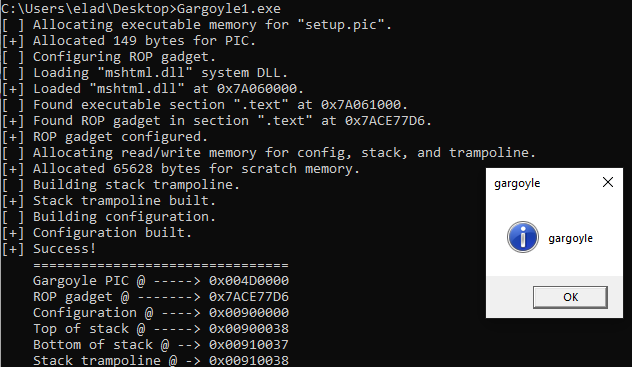
**Gargoyle Malfind Evasion**

We ran gargoyle on a 32bit windows 10 machine:



As you can see, the payload was a message box.

When gargoyle is sleeping, it marks the payload pages as read only:



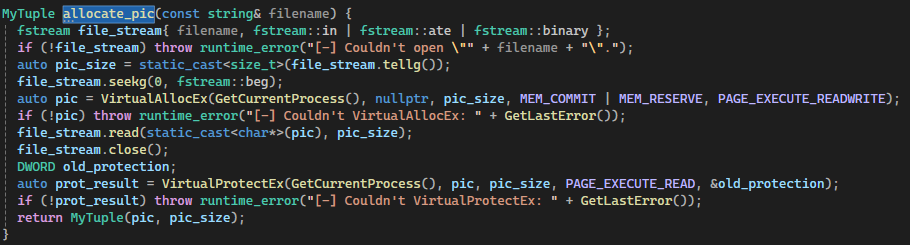
And while the payload is running, the pages are marked back to read execute:



However, although gargoyle does the r-rx switches as expected, it's being detected by volatility's malfind:



It even claims that the protections are EXECUTE\_READWRITE which is weird, because we saw that there is no state in which the protections are like that.

The reason for this is that gargoyle implemented in this way: it first allocates read-write-execute region, and then it changes its initial protection by calling VirtualProtectEx and removing the write protection:  
  
  
  
Then after finish operating, it does the rx to rw switch we described before.

Hence, malfind plugin, which searches for vad nodes with read-write-execute protections, detects the gargoyle payload even when it has only read-write protections (as vad node saves only the initial protection of the region when it was allocated).

However, with a little adjustment we can bypass malfind easily, as we don't need to have both write and execute protections at any stage. More specifically, we don't need to allocate the memory region with execute permission, because we only need to have write permission in order to initiate the region by writing the gargoyle payload into it.  
So all we need is to change the call to VirtualAllocEx from the previous figure to this call:



Indeed, after we applied this patch, malfind doesn't catch it anymore!