linux

pstree -l 1234

pgrep -aef vismon

pwdx 12345

gdb tran.ems core

vi :syntax off turns off the colors

getent passwd same as ypcat passwd

getent | cut -d : -f1 to see the first field

process groups are used for job control by shells, sessions are to keep track of all the processes in a interactive login session, there are cmds to set the session or process group ids, when you setsid, your tty becomes a ? forked children inherit the same pgid and sid as the parent

The first process, called init, is given a pid of 1

When a process is created, it inherits the process group ID and the session ID of its parent process. setpgrp() and setsid() create new proc group/session ids respectively.

ps –e –o “user pid”

you can specify the exact fields you want to display

ps –e –o “time args”

shows the total cpu time and the command

the args option is similar to comm option (check this out)

ps –e –o “stime”

start time

ps –e –o “osz”

virtual memory

ps –e –o “rss”

relative size

ps -aeo "user pid ppid pcpu pmem vsz stime time args"

CTL-Z and fg are used to suspend and return to a interactive job

nice -sets priority, - is high, + is low, i.e. –20 is the highest, +19 is the lowest priority

at 11am Jun 28 -submit job for this time, gives you a prompt to enter commands, finish with CTL-D

top – shows top jobs, has 1 letter cmds, type h to see

prstat –a and glance – more job cmds

users

jobs

last

uptime

free

truss traces system calls

aeries of proc cmds, psig, pstack, ptree also show job info

finger -check interactive jobs across network

who -shows log ons, aka w

anvil 19474 1 0 10:21:34 ? 10:52 isql -Urepo\_nyroot -Paardvark -w9999999999

when the parent (ppid) is 1, its an orphan, the parent is dead

ps -f will give this

UID PID PPID C STIME TTY TIME CMD

buonorat 781 21659 0 16:29:03 pts/17 0:00 ksh

in this example 781 was created as a new job via su

when i exited ps -f generated for job 21659

UID PID PPID C STIME TTY TIME CMD

anvil 21659 21640 0 Jan 03 pts/17 0:00 ksh

F S UID PID PPID C PRI NI ADDR SZ WCHAN STIME TTY TIME

8 Z anvil 6606 4447 0 0 0:00

the Z indicates its in a zombie state

buonorat 4453 1 0 17:13:41 pts/17 0:00 perl -w p3.pl

TTY is the controlling terminal, when you setsid, it becomes ?

as it has "no controlling tty"

WHEN YOU setsid, the pts/17 becomes a ?

buonorat 4512 1 0 17:15:01 ? 0:00 perl -w p3.pl

ps -j will give the session ID as well

ps

-a frequent processes

-e all

-d all exception session leaders

-f full listing

-l long listing

daemon - a process that does not belong to a terminal session. Many system services are daemons; network services, printing etc.

You can not just invoke a daemon, must detatch from terminal, below are 2 methods

method 1

1) fork() so the parent can exit, this returns control to the shell, and guarnetees that the new process is NOT a group leader

\* setsid() fails if you're a process group leader.

setsid() to become a process group and session group leader.

2) chdir("/") to ensure that our process doesn't keep any directory in use, otherwise you might interfere with SA work, i.e. unmounting

3) umask(0) - guarentees permissions -optional

4) close() fds 0, 1, and 2. This releases the standard in, out, and error we inherited from our parent process, then reopen

method 2

chdir '/' or die "cannot chdir to /: $!";

umask 0;

open STDIN, '/dev/null' or die "cannot read /dev/null: $!";

defined(my $pid = fork) or die "cannot fork: $!";

exit if $pid;

setsid or die "cannot start a new session: $!";

/etc/profile executes 1st

ps2asci

man -k searchterm

to remove duplicate records from a file

sort<file.txt | uniq

alias path="echo \${PATH} | awk -F: '{ for (i = 1; i <= NF; i++) printf(\"%02d:%s\n\", i, \$i) }'" # do this for CLASSPATH, PERL5LIB, etc..

for numbers, use –eq, -ne, -lt, -gt, -le

for strings, use =, !=, >, <, >=, <=

(opposite of perl)

to print to a printer

cat somefile | enscript -r -G -f Courier9 -b "Summary Report" -p uspoma0714

to allow a file to stay open

cat /dev/null > /tmp/somefile

ls -ltr | awk '{ print $9 }'

du -k | sort -n

find . -type f -mtime +30 -exec rm -rf {} \;

find . -xdev -size 100000000c -exec ls -l {} \; | sort -rn | head -10

rcp –r vision-nydev-a:/app/QA

ttsession starts a ToolTalk session to the ip address in $DISPLAY

pldd - like ldd for pids, shows linked libraries for an active job

/usr/ucb/ps –auwwwx - shows more of the cmd then /usr/bin/ps

ps –efl –o ruser –o args

ptree

sh is bourne, ksh/bash are extenstions of bourne (bash is called bourne again)

eval really means evaluate 1st, then invoke

exec really means invoke this command "in place" of whats running

in effect, the calling script exits, leaving the invoked cmd

running

exec also associates file handles

Citrix - type of server that speeds up access over a WAN, i.e users in Dublin

can use it to access an application running in New York

crypt 0220 encrypted.key encrypts whatever you type (stdin)

crypt 0220 < encrypted.key un-encrypt

windows ctl char problem ---> dos2unix, vi -b, cat -ev

windows line terminator is 1 byte more than unix, thus file sizes differ

to remove a unix file with leading metacharacter

rm ./-badfile or mv ./#badfile goodfile

shell cmds like cd wont show up in which/whereis, their man pages are part of ksh/sh

