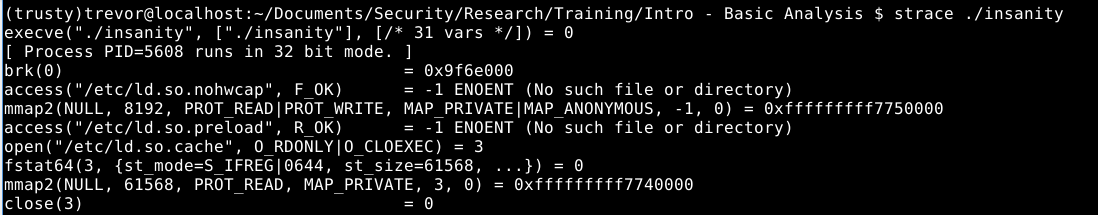
**Intro To Reverse Engineering - WhatAboutStrace Solution**

**Introduction:** WhatAboutStrace is an easy challenge designed to teach students additional steps to perform during a basic analysis against a target binary to learn more about the binary. Using the commands ‘strace’, and ‘ltrace’. These commands should be some of your first go-tos when trying to learn more about a target executable. This binary was originally written by the 9447 CTF team for their 2014 ctf competition.

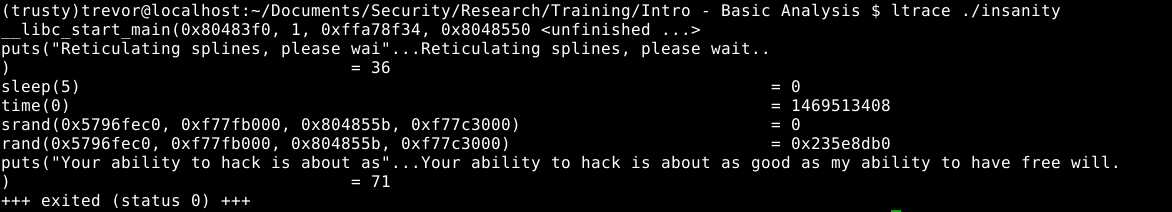
**Challenge:** Learn how to use the command ’strace’, and ‘ltrace’, and retrieve the flag from the binary. The flag is how many seconds the sleep() function is called for in the form: TXSTCS{seconds}

**Solving:** The last two commands we’ll use during our basic analysis are ‘[strace](http://man7.org/linux/man-pages/man1/strace.1.html)’, and ‘[ltrace](http://man7.org/linux/man-pages/man1/ltrace.1.html)’. These two commands are helpful in determining a program's general behavior as it executes. Strace is a program that will print out every system call or signal a program makes or receives during its execution. Strace’s basic syntax is: “strace ./binaryToExecute”. Go ahead and run Strace against the insanity binary.



Each line output by Strace represents either a system call or signal received by the ‘insanity’ program during execution. For instance we can see calls to ‘[execve()](http://man7.org/linux/man-pages/man2/execve.2.html)’, ‘[open()](http://linux.die.net/man/2/open)’, and ‘[mmap2()](http://man7.org/linux/man-pages/man2/mmap2.2.html)’. Scroll down a little further in the output and you’ll see the system call to ‘[write()](http://linux.die.net/man/2/write)’ that is responsible for printing out “Reticulating Splines….” to the screen.

Finally let's take a look at the Ltrace command. Ltrace functions similarly to Strace with the minor difference that Ltrace prints out library calls made by a program during execution. Its syntax is: “ltrace ./binaryToExecute”. Run the command and you should see:



If you’ve executed the binary normally before you should recognize some of these calls. ‘[Sleep()](http://linux.die.net/man/3/sleep)’ is responsible for that boring 5 second delay before printing a message which is the value of the flag!. You can also see a call to the ‘[puts()](http://www.tutorialspoint.com/c_standard_library/c_function_puts.htm)’ function responsible for printing “Reticulating splines”. Finally we can see information we normally can’t; there’s a call to puts() which should print out “Your ability to hack is….” right before the program exits. Now while none of this information is particularly useful for this challenge. I wanted to demonstrate the power behind the Strace and Ltrace commands when learning about how to approach a target binary.

Flag: TXSTCS{5}