**Damian Rozpedowski HW 2 Operating Systems**

**A**

**1)**

damianrozpedowski@Damians-MacBook-Air ~ % cd

damianrozpedowski@Damians-MacBook-Air ~ % ls -l

total 5016

drwx------@ 3 damianrozpedowski staff 96 Jan 26 2022 Applications

drwx------@ 8 damianrozpedowski staff 256 Apr 18 21:40 Desktop

drwx------+ 5 damianrozpedowski staff 160 Jun 2 06:54 Documents

drwx------+ 14 damianrozpedowski staff 448 Aug 29 14:42 Downloads

drwxr-xr-x 12 damianrozpedowski staff 384 Aug 25 2022 JPMC-tech-task-2

**2&3)**

damianrozpedowski@Damians-MacBook-Air ~ % cd /bin

damianrozpedowski@Damians-MacBook-Air /bin % ls -l

total 9544

-rwxr-xr-x 1 root wheel 150608 Dec 7 2021 [

-r-xr-xr-x 1 root wheel 1326560 Dec 7 2021 bash

-rwxr-xr-x 1 root wheel 151792 Dec 7 2021 cat

-rwxr-xr-x 1 root wheel 136944 Dec 7 2021 chmod

-rwxr-xr-x 1 root wheel 152640 Dec 7 2021 cp

**4)** List 6 that I recognize: **Bash, Echo, Ls, Sleep, Mkdir, and Kill**

**5)**

damianrozpedowski@Damians-MacBook-Air /bin % cd /dev

damianrozpedowski@Damians-MacBook-Air /dev % ls -l

total 1

crw-rw-rw- 1 root wheel 0x1e000000 Sep 25 14:20 aes\_0

crw------- 1 root wheel 0x16000001 Sep 25 14:20 afsc\_type5

brw------- 1 root operator 0x2000000 Sep 25 14:20 apfs-raw-device.2.0

crw------- 1 root wheel 0xb000000 Sep 25 14:20 auditpipe

crw-r--r-- 1 root wheel 0xa000002 Sep 25 14:20 auditsessions

I recognize some devices such as Galaxy Buds and Bluetooth, but a lot of them I don’t recognize.

**6-7)**

damianrozpedowski@Damians-MacBook-Air /dev % cd /etc

damianrozpedowski@Damians-MacBook-Air /etc % ls -l

total 832

-rw-r--r-- 1 root wheel 515 Jan 5 2022 afpovertcp.cfg

lrwxr-xr-x 1 root wheel 15 Dec 7 2021 aliases -> postfix/aliases

-rw-r----- 1 root wheel 16384 Jan 5 2022 aliases.db

drwxr-xr-x 9 root wheel 288 Dec 7 2021 apache2

drwxr-xr-x 17 root wheel 544 Dec 7 2021 asl

Heard about ssh and shells

**8)** Most used permission: rw-r--r--

rw- owner of file has read write permission

r- others have read-only permissions

**9)**

damianrozpedowski@Damians-MacBook-Air /etc % cat /etc/passwd

##

# User Database

#

# Note that this file is consulted directly only when the system is running

# in single-user mode. At other times this information is provided by

# Open Directory.

**This command did not work I had to use this instead,**

damianrozpedowski@Damians-MacBook-Air /etc % dscl . -read /Users/damianrozpedowski

NFSHomeDirectory: /Users/damianrozpedowski

Password: \*\*\*\*\*\*\*

Picture: /Library/User Pictures/Flowers/Lotus.tif

PrimaryGroupID: 20

RealName: Damian Rozpedowski

RecordName: damianrozpedowski

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

**B**

**10)**

damianrozpedowski@Damians-MacBook-Air /etc % echo $HOME

/Users/damianrozpedowski

**11)**

damianrozpedowski@Damians-MacBook-Air /etc % pwd

/etc

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

**C**

**1)**

damianrozpedowski@Damians-MacBook-Air /etc % echo $SHELL

/bin/zsh

**2) Had to use cat on Mac**

damianrozpedowski@Damians-MacBook-Air ~ % cat /etc/shells

# List of acceptable shells for chpass(1).

# Ftpd will not allow users to connect who are not using

# one of these shells.

/bin/bash

/bin/csh

/bin/dash

**3)**

damianrozpedowski@Damians-MacBook-Air ~ % chsh -s /bin/tcsh damianrozpedowski

Changing shell for damianrozpedowski.

Password for damianrozpedowski:

**4) Reopened and is listed as tcsh**

**5)**

[Damians-Air:~] damianrozpedowski% ps

PID TTY TIME CMD

865 ttys000 0:00.02 -tcsh

I notice the information of a running process, PID is the process ID, TTY is the terminal associated with the process, TIME is the CPU time used by the process, CMD is the command that was used to start the process.

