BBM459
Environment Variable and Set-UID Program

DENİZ ECE AKTAŞ 21626901

ATAKAN AK 21626862

TASK 1: Manipulating Environment Variables

The default shell we used “cat /etc/passwd” command to check the Bash.

```
[atknak22@localhost ~] $ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
```

Then we used “printenv” command to print the environment variables on the screen. Variables are shown with “variable = value” format.

```
[atknak22@localhost ~] $ printenv
SHELL=/bin/bash
SESSION_MANAGER=local/unix:@/tmp/.ICE-unix/3645,unix/unix:/tmp/.ICE-unix/3645
COLORTERM=truecolor
HISTCONTROL=ignoredups
XDG_MENU_PREFIX=gnome-
HISTSIZE=1000
HOSTNAME=fedora
GNOME_SHELL_SESSION_MODE=classic
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=gnome-classic
SSH_AGENT_PID=3575
PWD=/home/atknak22
LOGNAME=atknak22
XDG_SESSION_DESKTOP=gnome-classic
XDG_SESSION_TYPE=x11
MODULESHOME=/usr/share/Modules
MANPATH=:
XAUTHORITY=/run/user/1000/gdm/Xauthority
GJS_DEBUG_TOPICS=JS ERROR;JS LOG
WINDOWPATH=2
GDM_LANG=tr_TR.UTF-8
HOME=/home/atknak22
USERNAME=atknak22
LANG=tr_TR.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=01;37;41:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=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:*.tb
z=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:*.e
sd=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:*.svgz=01;35:*.mng=01;35:*.pcx=0
1;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:*.ogv=01;35:*.ogx=01;35:*.aac=01;36:*.au=01;36:*.flac=01;36:*.m4a=01;36:*.mid=01;36:*.midi=01;36:*.mka=01;36:*.mp3=01;36:*
.mpc=01;36:*.ogg=01;36:*.ra=01;36:*.wav=01;36:*.oga=01;36:*.opus=01;36:*.spx=01;36:*.xspf=01;36:
XDG_CURRENT_DESKTOP=GNOME-Classic:GNOME
VTE_VERSION=6203
GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/658ce2f2_80e6_4f78_a87e_49701121f9a3
GJS_DEBUG_OUTPUT=stderr
MOZ_GMP_PATH=/usr/lib64/mozilla/plugins/gmp-gmpopenh264/system-installed
MODULEPATH_modshare=/usr/share/modulefiles:1:/usr/share/Modules/modulefiles:1:/etc/modulefiles:1
XDG_SESSION_CLASS=user
TERM=xterm-256color
LESSOPEN=||/usr/bin/lesspipe.sh %s
USER=atknak22
GNOME_TERMINAL_SERVICE=:1.132
MODULES_RUN_QUARANTINE=LD_LIBRARY_PATH LD_PRELOAD
LOADEDMODULES=
DISPLAY=:0
SHLVL=1
QT_IM_MODULE=ibus
XDG_RUNTIME_DIR=/run/user/1000
LS_COLORS=auto
XDG_DATA_DIRS=/home/atknak22/.local/share/flatpak/exports/share:/var/lib/flatpak/exports/share:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
PATH=/home/atknak22/.local/bin:/home/atknak22/bin:/home/atknak22/.local/bin:/home/atknak22/bin:/usr/share/Modules/bin:/usr/lib64/ccache:/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/var/lib/snapd/snap/bin
MODULEPATH=/etc/scl/modulefiles:/etc/scl/modulefiles:/usr/share/Modules/modulefiles:/etc/modulefiles:/usr/share/modulefiles
GDMSESSION=gnome-classic
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
MAIL=/var/spool/mail/atknak22
MODULES_CMD=/usr/share/Modules/libexec/modulecmd.tcl
BASH_FUNC_ml%%=() {  module ml "$@"
}
BASH_FUNC_switchml%%=() {  typeset swfound=1;
 if [ "${MODULES_USE_COMPAT_VERSION:-0}" = '1' ]; then
 typeset swname='main';
 if [ -e /usr/share/Modules/libexec/modulecmd.tcl ]; then
 typeset swfound=0;
 unset MODULES_USE_COMPAT_VERSION;
 fi;
 else
 typeset swname='compatibility';
 if [ -e /usr/share/Modules/libexec/modulecmd-compat ]; then
 typeset swfound=0;
 MODULES_USE_COMPAT_VERSION=1;
 export MODULES_USE_COMPAT_VERSION;
 fi;
 fi;
 if [ $swfound -eq 0 ]; then
 echo "Switching to Modules $swname version";
 source /usr/share/Modules/init/bash;
 else
 echo "Cannot switch to Modules $swname version, command not found";
 return 1;
 fi
}
BASH_FUNC_module%%=() {  _module_raw "$@" 2>&1
}
BASH_FUNC_scl%%=() {  if [ "$1" = "load" -o "$1" = "unload" ]; then
 eval "module $@";
 else
 /usr/bin/scl "$@";
 fi
}
BASH_FUNC__module_raw%%=() {  unset _mlshdbg;
 if [ "${MODULES_SILENT_SHELL_DEBUG:-0}" = '1' ]; then
 case "$-" in
 *v**x*)
 set +vx;
 _mlshdbg='vx'
 ;;
 *v*)
 set +v;
 _mlshdbg='v'
```