special built in cmds will end the script if theres an error

find . -name '\*.[chly]' -print | sort > file

find . -type f -mtime -1 | xargs grep core:Fr | awk -F: '{ print $1 }' | xargs ls -l

find . -type f -mtime +30 -exec rm -rf {} \;

find . -type d -mtime +30 -exec rmdir {} \;

find . -type f -exec grep load {} \; -print | more

find . -type f -mtime -1 -exec tail -10 {} \;

find . -name Tib\\* -ctime -8 -exec grep XS0306733014 {} \;

find -name "\*.pl" # must quote a wild card or else the shell will expand them

find . -name '\*.pl' |xargs grep -n socket

print all files and line#s down the directory tree where 'socket' appears

find . -type f -atime -90 # files accessed less than 90 days ago

find . | wc to count the # of files in a file tree

find . -type f -mtime -1 -print find only files modified < 1 day

find . -type f -mtime 1 -print modified 1 day ago

find . -type f -mtime +1 -print > 1

xargs sends input stream as many parms to 1 cmd, this may not work for you so...

-n 1 enforces 1 parm per cmd

find . -type f | xargs -n 1 tail -10 # same as –exec

find . –name “srchstring” –print 2> /dev/null

find / -name '\*FISS\*' -print 2> /dev/null

ls -ltr /var/adm/crash/\*Edit\* | awk '{ print $6 " " $7 " " $8 ", nsc27n01" }'

find . -type f -mtime -1 | xargs grep core | awk -F: '{ print $1 }' | xargs ls –l

od -h

man ascii

a

to print out man pages

man man | col -b > temp

or man man | col -b | lp

col - reverse line-feeds filter

cat –ev shows cntl characters

lsof who is using this file

lsof –i who is using this port

tar cvf uat\_tar \* -create a tar file

tar xvf uat\_tar -unpack it

tar -tvf /tmp/LON\_Static.0520232049 # to see whats been tar'd

tar -xvf /tmp/USA\_Static.0521020845 IndexRate # extract only the IndexRate file

untar 2 files from a gzipped file

gzcat patches/b4.tar.gz | tar xvf - site/bin/fissrec site/bin/fissrecnew

tar -z filter thru gzip

tar cvfz archive.tar.gz \*

truss - trace system calls, signals, useful if your not sure what a cmd is doing

truss find . -name \*cpp -print >output 2>&1

truss –p 123 # identify it by pid

gdb binary corefile or gdb –bt corefile # analyze core files

r cmds/s cmds/.netrc/.rhosts

.netrc

holds ftp info, make sure authority is 600, to override ftp -n

machine nyyst-uat login systumgr password systr1mgr

machine nyyst-live login systcc password systr1mgr

rdist -vf distfile - verbose does not execute, good to do first

* if you don’t do it from the right root directory, it will delete everything!
* without -f default is ~/distfile
* the distfile is somewhat like a makefile w/targets

dev:

${FILES\_ARTS} -> ${HOSTS}

install -R ${DEST\_DEV} ;

except ./${ALL\_EXCL\_DEV} ;

except\_pat ${PRM\_EXCL\_PAT} ;

notify ${NOTIFY\_LIST} ;

where 1st line says these files go to these hosts

Line 2 install (-R means remove extra files) ${DEST\_DEV} - root path of remote system

3rd line says except for these files

4th line says same but with wildcard, i.e. \*.core

5th indicates where to mail the log

the .rhosts file on the remote machine must have the machine that is trying

rcp nickfury:/customer/wayne/mort/reports/Det.TERM .

remsh lsunds97 -l anvil -n /usr/openwin/bin/xterm -ls -fn 7x13 -sb -sl 200000 -display $DISPLAY -e /apps/arts/anvil/trees/live-copy/site/bin/LoadReutersPrices2

great way to alias to other servers

alias remote='remsh some.server.com -n /usr/openwin/bin/xterm -display -title '\''production'\'' -name '\''PRODUCTION'\'' -fg red -bg black -ls -sb -sl 1000 -geometry 72x24+10+10 &

remote cmds must have the local machine in the .rhosts file within the home directory of the user who is logging on (i.e. the –l parameter if specified)

**signals**

sent via kill command

kill –9 same as kill SIGKILL or kill KILL

kill -15 same as kill SIGTERM or kill TERM

kill –l lists them all

ctl-C is same as kill SIGINT or kill –2

the shell ignores SIGTERM and SIGQUIT (unless you trap them)

SIGHUP by default exits a shell and kills all submitted jobs (unless they were submitted via nohup)

to listen and respond to a signal

trap “echo ‘I just heard a SIGINT’” INT

to ignore a signal

trap “” INT

trap "" 2 # both INT and 2 are (ctl-C)

trap "rm -f $TMPFILES" 0 # this is at pgm end

perl uses $SIG{int} = \&my\_subroutine;

trap ‘ bunch of cmds….’ ERR # trap any error

nohup - when a user logs off, the HUP signal is sent to its child jobs, nohup cmd submits a job that will ignore that signal

some signals are o/s defined, i.e. the alarm() function sends a ALRM signal

USR1 and USR2 are user defined

SIGCHLD 18 Ignore Child process status changed

SIGPIPE 13 Exit Broken pipe

SIGSEGV 11 Core Segmentation fault -- address reference boundary error/stack overflow

SIGHUP 1 Exit Hangup

**autosys**

. /opt/autotree/autouser/autosys.LV2

the autosys config file sets AUTOSYS, AUTOUSER, AUTOSERV, SYBASE

autorep –J %SYSDEV% -shows jobs -q to see the job desc

autoflags –a -shows configurations

autotimezone -l shows all autosys timezones

systlmgr@LIVE $ autocal\_asc

Utility to Add/Delete or Print entries in Calendar.

