg`.

```
# lsvg rootvg
VOLUME GROUP:      rootvg
VG STATE:          active
VG PERMISSION:     read/write
MAX LVs:           256
LVs:               13
OPEN LVs:          12
TOTAL PVs:         1
STALE PVs:         0
ACTIVE PVs:        1
MAX PPs per VG:    32512
MAX PPs per PV:    2032
LTG size (Dynamic): 256 kilobyte(s)
HOT SPARE:         no
PV RESTRICTION:    none
DISK BLOCK SIZE:   512
FS SYNC OPTION:    no
ENCRYPTION:        no

VG IDENTIFIER:     00fa00d600004c000000001809352fa44
PP SIZE:           32 megabyte(s)
TOTAL PPs:         1599 (51168 megabytes)
FREE PPs:          23 (736 megabytes)
USED PPs:          1576 (50432 megabytes)
QUORUM:            2 (Enabled)
VG DESCRIPTORS:    2
STALE PPs:         0
AUTO ON:           yes

MAX PVs:           16
AUTO SYNC:         no
BB POLICY:          relocatable
INFINITE RETRY:    no
CRITICAL VG:       no
CRITICAL PVs:      no
```

So there are about 50 GB of space available (TOTAL PPs) to my LPAR, and most of that is allocated (USED PPs).

We have plenty of space on our LPARs, so let's shrink its `/home` partition to 5 GB.

```
# chfs -a size=5G /home
[=====] 100%
Filesystem size changed to 10485760
```

Check out the output of `df` again to make sure it worked. With volume groups, space can be dynamically reallocated between partitions as required.

10. Each of our AIX instances was configured with multiple virtual hard disks attached ...

```
# ls -l /dev/hdisk*
brw----- 1 root    system    13,  0 Sep 12 2023 /dev/hdisk0
brw----- 1 root    system    13,  2 May 05 2022 /dev/hdisk1
```

... though only one is currently in use by the root volume group.

```
# lspv
hdisk0          00fa00d6b56a99bb          rootvg          active
hdisk1          none                      None
```

Let's fix this by adding the other disk to the root volume group ...

```
# extendvg -f rootvg hdisk1
0516-1254 extendvg: Changing the PVID in the ODM.
```

... and then configure the root volume group to mirror to both disks.

```
# mirrorvg -S rootvg hdisk1
0516-1804 chvg: The quorum change takes effect immediately.
0516-1126 mirrorvg: rootvg successfully mirrored, user should perform
bosboot of system to initialize boot records. Then, user must modify
bootlist to include: hdisk1 hdisk0.
```

Since we're not rebooting our AIX instances here in this lab, we're not going to modify the boot records, but in real life, to get true failover of our root volume group including starting the computer this would have to be done.

As you can see, the contents of the root volume group are spread out across the two disks that constitute it ...

```
# lspv -l hdisk0
hdisk0:
LV NAME          LPs      PPs      DISTRIBUTION      MOUNT POINT
hd8              1        1        01..00..00..00..00 N/A
hd6              16       16       16..00..00..00..00 N/A
hd5              2        2        02..00..00..00..00 N/A
repo00          247      247      00..247..00..00..00 /usr/sys/inst.images
lg_dumplv       32       32       32..00..00..00..00 N/A
livedump        8        8        08..00..00..00..00 /var/adm/ras/livedump
hd11admin       4        4        04..00..00..00..00 /admin
hd3             32       32       04..28..00..00..00 /tmp
hd9var          38       38       01..06..31..00..00 /var
hd2            73       73       49..24..00..00..00 /usr
hd4            291      291      03..00..288..00..00 /
hd10opt        160      160      01..15..00..144..00 /opt
hd1            160      160      01..00..00..159..00 /home

# lspv -l hdisk1
hdisk1:
LV NAME          LPs      PPs      DISTRIBUTION      MOUNT POINT
hd8              1        1        00..00..01..00..00 N/A
hd6              16       16       00..16..00..00..00 N/A
hd5              2        2        02..00..00..00..00 N/A
repo00          247      247      247..00..00..00..00 /usr/sys/inst.images
livedump        8        8        08..00..00..00..00 /var/adm/ras/livedump
hd11admin       4        4        04..00..00..00..00 /admin
hd3            32       32       00..32..00..00..00 /tmp
```

hd9var	38	38	00..38..00..00..00	/var
hd2	73	73	00..73..00..00..00	/usr
hd4	291	291	00..00..291..00..00	/
hd10opt	160	160	00..00..27..133..00	/opt
hd1	160	160	00..160..00..00..00	/home

... although it is not instantaneous! A few gigs of data still have to be copied to the clone drive.

