Target Acquired
Malware
200 Points

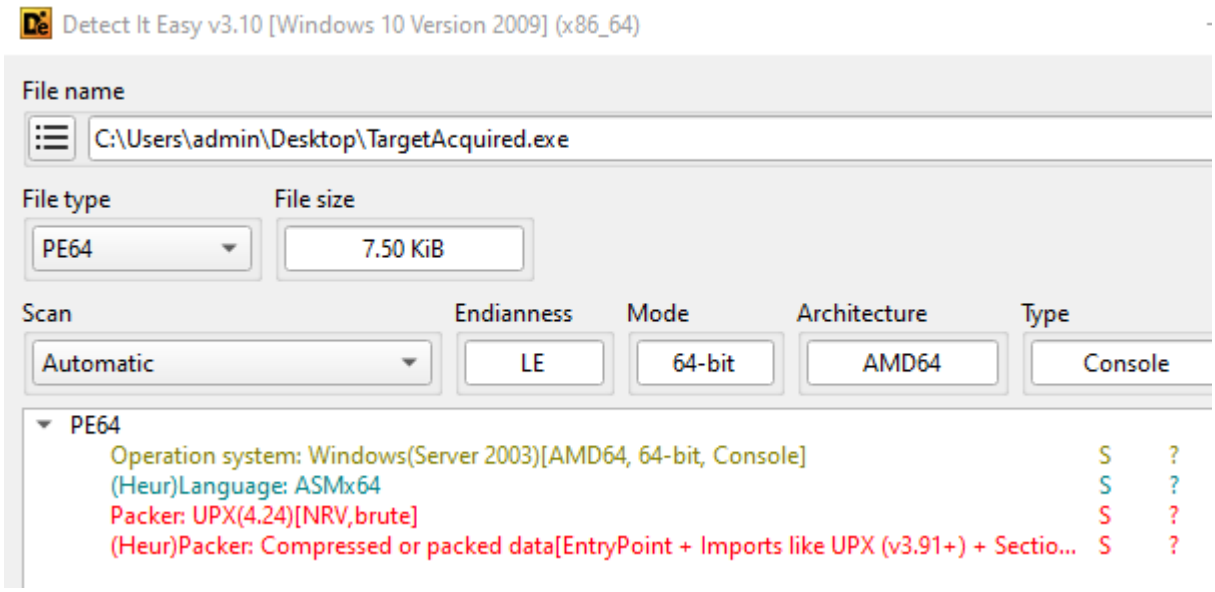
Both challenges in the Malware category were able to be solved using tools and steps in what is commonly referred to as the “Basic Analysis” phase. This means that no debugger, disassembler, or decompiler is necessary to solve these challenges.

All the tools required to solve this challenge are installed by default with FlareVM, one of the best malware analysis VMs out there, made by Google’s Mandiant: https://github.com/mandiant/flare-vm.