After these command, we looked up “PWD” which is “print working directory”, to archive this we put “printenv PWD” and “env | grev PWD” commands into the terminal. They gave the same result.
Afterwards, we used export to set and unset command to unset an environment value because these commands are internal Bash commands, when we tried to do these commands outside of seed the program put us into the seed to execute the command. The results of the terminal are shown below:

```
[atknak22@localhost ~] $ printenv PWD
/home/atknak22
[atknak22@localhost ~] $ env | grep PWD
PWD=/home/atknak22
[atknak22@localhost ~] $ unset PWD
PWD environment variable not set[atknak22@localhost ~] $ env | grep PWD
PWD environment variable not set[atknak22@localhost ~] $ export PWD=/home/seed
[atknak22@localhost seed] $ env | grep PWD
PWD=/home/seed
[atknak22@localhost seed] $ printenv
```

Task 2: Inheriting environment variables from variables:

The second task's goal was to find the differences between a child directory command and a parent directory command and we did this by putting the code in fork.c code file and executing it and then for the child directory we did the same thing but after putting the printenv() command in parent directory into comment and uncommented printenv() command in child directory.

fork.c

```c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
extern char **environ;
void printenv()
{
int i = 0;
while (environ[i] != NULL) {
printf("%s\n", environ[i]);
i++;
}
}
void main()

pid_t childPid;
switch(childPid = fork()) {
case 0: /* child process */
printenv();
exit(0);
default: /* parent process */
//printenv();
exit(0);
}

```

```
[atknak22@localhost ~] $ gcc fork.c -o out
[atknak22@localhost ~] $ ./out > a.out
[atknak22@localhost ~] $ gcc fork.c -o out
[atknak22@localhost ~] $ ./out > b.out
```

When we run the fork.c file which had our c code in it. The output file a.out was as shown below:

```
SHELL /bin/bash
SESSION_MANAGER=local/unix:@/tmp/.ICE-unix/3645,unix/unix:/tmp/.ICE-unix/3645
COLORTERM=truecolor
HISTCONTROL=ignoredups
XDG_MENU_PREFIX=gnome-
HISTSIZE=1000
HOSTNAME=fedora
GNOME_SHELL_SESSION_MODE=classic
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=gnome-classic
SSH_AGENT_PID=3575
PWD=/home/atknak22
LOGNAME=atknak22
XDG_SESSION_DESKTOP=gnome-classic
```