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

**D**

**1)**

[Damians-Air:~] damianrozpedowski% man ps

PS(1) General Commands Manual PS(1)

**NAME**

**ps** – process status

**SYNOPSIS**

**ps** [**-AaCcEefhjlMmrSTvwXx**] [**-O** fmt | **-o** fmt] [**-G** gid[,gid...]]

[**-g** grp[,grp...]] [**-u** uid[,uid...]] [**-p** pid[,pid...]] [**-t** tty[,tty...]]

**2)**

***List of possible states w/ significance:***

***R - a process is running/executing***

***S - sleep state - woken up when event its waiting for happens***

***T - Stopped, process is stopped***

***Z - Zombie, completed execution but waiting for parent to retrieve exit status***

Damians-MacBook-Air:~] damianrozpedowski% echo $SHELL

/bin/tcsh

**3)**

[Damians-Air:~] damianrozpedowski% ps -l

UID PID PPID F CPU PRI NI SZ RSS WCHAN S ADDR TTY TIME CMD

501 1077 1076 4006 0 31 0 408786864 3280 - S 0 ttys000 0:00.01 -tcsh

F - Flag, displays flags related to processes

S - State - #2 answers this

UID - User ID, shows user running process

PID - Process ID, each process has unique id

PPID - parent process id, shows relationship between child and parent process

C - CPU util, % of cpu utilization used by process

PRI- Priority, order of execution

NI - Nice value, higher value -> lower priority

ADDR- Mem address

SZ - size of process in memory

WCHAN - wait channel, why a process is in a state

TTY - terminal, which terminal process is running on

TIME - CPU time, amount of time cpu used for process

CMD - Command - name of command that started the process.

**4)** {Damians-Air:~] damianrozpedowski% top

Processes: 370 total, 2 running, 368 sleeping, 1738 threads 12:56:59

Load Avg: 1.80, 4.37, 3.44 CPU usage: 3.47% user, 3.59% sys, 92.93% idle

SharedLibs: 385M resident, 76M data, 21M linkedit.

MemRegions: 157455 total, 1420M resident, 145M private, 683M shared.

PhysMem: 7235M used (1002M wired), 397M unused.

VM: 152T vsize, 3778M framework vsize, 0(0) swapins, 0(0) swapouts.

**5)** **# Tasks: 370, # Running: 2, # Sleeping: 368, # Stopped & Zombies : 0**

**6)**

Damians-Air:~] damianrozpedowski% kill 1142

[Damians-Air:~] damianrozpedowski%

**7)**

Damians-Air:~] damianrozpedowski% kill -9 1142

[Process completed]

Kill -9 is a sure kill which forcefully terminates the process without allowing cleanup.

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

**E**

**1)** In windows you use “start” to create processes,

Syntax (from microsoft learn page): start <"title"> [/d <path>] [/i] [{/min | /max}] [{/separate | /shared}] [{/low | /normal | /high | /realtime | /abovenormal | /belownormal}] [/node <NUMA node>] [/affinity <hexaffinity>] [/wait] [/b] [/machine <x86|amd64|arm|arm64>] [<command> [<parameter>... ] | <program> [<parameter>... ]]

Some important parameter explanations, /d path specifies startup directory, /low | /normal |/high … starts an application in the specifies priority class, wait starts an application and waits for it to end, etc. All of these can be found on the site below. <https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/start>

**2)**

Damians-MacBook-Air:~/Desktop] damianrozpedowski% gcc parent.c -o parent

[Damians-MacBook-Air:~/Desktop] damianrozpedowski% gcc child.c -o child

[Damians-MacBook-Air:~/Desktop] damianrozpedowski% ./parent

Process[2880]: Parent in execution ...

Process[2881]: child in execution ...

Process[2881]: child terminating ...

Process[2880]: Parent detects terminating child

Process[2880]: Parent terminating ...

Damians-MacBook-Air:~/Desktop] damianrozpedowski% ./orphan

I'm the original process with PID 2888 and PPID 2840.

I'm the parent process with PID 2888 and PPID 2840.

my child's PID 2889

PID 2888 terminates.

[Damians-MacBook-Air:~/Desktop] damianrozpedowski% I'm the child process with PID 2889 and PPID 1.

PID 2889 terminates.

**Understanding the execution:**

Fork() creates new child process, if 0 its a child, else its a parent

Execve(“child,null,null) executes child program

wait() makes parent process wait until it child terminates

sleep() a delay/pause in seconds

Parent & Child

Compiles using gcc

When running parent ./parent created parent process and child process

Parent prints parent in execution followed by child in execution

Child terminates and prints child terminating

Parent is using wait() and detects termination and prints parent detects terminating child

Parent prints Parent terminating and terminates

Orphan

Compiles using gcc

Run ./orphan, creates original process and parent process

Original process forks a child process and prints my child’s PID #

After this it prints I’m the parent process…

Parent process is still running and the program exits (PID 2888 terminates)

Child continues running after parent has terminated

Child process prints “Im the child process with PID…”

PPID 1 shows that the parent is the initial process

Shows its an orphan process

Child process exits and termination message is printed.