**Capture the Flag (CTF)**

Challenge on IoT Forensics • Instructor • Challenge 1

**Index**

**Assumptions/Needs** 4

**Objective** 4

**Problem** 4

**Task 1** 4

**Task 2** 6

**Task 3** 7

**Task 4** 8

**Objective**

The user’s objective will be to use a filesystem debugger tool to read a baseline timestamp on a few files. After establishing the baseline timestamps, a malicious actor will make modifications to some of the files. The user will have to determine which files were affected, when the attack took place, and what was done.

**Problem**

You suspect that you have experienced a breach. Your job is to determine the extent of the breach and when the breach occurred.

**Assumptions/Needs**

* Hyper-V software for virtual Rasperry Pi
* Raspbian Stretch OS
* debugfs enabled in Linux.

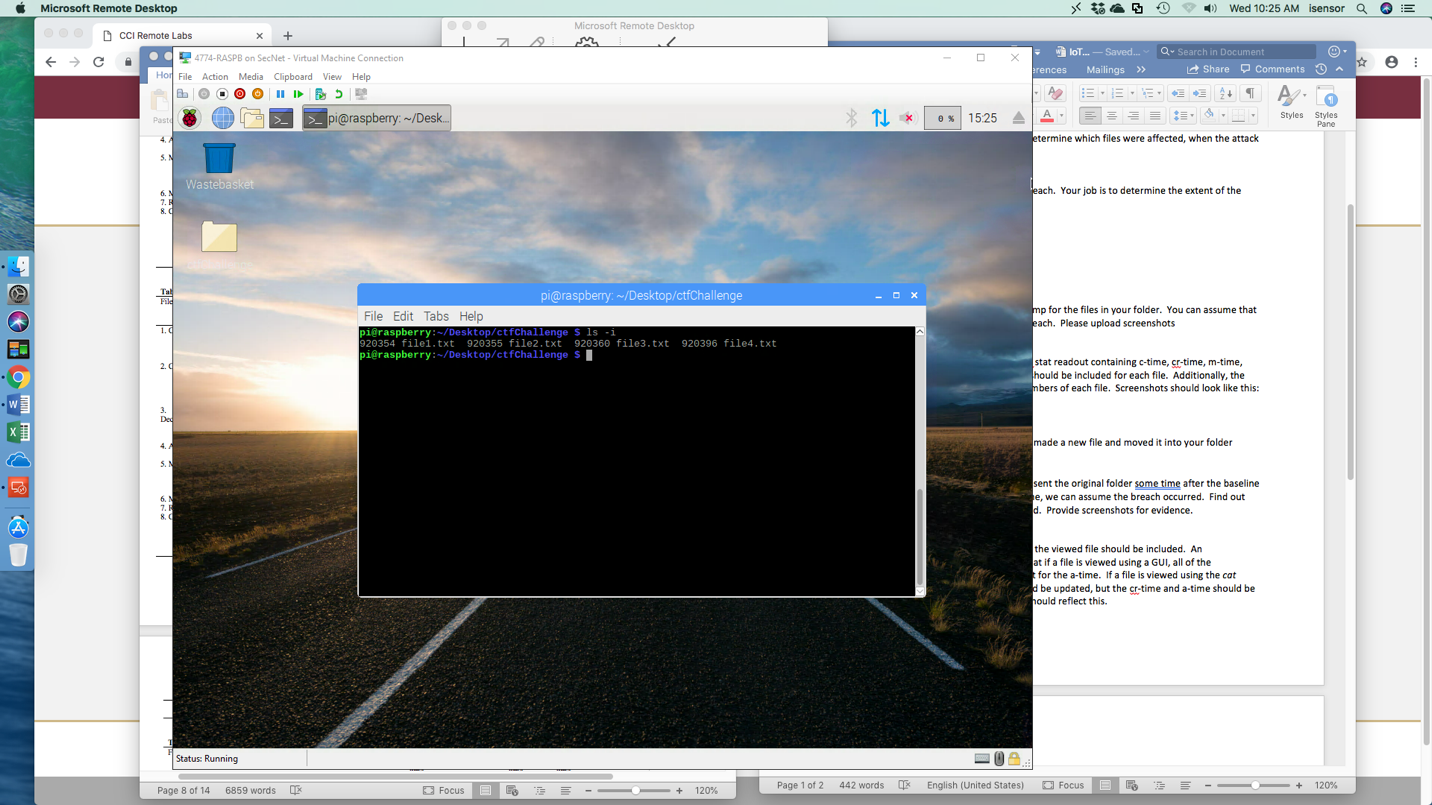
**Task 1**

**Description**

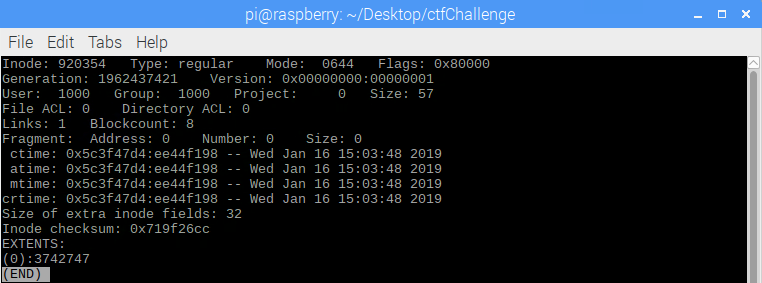
The first task is to create 3 files and collect baseline timestamps. You can assume that these timestamps were taken before the breach. Identify the **inode** of each file and upload screenshots of both the **inode** number and the metadata viewer.

