BAHIRDAR UNIVERSITY

Department of Software Engineering

Operating System and System Programming

Individual Assignment

“***ZORIN** OS **System call Implementation*** ”

Personal Information

- **First Name**: Yonas
- **Second Name**: Sera
- **Last Name**: Chalachew
- **ID No**: BDU1602854
- **Section**: B

Submitted To: Wendimu B.

Submission Date: 16/08/2017 E.C

System Call Implementation in Linux (C++ Version)

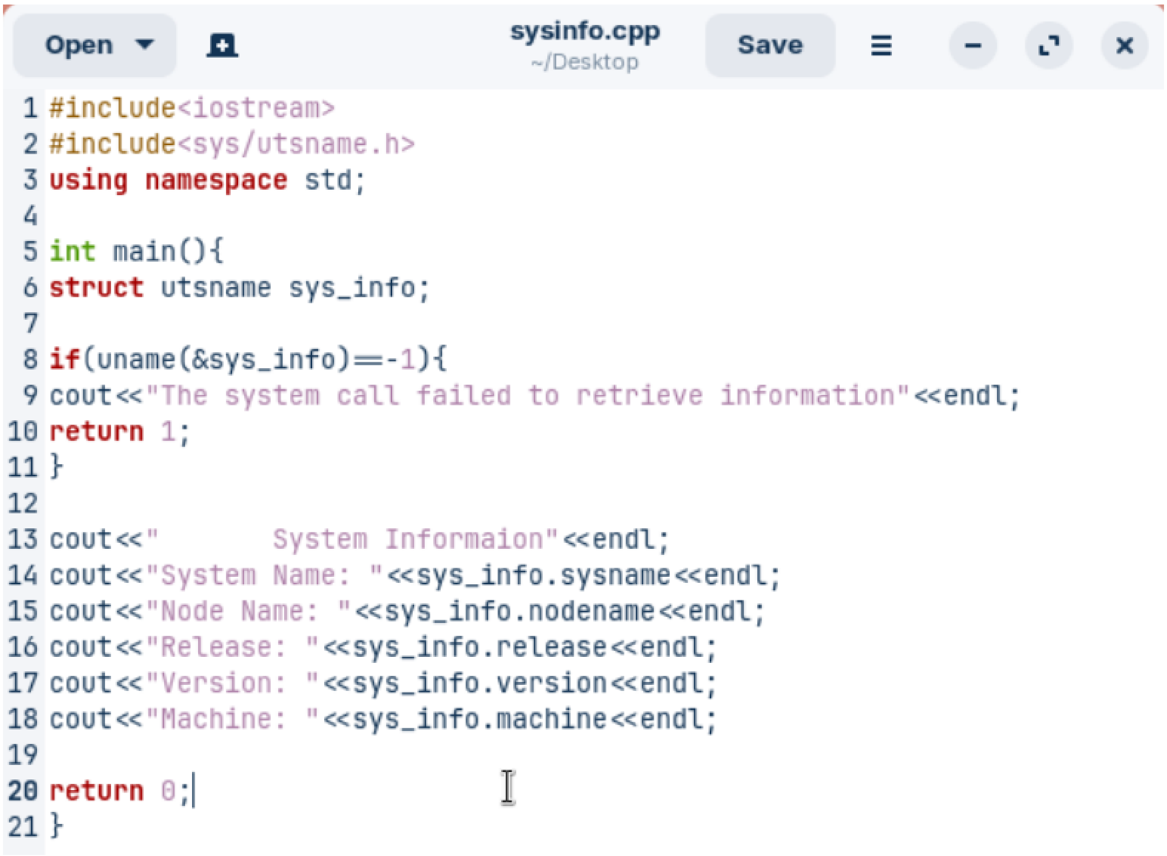
System calls are like a way for a program to ask the operating system for help. The operating system is what controls the computer's hardware, like the keyboard, mouse, memory, or hard drive. So, if a program wants to do something with these things—like read a file, open a folder, or save something—it can't do it all by itself. Instead, it uses a system call to ask the operating system to do it.

You can think of a system call as a bridge between software (the program) and hardware (the physical parts of the computer). It's like saying, "Hey OS, can you help me do this?" The operating system listens and takes care of the request, making sure everything is done safely and in the right way.

One example of a system call is called `uname()`. This one is used when a program wants to know more about the system it's running on—like what kind of operating system it is, what version, what machine type, and other details. It's useful for checking system info before doing some tasks.

