Note that anything from the presented slides + labs are fair game for exam questions!!

 \*\*\*

1. What should a sysadmin monitor system-wise?

Logs, users, performance, access, configure systems, document changes, backups, security, system capacity

1. Know the differences between the different types of software licenses (e.g., freeware, open source, etc)

Freeware: Free program

Shareware: Licensed, trial period, features disabled

Open Source: Give access to your own code, not proprietary code

Closed Source: Software sold commercially from a manufacturer

1. Know how to show files in a directory, absolute vs relative paths, showing hidden files, etc.

Ls

Ls –a

Absolute: starts from the root on down - /home/user

Relative: Short hand paths - ./../some directory (. Current dir, .. Previous dir)

1. Describe logging, logging facility/priority, how to use the logger bash command, what a sysadmin should do in terms of logging best practices.

Logging: It is the record keeping for actions done by the system, programs, scripts and users of a system, or multiple systems on a network or from a remote site. Events too!

Facility: source of the message

Priority: short hand description of log contents, severity and their importance.

Logger command: logger “Something happened” which is sent to the general log /var/log/messages

Best Practices:

* Don’t log everything, only what is needed for the current environment and company guidelines
* Log to a different partition
* Have scripts log actions
* Track all critical failures
* Separate debug and access logs
* Cleanup old logs

1. How do you compile and install a program from source, including a description of each step.
   1. Congifure - ./configure
      1. Check dependencies
      2. User preferences
      3. Generates intermediate files
   2. Make
      1. Compiles source code into binary
   3. Make install
      1. Makes program available system wide or locally (accessible to you)
2. Why would I want to use RPM instead of YUM?

I would use RPM to download and install specific or necessary packages.

1. What are the Debian versions of these commands?

apt-get – download and install DPM packages

apt-get remove/purge – remove a DPM package

apt-get update/upgrade – update/upgrade a DPM package

add-apt-repository – add new repositories

dpkg

1. What does RPM and YUM even do?

RPM: Package Manager that can easily install packages but lacks dependencies and version installation. Needs to install packages individually. Does actually check for dependencies, but doesn’t install them. Must have the .rpm file to install.

YUM: Package Manager that takes care of dependency problems when installing packages. Uses repository to install. (Metapackage Manager)

1. What are filesystems and why do we need them. Know at least one Linux and one Windows filesystem.

Organize the metadata of the system. We can’t utilize data or run programs without them.

Linux: EXT-4, XFS

Windows: NTFS, Fat32

1. Know the differences between journaling and copy-on-write in terms of filesystems.

Journaling: write to a temp file before commiting to storage/inode table. Deleted after write success.

Journaling Approaches:

* Data mode: inode and file data journaled
* Writeback mode: inode only written, no control over file data
* Ordered mode: inode only written, but not removed until file data is written

Copy-on-write: snapshots of data rather than writing to temp file. Creates new file rather than updating over the old file. Includes superblock and inode metadata

1. Schedulers (Crond, atd)

Crond: System daemon that executes tasks repetitively in the future.

atd: System daemon that executes tasks at a future time.

1. Know the difference between the cron and at daemons, including examples of what you would use each for.

Crond: Periodic, frequently done tasks. Check updates, schedule backups, schedule logging

Atd: task needed to be done at a specific time or later date, not frequently done. System restart, email another user after task completion, user log report before logging off/exiting terminal

1. Describe the different types of processes (parent, child, zombie).

Parent: Process that created another process (child process)

Child: Process created by another process (parent process)

Zombie: child process which doesn’t release its resources when terminated.

1. Know what the init process is, what its PID/PPID are, how to kill processes vs killall.   Understand why kill -9 (SIGKILL / absolute kill) is special with respect to killing processes.

Init process: init daemon starts most other daemons during the system initialization process. Includes those that allow for user logins. Login program starts a BASH shell, which will then interrupt user commands and start all user processes.

PID: process ID assigned to a process to uniquely ID it.

PPID: Parent Process ID

Killing processes: To kill a process, give a kill signal and PID. If no kill signal given, the default SIGTERM signal is used.

Killall: command that kills multiple processes of the same name. Takes kill signal number as an option. Uses process name instead of PID. Default signal is SIGTERM

Kill -9 cannot be ignored

1. Be able to convert permissions to their octal form, be able to assign various permission types to users/groups/everyone.

X - 1

R - 4

W – 2

Users:Groups:Everyone

Chmod g+x gives the group section execute permissions

Chmod 674 rw:rwx:r

1. Know about the special permission bits (sticky bit, suid, etc) and what it means to umask permissions.

Sticky bit: run with the privileges of who created it

Suid: set user ID, when set on a file, the user who executes the file becomes owner of the file during its execution. User inherits permissions.

Umask: Octal number. Customize default permissions. Run umask get 4 octal numbers which subtract from 7 to give default permissions to new files made.

1. Know the purpose of the /etc/passwd, /etc/group, and /etc/shadow files.

Passwd: Contains user information (Username:Password, not actually shown: UID: GID: UID info: home directory:shell 7 fields)

Group: Keep track of all groups in the system.

Cat etc/group | grep wheel

Shadow: Hashed passwords

1. Be able to write some basic bash code and scripts. For instance, how to define an array, print a value, play with environment variables, copy a file, take in command line arguments, write a basic for loop, etc (basically, the stuff we covered in lab).

#!/bin/bash

bash script arguments

Echo $1 – print first argument

$0 – name of the script itself

cp

mv

rm

useradd

cd

mkdir

man

cat

less

more

grep

echo

expr

date

GO OVER C PROGRAMMING NOTES AND FILES