**CEG 4424/6624 Security Attacks and Defenses**

**Lab/Project 2**

**(100 Points)**

1. The objective

The objective of this lab is for students to learn more about Buffer Overflow, and exhaustive key search. Students will gain hands-on experience on Buffer overflow and exhaustive key search.

1. Submission

A team can have up to 3 students. All students in the same team will receive the same grade.

* 1. Each team only submits one report or a copy of answers/files. (See the section of tasks).
  2. **Each team member needs to submit the list of the names of all team members**.

1. Tools

You will use the cs unix server, fry.cs.wright.edu for this project, you need to connect to this unix server remotely using a secure shell client, putty. You can remotely connect to this unix server, fry.cs.wright.edu, on campus from a Wright State computer or use your own laptop connecting to the WSU wifi network named “WSU-Secure”. Note that you cannot remotely connect to this computer using ssh using computers outside Wright State University without installing VPN or use the campus “WSU\_EZ\_CONNECT” wifi network.

If you want to connect to this server remotely off campus, you need to install VPN on your computer first (You can download the VPN from WSU, https://www.wright.edu/information-technology/virtual-private-network-vpn. )

You can use WinSCP, the secure file transfer client to transfer files between your local machine, and the server, fry.cs.wright.edu.

**How to use software tools on fry.cs.wright.edu**

(1). Connect to fry.cs.wright.edu using a VPN.

(2). Use putty or other secure shell clients to connect to fry.cs.wright.edu using your campus id (for example, w123abc) and password.

1. Tasks

Download Proj2.zip from Pilot.

**Task 1 - Examining the code** (15 Points)

* The BOFtest.c is a source C code program. Use a text editor to examine the C source code file (but don’t change it).
* Which line of code in this file is the source of the buffer overflow vulnerability? (8 points)
* Which variable is subject to overflow? (7 points)

**Task 2 - Overflowing the buffer** (25 Points)

* Compile the BOFtest.c code with the following command: ***gcc BOFtest.c -o BOFtest*** What does this command do? (5 points)
* Now run the program a few times with various inputs (various number of characters).
* Are you able to cause the program to halt in an error state? What is the number of input characters when the error occurred? Why? (20 points) (Please include the screenshot for the cases when the program runs correctly, and when the error occurred)

**Task 3 – Examine the code** (15 Points)

* The Proj2-test.cpp is a source C++ code program that used cryptopp library functions. Use a text editor to examine the C++ source code file (but don’t change it).
* (15 points) What is the functionality of the source c++ code? Write a short paragraph to answer this question, not just one sentence.

**Task 4 – Compile and execute** (15 Points)

Unzip Proj2.zip to get 00\_e, 01\_e, 02\_e, 03\_e.

Compile the Proj2-test.cpp code with the following command: cryptog++  Proj2-test.cpp  -lcryptopp -o Proj2.

execute Proj2 using the following command:

./Proj2 00\_e 00\_p

Display the contents of 00\_p using cat 00\_p.

* What is the output? Display the output using the command: cat 00\_p (Please include the screenshots of the output in your submission for the input files 00\_e, 01\_e, 02\_e.)

**Task 4 – Add code to record the number of keys that have been tested** (30 points, for graduate students)

* How many key combinations have been tried before the correct key is found? Please add code to record the number of the keys been tried? (The graduate students need to submit the modified source code program for this task. The graduate students also need to report the number of the keys tested for the input files 00\_e, 01\_e, 02\_e. It would be great if the graduate students can include the screenshots to show the number of the keys tested. )

How to compile your programs on fry.cs.wright.edu

fry.cs.wright.edu is a unix server and you can use the cryptopp installed on it.

1. Compiling method 1:

Once you log into fry.cs.wright.edu, use an editor, such as vim, to add the follow lines into the end of your .bashrc.

