Lab: Detecting and Unpacking

Environment

- Windows Virtual Machine
- PEiD
- IDA, Cheat Engine, Immunity, etc.

Links for the required tools are available in the repository under the 'tools' directory.

Target

- Dark Basic Pro

Provided in the 'target' directory.

Outline

1. In this folder, you will find two versions of the Dark-Basic-Pro program we've been examining in prior labs, TGCSerial.exe, and TGCSerial.orig.exe.
2. Compare the original executable (TGCSerial.orig.exe) to the new version (TGCSerial.exe):
    a. Do they have a similar file size?
    b. Do they appear to run the same?
    c. Drop the new exe (TGCSerial.exe) into IDA:
        i. Does it look the same as it did in the IDA lab?
        ii. Are there any strings in the strings window?

Task 1

- Use the skills you have developed so far, along with IDA, CheatEngine, Immunity, etc., to crack this new version of TGCSerial.exe.
- CHALLENGE: This *can* be done, but is difficult. See if you're able to do it, but limit yourself to 20 minutes for now.

Task 2

- Launch PEiD, and load TGCSerial.orig.exe into the tool. Observe the results.
- Load TGCSerial.exe into PEiD next. Compare the results of this new version of the executable to the original version. What is different?
- The new TGCSerial has been *packed*. PEiD can shed light on what packer was used.
- There are some common approaches to cracking a packed binary:
    - In some cases, tools exist to *unpack* the binary. Always try this first.
    - In other cases, no such tool is available. However, the program generally *must* unpack *itself* in memory in order to run. Tools like IDA, Immunity, and CheatEngine can be used to capture the unpacked program after it has extracted itself to memory. Dynamic analysis is crucial here – by following execution and watching memory contents, we can watch for strings and other structured data to begin to appear as the program unpacks itself. Some automated tools exist to help here as well.
- Using the PEiD tool and internet sleuthing, determine how the new TGCSerial has been packed. What tool was used? Is there a way to unpack it?

ISBN: 139419988

- Based on your findings, unpack, patch, and repack TGCSerial.exe.

Optional Bonus Tasks

- Without using an existing unpacker, create an unpacked version of TGCSerial.exe using IDA, Cheat Engine, Immunity, etc. (Caution: challenging)
- Explore other automated approaches to unpacking this target, such as https://github.com/unipacker .

ISBN: 139419988

Task 2 Hints

Tip: Use hints sequentially; once you've unblocked yourself, try to continue without using the subsequent hints.

- Follow the string "%c%c%c%c%c%c%c%c" in the Strings window.
- Next to the string in the main window, you will see cross-references to the string, in the form "DATA XREF:". Double click a cross-reference to find where that string is used in code.
- Rather than patching this function (which is the serial check function), it is often easier to patch the *call* to the check function.
- Scroll to the top of the serial check function, and, next to the subroutine name, look for code cross-references in the form "CODE XREF:". Double click the code cross-reference to find who calls the serial check function.
- Use 010 to patch out the call to this function (remember nops), or modify the conditional "jz" that checks the return value from the function.

ISBN: 139419988