Our root volume group now shows it has more space.

```
# lsvg rootvg
VOLUME GROUP:      rootvg
VG STATE:          active
VG PERMISSION:     read/write
MAX LVs:           256
LVs:               13
OPEN LVs:          12
TOTAL PVs:         2
STALE PVs:         1
ACTIVE PVs:        2
MAX PPs per VG:    32512
MAX PPs per PV:    2032
LTG size (Dynamic): 256 kilobyte(s)
HOT SPARE:         no
PV RESTRICTION:    none
DISK BLOCK SIZE:   512
FS SYNC OPTION:    no
ENCRYPTION:        no

VG IDENTIFIER:      00fa00d600004c000000001809352fa44
PP SIZE:            32 megabyte(s)
TOTAL PPs:          3198 (102336 megabytes)
FREE PPs:           1102 (35264 megabytes)
USED PPs:           2096 (67072 megabytes)
QUORUM:             1 (Disabled)
VG DESCRIPTORS:     3
STALE PPs:          309
AUTO ON:            yes

MAX PVs:            16
AUTO SYNC:          no
BB POLICY:          relocatable
INFINITE RETRY:     no
CRITICAL VG:        no
CRITICAL PVs:       no
```

Good job; aside from the small piece we didn't do for booting, you've just set up fault-tolerance for your root volume group.

part four: third-party packages

Just like on other operating systems, we want a package manager for installing third party software. Fortunately, IBM knows this, and they've put up a site to actively support the installation of free and open-source software on AIX. It's called the "AIX toolbox" ... a lot of OSs have their own third-party value-add systems for adding free software. IBM is a big fan of Linux – remember, they own Red Hat – and AIX is no exception. The "AIX Toolbox" is really just Red Hat Linux's package manager (`yum` or `dnf`), ported to AIX, with RPMs of AIX binaries, so you can install IBM-curated ports of common freeware.

<https://www.ibm.com/support/pages/aix-toolbox-linux-applications-overview>

Our immediate goal here will be to make AIX more comfortable with a modern shell (`bash`). After that, we want to bring our AIX instance up to parity with our other systems, and get `sudo`, `tcsh`, `gnupg`, `wget`, and `gcc` running.

Another option here would be NetBSD's `pkgsrc` system. NetBSD, again, is the ancestor to OpenBSD, and `pkgsrc` is NetBSD's name for its third party software build system, the ancestor of the "ports tree" on FreeBSD and OpenBSD. See https://wiki.netbsd.org/pkgsrc/how_to_use_pkgsrc_on_aix/ ... for some reason, the NetBSD project briefly ported their `pkgsrc` tree to AIX. No idea why here. We're not going to use it.

11. We used to install `yum` for AIX from the internet using the directions here ...

<https://public.dhe.ibm.com/aix/freeSoftware/aixtoolbox/README.txt>

... and I recommend you skim through it to see what we used to go through, but `yum` is no longer updated, so `dnf` it is.

Make a directory `/tmp/dnf` and `cd` to it to download and install it.

Remember `curl` from a few steps ago? Use it again, with `-O` to write output to a file, to download the tarball of the latest version of the `dnf` bundle for AIX 7.2 from the `ezinstall/ppc` folder in the `README.txt` link above, and extract it with `tar`.

If you get an SSL cert hostname error when running `curl` with the `dnf` bundle, try replacing the “ftp.software” in the URL with “www3.software” ... they’re the same repository, but the certificate provided by “ftp.software.ibm.com,” which you can see by navigating to it in a web browser and clicking that lock icon in the URL bar, has its common name set to “www3.software.ibm.com” for some reason. The mismatch between the fully-qualified domain name and the cert’s common name can – and should – cause HTTP clients like `curl` or Firefox to freak out during the TLS handshake.



One of the files that you unpacked is called `install_dnf.sh` ... run it without any switches to see its options. We don’t care about having a `yum` command, so this seems correct ...

```
# ./install_dnf.sh -d
```

Note that AIX is using `rpm`, the base Red Hat Package Manager, and that it is part of the base AIX installation.

```
# which rpm
/usr/bin/rpm
```

If you don’t get this error, you can skip ahead to the next step ...

Please install openssl 1.1.x and higher version.
 You can download and install latest openssl from AIX web download site
<https://www-01.ibm.com/marketing/iwm/platform/mrs/assets?source=aixbp>

... if you did, make a directory `/tmp/openssl` and `cd` to it.

I'm hosting the package here; use `urlgrabber` again ...