default global configuration in $AUTOUSER/config.$AUTOSERV

export PS1="`whoami`@`hostname` :"

**printing**

export PATH=$PATH:/opt/enscript/bin

export MANPATH=$MANPATH:/opt/enscript/man/

enscript -G -r -f Courier8 -F Courier12 -a 30-31 price\_variance

to print in landscape

lp -d systr\_prn1 -o landscape -o c catg.rpt

to wrap at 80 columns

fold –w 80 catg.rpt | lp

lpstat –p - printer status

lpstat - by itself is all user requests, try options o,d,R,s,t

echo $PRINTER (default lp)

to kill a print job

cancel -u buonorat (kills mine)

lpr –Pvismux (vismux is a script contained in /etc/lp/interfaces)

lpadmin -pvismux -v/dev/null -ivismuxif-hp (applies vismux to lpr)

lp –dnewprod similar, newprod was a destination, but it really was a script in

/var/spool/lp/admins/lp/interfaces

**various unix**

date '+ 20%y%m%d' # 20080325

which -searches your path

whereis -searches all paths

apropos

man -k -both take a search parameter for related cmds

database of commands is in /usr/local/man/whatis

man directories

/usr/share/man /usr/contrib/man /usr/local/man /apps/systr/perl5/man

ls -l | awk '{ print $5 " " $9 }' | sort -n # to sort by file size

du –r | sort –n - to isolate disk usage issues

uname gives OS (-n gives cluster/node -X gives all info –r gives version)

nslookup (give name or IP to run in non-interactive mode)

hostname (gives cluster and node)

bdf (df -k only on sun)

wc –l # of lines

no options = wc –lwc

ls | wc -l -tells you how many files are in your directory

rm –rf - all files and subdirectories, no prompt

rm –i - prompt

sort cxlamd.ext >| cxlamd.sorted.ext -execute command despite noclobber option

tee will output to the the indicated file as well as to the stdout

sort cxlamd.ext | tee cxlamd.sorted.ext

jil < systr\_autojob\_batch\_dev.txt | tee ~/autosys.log

to create a linked file in unix

ln -s /apps/yst-live/profile/buildenv buildenv # rm to remove it

cmp - compares files

cksum - does some algorithm, for file transmission check

multiple makes can cause different check sums and even byte counts

**isql**

- simple isql script – make sure its left justified

isql -SDSARTLDD03 -Ureporoot -Paardvark << EOF

select count(\*) from PartialTrans

go

EOF

isql -U$SYBASEUSER -P$SYBASEPASSWD -S$DSQUERY -o test.out << !

sp\_who

go

!

-similar method

cat <<-! | isql -U$SYBASEUSER -P$SYBASEPASSWD -w9999999999 -s"," -o $OUTPUT

sp\_who

go

!

-send it directly to sed

isql -U$SYBASEUSER -P$SYBASEPASSWD -S$DSQUERY << ! | sed -n '1,4 p'

sp\_who

go

!

-another method, anything in $() can be assigned to a variable

CUTOFF=$(isql -U$SYBASEUSER -P$SYBASEPASSWD -S$DSQUERY << EOF

select convert(char(8),dateadd(hour, -5, getdate()),8)

go

EOF)

CUTOFF=$(echo $CUTOFF | awk '{ print $2 }')

-add the -w 9999999999 option, otherwise 1 record gets cut up into 3 lines

- VERY USEFUL

isql -U$SYBASEUSER -P$SYBASEPASSWD -S$DSQUERY -w 99999999 << ! | sed -n '/repo\_ny/ p'

sp\_who

go

!

**exceed**

hummingbird is a subsidiary of opentext, makes exceed lets you start “X clients” via protocals such as telnet, ssh, rlogin, FTP

@u or @U username @d or @D display

/usr/openwin/bin/xterm -d @D -ls +rv -sb -sl 2000 -fg yellow -bg blue

Xconfig parameters enable you to paste

to copy drop down menus, set xceed config to unclick "copy on focus lost"

**mail**

mail vs. mailx (mailx is for the network)

mailx -s "Test subject" tom.buonora@barcap.com $datafile

mailx -s "Bony Wire" bonywire@barcap.com < $userfile

cat $file | mail tom.buonora@barcap.com

echo "To: tom@abc.com\nSubject: test subject\nthis is the msg" | mail [tom@abc.com](mailto:tom@abc.com)

-make sure no leading spaces

“d” delete in mail, but you can also vi the /var/ mailfile

mail tom.buonora@barcap.com <<- EOF

$(date)

Hello

EOF

$MAIL contains the directory of the file that contains the email

--- /var/spool/mail/$USER

<<- the dash indicates to ignore the leading tab

mail

-p print all msgs

at the ? prompt

# prints the msg#

- prints previous msg

+ or n next

d delete current

dp delete and print next

when you send input to the mail pgm, you can delineate header, subject, and text,

the header can contain keywords such as content type, mime version, etc..

this header can instruct the mail pgm to treat the subsequent data as a binary, text, or html attachment

telnet smtphost.ldn.bzwint.com 25 ---use cmds helo, mail from: rcpt to: data

to see clearcase views

buonorat@DEV $ ct lsview | grep 'systr'

\* systr\_dev\_view /VIEWSTORE\_2/BO\_LIBERTY/systr\_dev\_view.vws

# manpages

ct man lstype

# show all labels

ct lstype -kind lbtype

# show all non clearcase objects in your view

ct lsprivate

# show code from branch

ct find . -ver "brtype(CFGIT00002385)" -print

# show code from label (release)

ct find . -ver "lbtype (REL0131-20051105)" -print

# see history (lsvtree/lshist/desc)

ct lsvtree ff\_rates.pl

# see whats viewable and the rule thats letting you view it

ct ls -l

cleartool find . –all –version $cq\_num #show\_change\_set

cleartool desc –attype $cq\_num #show\_bug\_status

ct rmname - to remove from the view

ct mv - to rename it

If you want to see everything that's checked out, try:

alias allco="cleartool lscheckout -recurse /vobs/BO\_LIBERTY/SySTR"

to see the differences in a cleartool version

ct diff -pred systr\_autojob\_batch\_dev.txt

ct pwv - to see your current view

ct lsco - to see whats checked out

ct setview

ct man for help

ct mount /vobs/BO\_LIBERY if you cant see it..

