Codio Activity - Buffer Overflow

Buffer Overflow Part 1

In this example, you will compile and run a program in C using the **Codio workspace** provided (Buffer Overflow in C). The program is already provided as bufoverflow.c - a simple program that creates a buffer and then asks you for a name, and prints it back out to the screen.

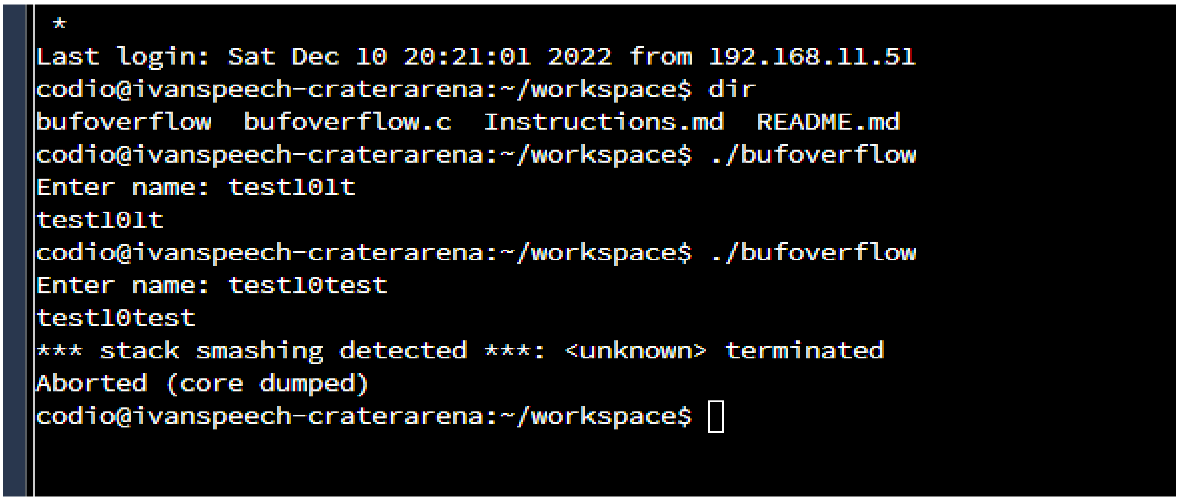
This is the code in bufoverflow.c:

```
#include <stdio.h>
int main(int argc, char **argv)
{
char buf[8]; // buffer for eight characters
printf("enter name:");
gets(buf); // read from stdio (sensitive function!)
printf("%s\n", buf); // print out data stored in buf
return 0; // 0 as return value
{
```

Now use the rocket icon to compile and run the code. To test it, enter your first name (or at least the first 8 characters of it) you should get the output which is just your name repeated back to you.

Run the code a second time (from the command window this can be achieved by entering ./bufoverflow on the command line). This time, enter a string of 10 or more characters.

Output



```
 *
Last login: Sat Dec 10 20:21:01 2022 from 192.168.11.51
codio@ivanspeech-craterarena:~/workspace$ dir
bufoverflow  bufoverflow.c  Instructions.md  README.md
codio@ivanspeech-craterarena:~/workspace$ ./bufoverflow
Enter name: test101t
test101t
codio@ivanspeech-craterarena:~/workspace$ ./bufoverflow
Enter name: test10test
test10test
*** stack smashing detected ***: <unknown> terminated
Aborted (core dumped)
codio@ivanspeech-craterarena:~/workspace$ 
```

- What happens?

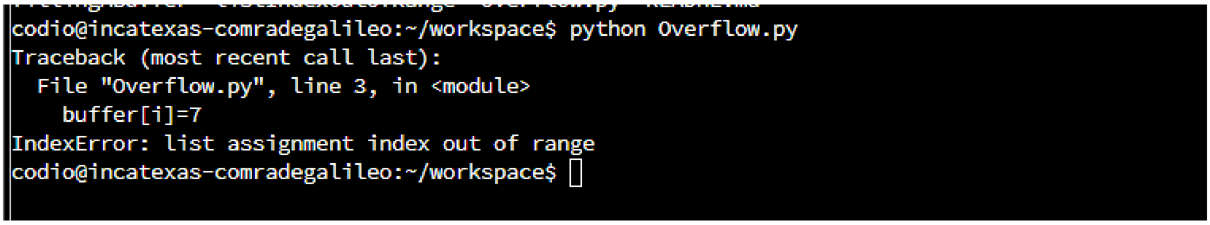
  I was able to find a good guide on the below url, which discusses buffer overflow. In our example when we exceed 8 characters we exceed the buffer capacity resulting in an overflow.

