Assignment 4A: Re-Hosting

Objectives for this assignment:

- Learn how to re-host parts of the firmware to emulate and instrument it.

Bill of Materials

To complete this lab you will need:

- STM32 NUCLEOL152RE evaluation board
- 1x USB cable Type-A to type-B mini
- hwsec.elf (on canvas)

Part 1: Rehosting (4):

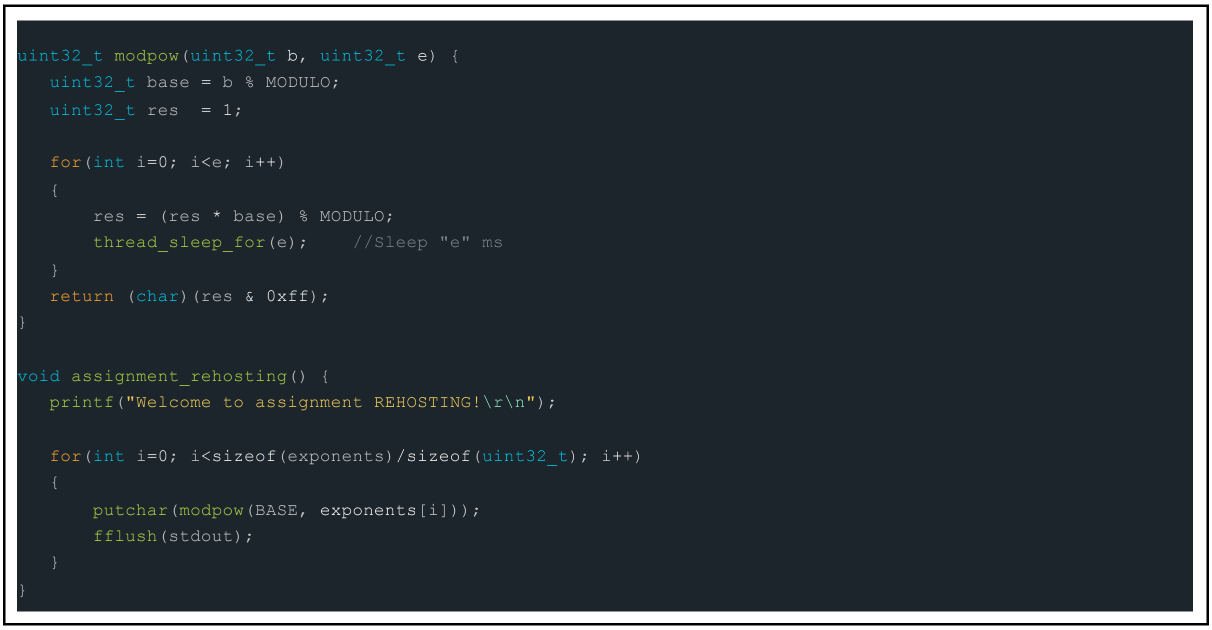
Rehosting is a common method for advanced firmware analysis. In this assignment, you will carry out manual rehosting on function level, to get familiar with the main concepts and challenges of rehosting.

1. Assignment Overview

To start this assignment, connect to the UART interface of the board and type 2 to select “Assignment 4A: RE-HOSTING”.

After a welcome message, you will see some characters being printed. If you wait long enough (~3 months) the flag will be printed.
This is obviously too slow, indeed, the objective of this assignment is to rehost part of the firmware to speed up this process.

Below you can find the code for this assignment:



```c
uint32_t modpow(uint32_t b, uint32_t e) {
   uint32_t base = b % MODULO;
   uint32_t res  = 1;

   for(int i=0; i<e; i++)
   {
       res = (res * base) % MODULO;
       thread_sleep_for(e);    //Sleep "e" ms
   }
   return (char)(res & 0xff);
}

void assignment_rehosting() {
   printf("Welcome to assignment REHOSTING!\r\n");

   for(int i=0; i<sizeof(exponents)/sizeof(uint32_t); i++)
   {
       putchar(modpow(BASE, exponents[i]));
       fflush(stdout);
   }
}
```

As you can see, the slowdown is caused by `thread_sleep_for()` function.
Your objective is to rehost `assignment_rehosting` and `modpow` while hooking `thread_sleep_for` to skip it. By emulating without sleeps, you will obtain the flag in a few seconds.

2. Setting up unicorn

To emulate the firmware you will use unicorn (https://github.com/unicorn-engine/unicorn).
The easiest way for installing unicorn, alongside with python bindings, is using pip:

```
pip install unicorn
```

Alternative installation methods (with binding for other languages) can be found at:
https://www.unicorn-engine.org/docs/Install

To get a feeling how unicorn is used, we suggest to follow the following tutorial:
https://www.unicorn-engine.org/docs/tutorial.html

3. Rehost assignment_rehosting & modpow

As a template we suggest starting with the one provided by the engine itself:
https://github.com/unicorn-engine/unicorn/blob/master/bindings/python/sample_arm.py

The steps to be performed can be summarized as below:

1. Start a GDB server, set a breakpoint to assignment_rehosting, and reach it.
2. Dump the firmware, the ram, and the register values to create a snapshot of the current execution context.
3. Load the snapshot in unicorn engine (refer to sample_arm.py for this step).
4. Create a hook for putchar to print on your console the flag character.
5. Create a hook for thread_sleep_for to skip it.
6. Create hooks for all the remaining code that you want to skip (eg fflush)
7. Start emulation and get the flag!

Deliverable:

- A text file containing the flag.
- A log of your gdb-session (or a script) to dump the state to be fed to unicorn.
- Your program for rehosting the target functions.
- The binary files required to run this program.

Hints:

- You should have received a firmware.elf file. By passing it as argument to gdb-multiarch (or loading it using the “file” command), you can make use of the symbols contained in the ELF!
- These symbols allow you also to identify the addresses of the functions! Perfect, for determining hook locations.

Part 2: Alternative solutions (2pts):

Rehosting with unicorn is just one way to obtain the flag for this challenge and there are multiple other ways. Get creative and show us other ways to obtain the flag without using unicorn!
Each alternative way will give you one point, up to a maximum of two points.

Deliverable:

- A text file describing your alternative solutions to obtain the flag
- Supporting code and binary files for each approach

Hints:

- You could use other rehosting frameworks, static analysis, or even a re-implementation of the assignment on your host! Any approach could qualify for points (EXCEPT copying from other students)!