When we run our code again but with printenv() command in parent process not as a comment and the printenv() in child process was put into a comment. b.out which is the output file is as shown below:

```
SHELL / /
SESSION_MANAGER=local/unix:@/tmp/.ICE-unix/3645,unix/unix:/tmp/.ICE-unix/3645
COLORTERM=truecolor
HISTCONTROL=ignoredups
XDG_MENU_PREFIX=gnome-
HISTSIZE=1000
HOSTNAME=fedora
GNOME_SHELL_SESSION_MODE=classic
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=gnome-classic
SSH_AGENT_PID=3575
PWD=/home/atknak22
```

Afterwards we found the differences between two output files, a.out and b.out, with diff command .

```
[atknak22@localhost ~] $ gcc fork.c -o out
[atknak22@localhost ~] $ ./out > a.out
[atknak22@localhost ~] $ gcc fork.c -o out
[atknak22@localhost ~] $ ./out > b.out
[atknak22@localhost ~] $ diff a.out b.out
```

Our terminal didn't show anything in the terminal because the two files are identical. This shows that there is no difference between parent and child processes and they are the same.

Task 3: Environment Variables And Execve() : (!!!!!!!!!!!!!!!!!!!!!!!)

Execve () is a function that takes environment variables but before we edited out the NULL value which was the third variable of the execve() function, the output was an empty file but after we edited the NULL to be environ, the output file showed environment variables because the third variable in execve() function specifies the running process so environment variables are not automatically inherited.

execve.c

```c
#include <stdio.h>
#include <stdlib.h>
extern char **environ;
int main()
{
        char *argv[2];
        argv[0] = "/usr/bin/env";
        argv[1] = NULL;
        execve("/usr/bin/env", argv, NULL);
        return 0 ;
}
```

task3.c

```c
#include <stdio.h>
#include <stdlib.h>
extern char** environ;
int main()
{
        char *argv[2];
        argv[0] = "/usr/bin/env";
        argv[1] = NULL;
        execve("/usr/bin/env", argv, environ);
        return 0 ;
}
```

```
[atknak22@localhost ~] $ gcc task3.c -o task3out
task3.c: 'main' işlevinde:
task3.c:9:2: UYARI: 'execve' işlevinin örtük bildirimi [-Wimplicit-function-declaration]
    9 |  execve("/usr/bin/env", argv, NULL);
      |  ^~~~~~
[atknak22@localhost ~] $ gcc -o environ task3.c
task3.c: 'main' işlevinde:
task3.c:9:2: UYARI: 'execve' işlevinin örtük bildirimi [-Wimplicit-function-declaration]
    9 |  execve("/usr/bin/env", argv, environ);
      |  ^~~~~~
[atknak22@localhost ~] $ ./environ > environ.txt
[atknak22@localhost ~] $
```

task3.c

```
SHELL=/bin/bash
SESSION_MANAGER=local/unix:@/tmp/.ICE-unix/22677,unix/unix:/tmp/.ICE-unix/22677
COLORTERM=truecolor
HISTCONTROL=ignoredups
XDG_MENU_PREFIX=gnome-
HISTSIZE=1000
HOSTNAME=fedora
GNOME_SHELL_SESSION_MODE=classic
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=gnome-classic
SSH_AGENT_PID=22121
PWD=/home/atknak22
LOGNAME=atknak22
XDG_SESSION_DESKTOP=gnome-classic
XDG_SESSION_TYPE=x11
MODULESHOME=/usr/share/Modules
MANPATH=:
XAUTHORITY=/run/user/1000/gdm/Xauthority
GJS_DEBUG_TOPICS=JS ERROR;JS LOG
```

Task 4: Environment Variables And System() :

The system() function takes variables directly to bin/sh and, this is why even if we didn't specify or give environ external variable as an argument, unlike exec() function, the program still executes and gives output.