<http://slagheap.net/media/cs470/aix/openssl-1.1.1.2000.tar.Z>

... and the README for the package here:

<http://slagheap.net/media/cs470/aix/Readme-1.1.1.2000.txt>

As we covered in lecture just recently, archives with the `.Z` extension use `compress` and `uncompress`, the very oldest of Unix compression tools. Once you have that figured out, you should have a subdirectory with `openssl.base`, `openssl.license`, and `openssl.man.en_US` in it. `cd` there, and then to install it, we're back to running SMIT.

```
# smitty install_all .
```

In the field to enter the directory, enter `.` for the current directory.

In the field for software to install, enter `openssl.base`.

Change the value for accepting licenses to yes, then proceed.

Finally, this command updates the backing package databases.

```
# updtvpkg
```

The command from the prior step to install `dnf` should now work.

```
dnf installed successfully.  
Please run 'dnf update' to update packages to the latest level.
```

```
Please note, RPM packages are downloaded in dnf cache /var/cache/dnf.  
RPM packages install files go under the path /opt.  
Hence it is recommended to always keep at least 512MB of free space in /var & /opt  
to avoid any download and installation/update failures.
```

Make sure you have the free space under `/var` and `/opt` that it recommends ... we're about to use it.

12. `dnf` is now installed at `/opt/freeware/bin/dnf` ... take a look at the contents of `/opt/freeware` and `/opt/freeware/bin` at a minimum ... `/opt/freeware` is the file tree used for things installed by this build of `dnf` on AIX.

Note how `/opt/freeware/bin` contains both 32- and 64-bit versions of selected utilities.

```
# ls -l /opt/freeware/bin/clear*
lrwxrwxrwx    1 root    system           8 Sep 13 2023 /opt/freeware/bin/clear -> clear_64
-rwxr-xr-x    1 root    system    114632 Jun 08 2023 /opt/freeware/bin/clear_32
-rwxr-xr-x    1 root    system    120609 Jun 08 2023 /opt/freeware/bin/clear_64
# file /opt/freeware/bin/clear*
/opt/freeware/bin/clear: 64-bit XCOFF executable or object module
/opt/freeware/bin/clear_32: executable (RISC System/6000) or object module
/opt/freeware/bin/clear_64: 64-bit XCOFF executable or object module
```

The [RS/6000](#) was IBM's first line of PowerPC-based systems using 32-bit PowerPC-based CPUs. IBM hasn't sold a 32-bit CPU since before year 2000, so it's unclear to me why 32-bit binaries are being installed here.

Also examine the file `/opt/freeware/etc/dnf/dnf.conf` ... this is the main `dnf` configuration file, and IBM has placed all its repositories directly into this file.

Add `/opt/freeware/bin` to your `PATH` if it isn't already there.

Since we don't have to install it, update the `dnf` databases from the repositories, and invite it to update its core packages.

```
# dnf -y update
```

You should see the familiar dance of `dnf` reaching out to repositories (at IBM this time), updating its list of packages, resolving dependencies for packages it intends to install, and finally, updating packages already installed along with the package manager. For me, this included the installation of `bash`.

If you see warnings about failing to remove older copies of `bash`, you can ignore them. If you see errors about not being able to install the absolute best update candidate for a package, you may pass a certain `dnf` flag in order to relax that requirement.

13. I'll save you some pain; even though the installation of `bash` has added it to `/etc/shells`, `chsh` (change shell) **still** doesn't honor it as a valid shell. BE VERY CAREFUL as you manually edit `/etc/passwd` and change your non-root user shell manually to `/opt/freeware/bin/bash`.

14. Now that we have a real shell installed, let's install `sudo`, so that we can finally stop running around as root, and start using our mortal user account.

```
# dnf -y install sudo
```

Edit `/etc/sudoers` and allow the wheel group to run things as root, if they provide a password.

Then, edit `/etc/group` and create a group called `wheel` with any GID of your choice, and add your non-root user to that group.

15. Log out of the root account, log back in as your regular user.

Add `/opt/freeware/bin` to your path for your regular user now too.

Create a personal `.ssh` directory with the appropriate owner-only permissions in your non-root user account's home directory on AIX. Copy your SSH public key into it as the file `authorized_keys`, as always, and test that SSH key-based authentication works.

16. Finally, let's install some more packages, and call it a lab.

```
$ sudo dnf install gcc tcsh wget
```

17. Later, when we grade this lab, we're going to grade it via your lab 5 VM. Go on your lab 5 Solaris VM and install `openconnect`.

```
$ sudo pkg install openconnect
```

Note that we gave you syntax for using `openconnect` at the top of this lab. If you're on campus when you're trying to grade your VMs at the end of the class, you won't be able to use both the campus VPN to fix DNS and the IBM VPN for AIX at the same time, on the same system.

