HOW TO: unboringly tease Google CTF 2019

HOW NOT TO: introduce into python

1 Introduction

Last year's Google CTF's Beginners Quest[1] did not introduce into reverse engineering very well. Unfortunately there were two RE-challenges and only one of them, called GATEKEEPER[2] , with the potential to get you in touch with a disassembler. Most video write-ups, I have seen [3][4][5], did not take a look into the assembly or the algorithm itself, because it was not necessary and they caught the password almost immediately. What a pity! How could this be and does it give a good introduction into the topic of reverse engineering?

2 Problem

Because the encoded password was stored within the binary, you got your attack vector. In my opinion, the chosen password was way too trivial and so the reversed leetspeak phrase „zLl1ks_d4m_t0g_I" kind of alerted everybody. Not real reversing but literally simple reversing was involved to get to the flag. There was a big unused potential within this task. It was small and commonly compiled code, to easily reverse and understand the algorithm, instead of guessing the right answer. I heavily thought about how to use this good potential and gave it a try patching it.

[1] https://github.com/google/google-ctf/tree/master/2018/beginners

[2] https://github.com/google/google-ctf/blob/master/2018/beginners/re-gatekeeper/attachments/gatekeeper

[3] https://www.youtube.com/watch?v=bshuAGkgY3M

[4] https://www.youtube.com/watch?v=qDYwcIf0LZw

[5] https://www.youtube.com/watch?v=WUOMnLWKFrc

3 Solution

I just tinkered a little bit inside the binary, closed the backdoor and let you peek into crucial changes being made. You should be unable to simply reverse the patch. I think it is still easy but hopefully not as quickly solvable as last time. Perhaps you will learn at least something new from the modified challenge.

4 Task

The home owners put another cake in the fridge, not before fixing some issues and patching the software. Thanks to our surveillance team, we just intercepted some parts of the current patch.

```
#! /us..bin/..thon
f = open('gatekeeper', 'r+b')
f.s.ek(0xde0)
f.wr..e(b'S..Wh..e')
f.seek(0xe01)
f..rite(b's..cr..E..1k..rc')
..see..0xb29)
f.write.b.\x..')
```

Good luck and lots of fun using your prefered disassembler to reverse some x86[6] opcodes. Experienced players must not use the given link and instead disassemble the binary stored in olly's magical backup patterns. With pen and paper only, of course! ;P. Solutions you could mailto:idandre@hotmail.de. Do you feel like playing more CTFs? Let's meet June 22 at Google CTF 2019[7]!

[6] https://github.com/idandre/gatekeeper-2.git

[7] https://g.co/ctf