  https://www.educative.io/answers/what-is-the-stack-smashing-detected-error

- What does the output message mean? A Defence against buffer over flows.

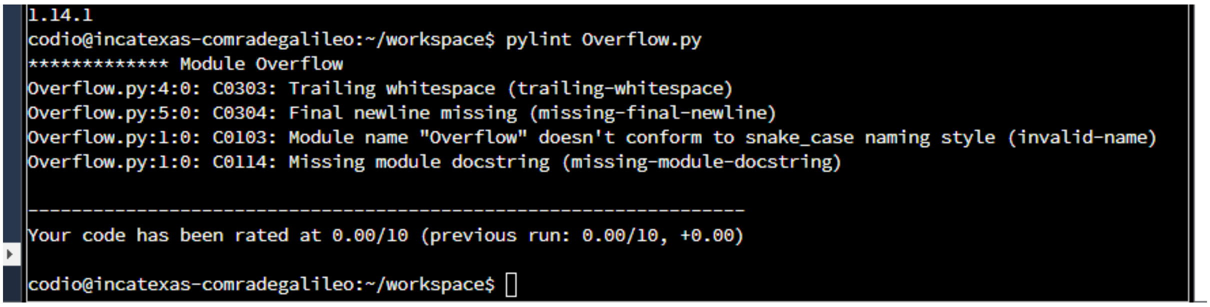
Buffer Overflow Part II

- Run the code using `python overflow.py` (or use the **rocket icon**)
- What is the result?



```
codio@incatexas-comradegalileo:~/workspace$ python Overflow.py
Traceback (most recent call last):
  File "Overflow.py", line 3, in <module>
    buffer[i]=7
IndexError: list assignment index out of range
codio@incatexas-comradegalileo:~/workspace$ 
```

- Install `pylint` using the following command:



```
1.14.1
codio@incatexas-comradegalileo:~/workspace$ pylint Overflow.py
************* Module Overflow
Overflow.py:4:0: C0303: Trailing whitespace (trailing-whitespace)
Overflow.py:5:0: C0304: Final newline missing (missing-final-newline)
Overflow.py:1:0: C0103: Module name "Overflow" doesn't conform to snake_case naming style (invalid-name)
Overflow.py:1:0: C0114: Missing module docstring (missing-module-docstring)

------------------------------------------------------------------
Your code has been rated at 0.00/10 (previous run: 0.00/10, +0.00)

codio@incatexas-comradegalileo:~/workspace$ 
```

Team Activity

You should read Chapter 2,6,7,8 of the course text (Pillai, 2017) and Cifuentes & Bierman (2019) and then answer the questions below, adding them as evidence to your e-portfolio.

1. What factors determine whether a programming language is secure or not?

   The main principle of being secure is data has confidentiality, has restriction in place so that it is not exposed to un-authorized entities, ensure integrity that is that the data can be trusted and finally available by people or systems that authorized to read/write the data. Secure code falls within the same principles. Cifuentes & Bierman (2019) have found that the top three vulnerability fall in buffer error, injection, and information leakage. This violate the principal, buffer error would mean that the system is not available, Injection would result in data tempering and information leakage mean the data is not safe in transit. For a programming language to be secure these principles need to be maintained.

2. Could Python be classed as a secure language? Justify your answer.

   Any programming language can have vulnerabilities if the code is not maintained. Maintenance is required periodically by upgrade the libraries if any vulnerabilities exist. Ensuring passwords police in place and are not shared. Make use of prepared statements if querying a database. Do not expose information to the users when an error occurs. If these principles are missed any coding language can become weak.