==========================

if [ -f ~/.bash\_aliases ]; then  
    . ~/.bash\_aliases  
fi

Then you can use the following command,  cat /home/containers/.bash\_aliases, to see if the output is as follows:

[w901mxl@login01 ~]$ cat /home/containers/.bash\_aliases

## container aliases

alias cryptog++='srun singularity exec /home/containers/cryptopp.sif g++'

alias cryptorun='srun singularity exec /home/containers/cryptopp.sif'

If the output contains the commands as shown above, you can compile your c++ program source.cpp using the following command:

cryptog++  <sourcefile.cpp>  -lcryptopp -o desenc1

1. Compiling method 2:

Every time, you log into fry.cs.wright.edu, run the command,

alias cryptog++='srun singularity exec /home/containers/cryptopp.sif g++'

then you can run the command, alias,

If the output contains the commands as shown below

alias cryptog++='srun singularity exec /home/containers/cryptopp.sif g++'

Then, you can compile your c++ program source.cpp using the following command:

cryptog++  <sourcefile.cpp>  -lcryptopp -o desenc1

1. Compiling method 3:

Once you log into fry.cs.wright.edu, you can compile your c++ program source.cpp using the following command without modifying the .bashrc.

srun singularity exec /home/containers/cryptopp.sif g++ des\_encode\_SP2023.cpp -lcryptopp -o desencl

1. Execution:

You can use the following command to execute your program.

./desenc1

Tutorial to use the AES function in the crypto library [Crypto++](http://www.cryptopp.com/):

On fry.cs.wright.edu, the crypto++ library is installed, which you can use directly. You can log into fry.cs.wright.edu using you school wid (w123abc) and corresponding password. You can remotely connect to fry.cs.wright.edu on campus from a Wright State computer or use your own laptop connecting to the WSU wifi network named “WSU-Secure”. Note that you cannot connect to fry.cs.wright.edu using computers outside Wright State University without installing VPN or use the campus “WSU\_EZ\_CONNECT” wifi network, i.e., you need to follow the instructions to install VPN on your computer first.

I have created a discussion forum on pilot. If you have any questions, you can post your questions on the discussion forum first.

1. In order to use the crypto library, in your C++ source program (e.g., test1.cpp), you need to include the right library files, and use the right namespace as follows.

#include "cryptopp/cryptlib.h"

#include "cryptopp/hex.h"

#include "cryptopp/filters.h"

#include "cryptopp/des.h"

#include "cryptopp/aes.h"

#include "cryptopp/modes.h"

2. How to compile your source program:

cryptog++  <sourcefile.cpp>  -lcryptopp -o test

-lcryptopp : link CryptoPP library.

3. How to execute your program:

cryptoexec ./test

1. Encryption with AES:

A text string can be encoded with AES algorithm using the following tool function:

string aes\_encode(string & plain, byte key[])

{

string cipher;

try{

ECB\_Mode<AES>::Encryption enc;

enc.SetKey(key, AES::DEFAULT\_KEYLENGTH);

StringSource(plain, true, new StreamTransformationFilter(enc, new

StringSink(cipher))); //add padding by StreamTransformationFilter

}

catch(const CryptoPP::Exception & e)

{

}

return cipher;

}

The input parameter "plain" is the plain text that you want to encrypt with AES encryption algorithm. The input parameter "key" is a byte array that stores the key you want to use during the cipher process of AES. The length of the key array is defined by constant AES::DEFAULT\_KEYLENGTH (AES::DEFAULT\_KEYLENGTH is 16 for now).The string returned by this function is the output cipher text. In this program assignment, for simplicity, we use AES in ECB mode (In ECB mode, each block of 128 bits of plaintext is encrypted independently using the same key). The input plaintext is padded by StreamTransformationFilter tool if the number of bits in the input plain text file is not a multiple of 128 bits (16 bytes).

1. Decryption with AES:

A cipher text can be decrypted with AES algorithm using the following tool function:

string aes\_decode(string & cipher,byte key[])

{

string plain;

try{

ECB\_Mode< AES >::Decryption dec;

dec.SetKey(key, AES::DEFAULT\_KEYLENGTH);

StringSource s(cipher, true, new StreamTransformationFilter(dec, new

StringSink(plain)));

}

catch(const CryptoPP::Exception& e){

}

return plain;

}

The input parameter "cipher" is the ciphertext that you want to decrypt with AES in ECB mode. The input parameter "key" is a byte array that stores the key you want to use during the decryption process of AES. The length of key array is defined by constant AES::DEFAULT\_KEYLENGTH.