Aç

task4.c

```
#include <stdio.h>
#include <stdlib.h>
int main()

system("/usr/bin/env");
return 0 ;

```

```
[atknak22@localhost seed] $ gcc task4.c -o task4out
[atknak22@localhost seed] $ ./task4out > task4.txt
[atknak22@localhost seed] $
```

The output is shown below:

```
task3.c                                                    task4.c
SHELL=/bin/bash
SESSION_MANAGER=local/unix:@/tmp/.ICE-unix/3645,unix/unix:/tmp/.ICE-unix/3645
COLORTERM=truecolor
HISTCONTROL=ignoredups
XDG_MENU_PREFIX=gnome-
HOSTNAME=fedora
HISTSIZE=1000
GNOME_SHELL_SESSION_MODE=classic
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=gnome-classic
SSH_AGENT_PID=3575
PWD=/home/atknak22
XDG_SESSION_DESKTOP=gnome-classic
LOGNAME=atknak22
XDG_SESSION_TYPE=x11
MODULESHOME=/usr/share/Modules
MANPATH=:
_=/usr/bin/env
XAUTHORITY=/run/user/1000/gdm/Xauthority
GJS_DEBUG_TOPICS=JS ERROR;JS LOG
WINDOWPATH=2
GDM_LANG=tr_TR.UTF-8
HOME=/home/atknak22
USERNAME=atknak22
LANG=tr_TR.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=01;37;41:su=37;41:sg=30;
```

Task 5: Environment variable and SET-UID Programs

First of all we wrote the specified C code, compiled and run it. In the first output we run the code with the ownership on the user account and the environment variables are from there.

```
task5.c
#include <stdio.h>
#include <stdlib.h>
extern char **environ;
void main()
{
int i = 0;
while (environ[i] != NULL) {
printf("%s\n", environ[i]);
i++;
}
}
```

```
[atknak22@localhost ~] $ gcc task5.c -o output5
[atknak22@localhost ~] $ ./output5 > output5.txt
[atknak22@localhost ~] $ sudo chown root output5
[sudo] password for atknak22:
[atknak22@localhost ~] $ sudo chmod 4755 output5
[atknak22@localhost ~] $ export PATH
[atknak22@localhost ~] $ printenv PATH
/home/atknak22/.local/bin:/home/atknak22/bin:/home/atknak22/.local/bin:/home/atknak22/bin:/usr/share/Modules/bin:/usr/lib64/cca
[atknak22@localhost ~] $ export PATH=/home/atknak22/output5:$PATH
[atknak22@localhost ~] $ env | grep PATH
MANPATH=:
WINDOWPATH=2
MOZ_GMP_PATH=/usr/lib64/mozilla/plugins/gmp-gmpopenh264/system-installed
MODULEPATH_modshare=/usr/share/modulefiles:1:/usr/share/Modules/modulefiles:1:/etc/modulefiles:1
MODULES_RUN_QUARANTINE=LD_LIBRARY_PATH LD_PRELOAD
PATH=/home/atknak22/output5:/home/atknak22/.local/bin:/home/atknak22/bin:/home/atknak22/.local/bin:/home/atknak22/bin:/usr/shar
MODULEPATH=/etc/scl/modulefiles:/etc/scl/modulefiles:/usr/share/Modules/modulefiles:/etc/modulefiles:/usr/share/modulefiles
[atknak22@localhost ~] $
```

```
task5.c
SHELL=/bin/bash
SESSION_MANAGER=local/unix:@/tmp/.ICE-unix/22677,unix/unix:/tmp/.ICE-unix/22677
COLORTERM=truecolor
HISTCONTROL=ignoredups
XDG_MENU_PREFIX=gnome-
HISTSIZE=1000
HOSTNAME=fedora
GNOME_SHELL_SESSION_MODE=classic
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=gnome-classic
SSH_AGENT_PID=22121
PWD=/home/atknak22
LOGNAME=atknak22
XDG_SESSION_DESKTOP=gnome-classic
XDG_SESSION_TYPE=x11
MODULESHOME=/usr/share/Modules
MANPATH=:
XAUTHORITY=/run/user/1000/gdm/Xauthority
GJS_DEBUG_TOPICS=JS ERROR;JS LOG
WINDOWPATH=2
GDM_LANG=tr_TR.UTF-8
HOME=/home/atknak22
```