3 ways to see code….

cleartool find . -ver "brtype(CFGIT00002074)" -print

cleartool lsvtree somefile # shows all branches

cleartool ls # also gives useful information

\* a perplexing issue was resolved with ct unco

if a bad umask, screws up permissions on a vob, do this (must be owner)

cleartool protect -chmod g+w include src

ct findmerge . -fversion /main/CFGIT00002122 -print -long -okgmerge

ct lsco -me -all --> shows whats merged to main, is still checked out

ct lsco -me -rec --> shows only whats checked out, not to main

ct lshist -all -since 19-Nov-04.03:00 // shows everyones change

ct lsco -rec | grep checkout | grep buonorat | sed -e 's/directory version/oneword/g' | awk -F" " '{ printf "%s\n", $5}' > ~/mine

bdiff report\_email\_lists report\_email\_lists@@/main/LATEST // changes you made

ct setview buonorat1\_unix\_view1

ct edcs - to edit your branches

ct catcs to just see it

ct mkbrtype CFGIT00002193

ct lshist somefile

cleartool find . -ver "brtype(CFGIT00002362)" -print

or

ct lsco -me -all

ct lshist -all -since 19-Nov-04.03:00 // shows everyones changes

ct lspriv - to see what youve got in your view

see whats out

ct lsco -rec | grep checkout | grep buonorat | sed -e 's/directory version/oneword/g' | awk -F" " '{ printf "%s\n", $5}' > ~/mine

ct lsco -rec -brtype CFGIT00002074 | grep checkout | sed -e 's/directory version/oneword/g' | awk -F" " '{ printf "%s\n", $5}' > ~/mine

merging thru the gui, in exceed

cleartool lsvtree -g reconcile\_adp

version-->merge-->from version

click on the 1

this supposed to work but it takes forever from ny

cleartool findmerge . -fversion /main/CFGIT00002193 -print -long -okmerge -abort

2 ways to merge branches

cleartool findmerge . -fversion /main/CFGIT00002193 -print -long -okmerge -abort

ct lsvtree -gra filename e.g. cleartool lsvtree -gra autosys

to find versions by branch and user and date

cleartool find . -ver "created\_by(buonorat)&&brtype(CFGIT00002193)&&created\_since(24-Nov)"

users setview and see /vobs/BO\_LIBERTY

autosys will see /view/systr\_dev\_view/vobs/BO\_LIBERTY

* believe set view is just a way of linking the /vobs to view/...

alias -x rm="rm -i"

to find this string in all files in current directory

grep 'DBH->' \* > find\_dbh.out

to grab fields 1 and 2 from a tilde delimited file

cat dnl\_fift.load | awk -F~ '{ print $1 $2 }' > dftest

and to find the duplicates...

uniq -d dftest

To insert a new line in the middle of a file

cut -b 1-48, "\n", 48-90 fix > testme

to parse out a few fields from a file (by byte position)

cut -b 70-84,536-540,585-616 posdldep > pos2

AWK -editor + programming language

-$0 refers to current record

-$1 refers to first field

-$2 refers to 2nd field

awk /credit/ {print $1} myfile

-this prints the first field for every record that has the string 'credit'

-default field separator is tab or space

awk /credit/ myfile

-this will print all records with string 'credit'

awk /^H/ myfile

-print all records that begin with H

awk '/P100/, /P103/' myfile

-print all records with P100 through P103 in it

grep 912828 becdldep | nawk '{ print substr($0,203,31) " , " substr($0,830,15) }'

awk 'pattern {action}' file

awk '$2~/foo/{print $3,$1}'

print the 3rd and 1st fields if the 2nd field contains foo

print is a default action

awk '$1 > $2' file

awk '$1 > $2'{print}' file

awk '$1 > $2' {print $0}' file

are all the same

BEGIN{} AND END{} blocks are available to run cmds

awk -f myscript.awk # runs a script

$NF # of fields

$NR current rec#

awk -F: use : as a field separator

to substring in a shell pgm

YYMMDD="`echo ${DDMMYY} | awk '{ print substr($0,5,2) substr($0,3,2) substr($0,1,2) }'`"

To view only the file names in columnar format

ll | awk '{ print $9 }'

sed has single character functions whose types are:

line-oriented, substitute, input-output, central, hold, get

d, n, a, i, c -line oriented functions

d (delete) n (replace) a (append) i (insert) c (replace)

sed '/q1/d' EMPFILE -deletes all lines matched by q1

sed s/old/new/g EMPFILE -replaces all old with new

to remove a full line from quotes

sed s/\"\$//g cuswrk | sed s/^\"//g > cusfilep

Pattern Matching (regular expressions)

"Old[abc]" -matches Olda, Oldb, Oldc

"^[abc]" -matches lines that begin with a or b or c

"Old[^abc]" -matches Oldd, Olde but not Olda (the ^ inside brackets means do not match these)

