**PATH**

*This is one of the most important lessons in this entire course!* **Listen up**.

There are environment variables that are sort of "built-in" to your shell. By "built-in" I just mean that different programs and parts of your system know about them and use them. The PATH variable is one of those.

**Why Do We Care About the PATH?**

If it weren't for the PATH, you'd have to remember the filesystem path of every executable you wanted to run in your shell. Instead of just running ls, you'd have to run /bin/ls (or whatever the location of the ls executable is on your system). That's not very convenient.

The PATH variable is a list of directories that your shell will look into when you try to run a command. If you type ls, your shell will look in each directory listed in your PATH variable for an executable called ls. If it finds one, it will just run it. If it doesn't, it will give you an error like: "command not found".

**What's in the PATH Variable?**

Take a look at your current PATH variable:

echo $PATH

You should see a giant list of directories separated by colons (:). Each of those directories is a place where your shell will look for executables. For example, with a PATH like this:

/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin

Your shell will look for executables in the following directories:

* /usr/local/bin
* /usr/bin
* /bin
* /usr/sbin
* /sbin

**Assignment**

As something of a security engineer yourself, you want to temporarily disable the PATH variable so that you can only run executables by using their full path. You know, just so you don't accidentally run something you don't mean to.

1. Reset your PATH variable to an empty string:

export PATH=""

*This will only affect your current shell session. If you open a new shell, it will have the default PATH variable again.*

1. Try running some simple commands like ls, pwd, echo, etc – some will no longer work, while others that do not rely on PATH will.

“The command is not found”.