And then we used export commands to export PATH, LD_LIBRARY_PATH and HOSTNAME,as our ANY_NAME variable) variables.

```
[atknak22@localhost ~] $ export LD_LIBRARY_PATH=/home/atknak22/output5:$LD_LIBRARY_PATH
[atknak22@localhost ~] $ env | grep LD_LIBRARY_PATH
MODULES_RUN_QUARANTINE=LD_LIBRARY_PATH LD_PRELOAD
LD_LIBRARY_PATH=/home/atknak22/output5:
[atknak22@localhost ~] $
```

```
[atknak22@localhost ~] $ export HOSTNAME=/home/atknak22/output5:$HOSTNAME
[atknak22@localhost ~] $ env | grep HOSTNAME
HOSTNAME=/home/atknak22/output5:fedora
[atknak22@localhost ~] $
```

After doing SET_UID operations we run the code again and when we do this the program forks and opens a child processor. When we run the code on this child processor, we get the output5-2.txt file as our output. And then to see the differences between child processor and the previous one, we used diff and saw the differences, as shown below. Because the child process to access the environment variables reaches home/usr/bin, it adds this to the start of the environment variables.

```
[atknak22@localhost ~] $ gcc task5.c -o output5
[atknak22@localhost ~] $ ./output5 > output5.txt
[atknak22@localhost ~] $ sudo chown root output5
[sudo] password for atknak22:
[atknak22@localhost ~] $ sudo chmod 4755 output5
[atknak22@localhost ~] $ export PATH
[atknak22@localhost ~] $ printenv PATH
/home/atknak22/.local/bin:/home/atknak22/bin:/home/atknak22/.local/bin:/home/atknak22/bin:/usr/share/Modules/bin:/usr/lib64/ccache:/usr/local/bin:/usr/local/sbin
[atknak22@localhost ~] $ export PATH=/home/atknak22/output5:$PATH
[atknak22@localhost ~] $ env | grep PATH
MANPATH=:
WINDOWPATH=2
MOZ_GMP_PATH=/usr/lib64/mozilla/plugins/gmp-gmpopenh264/system-installed
MODULEPATH_modshare=/usr/share/modulefiles:1:/usr/share/Modules/modulefiles:1:/etc/modulefiles:1
MODULES_RUN_QUARANTINE=LD_LIBRARY_PATH LD_PRELOAD
PATH=/home/atknak22/output5:/home/atknak22/.local/bin:/home/atknak22/bin:/home/atknak22/.local/bin:/home/atknak22/bin:/usr/share/Modules/bin:/usr/lib64/ccache:/u
MODULEPATH=/etc/scl/modulefiles:/etc/scl/modulefiles:/usr/share/Modules/modulefiles:/etc/modulefiles:/usr/share/modulefiles
[atknak22@localhost ~] $ export LD_LIBRARY_PATH=/home/atknak22/output5:$LD_LIBRARY_PATH
[atknak22@localhost ~] $ env | grep LD_LIBRARY_PATH
MODULES_RUN_QUARANTINE=LD_LIBRARY_PATH LD_PRELOAD
LD_LIBRARY_PATH=/home/atknak22/output5:
[atknak22@localhost ~] $ export HOSTNAME=/home/atknak22/output5:$HOSTNAME
[atknak22@localhost ~] $ env | grep HOSTNAME
HOSTNAME=/home/atknak22/output5:fedora
[atknak22@localhost ~] $ sudo chown root output5
[sudo] password for atknak22:
[atknak22@localhost ~] $ sudo chmod 4755 output5
[atknak22@localhost ~] $ ./output5 > output5-2.txt
[atknak22@localhost ~] $ diff output5.txt output5-2.txt
7c7
< HOSTNAME=fedora
---
> HOSTNAME=/home/atknak22/output5:fedora
46c46
< PATH=/home/atknak22/.local/bin:/home/atknak22/bin:/home/atknak22/.local/bin:/home/atknak22/bin:/usr/share/Modules/bin:/usr/lib64/ccache:/usr/local/bin:/usr/loc
---
> PATH=/home/atknak22/output5:/home/atknak22/.local/bin:/home/atknak22/bin:/home/atknak22/.local/bin:/home/atknak22/bin:/usr/share/Modules/bin:/usr/lib64/ccache:
[atknak22@localhost ~] $
```