"[abc]$" -matches lines that end with a or b or c

"[abc]." -matches a or b or c followed by any one character

"\[abc\]" -matches explicitly [abc] (backslash suppresses the control

character)

sed

sed 's/AS400/UNIX/' tomtest1 > tomtest2 # search and replace

-n do not print

/p print (overrides -n)

-f run sed script

print 1st 2 lines

sed -n '1,2 p' test.out

sed '11,$ d' <file delete lines 11 to the end (prints 1st 10 lines)

these 2 are the same

sed -n '/repo/ p' < test.out

sed -n '/repo/ p' test.out

perplexingly you must use the -n to make it work

sed '/repo/' test.out # fails

p print use with -n

d dont print

-------------this prints the 1st 2 lines from sp\_who (headers), then greps for repo\_ny

isql -U$SYBASEUSER -P$SYBASEPASSWD -S$DSQUERY -w 99999999 -o test.out << !

sp\_who

go

!

sed -n '1,2 p' test.out

sed -n '/repo\_ny/ p' test.out

exit 0 -- by convention 0 is success, 1 is not found

diff returns 0 if no differences

grep returns 0 if found, 1 if not found, 2 if error

grep -i -E 'bcp|manual' \*

case insensitive, process the regex

same as egrep -i 'bcp|manual' \*

grep -c 'bcp' \*

gives file name and the # of matches

grep -n bcp \*

gives file and line#

grep -v bcp \*

print all lines not matching, useful for stripping out records

l

-i ignore case

-n print each line and line#

-c prints only the total # of lines

-v prints lines that do not match

fgrep -scan for multiple strings, multiple files

egrep -scan for multiple strings (uses regular expressions too)

- grep –e does this also

grep AST[PR] impactticket.pl.nyuarp02.log

egrep 'ASTPAC|ASETID' impactticket.pl.nyuarp02.log # multiple searches

F+ -matches one or more F's

F? -matches zero or one F

egrep '[Pp]erfect|[Ff]aulty' testfile -matches Perfect, perfect, Faulty, faulty

man –l output -searches man pages for string output

(you have to sort a file before using uniq)

uniq -u testfile -displays unique lines in the file

uniq -d testfile -displays duplicate lines

uniq -c testfile -displays every line with a count of its occurence

script -contains commands and UNIX responses from a session (job log)

script mylogfile -records your job

script -a mylogfile -appends to your file

split -300 cusfilep cus -this splits cusfilep into many files, 300 lines each, cusaa, cusab,

-the new files will be found in the /temp directory

csplit tranfilp '/CBNK/' '/GBON/' -creates 1 file with all CBNK trades, 1 with all GBON

pack/compress -compresses myfile into myfile.z

-pack compresses a bit more but takes longer to run

pcat -displays contents of a packed file

zcat -displays contents of a compressed file

du -displays disk space usage

ls -s buonora -shows file sizes for files in directory buonora

df (generic) - report number of free file system disk blocks

bdf - report number of free disk blocks (Berkeley version)

set or simple assignment (fldA=”Hello”)

-these are valid for your shell only

-view in set

export

-this is true for all child processes also

-a child can change it only for itself and its children

-view in env

typeset –x name=Tom

-this is the old Bourne way of exporting

$$ process ID

$\_ last variable

$? last retst (0 = success, 1= failure)

- some commands may put unexpected values in $?

$0 name of process

$1 1st parameter or set variable

$2 2nd

print "$PWD is evaluated so is \n so is `ls`"

buonorat@DEV $ print '$PWD is not evaluated neither is `ls`'

$PWD is not evaluated neither is `ls`

set –x # turn on trace

date > /dev/null # redirect output to never never land

MetaCharacters

[] matches 1 character only

[1-49] matches 1 thru 4 or 9

[!5] matches anything not 5

ls \*[!h] does a ls on all files not ending in h

ls [BD]\* does ls on all files starting with B or D

delimit with the colon (-d:), cut field 1 (-f1), sort, use more

cut -d: -f1 /etc/passwd | sort | more

**vi**

H,M,L -go to top, middle, bottom of screen

3H -go to 3 lines from top

-any # b4 a command repeats that command that # of times

to copy in a section of a file in vi

:e filename

:160,220ya f

ctl ^

:pu f

dw -delete word

D or d$ -delete rest of line

:se nu -show line #s

:e *file* open up new file

:r *file* copy in file after cursor

:!ksh to bring up korn shell

f5yy -yank in 5 lines into buffer f

:e# -go to other file

fp -put buffer f

:160,224ya f -yank in lines 160 \_ 224 into buffer f

:e#

:pu f -put buffer f

Semaphore hardware or software flag used to indicate the status of some activity. (2) A shared space for interprocess communications (IPC) controlled by "wake up" and "sleep" commands. The source process fills a queue and goes to sleep until the destination process uses the data and tells the source process to wake up.

shell options

-norc do not read .bashrc upon login

-noprofile dont read .bash\_profile

-rcfile "file" read "file" upon login

bourne shell (sh) is the default on most systems

doesnt have the line editing feature of ksh and bash

ksh adds the ctl-Z and fg job# functionality

also supports numeric data types

bash and sh require an eval() function to do math

uses .profile, .kshrc and .sh\_history files

**file systems**

/etc is for system configuration files

/home is for user home directories

/opt and /usr are read-only for software apps ( /opt is considered a “add-on” )

our autosys and mq was in /opt

/usr/local for apps to be protected from system upgrades

/var is for variably sized files, logs, or other output files go to /var or /tmp

/tmp gets cleaned out when the box gets bounced /var/tmp does not

/sbin binaries for super user to run system, it is not part of $PATH

where /bin and /sbin contain executables, /lib and /usr/lib contain modules, (.so)

