TimeStomper Evaluation

Contents

[Introduction 3](#_Toc42955973)

[Installation 3](#_Toc42955974)

[Using TimeStomper 4](#_Toc42955975)

[Conclusions 9](#_Toc42955976)

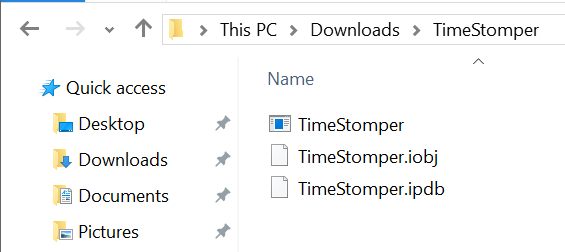
# Introduction

TimeStomper is a proof of concept tool that uses the Windows SetFileTime() API to modify the MACE (Modified-Accessed-Created-Entry) values of files. This technique is commonly called timestomping, named after TimeStomp, an executable that popularized the technique. However, TimeStomp no longer runs on Windows 10, so I am turning to a more up to date alternative tool.

# Installation

TimeStomper is a publicly available PoC program created by the github user slyd0g. You can find it at this link <https://github.com/slyd0g/TimeStomper>

From this github page, brose into the release folder. There you’ll find the three files needed to run TimeStomper. I downloaded a copy of each and placed them within a folder named “TimeStomper” to ensure they would stay grouped together should I need to move this tool’s location.



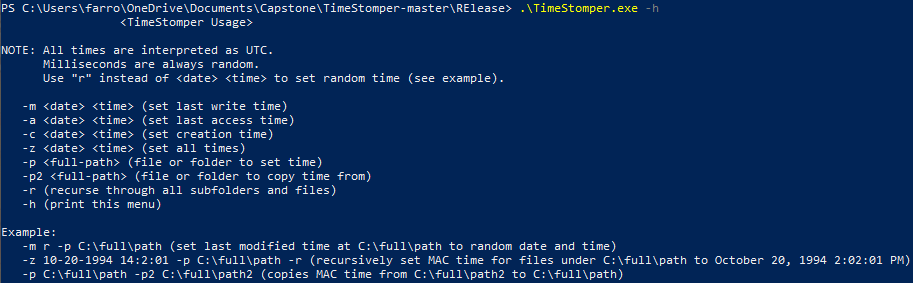
*Fig 1. The contents of my TimeStomper directory*

With these files downloaded, TimeStomper was ready to be used.

# Using TimeStomper

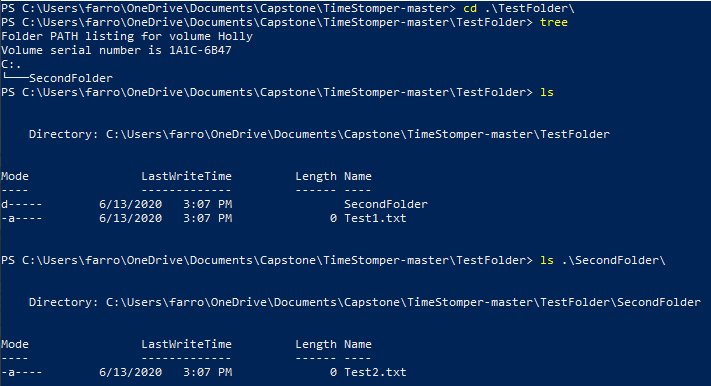
*Note: I was unable to get this tool working on my SEC335 VM. However, it ran just fine on my host Windows 10 device. Some file paths may have changed as a result.*

Using Timestomper is simple. It is a command line only tool with only 8 possible arguments, as shown in this screenshot of its help page.

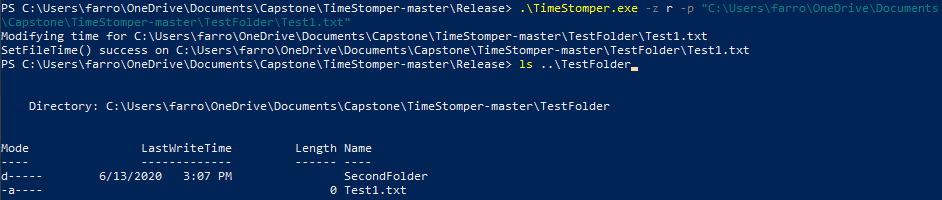


*Fig 2. The TimeStomper help page*

To explore TimeStomper’s settings, I created a test file, as well as a test directory to explore its functionality. The structure of these files and directory are shown here.

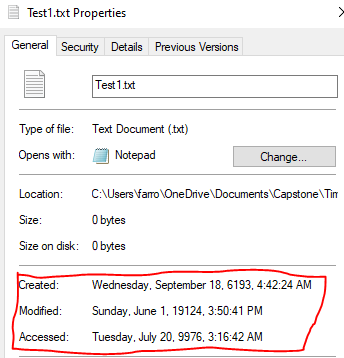


*Fig 3. The contents of the TestFolder directory.*

The first thing I tested was randomly setting all MACE values for the “Test1.txt” file. To do this, I used the -z r argument. You can see the full command and the result in the screenshot below.

*Fig 4. The results of randomly setting all MACE attributes as viewed through the command line*

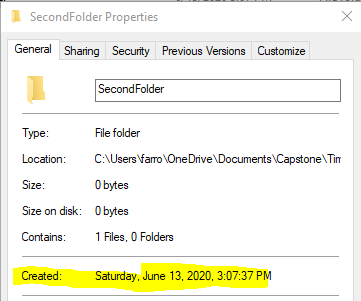
As you can see in the above screenshot the date and time that the file were edited do not appear in the command line. However, if you view the file properties using the Windows GUI you can see what the values were changed to.