Task 6: The LD_PRELOAD Environment Variable And Set_UID Programs

We wrote the necessary codes and compiled and run them.

mylib.c

```c
#include <stdio.h>
void sleep (int s)
{
/* If this is invoked by a privileged program,
you can do damages here! */
printf("I am not sleeping!\n");
}
```

myprog.c

```c
/* myprog.c */
int main()

sleep(1);
return 0;
```

When we compile the code without any modification, the output is as printed as such;

output6-1.txt

```
I am not sleeping!
```

atknak22@fedora:~

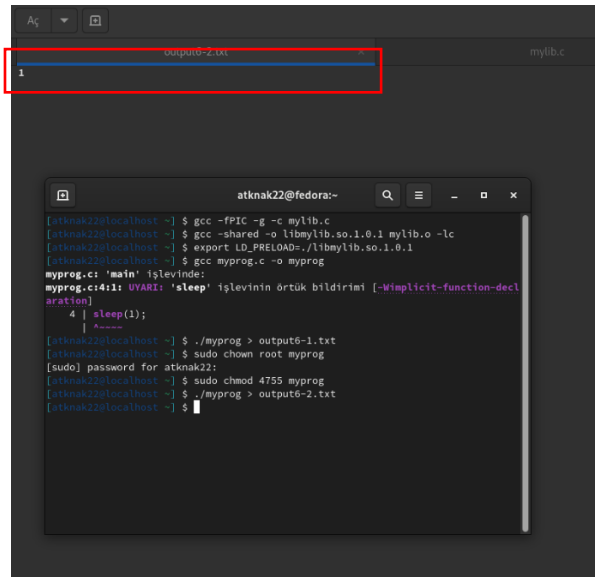
```
[atknak22@localhost ~] $ gcc -fPIC -g -c mylib.c
[atknak22@localhost ~] $ gcc -shared -o libmylib.so.1.0.1 mylib.o -lc
[atknak22@localhost ~] $ export LD_PRELOAD=./libmylib.so.1.0.1
[atknak22@localhost ~] $ gcc myprog.c -o myprog
myprog.c: 'main' işlevinde:
myprog.c:4:1: UYARI: 'sleep' işlevinin örtük bildirimi [-Wimplicit-function-decl
aration]
    4 | sleep(1);
      | ^~~~~
[atknak22@localhost ~] $ ./myprog > output6-1.txt
[atknak22@localhost ~] $
```

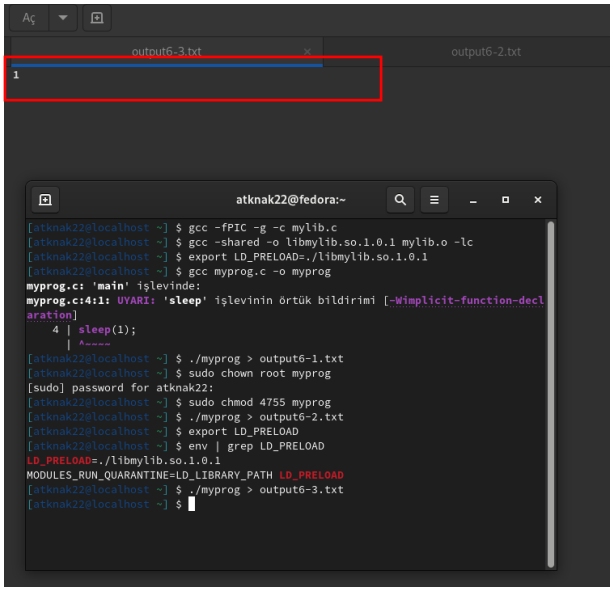
This shows that when the child process opened up, it did not inherit LD_PRELOAD environment variable so the sleep function is used.