/dev is devices, i.e. cat myfile > /dev/lp0 sends it to your printer

/home /tmp /usr /var are mountable, /bin /dev /etc /lib /sbin are required by the boot process, considerations about directory topology, NFS, shareable, etc..

i think clustering means /home /tmp /usr /opt are shared, /bin /sbin are not

cluster - contains several machines (nodes) that share file system and accounts, the login process will route you

Network File Systems – any software that shares recources over a network, an ever evolving area....some info on wiki

SAN - Storage Area Network vs. NAS - Network Attached Storage (one of them you could only see the local files)

NFS – Network File System is considered NAS

nas directories can have .ckpt directory (unlike san, nas visible across both nodes)

find checkpoint backups (cpio is a sort of file archive)

find ./bin | cpio -pumvd /opt/repo/fifm-nas/domains/finman-1.0.0-dev

outgoing connections use ports > 1024

/etc/inetd.conf has daemons such as pop-3 or apache, unnecessary daemons s/b commented out

openlog, syslog, closelog (system functions, there are man pages for them)

system log is in /var/adm/syslog/syslog.log (on sun try /var/adm/messages )

ITO reads this to look for remedy errors

logger –t $APPCRIT “Batch failure”

ito environments may be in /etc/profile.d itostub.rc and arts\_ito.profile

to edit the cron file

crontab -e

at 14:20 command executes once

crontab jobs are stored in /var/spool/cron/crontabs

at one time only jobs stored in /var/spool/cron/atjobs/

cron format is mm hh (day of month 1-31) (month of year 1-12) (weekday 0-6)

55 03 \* \* 6 is 3:55am every saturday

SECURITY

umask 027 gives

- user all permissions ( 0 )

- users group read and execute permission ( 2 )

* everyone else nothing ( 7 )

umask subtracts from the system default i.e., umask 027 subtracts 2 from the group and 7 from the others group

chmod tips

chmod u+s ndma\* sets on sticky bit to owner

chmod g+r ndma\* sets on read to group

chmod o+x ndma\* sets on execution to others

chmod a+x some unix permissions are only covered by "a"

the s code is defined as "set-owner-ID-on-file-

execution or set-group-ID-on-file-execution

permission for who. Useful only if u or g

is expressed or implied in who.

fuser tells you what pid (job) has a file opened

cat ftpit.sh | nl -- nl numbers output

> newfile -- same as touch newfile

>&- -- closes STDOUT (suppresses output)

> /dev/null -- the same but different

<&- -- close STDIN

>&2 -- send STDOUT to STDERR

2>ls.out -- send STDERR to ls.out

>| -- override noclobber

|& -- submit cmd as background job and share its stdin/stdout

co-process, its stdout is your stdin

date |& # create a coprocess

read –p DATE -- reads the stdout of the coprocess into $DATE

print –p -- prints the stdout of a coprocess

<&p The input from the co-process is moved to stdin

>&p The output to the co-process is moved to stdout

cat >> .profile << END -- append all this to .profile

export A=B

export C=D

END

-- << -END is the same except it ignores leading whitespace

file descriptors 3-9 should be opened w/exec

exec 4

read -u4 -- read from file 4

print -u6 -- print to 6

but you can redirect a file into stdin like this

exec 0< somefile

while read LINE

read ANSWER?”Enter answer” -- one line to prompt and read

read -- by itself, answer goes to $REPLY

exec -- either opens a file OR runs a cmd ***in place of*** the current process

exec 0<myfile # opens myfile for input

while read LINE # reads each line into $LINE

while read -r fld1 fld2 fld3 # reads and parses fields based on $IFS

set -o noglob -- turns off special characters

set -f -- ditto

echo $(ls -l) -- command substition, put in parenthesis to evaluate

echo `ls -l` -- ditto

echo $(< myfile) -- same as cat myfile

~+ -- same as $PWD

: is the null cmd like a comment, except after a ; the rest is executed

to debug a pipe, use tee

awk 'some shit' | tee awk 'more crap'

in a series of pipes the right most function controls the exit status

do not put spaces around = in an assignment, i.e. X=r

${Variable} is useful to allow concatenations, i.e. ${prefix}rootword

command substition can be either 2 backticks or $()

find $HOME -name \*.tmp -exec rm {} \;

- the backslash prevent unix from interpreting the ;,

as opposed to properly passing it to the rm command

2 double quotes will always send every thing within as 1 parameter

"$\*" will send all as one parm, $@ is more appropriate but without

the quotes they are the same

like perl and $@, a for loop defaults to $\*

for o

do

echo $o

done

other for loop

for file in \*.jil

do

sed -e 's/apps/anvil\/trees\/ny-dev/g' $file > $file.tmp

sed -e 's/nsc27n01/nsc77n02/g' $file.tmp > $file

done

in shell, must eliminate blanks in assignments, i.e.

fld1="abc"

[ ] is same as test, opposite of = in that must have spaces on both sides

read fld1 fld2 fld3 - read defaults \s as field terminator ($IFS)

read without fields puts the whole line in $REPLY

$! pid of last background cmd

$- last argument to previous cmd (like cd -)

$TMOUT seconds for shell to timeout during non-activity (0 means unlimited)

set -o noexec - to syntax check a shell script

ksh -v verbose mode

ksh -x expands and prints each cmd before running

result=${var1:-word} # sets $result to "word" if var1 not defined

result=${var1:+word} # opposite, set $result to "word" if var1 IS defined

result=${var1:=word} # same thing but will also assing "word" to $var1 if $var1 is not defined

result=${var1:? too bad} # prints too bad and exits script (if $var1 not defined)

note by doing var1="" or var1=$1 where no parm was passed both result in $var1 being "not defined"

