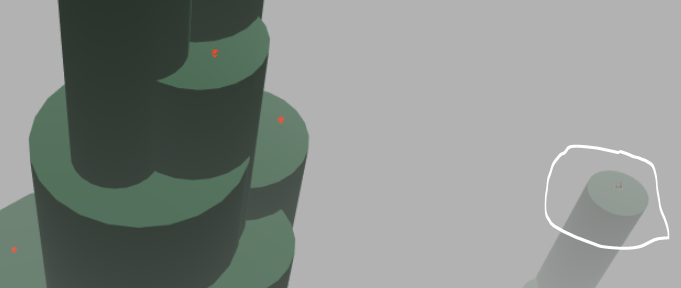
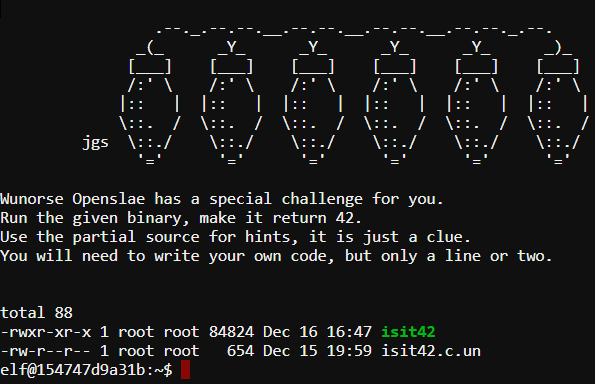
# We’re off to see the… or The isit42 Terminal Challenge

This challenge shows how you can change a compiled binary program by forcing it to call functions that you write, rather than functions from the system library. It is based on a mistake a programmer may make in coding, by naming variables or functions with names that are already in use. (Of course, I’ve never done that…much.) This error in duplicating reserved names can cause erratic results, especially when other programmers attempt to incorporate that code into their work. In this challenge we do it purposely to subvert an already compiled program. This can be done in just about any language that calls library functions.

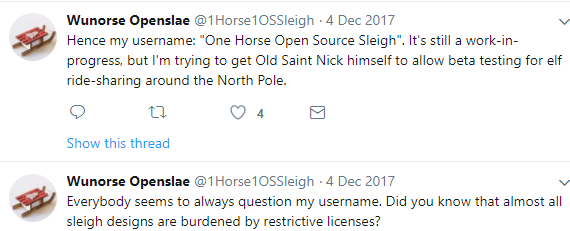
The challenge requires you to create and compile a small C program, but don’t worry. When you remove the lines required in any C program, the code is very short.

The terminal is in the “We’re off to see the…” game. It sits on a pillar next to the main pillar where the game action takes place. If you use the SVGS account, it is in the stocking as the “isit42 terminal challenge.”

The terminal has the compiled binary, isit42, and a fragment of code, isit42.c.un. 

## Hint

Wunorse doesn’t give us any help in his tweets.



However, there is a SANS Penetration Testing Blog that will lead you through the process.   
<https://pen-testing.sans.org/blog/2017/12/06/go-to-the-head-of-the-class-ld-preload-for-the-win>

Since this is the last challenge there will be fewer hints. There are a few things that may be helpful for people who have never compiled C on Linux before.

1. Most Linux distributions have easy access to a C compiler. In Ubuntu, for example, the command apt-get install build-essentials will install the gcc compiler and other utilities. In our case, gcc (GNU Compiler Collection) is already installed so you don’t have to worry about it. When you see this line in the blog,  
   $ gcc hacking\_time.c -o hacking\_time -shared -fPIC ,  
   he is compiling his hacking\_time.c source code (text file) to create an executable (binary) file hacking\_time. You will need to do something similar.
2. In the paragraphs just above “Common Issues”, he tells you how he decided what the variable types for the input and output of his function should be. In the fragment of C code in the challenge there’s a line that says, “The prototype for xxx is…”. That does the work for you and tells you what the function needs to look like. You can drop that line directly into the code you write. It says the output of the function will be an integer, and the function does not require input (i.e., void).

# Hand in

1. Your C source code, and the command you used to execute it.
2. A screenshot of the terminal showing your success.

# Solution

#include <stdio.h>

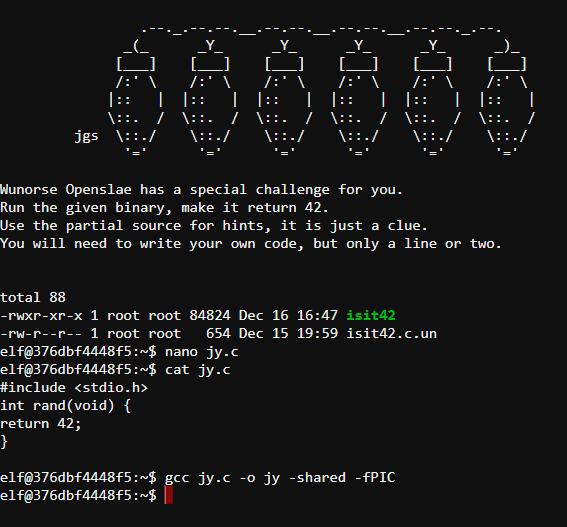
int rand(void) {

return 42;

}

We need to be sure to compile it with the switches listed in the blog:

gcc jy.c -o jy -shared -fPIC



Then we just execute ./jy using the format in the blog:

LD\_PRELOAD="$PWD/jy" ./isit42