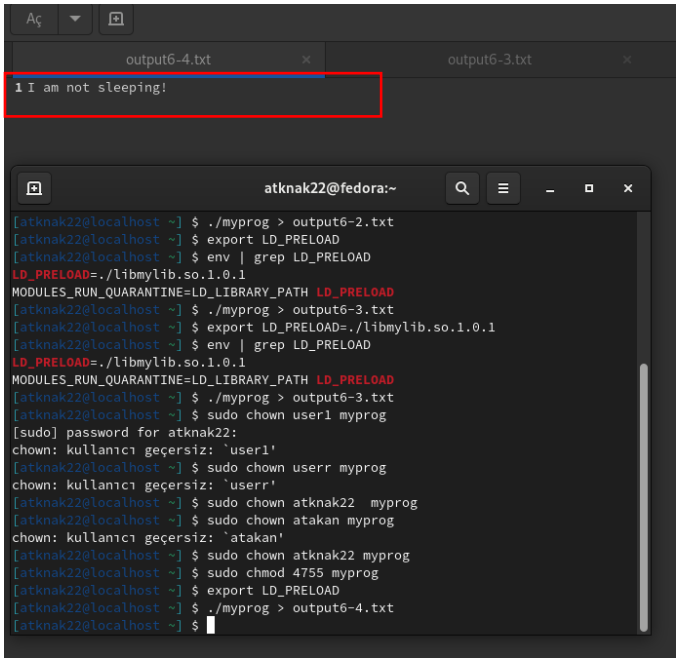
When we Set-UID root program and run it the output is different and when we export the LD_PRELOAD the output is same as Set-UID operation but when we run the code as a different user with Set-UID operations the output is same as running the code normally, without any modifications.

When we Set-UID root program because we are in the root account the program can see the LD_PRELOAD variable even though child process opens up so sleep function is not called.

When we export the variable we are still in the root account the program can see the LD_PRELOAD variable even though child process opens up so sleep function is not called.







When we change the user the child process opened up, it did not inherit LD_PRELOAD environment variable so the sleep function is used because the new user's code could not reach root to see the environment variable.

Task 7: Capability Leaking

In this task we compiled and run the code and then processed Set-UID operations and compiled and run the code again.

task7.c

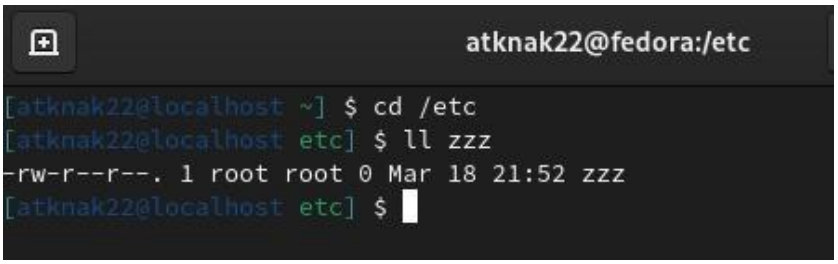
```c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
void main()
 int fd;
/* Assume that /etc/zzz is an important system file,
* and it is owned by root with permission 0644.
* Before running this program, you should creat
* the file /etc/zzz first. */
fd = open("/etc/zzz", O_RDWR | O_APPEND);
if (fd == -1) {
printf("Cannot open /etc/zzz\n");
exit(0);
}
/* Simulate the tasks conducted by the program */
sleep(1);
/* After the task, the root privileges are no longer needed,
it's time to relinquish the root privileges permanently. */
setuid(getuid()); /* getuid() returns the real uid */
if (fork()) { /* In the parent process */
close (fd);
exit(0);
} else { /* in the child process */
/* Now, assume that the child process is compromised, malicious
attackers have injected the following statements
into this process */
write (fd, "Malicious Data\n", 15);
close (fd);
}
```

