Cheatography

Linux, Bash, and System Administration Cheat Sheet

by beersj02 via cheatography.com/89657/cs/20407/

Bash Scripting Basics

#!/bin/env bash — the 'shebang' used to tell the operating system the path it should use to interpret the file

bash *file-name*.sh — run the bash script in terminal

./ *file-name*.sh — run the bash script in terminal if set to executable

<*parameter*> — use in documentation to specify if a parameter is required when running script

[*parameter*] — use in documentation to specify is a parameter is optional when running script

— used to make comments throughout script

|| — logical OR

&& — logical AND

$# — resolved to the number of arguments that have been passed to the script

$0 — refer back to the script name

$1, $2, etc. — refer to user input (parameters) that user can add when running script, separated by a space

exit [0-255] — exit script and return number from 0 to 255. 0 means everything worked as intended, but other values can be used to denote errors that the script ran into

Bash Loops and Conditions

if fi — basic structure of all if-then-exit, if-then-else, or if-elif-else statements

if *condition* ; then *do-something* — if condition is met, do something

if *condition* ; then *do-something* else *do-something-else* — if condition is met, do something, otherwise do something else

if *condition* ; then *do-something* elif *condition2* ; then *do-something-else* else *do-final-thing* — if condition is met, do something; however if a different condition is met, then do something else; otherwise do the final thing

while *condition-is-true* ; do *action* done — perform the action as long as the condition is true

until *condition-is-true* ; do *action* done — opposite of while loop, perform the action until the condition becomes true

sleep *time* — sleep or wait for a specified number of second before continuing through script, usually performed within loops

Bash Loops and Conditions (cont)

for *value* in *list-of-values* ; do *thing-with-value* done — iterate over a list of values

for ((*counter=number* ; *counter<=number* ; *counter++*)); do *something* done — start at counter is equal to a number, then do something and increment the counter by 1 until the counter is greater than another number

for counter in {*starting-value..ending-value*}; do *something* done — brace expansion that iterates over a number range or character range from starting value to the ending value

{*starting-value..ending-value..increment-value*} — specify the increment value in a for loop, otherwise the default is 1

for ((; ;)); do *something* done — infinite loop

break — can add to while or for loops to exit from the loop but continue the rest of the script

continue — used to skip current iteration of a loop and continue to the next iteration of the loop

cut — cut different parts of a string

basename *path* — get the filename from a given path

Bash Arrays and Functions

***array*=(*"elements" "of" "array"*)** — create an array of strings

${*array*[0]} — get the first element of the array

${*array*[*]} — get all values in the array

${*array*[-1]} — get the last value in the array

${*array*[@]} — expand all of the array elements

declare -A *associative-array* — declare an associative array that allows string indices, similar to a dictionary in Python

***associative-array*=([*"association"*]=*"string"*)** — add an association to an associative array

***array*+=(*"new" "elements"*)** — append elements to the end of an array

shift — move argument $2 to $1

***function()* { *content-of-function* }** — define a function

alias — list all aliases defined in the current session

alias *alias*='*bash-command* ' — define an alias

type -a *command* — tells us if command is an alias

Cheatography

Linux, Bash, and System Administration Cheat Sheet

by beersj02 via cheatography.com/89657/cs/20407/

Automated Commands

man 5 crontab — view manual page for crontab

crontab -e — edit scheduled tasks in the */var/spool/cron/crontabs* file

crontab -l — list scheduled tasks

*** * * * * find *directory* -exec ls -l {} \;** — find files on *directory*

*** * * * *** - cron format (0-59 minutes, 0-23 hours, 1-31 day of month, 1-12 month, 0-7 day of week)

0 1 1 * * find /temp -atime 3 -exec ls -l {} \; — run the command just on the first day of each month

0 1 * * mon find /temp -atime 3 -exec ls -l {} \; — run the command once a week on a Monday

0 1 1,15 * * find /temp -atime 3 -exec ls -l {} \; — run the command on the 1st and 15th day of each month

0 1 1-15 * * find /temp -atime 3 -exec ls -l {} \; — run the command every day from the 1st through the 15th, inclusive

0 1 */5 * * find /temp -atime 3 -exec ls -l {} \; — run the command every fifth dat (1st, 6th, 11th, etc.)

at — reads commands to be executed from a file or from standard input

atq — show which commands you have in the *at* queue, displays job number, date of planned execution and job owner

atrm *job-num* — delete a job from the queue by specifying *job-num*

System

& — puts command into the background, allowing you to continue executing other commands

du — display disk usage statistics

df — display free disk space

free — display amount of free and used memory in the system

kill — get rid of a command in the background

man *command* — show manual for *command*

shutdown now — shutdown machine

Download and Unpack

wget *file-url* — download a file

tar -xzf *tar-file* — extract a tar file

Package Management

dnf upgrade — update the system and all of its packages

dnf search *package-name* — search for new software called *package-name*

dnf provides *package-name* — check package name to install

dnf install *package-name* — install new software packages

dnf remove *package-name* — remove a package from the system

System Logs

who — produce information on who is logged in

w — produce information on who is logged in

finger — produce information on who is logged in

id -u *username* — get the user ID for a specific user

journalctl — view the log of the entire system

Q — quit from journalctl log

journalctl -f — follow the logs in real time

journalctl -u sshd — view only log entries for ssh unit

journalctl -u httpd -n 3 — vie a specific number of log entries (i.e. 3)