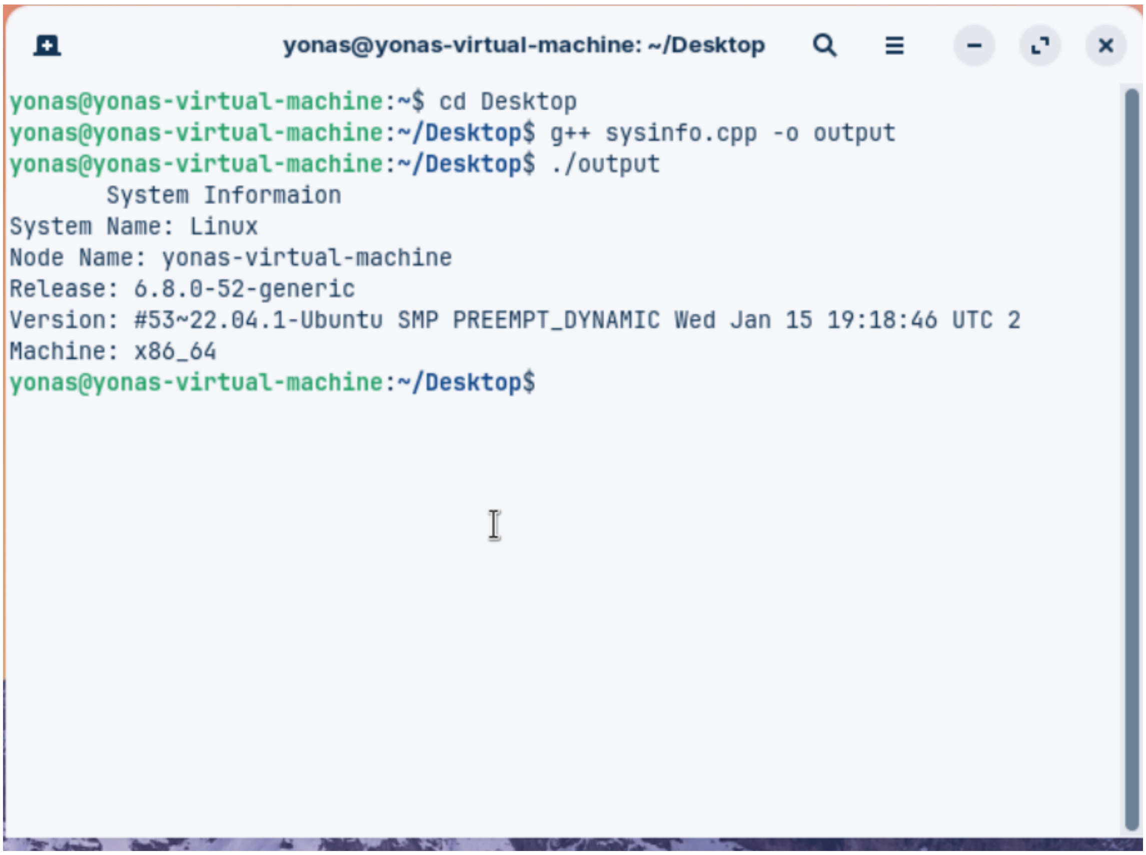
Now, let's see how we can use this `uname()` system call in a C++ program to get and show some basic information about the system.

C++ Code



```cpp
#include<iostream>
#include<sys/utsname.h>
using namespace std;

int main(){
struct utsname sys_info;

if(uname(&sys_info)==-1){
cout<<"The system call failed to retrieve information"<<endl;
return 1;
}

cout<<"        System Informaion"<<endl;
cout<<"System Name: "<<sys_info.sysname<<endl;
cout<<"Node Name: "<<sys_info.nodename<<endl;
cout<<"Release: "<<sys_info.release<<endl;
cout<<"Version: "<<sys_info.version<<endl;
cout<<"Machine: "<<sys_info.machine<<endl;

return 0;
}
```

Output



```
yonas@yonas-virtual-machine:~$ cd Desktop
yonas@yonas-virtual-machine:~/Desktop$ g++ sysinfo.cpp -o output
yonas@yonas-virtual-machine:~/Desktop$ ./output
       System Informaion
System Name: Linux
Node Name: yonas-virtual-machine
Release: 6.8.0-52-generic
Version: #53~22.04.1-Ubuntu SMP PREEMPT_DYNAMIC Wed Jan 15 19:18:46 UTC 2
Machine: x86_64
yonas@yonas-virtual-machine:~/Desktop$
```

- In the above code I have implemented a system call uname(), which retrieves system information. To do that I used <sys/utsname.h> header to include uname() function into the code.