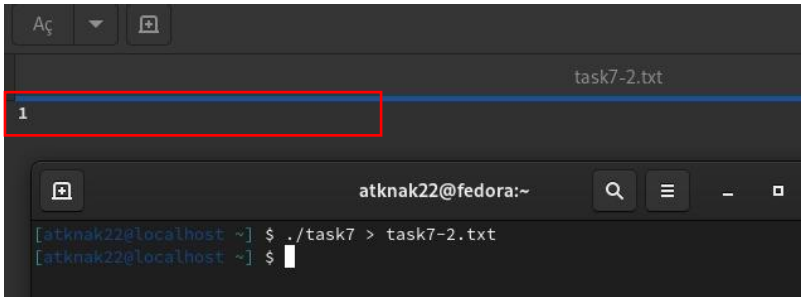
output7.txt

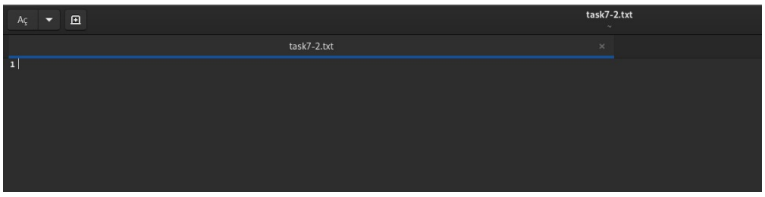
```
Cannot open /etc/zzz
```

atknak22@fedora:~

```
[atknak22@localhost ~] $ ./myprog > output6-4.txt
[atknak22@localhost ~] $ gcc task7.c -o task7
task7.c: 'main' işlevinde:
task7.c:16:1: UYARI: 'sleep' işlevinin örtük bildirimi [-Wimplicit-function-declaration]
   16 | sleep(1);
      | ^~~~~
task7.c:19:1: UYARI: 'setuid' işlevinin örtük bildirimi [-Wimplicit-function-declaration]
   19 | setuid(getuid()); /* getuid() returns the real uid */
      | ^~~~~~
task7.c:19:8: UYARI: 'getuid' işlevinin örtük bildirimi [-Wimplicit-function-declaration]
   19 | setuid(getuid()); /* getuid() returns the real uid */
      |        ^~~~~~
task7.c:20:5: UYARI: 'fork' işlevinin örtük bildirimi [-Wimplicit-function-declaration]
   20 | if (fork()) { /* In the parent process */
      |     ^~~~
task7.c:21:1: UYARI: implicit declaration of function 'close'; did you mean 'pclose'? [-Wimplicit-function-dec
laration]
   21 | close (fd);
      | ^~~~~
      | pclose
task7.c:27:1: UYARI: implicit declaration of function 'write'; did you mean 'fwrite'? [-Wimplicit-function-dec
laration]
   27 | write (fd, "Malicious Data\n", 15);
      | ^~~~~
      | fwrite
[atknak22@localhost ~] $ sudo chown root task7
[sudo] password for atknak22:
[atknak22@localhost ~] $ sudo chmod 4755 task7
[atknak22@localhost ~] $ ./task7 > output7.txt
[atknak22@localhost ~] $
```







As the program looks into zzz file the output prints but because fork is called the child processor can change the contents of the output file, to be sure that no modification happened before calling fork the previous program should be closed.