**Expected Answer**

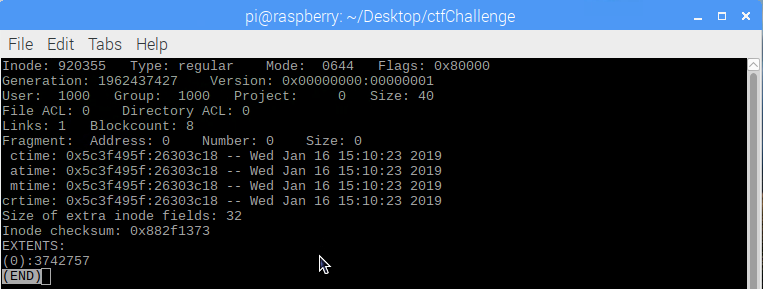
The timestamps should include the debugfs stat readout containing **c-time**, **cr-time**, **m-time**, and **a-time**. Separate timestamp readouts should be included for each file. Additionally, the user should run **ls -i** to display the **inode** numbers of each file.

**Screenshots should look like this:**

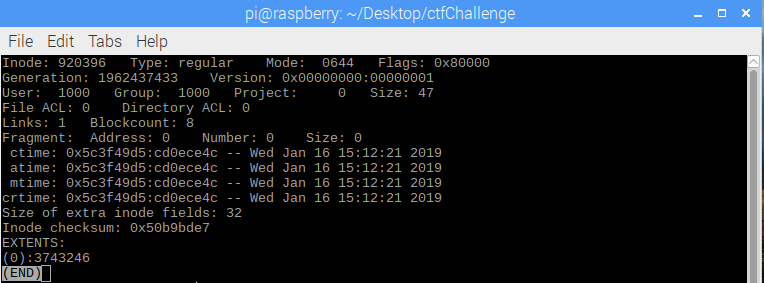
Initial run of **ls -i** to see the inode numbers on all files.



Base line run of **file1.txt**



Baseline timestamp of **file3.txt**



Baseline timestamps for **file4.txt**

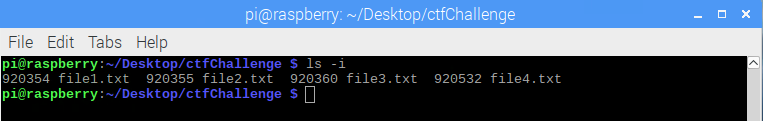
**Task 2**

**Description**

The baseline timestamps for a system you are in charge of are given. Since then, however, the system has been breached. In an effort to cover his tracks, the attacker made a new file and moved it into your folder under the original name. Find out which file has been deleted and replaced.

**Expected Answer:**

A screenshot of a run of ls -i should indicate that a file has been created under a different inode. An accompanying paragraph should indicate that a file’s inode number remains constant, even if the file is moved or deleted. A file with an identical name and pathway will not generate the same **inode** number.



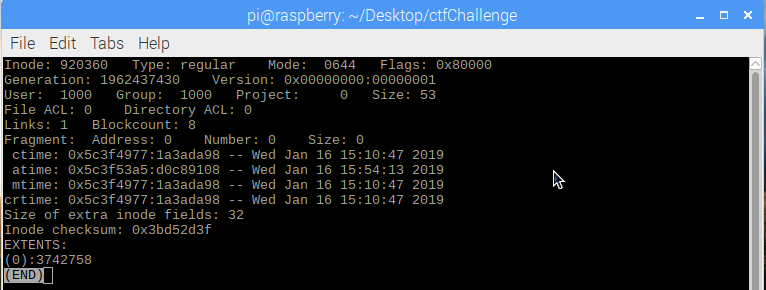
**Task 3**

**Description**

You discovered which file the attacker deleted and replaced, but you still suspect that the attacker has seen a file that he shouldn’t have, even if no changes were made. Find out which file has been viewed, but not modified. Provide screenshots for evidence.

**Expected Answer:**

A screenshot of the debugfs stat readout of the viewed file should be included. An accompanying paragraph should indicate that if a file is viewed using a GUI, all of the timestamps should remain the same, except for the a-time. If a file is viewed using the *cat* command, the c-time and the m-time should be updated, but the cr-time and a-time should be the same as the baseline. The screenshot should reflect this.



Here, only the a-time for **file3.txt** has changed. This indicates that the file has been viewed, but not modified.

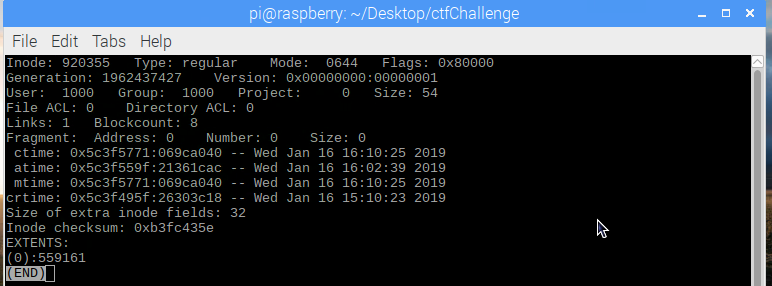
**Task 4**

**Description**

A different file within this folder has been modified without your permission. Determine which file was modified, and when the breach occurred.

**Expected Answer:**

A screenshot of the debugfs stat readout of the modified file should be included. An accompanying paragraph should indicate that modified files should have a different c-time, m-time, and a-time. The new timestamp reflects the time at which the attack occurred. The fact that the cr-time has not changed indicates that it is, in fact, the original file.



Here, the c-time and m-times are both modified, reflecting the time that the modifications were made. The a-time reflects when the file was opened, so the attacker would have had the file open from the time the a-time was updated to when the c-time and m-times were updated.