journalctl _UID=1000 — view log entries for a specific user by giving user ID

journalctl --since "YYYY-MM-DD HH:MM" --until "YYYY-MM-DD HH:MM" — filter and display log entries for a certain time period

dmesg — view all kernel messages from the last boot of the machine

last — display last user logins

history — list previous commands used

history | grep *keyword* — search for a command by *keyword* in history

!*command-num* — repeat a command from history and run the command

script — record all output for the session in a file

exit — exit from scripting session

Linux, Bash, and System Administration Cheat Sheet

by beersj02 via cheatography.com/89657/cs/20407/

Secure Shell

ssh — gives *ssh* command information

ssh *username@ip-address* — log into remote system

ssh-keygen — generate public/private key pair

ssh-add — command for adding SSH private keys into the SSH authentication agent for implementing single sign-on with SSH

ssh-keyscan — for retrieving public keys from servers

scp *file-path username@ip-address*: — copy a file from your local system to remote system

scp *username@ip-address*:*file-path* — copy a file from the remote system to your own system

scp -r *username@ip-address*:*directory* — copy a directory from the remote system to your own system

exit — terminate the shell

~ + Ctrl-Z — suspend the remote login session

File Searching

find — search for a file or directory on your file system

find /home -name *.jpg — find all *.jpg* files in the */home* and sub-directories

grep *options pattern files* — searches through *files* for a particular pattern of characters, and displays all lines that contain that pattern

grep -r *pattern dir* — search recursively for pattern in *dir*

locate *file* - locate a file

Important Directories

/ — root directory

/bin — the most essential Unix commands (such as *ls*)

/boot — location where the kernel and other files used during booting are sometimes stored

/dev — contains device files, the interface between the filesystem and the hardware

/etc — contains configuration files, which can generally be edited by hand in a text editor

/etc/passwd — contains user information in a certain format (*username*:*password*:*uid*:*gid*:*gecos*:*homedir*:*shell*)

Important Directories (cont)

/etc/skel — sample startup files you can place in home directories for new users

/home — contains a home folder for each user

/lib — contains libraries needed by the essential binaries in the */bin* and */sbin* folder

/opt — contains subdirectories for optional software packages

/proc — the interface between the filesystem and the running processes, the CPU and memory

/root — the home directory of the root user

/sbin — very common commands used by the superuser for system administration

/tmp — temporary files stored by applications

/usr — contains applications and files used by users

/usr/bin — application/distribution binaries meant to be accessed by locally logged in users

/usr/sbin — application/distribution binaries that support or configure stuff in */sbin*

/usr/include — standard location of include files used in C programs (such as <*stdio.h*>)

/usr/src — location of sources to programs built on the system

/usr/local — programs and data files that have been added locally by the system administrator

/var — administrative files such as log files, used for various utilities

/var/spool — temporary storage for files being printed, sent by UUCP

Ownership and Permissions

sudo — log in or run program as root user

ls -l — display ownership and permissions

adduser — create a user account (as root)

passwd *account* — set password for *account* (as root)

userdel -r *account* — delete an account and account's home directory (as root)

chown — change owner of a file

chown *userid* /home/*userid*/ — make user account owner of home directory (as root)

chgrp — change group

Ownership and Permissions (cont)

chmod ugo *file* — change the user, group, and others permissions for *file* (ugo given in base 8, where u is the user, g is the group, and o is others)

chmod [ugo][+-=][rwx] *file* — give, take away, or set the read, write, and/or execute permissions for user, group and/or others for *file*

7 — read, write and execute permissions

6 — read and write permissions

5 — read and execute permissions

4 — read permissions

3 — write and execute permissions

2 — write permissions

1 — execute permissions

0 — no permissions

chmod 644 *file* — standard permissions for files

chmod 755 *dir* — standard permissions for directories

find / -user *username* -ls — find files associated with a user

File Management

ls — list items in your current directory

ls -a — list all items and hidden files in your current directory

ls -l — list items, including their size and permissions, in your current directory

pwd — prints path of current working directory

cd — change directory to home directory

cd *dir* — change directory to *dir*

cd .. — go up one directory

cp *file1 file2* — copy *file1* to *file2*

cp -r *dir1 dir2* — copy *dir1* to *dir2*, recursively

mv *file1 file2* — move *file1* to *file2*, or just change file name

rm *file* — remove *file*

rm -r *dir* — remove directory *dir*, recursively

echo *text* — outputs *text* to standard output

echo "*text*" > *file* — redirect *text* to *file*

touch *file* — create *file*, such as an empty txt or zip

cat *file* — concatenate *file* and print to standard output

head *file* — output first 10 lines of *file*

tail *file* — output last 10 lines of *file*

File Management (cont)

less *file* — view *file* instead of opening in an editor, allowing page navigation

sort *file* — used to sort a file, arranging the records in a particular order

ln -S *target new-name* — make links between files

nano *file* — open *file* in nano text editor

nano -v *file* — open *file* for read only in nano text editor

Git Commands

git clone */path/to/repository* — create a working copy of a local repository

git add * — add all edited files to staging

git add *filename* — add specific filename to staging

git commit -am "*commit message*" — commit changes to head (but not yet to the remote repository)

git push — send changes to the master branch of your remote repository

git status — list the files you've changed and those you still need to add or commit

Miscellaneous

yes "*string*" — echo *string* in infinite loop

cal — prints an ASCII calendar of the given month or year

date — display current system time

true — does nothing and finishes with zero exit code, indicating success

false — does nothing and finishes with non-zero exit code (often 1), indicating failure

clear — clears the screen of the terminal