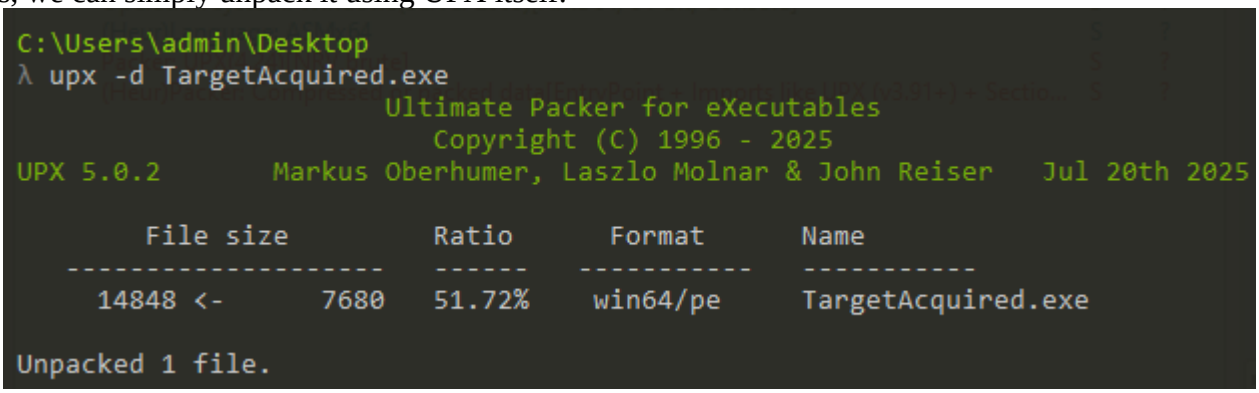
The easiest way to solve this challenge is using the strings utility, or rather Floss, since this is a Windows executable. Floss is another tool made by Mandiant that has several string decryption techniques built into it that the regular strings utility doesn’t have, but it only works on PE executables for Windows. Using Floss on the file:

```
C:\Users\admin\Desktop
λ floss -n 5 TargetAcquired.exe > floss.txt
INFO: floss: extracting static strings
finding decoding function features: 100%| | 4/4 [00:00<?, ? functi
INFO: floss.stackstrings: extracting stackstrings from 3 functions
extracting stackstrings: 100%| | 3/3
INFO: floss.tightstrings: extracting tightstrings from 1 functions...
extracting tightstrings from function 0x14000c742: 100%| | 1/
INFO: floss.string_decoder: decoding strings
emulating function 0x14000c742 (call 1/1): 100%| | 4/
INFO: floss: finished execution after 2.73 seconds
INFO: floss: rendering results
```

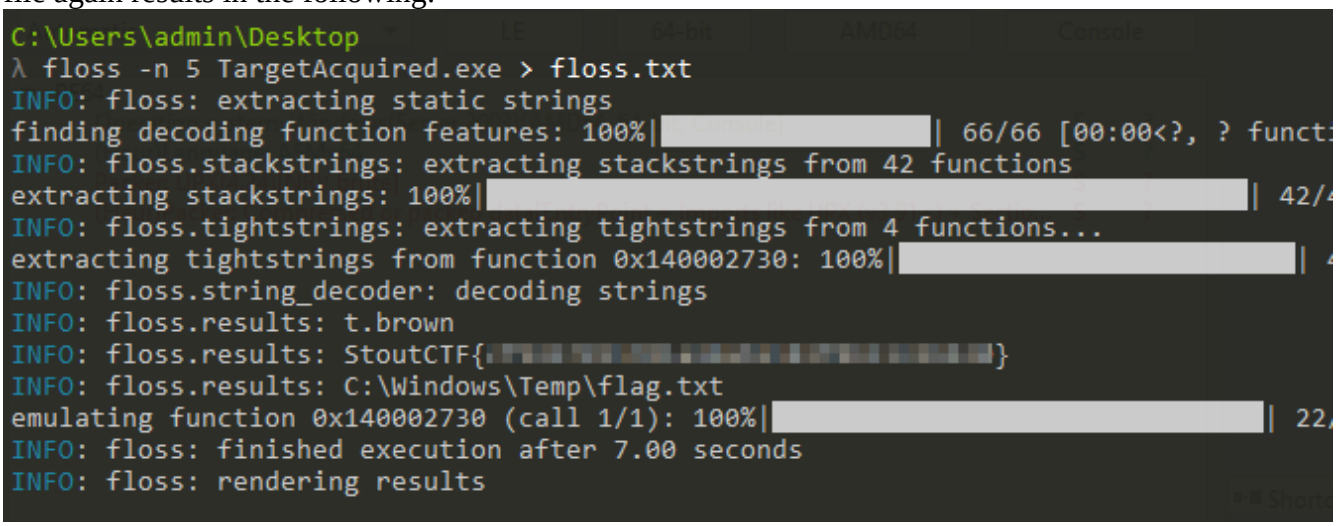
Looking through the floss.txt results file, nothing of interest can be found. Having very few strings like this is usually a sign that the sample is packed, as there are commonly strings left over from compiling programs that are dug up with Floss. To check whether it is, we can use Detect it Easy (DiE):



From the output, we can see that DiE detects that the sample is packed using UPX. Now that we know this, we can simply unpack it using UPX itself:



Now that it's unpacked, Floss should be able to give us a little more information. Running Floss on the file again results in the following:



We can see that Floss was able to decode the flag now that the file was unpacked, along with a few other strings.