Assignment 6

☕ Coffee and Donuts 🍩

Secure Systems Engineering

Prof. Chester Rebeiro

April 12, 2022

Hiding the assigned function

We can employ packing here. Packing encrypts portions of the code to make it invisible during static analysis, and decrypts those portions only at runtime. This way it would not be possible to statically analyze the code and look for the function.

Step 1: Write a function to calculate the minimum of two numbers.

Step 2: Get an object file for this function, and store the corresponding representation in a string. This string is stored as a character buffer. We can execute the function by typecasting the buffer to a function pointer.

Step 3: Encrypt the string. The key is part of the password, and is necessary to correctly decrypt the the payload string. The encryption and decryption portions were achieved using the TinyAES library.

Thwarting GDB

Only one process is allowed to use `ptrace` to attach to another process at any given time. Hence we can use `ptrace` to try to find out if something is already stuck to us, and deal with it appropriately. But you can skip the `ptrace` function in GDB. To make it difficult to find, we packed the `ptrace` function call as well, and kept it in between junk code.

If one manages to bypass the `ptrace` function call, we also used multithreading in the critical portions of the code so that debugging is harder.

Concealing the password

We didn't keep the password available as a single string literal anywhere in the code. Instead, we dispersed its characters among 1000 strings of 1000 characters each. The ith character of the password will be stored at string[$f(i)$][i]. We randomized the order in which we assigned these 1000 strings as well, for an additional layer of protection. Therefore, when making comparisons to check if the given password is valid, it would be very hard for an external observer to decipher the sequence of operations being performed.

Making static analysis suicidal

First off, in the compiler options, we included the `-s` flag. This removed all function names, except for library functions. This was preceded by adding lots of junk code, to mask the critical sections of the program.

CONFIDENTIAL

Running the executable with the correct password:

```
./CS19B080_CS19B081 <input1> <input2>
5D3F93FB9F70CED6DF8595E03295F7878597A42D978D65B7A2E66F6E7CB1DA66
5FEBC3F62429F4287913392AE5D0672151BD87C67C08A4D5B75B5F751CE44E38
DFAFC0720F7C71D75BCC1D3057E862005B641BD43370E54B2D209C244BF5E06E
FEDCB4A71B000B7030F5BD44858BC16C40431384D3FFBBEF0F1702839C18F1B3
5452C935B3EE8F71957E15BFE6A0B51EB28728A66547EA084897C6C0294545A5
C33A8A28D22A587DCD885946C97FA5276CAA9883374E5119798BC3A578418DB6
FDA527B6E58DD5CA5C7C7DF03C8E2FB2CD04BDFC9FD54A8DE5FCCD3D79B74848
CFFBEA6272FDB42F9179424CA0FBB7270BE25CE6641D9B11CF68E7B2BA542D40
23A57EE56C6D33044A2A78E660982BE71A704E9328831A8A75B718E346457CF4
F940B775947623015DA52B96B0411068CA596C20045E97710332212B286B2A6D
152CDA4993C547BFEAF750E8BF03FEA41E023495443EF21F1B1EFB406EB7DF27
28D4EE76F07AF69F20EF965A9FBAC2D4290DA37A9F128C6CD9A9606524F7EF90
1F447542BABB6158DF2FC2B0D9C58997FD50682CEF781BD78682E7C734CB5529
```