CS4379/CS5375
Dr. Jaime C. Acosta
Software Reverse Engineering

Homework Assignment – Process Injection

Due Date: **November 3, 2022 before 11:59pm**
Late assignments are accepted until November 5, 11:59pm (with a 2 letter grade deduction)

This assignment will be completed in groups of size **2-3**. At least one group member must have a personal laptop with VirtualBox installed.

In class we talked about process injection. In this assignment, you will learn how to use DLL injection to modify a remote process' behavior at runtime. You will have to install VirtualBox on your personal computer and will also have to download a virtual machine from here:

https://cs5375.cs.utep.edu/software/SREToolsVM.ova

Please password encrypt your zip files using the password: *infected*

PR I – Setting up Your Computer

1. Download and import the Virtual Machine from the following URL:

https://cs5375.cs.utep.edu/software/SREToolsVM.ova

2. Boot the Virtual Machine and use the following credentials to login:

username: cyber

password: arlsouth

(Windows Defender has been disabled on this VM. If it turns back on it may delete your .exe files. Instruction for turning it off are here: https://www.windowscentral.com/how-permanently-disable-windows-defender-windows-10)

PR II – Familiarization with Cheat Engine

In this section you will become familiar with the Cheat Engine.

1. Double click on the Cheat Engine shortcut from the Desktop and the Select ***Help*** -> ***Cheat Engine Tutorial***
2. Complete the tutorial steps 1-5. See https://wiki.cheatengine.org/index.php?title=Tutorials:Cheat_Engine_Tutorial_Guide_x64 for the answers. This may also help: https://www.youtube.com/watch?v=mCHYhOxwXuQ

PR III – Inject a DLL

In this section you will execute a victim process and then inject a DLL into that process.

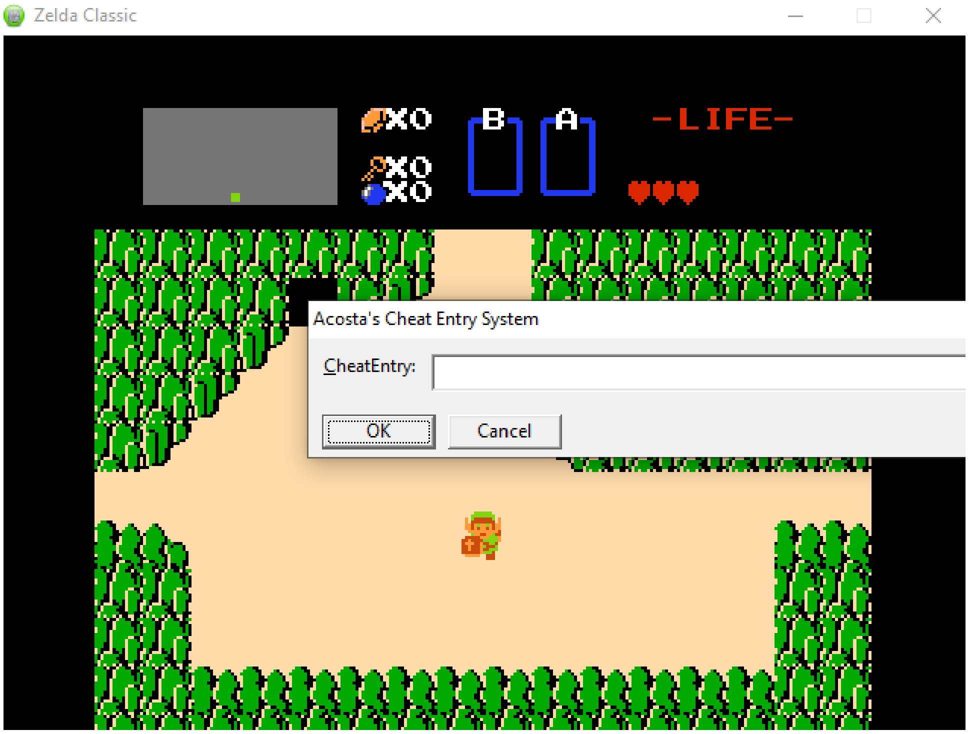
1. You'll notice the following files on your Desktop:
   a. Injector/solution/Injector_answer.exe – contains the .exe that will inject a dll into a process
   b. Injectee/solution/Injectee_answer.dll – contains a dll that can be injected into a process
   c. Injector/Injector.cpp – contains the partially implemented source code of (a)
   d. Injectee/Injectee.cpp – contains the partially implemented source code of (b)

2. Start the Zelda game by running zelda-w.exe (there is a shortcut on the desktop: zelda.exe - Shortcut).

3. Open a mingw32 terminal by clicking on the mingw32.exe icon on the Desktop. Navigate to Injector.exe and run it with the required parameters (usage information will be given when you execute the file) as shown below:

```
cd Injector/solution/
./Injector_answer.exe c:/Users/cyber/Desktop/Injectee/solution/Injectee_answer.dll zelda-w.exe
```

You should have the Zelda game running ☺ and an entry window (it may be hidden behind other windows) on your screen. As shown in class, obtain a sword and collect some rupees. Type *cheat on* and click submit to see what happens when you collect (or spend) rupees. You may switch off the cheat by entering *cheat off*.