Whether or not you're on campus or using the campus VPN, disconnect the IBM VPN on your host system and connect it from your lab 5 VM with `openconnect`. Make sure you can SSH into your AIX instance directly from your Solaris VM.

part five: updates

!! IMPORTANT NOTE: THE REST HERE IS JUST FOR YOUR READING PLEASURE. IBM removed the upgrade media from their educational cloud, sadly, and this will not work, but I wanted you to see what it looks like. Screenshots and version references in this section are from 2019, when they last had upgrade media available online, and we were using AIX 7.2.

Updates on AIX are generally done with one of two commands: `instfix`, which is typically for small patches, and `smit`, which is used for everything in AIX.

```
# oslevel -s
7200-00-02-1614
```

The command above shows we're running AIX 7.2, technology level 0 (no major update releases since the original release of AIX 7.2), service pack 2, from 2016 week 14. We don't, at least that I'm aware of, have any hotfix repositories made available to us to use with `instfix`, but we can use `smit` to update to the most recent "technology level" of AIX 7.2, AIX 7.2 TL 3, or 7200-03.

```
# smit suma
```

Select "Download Updates Now (Easy)," then select "Download Maintenance Level or Technology Level."

```

ttys003 — C#3
-tcsh — ttys0... -tcsh — ttys0... ssh kali-ma... ssh -x root@... ssh falcor — t... -tcsh — ttys0... ssh + scp -p... +
Download Maintenance Level or Technology Level
Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[Entry Fields]
* Maintenance Level or Technology Level to download 7200-03 +

F1=Help      F2=Refresh   F3=Cancel    F4=List
F5=Reset     F6=Command   F7=Edit      F8=Image
F9=Shell     F10=Exit    Enter=Do

```

On the next screen, enter 7200-03 and hit `return`. You'll see your session appear to drop out of `smit` — `smit` is often not very terminal-aware or SSH-friendly, I'm not sure which yet — and start showing output of downloading packages.

```

ttys003 — C#3
-tcsh — ttys0... -tcsh — ttys0... ssh kali-ma... ssh -x root@... ssh falcor — t... -tcsh — ttys0... ssh + scp -p... +
*
* Please see the README file in /usr/lpp/bos for information pertinent to
* this release of the AIX Operating System.
*
*
*****
# smit suma
# smit suma
Partition id was unassigned; will attempt to assign it.
Partition id assigned value 97
Storing auth proxy creds for SAS
successfully stored auth proxy creds for SAS
Storing auth proxy creds for PROFILE_URI_LENGTH
successfully stored auth proxy creds for PROFILE_URI_LENGTH
Storing auth proxy creds for PROFILE_URI_0
successfully stored auth proxy creds for PROFILE_URI_0
Filesystem size changed to 16515072
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/7200-03.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/U880005.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/U879963.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/U879943.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/U877276.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/U877198.bff

```

When it's finished (it'll take a while), you'll be back in `smit` again. Hit `F10` to exit out to a shell.

```

# cd /usr/sys/inst.images/installp/ppc
# inutoc
# ls -l .toc
-rw-r--r-- 1 root system 1120999 Aug 21 05:02 .toc

```

Then back into `smit` to run the update.

```
# smit update_all
```

In the first screen that pops up, provide `/usr/sys/inst.images/installp/ppc` as the directory, then you should see this screen after hitting `return`:

```

Update Installed Software to Latest Level (Update All)

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]
* INPUT device / directory for software      [Entry Fields]
* SOFTWARE to update                        /usr/sys/inst.images/>
PREVIEW only? (update operation will NOT occur)  no
COMMIT software updates?                      yes
SAVE replaced files?                          no
AUTOMATICALLY install requisite software?      yes
EXTEND file systems if space needed?           yes
VERIFY install and check file sizes?          no
DETAILED output?                             no
Process multiple volumes?                     yes
ACCEPT new license agreements?                no
Preview new LICENSE agreements?               no

WPAR Management
[MORE...5]

F1=Help      F2=Refresh    F3=Cancel    F4=List
F5=Reset     F6=Command    F7=Edit     F8=Image
F9=Shell     F10=Exit     Enter=Do

```

Use the arrow keys to go down to “accept new license agreements?” and hit space to change the value here. Hit tab, per the on-screen directions, to change the value to “yes.”

Then hit `return` twice to start the update.

After that is done and you hit `F10` to exit `smit`, the upgrade is finished, and so are we.

```
# oslevel -s
7200-03-01-1838
```

<https://www.ibm.com/developerworks/aix/library/au-aixservicepacks/index.html>