*Fig 5. The MACE data as viewed through the Windows GUI*

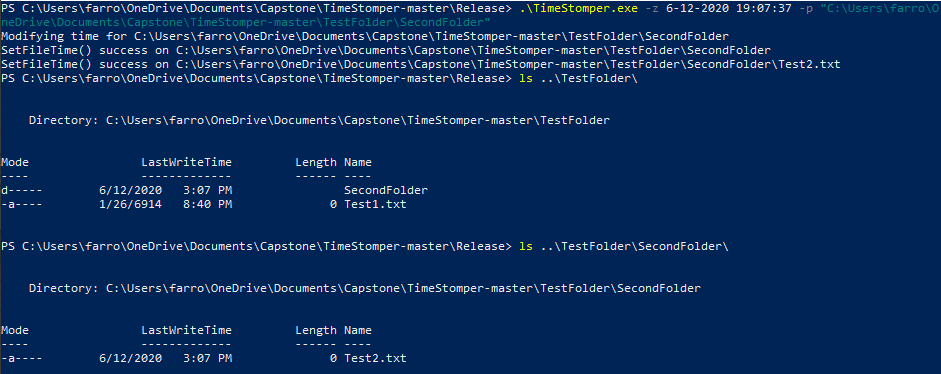
I believe that the reason these values are not able to be displayed within the windows CLI is that they are simply too far outside the expected range of dates. Having repeated this command several times the random function always seems to generate dates with these types of ranges. While this would be useful to tamper with timeline analysis, it is an incredibly obvious technique.

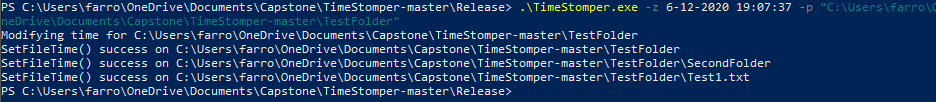
The next technique I tried was manually setting the date and time for a folder. In order to test this, I decided that I would make the “SecondFolder” directory appear as if I had created it one day earlier.



*Fig. 6. The Created time of the SecondFolder directory before TimeStomper*

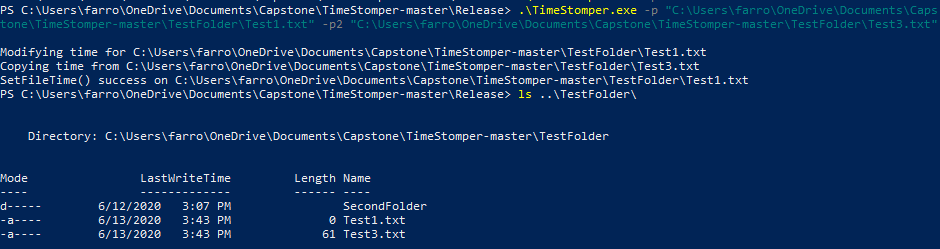
Again, I used the -z flag, this time specifying the date and time that I wished to be used. The full command as well as the results can be found below.

*Fig 7. The Results of running TimeStomper targeting a folder using a specific Date and Time.*

As you can see while the program was successful in editing the date and time of the file to be exactly one day earlier, TimeStomper also recursively changed the date and time of all the files contained with in SecondFolder. However, when I tried the same process on the main TestFolder directory itself, it only performed the edits one layer deep. In other words, when TimeStomper targets a directory it by default will always change the MACE values for all contained files and subdirectories but will not change values for any files or subdirectories within its own subdirectories. This manual setting of a files MACE data could be useful for a variety of tasks. For example, hiding a shell in System32 could become easier, as you could change the MACE information of your backdoor to match the information of other files in that folder. To an unsuspecting user, it may appear as if the file belongs there.

*Fig 8. The Results of running TimeStomper targeting a directory containing subdirectories.*

The final functionality left to test was TimeStomper’s ability to edit the MACE data of one file to be the same as the MACE data of another. To perform this, I decided to create a new file, Test3.txt, and set the value of Test1.txt’s MACE information to the value of Test3’s.

I first created Test3.txt simply by right clicking and using the “New > Text File” dialog prompt within windows. I also then manually edited the contents of the file to update the modified and accessed values of the file. I then used the -p2 option in TimeStomper to specify Test3.txt as the basic template for the copy procedure.

*Fig 9. The results of using the -p2 argument in TimeStomper*

The command worked as expected, and the values of both Test3.txt and Test1.txt were set identical. This could be an incredibly useful tactic for someone performing an offensive security investigation. For example, if a configuration file needed to be changed to allow the attacker to perform some basic command, they could first set the value of any other file on the system to the MACE data of the target file. Then, after changing the file, simply copy back the correct MACE data so there would be no suspicious write behavior apparent in it. This would only take two TimeStomper commands.

# Conclusions

TimeStomper is a powerful tool that gives attackers all the options that they need in order to make timelines hard to construct. While I have found that it’s random time generation function is simply too obvious to an investigator, the fact that it allows users to manually set each property, as well as to copy values from other files on the system gives it flexibility and real world applicability. It also does not require escalated privileges to run, a big help when you are making the initial entry to a system. Overall, it is an incredibly useful replacement to TimeStomp, and should be something that any forensics investigator is aware of.