Task 1 – Complete the Injector_Template Source Code

Your task is to finish the partially implemented Injector template by reverse engineering the Injector_answer.exe you unzipped in step **PR III part 1a**.

1. On the Desktop, double click on Injector/Injector.cpp to open it in Visual Studio Code.

2. Open a new MSYS2 terminal and navigate to the template directory and compile it as shown below. This will generate a non-working Injector.exe file.

```
cd Injector/
make
```

3. Using IDA Pro (there is a shortcut on the desktop) reverse engineer Injector_answer.exe and fill in sections of Injector.cpp labeled TODO (function calls, parameters, etc.). Your compiled Injector.exe should have the same functionality as the Injector_answer.exe (you can still test with Injectee_answer.dll from **PR III part 1a** at this point).

* When you're done, rename you IDA idb file to deliverable1.idb. Also, name your tutorial-style write-up for Task 1: deliverable1.doc. Make sure that the write-up includes your process and a description of every line of code, parameters, etc... that you added. Create a zip file with: (1) your **Injector folder** that contains your source code, (2) **deliverable1.idb**, and (3) **deliverable1.doc**. Name your zip file **deliverable1.zip** *

Task 2 – Complete the Injectee_Template Source Code

Your task is to write in correct values in the Injectee dll template source code to toggle the unchanging rupees cheat for Zelda.

1. On the Desktop, double click on Injectee/Injectee.cpp to open it in Visual Studio Code.

2. Open a new MSYS2 terminal and navigate to the template directory and compile it as shown below. The compilation will fail with a few errors that you'll have to fix, as described in the next step.

```
cd Injectee/
make
```

3. Use Cheat Engine (there is a shortcut on the desktop) to identify and fix the values that should be in sections of the Injectee.cpp source code labeled ***TODO***. You will also have to add a call to the memcpy function (see https://msdn.microsoft.com/en-us/library/dswaw1wk.aspx) among others. Recompile the program to generate Injectee.dll

* When you're done, name your tutorial-style write-up for Task 2: deliverable2.doc. Make sure that the write-up includes your process and a description of every line of code, parameters, etc... that you added. Create a zip file with: (1) your **Injectee folder** that contains your Injectee.cpp source code, and (2) **deliverable2.doc**. Name your zip file **deliverable2.zip** *

Task 3 – *Set Rupee to 255 when any rupee is collected*

Your task is to change the game behavior such that when a rupee is collected, the rupee amount will become 255 (or 0xFF).

Note: Do not assume that simply changing a memory value will solve this task -- this must work even after the game is restarted and/or the OS is rebooted.

* When you're done, name your tutorial-style write-up for Task 3: deliverable3.doc. Make sure that the write-up includes your process and a description of every line of code, parameters, etc... that you added. Create a zip file with: (1) your **Injectee folder** that contains your Injectee.cpp source code, and (2) **deliverable3.doc**. Name your zip file **deliverable3.zip** *

Task 4 – Invulnerability

Your task is to change the game behavior so that the player will never lose health.

Note: Do not assume that simply changing a memory value will solve this task -- this must work even after the game is restarted and/or the OS is rebooted.

Each task is more difficult; you will have to trace through some assembly (it is recommended that you use Cheat Engine's *memory view* window to do this, but you may use any tool of your choice).

* When you're done, name your tutorial-style write-up for Task 4 deliverable4.doc. Make sure that the write-up includes your process and a description of every line of code, parameters, etc... that you added. Create a zip file with: (1) your **Injectee folder** that contains your Injectee.cpp source code, and (2) **deliverable4.doc**. Name your zip file **deliverable4.zip** *

Task 5 – Add Another Functionality of your Choice

Your task is to add the ability to toggle a functionality of your choice to empower a player.

You may receive extra credit if this functionality is considered above and beyond the call of duty.

* When you're done, name your tutorial-style write-up for Task 5: deliverable5.doc. Make sure that the write-up includes your process and a description of every line of code, parameters, etc...

that you added. Create a zip file with: (1) your **Injectee folder** that contains your Injectee.cpp source code, and (2) **deliverable5.doc**. Name your zip file **deliverable5.zip** *

Deliverables:

- Create a single, password-protected, zip file called ***Assignment-Injection.zip*** that contains the other zip files associated with your deliverables. Use the password: infected

Store your deliverables on cloud storage (preferably google drive). Submit ***a link*** to the folder with your **password-protected** Assignment-Injection.zip file using the subject **Assignment-Injection** to fall22cs5375@gmail.com. Make sure the TAs and I have read access to the folder. ***You will not be able to send a password protect zip directly through email.***