

## ICS 332 Fall 2021

[Home](#) / [Modules](#) / [OS Overview and Interfaces](#) / Homework Assignment #2

# Homework #2 – Using strace to spy on system calls [30 pts]

You are expected to do your own work on all homework assignments. (See the statement of Academic Dishonesty on the [Syllabus](#).)

Check the [Syllabus](#) for the late assignment policy for this course.

## How to turn in?

Assignments need to be turned in via [Laulima](#). Check the [Syllabus](#) for the late assignment policy for the course.

## What to turn in?

You should turn in single **plain text** file named README.txt with your answers to the assignment's questions. Your file must be readable "as is" and points will be removed if the report is not readable.

## Environment

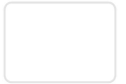
For this assignment **you need a Linux environment** (see [Assignment #0](#)).

## Hints

- You can look at our small [Linux refresher](#) in case you don't feel comfortable on how to filter (`grep`) and count (`wc`) lines in a Linux environment. Part of the goal of this assignment is for those of you who haven't used the command-line much to start doing so.
- Reading the `strace` man page is actually a good idea as well as the output can be formatted in different ways based on command-line arguments.
- There are also man pages for the system calls. For example `man 2 open` will show the manual page for the `open` system call.

## Exercise #1: Counting system calls in "Hello World" programs [10 pts]

In this question, you use `strace` to investigate the system calls placed when running HelloWorld programs in C and Java, i.e., programs that do nothing at all. Download the following two programs to your Linux box:



## ICS 332 Fall 2021

(To install java on Ubuntu: `sudo apt-get install openjdk-8-jdk`)

Use the `strace` tool we have mentioned in class, with the following command-line options for each program:

```
strace -xf -otracefile_java java HelloWorld
```

and

```
strace -xf -otracefile_c ./HelloWorld
```

The list of system calls (with argument values and return values) will be output into the trace files once each command completes. You can read the manual page for `strace` (type `man strace`) to find out what other command-line arguments are available.

**WARNING:** Some lines in the `strace` output may appear with the word “resumed” in them, or with a message like “+++ exited with 0 +++”; **IGNORE** these lines.

### Question #1 [3 pts]:

In your report give how many system calls are placed in total by each HelloWorld program, and briefly explain how you counted them (e.g., if using the command-line, just give the command(s) you used). Why do you think these numbers are so different?

### Question #2 [3 pts]:

How many *different* system calls (i.e., counting only one occurrence of a particular system call) are placed by each HelloWorld program? Briefly describe how you obtained this information.

### Question #3 [4 pts]:

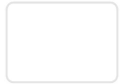
Based on your intuition and on reading man pages (make sure you read the correct man page, e.g. `man 2 write` and not `man write`), and on the `strace` output, for each of the broad “OS management” areas below give in your report the name of **2** system calls (8 total) placed by the Java HelloWorld program that fall in this area:

- Process Management
- Memory Management
- File System and I/O
- Protection and Security

### Exercise #2 [20 pts]: Is `wget` getting what it’s asking for?

The `/usr/bin/wget` system program is used to download files from a Web site given a URL, all on the command-line (To install `wget` on Ubuntu: `sudo apt-get install wget`). For

# ICS 332 Fall 2021



wget https://www.chadmorita.com/ics332\_fall2021/morea/GettingStarted/LaTeX/linuxshell\_slid

Like any program that does I/O, `wget` does *buffering*: repeatedly reading chunks of bytes from the “source” into a *buffer* (i.e., an array of bytes in RAM) and writing these bytes to some “destination”. In the case of `wget`, the source is the network (i.e., the Web server) and the destination is the local disk (i.e., the file). As you might expect, reading is done with the `read` system call, and writing is done with the `write` system call.

Conveniently, Peter Norvig hosts a large (6.2 MiB) text file at: <http://norvig.com/big.txt>. Use the provided output from `strace` and `wget` to answer the following questions. Explain the methods used to answer the question (commands/arguments) and included any interesting observations. You must use command line tools to answer the questions. The `norvig_outfile` file was downloaded from the Norvig server and the `uh_outfile` file was downloaded from a server at UH.

- [`./norvig\_outfile`](#)
- [`./uh\_outfile`](#)

**WARNING:** Some lines in the `strace` output may appear with the word “resumed” in them, or with a message like “+++ exited with 0 +++”; **IGNORE** these lines.

Answer the following questions using `norvig_outfile`:

- [q1] How many system calls are placed in total?
- [q2] How many system calls are placed by `wget` before it actually starts getting the file content?
- [q3] How many bytes does `wget` attempt to read from the Web server at a time typically (e.g., the buffer size)? Justify your answer by giving in your report one line of the `strace` output as an example.
- [q4] How many times does `wget` attempt to read pieces of file content, in total?
- [q5] Out of these, how many times does `wget` NOT receive the number of bytes it wants?
- [q6] Do you conclude that `wget` typically fills its buffer or not?

Answer the following questions using `uh_outfile`:

- [q7] How many times does `wget` attempt to read pieces of file content, in total?
- [q8] Out of these, how many times does `wget` NOT receive the number of bytes it wants?
- [q9] Do you conclude that `wget` typically fills its buffer or not?
- [q10] Why do you think results are different than in the previous experiment

# ICS 332 Fall 2021



4 modules | 10 outcomes | 15 experiences