X=abc

buonorat-lsg31n02: echo $X

abc

buonorat-lsg31n02: print ${#X} -- get length

3

unset X

X=${Z:-xyz}

X=${Z-xyz} -- both set X to Z but if Z is not set, then to "xyz"

print ${USER:-${LOGNAME:-${X}}}

print ${USER:=${X}} -- if $USER is not set then set it to X

cd pl cc # replace pl with cc in the path and cd to it

cd - # cd to previous dir and output it

$ENV should hold the script to run when a new shell is executed

$TERM is vt100 on systr, but xterm on arts

set -o shows all options

set -x show cmd while it is executed (set +x to shut it off)

whence -v shows the origin of a cmd, i.e. is it a alias of what

some aliases are preset

echo is an alias of print -

r is an alias of fc -e (execute cmd#)

history -r grep shows recent cmds with grep

alias -t track an alias to a path, so path lookup is unnecesary

subshells

within parenthesis, a new shell is generated

changed values are local, but all values from the "parent" are copied in

( print $oldvalue; $oldvalue = $newvalue )

$0 name of sheld $1 -> $9 positional parameters (shift moves them over)

$# the # of parms

$@ and $\* are both all parms, @ is a pattern match for exactly 1, \* matches all

"$\*" enclose all in quotes

[[ $X == $Y ]] - conditional, must have whitespace, may require quotes

[[ $X = ??? ]] && print "$X has exactly 3 characters"

[[ -a $file ]] && print "$file exists"

[[ -z $file ]] && print "$file is empty"

test and [ .. ] are also used but dont work as well

grep Snapple S\*

- the shell translates metacharacters 1st

grep Snapple 'S\*'

- it wont

return status

a cmd like who normally returns a 0

! who

would return 1

call with abc def

last=\$$#

print $last # gives $2

eval last=\$$# # makes a 2nd pass, translates $2 to def

print $last

cp "$@" $PWD

assignments

let x=y

(( $x = $y ))

typeset gives attributes to fields

-u make it uppercas

-l lower

-x export

${#b} length of $b

:= assign if it is null or unset

print ${var1 := $var2}

sets var1 to var2

- use if unset

:- use if unset or null

> myfile will "empty" the file

rm will "remove" it from your view

but will not deallocate the disk space

set -n same as set -o noexec

set -x same as set -o xtrace

set -v same as set -o verbose

ksh -v mypgm # call pgm in verbose mode

ksh -n mypgm # syntax check it, i.e dont execute

"here documents" refer to bundling input i.e.

"co-process" refers to a submitted command with |&

its IO can be redefined

> file is better than rm file

1st one will empty the file without deleting it

2nd one will delete it without emptying it from the system!

TESTVAR=abc: is not an identifier # typically a bourne shell error on

export TESTVAR='abc' fix this by assigning/exporting in separate cmds

login/environment

------------------------------------------------------------------------------

/etc/profile executes 1st

~/.profile executes 2nd (set up terminal, PATH, buildenv, exports)

bash will also execute .bashrc

~/.kshrc executes for every shell (aliases, export)

autosys, crontab, ftp do not source anything

su and rlogin will source only with a parameter

passwd file contains the default shell and home directory

ypcat passwd | grep buonora

a shell can be login (reads .profile) or non-login

i.e. a ftp connection starts a non-login shell

xterm -ls starts a login shell, typing ksh, or su do not

commands within ( ) are executed as a subshell, they are like a local block

( cd $HOME; pwd ) // new path

pwd // original path

when you login, ksh executes .profile and then $ENV if it is set

each time you start a subshell, it executes $ENV

\* wise to put your .kshrc into $ENV or else with each new shell, youll lose them

if a subshell or function returns assignments but does not execute them,

use eval, i.e.

eval ( tset -sun ) - executes the assignments within the subshell so they

are applied to the outer shell

set -o vi - to set your control characters

set -o noclobber -do not automatically overwrite existing files

set –o - to see how options are set

set shows all assigned variables, exported or not

unset removes an assignment

env shows all exported variables

printenv -shows “whole” environment

Xwindows/tty

------------------------------------------------------------------------------

from unix

xterm -display 30.254.31.145:0.0 -title 'nyyst-live.ny (systlmgr)' -name 'nyyst-live.ny' -fg white -bg black -ls -sb -sl 1000 -geometry 158x64+10+10 &

from exceed icon

/usr/openwin/bin/xterm -d @D -ls +rv -sb -sl 20000 -fg yellow -bg darkblue

/usr/openwin/bin/xterm -d @D -ls +rv -sb -sl 20000 -fg yellow -bg darkgreen

you can put shell cmds in the cmd line...

you can execute cmds, set environment

export TRADER\_EMAIL='tom.buonora@barclayscapital.com'; export TRADER\_EMAIL; $REPO/bin/anvilrepo -d @D

some troublesooting tips, try REXEC vs RLOGIN, .rhosts may matter, click host feedback box

rsh/rksh there are restrictions to what the user can do, usually used to run 1 pgm

fn is font

-n is the xterm description, can see this when window is minimized

-T is the title - this goes to top of window

-sb use scroll bar

-sl the # of lines to save off (default = 64)

@a@ is apparently my IP address

-s vs. +s +s is scroll synchronously, -s is faster but screen is not "up to date"

-aw allow autowrap

xprop, xwininfo, xset show or set properties

.Xdefaults in your home directory is very useful

format is name.class.resource: value

mwm\*activeForegorund: ghostwhite

mwm would be the –name parameter in the xterm command

mwm\*Tokyo\*iconImageForeground:

Tokyo is a class

type xprop to see

WM\_CLASS(STRING)=”mwm”, “XTerm”

to re-read your .XDefaults file,

xrdb ~/.XDefaults

on reflections who am i gives pts/ta Mar 29 20:19

tty gives /dev/pts/ta

in exceed who am i gives pts/1 Mar 29 14:13 (167.203.82.142)

tty gives /dev/pts/1

\* who am i is essentially the who cmd accepting “am i” as 2 parms, it lists the user as the device, whatever file is in the /dev directory

stty remaps your keyboard

stty erase ^? # sets the delete key to erase characters

stty erase ^h # sets backspace key to erase

stty -a # shows all settings,

stty can redefine the keyboard ^H is backspace

$DB::fork\_TTY=`tty`; # allows you to debug forks

tty - cmd that prints what file is connected to stdin, i.e. /dev/pts/4

- in unix terminals are given files in /dev/tty

- a fork cmd, the job is given a different tty, so you must tell the debugger

--------------------------------------------------------------------------------------

$EXINIT shows your default editor's options

vi options can be set or unset, i.e. set ignorecase, no autoindent, showmode

:se nu to see line#s

set –o vi

vi has a fmt function

:%!fmt will format a whole file

5!!fmt will format the next 5 lines

:map! a A will map all a to A (so you dont need upper case)

xterm & # very useful to open up new window in child shell

oclock and xcalc # fun

double click on a word to highlight it

xman gives man pages related to x (in a window!)

LD\_LIBRARY\_PATH is an environment variable you set to give the run-time shared library loader (ld.so) an extra set of directories to look for when searching for shared libraries.

SRV4 - ATT unix software distribuition (System V v.4 )

BSD - a unix software distribution from group at UCAL Berkely

most unix is considered SRV4 with added BSD

### Messaging

Synchronys communications require pgms to be running at same time

Asynchronous - pgms are independent of each other, messages may be chained via some transaction ID

Q Manager – one per machine, administers the q’s, transmission q’s, channels

Q/Transmission Q if the channel is down, a persistant msg stays on the transmission Q

Channel - connects local and remote Q managers, the "socket" between two MQ managers

* one way, so referred to as either receiving/sending

Binding connection – when app runs on same server as the QM

Client connection – when app is on a different server

Barclays used config, rules, and default files to control how MQ was read and acted upon, i.e. a msg was read, and the rules would tell the program to parse the data and call a stored procedure, if the proc failed, the msg would go to a defined error Q

dead-letter queue - a.k.a. undelivered-message queue, if a q is full or if the msg is invalid length or characters, the msg will go here

The "listener" has the function of detecting connections from incoming channels and managing them

Persistant messages are logged, and can be resent in the event of a failure

Non-persistant messages are not, are lost if failure

Languages such as C, RPG use MQI (MQ Interface) to use cmds such as MQOPEN, MQCONN, MQPUT, perl ok too

JMS

Provider - MQ or RV, the messaging app

Client - the app then sends or receives the msg

Producer/Consumer - the JMS client that sends/receives

Queues - FIFO

Topics - sends to all consumers (synchronous, app must be up)

distributed topics/Qs send every message to all managed servers

server affinity on will send all to one

Point to Point/Queuing asynchronous

Publish/Subscribe the msg is delivered to all subscribors who are active

if the subscribor is durable, it is asynchronous

A queue/topic may filter by definition, or the message ejb may define a filter

bridges connect 1 destination to another, a destination could be external Qs, i.e. MQ/tibco, or a jms q or topic

Others: TIBCO Rendevous, Tuxedo, and even OpenAdaptor are considered Message-Oriented-Middleware

good way to add all jar files to classpath, and exclude some

jarpath=$SANDBOX/lib

JARS=`find $jarpath -name "\*.jar"`

for i in $JARS

do

ls -l $i | fgrep -e weblogic.jar -e webservices.jar -e webserviceclient

res=$?

if [ "${res}" -ne 0 ]

then

echo "adding to classpath $i"

CLASSPATH="$CLASSPATH:$i"

else

echo "not adding $i"

fi

done

**function calling**

somefunc()

{

if [ -z "$1" ]

then

echo "no parm passed"

else

echo "$1 passed"

fi

}

somefunc "Abc"

somefunc

functions can be sourced in so they can be called by any script

they can also be exported so they can be called by a subshell

**example of testing for interactiveness as well as how to set clearcase view**

PATH=$PATH:/usr/atria/bin:.

bg\_test=`ps -efl | grep $$ | head -1 | awk '{ print substr($13,0,3) }'`

if [ "$bg\_test" == "pts" ]

then

echo "`date` :interactive " >> $HOME/.login.hist

cv=`cleartool pwv -short`

if [ "$cv" != "fifmbldusr\_dyn\_view" ]

then

echo "setting clearcase view to fifmbldusr\_dyn\_view"

cleartool setview -log fifmbldusr\_dyn\_view

exit

fi

else

echo "`date` :batch " >> $HOME/.login.hist

fi

export PS1="`whoami`@`hostname` :"

ftp –n hostname <<-!

user $1 $2

cd $3

get file

quit

!

case `date +%a` in # previous business date

Sun) += -172800

Mon) += -259200

\*) += -86400

esac

ssh

====================

- 3 step process to enable

on client

ssh-keygen -b 1024 -P "" -t dsa

on server

scp client\_mach:homedir/.ssh/id\_dsa.pub .

cat id\_dsa.pub > authorized\_keys

CDE (Common Desktop Environment) is a window system, ~/.dt/.dtprofile is executed first

**Caching**

We used gemfire 4.1, 5.1. Gemfire over kodo. Databases have their own cache. Who doesnt have a cache?