**RE lab 09 - reversing from other source languages**

**Lab files and setup**

Download the lab files from [here](https://pwnthybytes.ro/unibuc_re/09-lab-files.zip). The archive password is infected.

* Install CFF Explorer from [here](https://ntcore.com/files/CFF_Explorer.zip) in your Windows VM. You will use it to study the EXE file format.
* Install dnSpy from [here](https://github.com/0xd4d/dnSpy/releases) in your Windows VM. You will use it for reversing and debugging .NET code.
* Follow the steps [here](https://developer.android.com/studio/run/emulator) to get an Android emulator up and running.
* Install Bytecode Viewer from [here](https://github.com/Konloch/bytecode-viewer).

**Tasks**

**Task 1**

* Investigate task1.exe however you see fit. Spend NO MORE THAN 10-15 minutes on trying to approach the exe as usual, with IDA Pro.
* Next, open in CFF Explorer and look for the “FileDescription” field. What value does the binary have and what does it mean?
* Open the exe in 7z or Winrar, extract the underlying executable and open in IDA Pro. Explain why you think the extraction works. **(2p)**
* What type of file is recognized in this executable by IDA Pro ?
* Notice that decompilation does not work and reading the assembly is pretty hard
* Now use dnSpy (64 bit) to open the exe and poke around in btnDecode\_Click. Find the correct output (either through static analysis or dynamic analysis, both using dnSpy). **(4p)**

**Task 2**

* Investigate task2. What type of file is it? How can you unpack its contents? **(1p)**
* Run the application in an emulator **(2p)**
* After unpacking it, use Bytecode Viewer to open the same file.
* Look under “com” through the Activity classes. Where is the password checked?
* What does Loadlibrary do and where is the library file?
* Open it in IDA, look in the “Java\_com\_flareon\_flare\_ValidateActivity\_validate” function.
* Load jni.h using File/Load File/Parse C header file
* Retype the function as "int \_\_fastcall Java\_com\_flareon\_flare\_ValidateActivity\_validate(JNIEnv \*jnienv, int a2, jstring input)"
* Reverse the function and find the correct input. **(7p)**