

An Investigation into Timestamps in NTFS

LK049 – Bachelor of Science in Cyber Security and IT Forensics

Project Interim Report

Nathaniel Teskey

20247672

Dr. Jacqueline Walker

27/10/2023

**Abstract**

Over the course of this report, I will present my current findings and plan of action for the future. I am undertaking an investigation into the complex world of timestamps in the Windows New Technology File System (NTFS). NTFS is the backbone of the most world’s most popular operating system, Windows. The process of hiding a trail using timeline forgery is now more prevalent than ever. A simple Google search will reveal lots of free, user friendly, timestamp altering tools that make it harder for digital forensics investigators to reconstruct a timeline of events that occurred on an electronic device. Once upon a time, timestamps were credible evidence that could be relied upon, but not anymore. Now investigators need to prove that the timestamp is legitimate by finding evidence supporting their claim. On the other hand criminals may use timestamp modification software to hide their steps, which make it difficult for investigators to follow the trail.

My interest in this project was piqued by the complex nature of this project which requires a deep understanding of the NTFS file system within Windows. I have always been a Windows user and I find it very interesting that everything I do on my laptop leaves a digital footprint. Although I did not know how important or intricate timestamps were before I started my research, I always had an interest in how they worked. The knowledge that I will gain from this project will undoubtably stick with me throughout my career and I love that this project has such a “real-world” application. Unfortunately, cybercrime is surging in popularity worldwide so, by basing my FYP on such a current topic I feel I will gain a deeper knowledge of digital forensics to kickstart my career. I would also love to work in this field someday with either An Garda Siochana or a private company.

My plan for my report is to investigate timestamp patterns when common user operations are executed. I will layout my findings in evidence supported by screenshots and diagrams. I will show how different legitimate operations on Windows can cause different and sometimes surprising changes to the different timestamp stores in the filesystem. I will then explain my opinion on why these different operations, which should in theory produce the same result, produce different results. I will then take a dive into the world of timestamp modification and “timestomping”. [1]

**Declaration**

This interim report is presented in part fulfilment of the requirements for the LK049 Bachelor of Science in Cyber Security and IT Forensics Project.

It is entirely my own work and has not been submitted to any other University or Higher Education Institution or for any other academic award within the University of Limerick.

Where there has been made use of work of other people it has been fully acknowledged and referenced.

**Name**  Nathaniel Teskey

**Signature A close up of a sign

Description automatically generated**

**Date**  27 October 2023

**Table of Contents**

**ABSTRACT** 3

**Declaration** 5

**Table of Contents** 7

**List of Figures** 9

**Chapter 1 INTRODUCTION TO TIMESTAMPS IN NTFS** 11

**Chapter 2 MY MOTIVATION IN CHOOSING THIS FYP** 13

**Chapter 3 THE HISTORY AND FUTURE OF NTFS** 15

**Chapter 4 The Master File Table in NTFS** 17

**Chapter 5 Timestamp Information Storage in NTFS** 21

**Chapter 6 PLAN FOr future work** 25

**References** 7

**List of Appendices** 29

**Appendix A: Project Gantt chart** 31

**List of Figures**

**Figure 1.** Structure of an MFT entry…………………………………………………………………………………… 17

**Figure 2.** How the boot sector is used to find $MFT…………………………………………………………… 18

**Figure 3.** Forming the file reference address………………………………………………………………..……. 19

**Figure 4.** $FN Timestamp rules from SANS……………………………………………………………………….… 22

**Figure 5**. $SI Timestamp rules from SANS………………………………………………………………………..…. 23

**Chapter 1: Introduction to Timestamps in NFTS**

Imagine a world where everything you do on your computer leaves a hidden digital mark, almost like faded footsteps when walking through a forest, where an investigator can follow these footsteps and find you. This is exactly what happens when using a computer running a Microsoft Windows operating system. At the core of the operating system lies the New Technology File System (NTFS). [2] Thanks to NTFS, these footsteps are known as timestamps. Timestamps are a set of digital markers that record metadata about operations on files. [2] These timestamps are hidden deep within the file system in multiple locations. Investigators can compare differences in the values stored in these different locations to see if these pieces of metadata are legitimate or have been artificially altered. What makes NTFS so effective is that it records everything as a file. The user’s engagement with these files is then recorded as timestamps.

Attackers can and do alter these timestamps in order to obscure their activities. They try to cover their traces in order to make it as difficult as possible for digital forensics experts to find evidence on their activities surrounding an illegal incident. As I said previously, it is now easier than ever for these bad actors to modify timestamps and try to hide their trail using timestamp manipulation tools that can be widely found only. [3] These tools are becoming easier and easier to use with the help of simple language and intuitive graphical user interfaces (GUI).

In order to find discrepancies in timestamps we need to be able to understand the intricacies in how timestamps interact with each other and how they are created and stored. My investigation focuses on timestamps in Windows operating systems based on the underlying filesystem known as NTFS. There is limited information on this topic as Microsoft do not release comprehensive documentation on their process of recording, storing or modifying timestamps – presumably due to maintaining a high-level of security. There are some helpful pieces of documentation online however from my research so far [2], I have found that much of this information is now outdated as NTFS behaves differently in some more recent versions of Windows [4], as the timestamping rules have been updated. Due to this I will conduct my own experiments on a Windows 11 virtual machine to display the current behaviour of timestamps. This should help me figure out the current rules that are in place. What we can rely on is that Microsoft documentation [5] tells us that timestamps are changed when applications create, access, write to or modify files in a number of different ways. This is normally done through an application, for example this Word document will have its “modified” timestamp updated when I save the file. But what if I close the application with the red “X” in the top right corner, will this produce the same modification of the timestamp? These are all interesting factors which I will explore in my investigation. Another question is, after performing various operations on different filetypes (.exe, .txt, .mp3, .mp4, .pdf, .pptx), will the timestamps update differently? I will explore these questions over the course of my FYP, so stay tuned for the answers in a couple of months! I am just as excited as you to find out.

**Chapter 2: My Motivation in Choosing this FYP**

Even before I began studying Cybersecurity and IT Forensics at the University of Limerick, I have had keen interest in these fields. Since I was in fifth year at school, I have wanted to pursue a career in cybersecurity and digital forensics. The process and the challenge of retracing the steps of criminals and piecing together small pieces of information in order to build a bigger picture of the threat is very interesting. I enjoy the challenge; however hard it may be. I find that the greater the challenge, the greater the reward at the end. When the time came for me to pick my top 10 final year project ideas, I knew immediately that this was the one I wanted to pursue. I contemplated the others as good practice, however I kept returning to this one. I find the subject really fascinating especially in the past few years as cybersecurity is at the forefront of everyone’s minds. This is probably due to the HSE ransomware attack which occurred on the 14th of May 2021 [6]. This highlighted the importance of good cybersecurity procedures in not only the eye of corporations and government organisations but also in the eyes of the general public. In the future after I finish college I would love to be on the team of the “good guys” helping to fight the cyber-attacks.

This project perfectly aligns with my goals and future career ideas. In terms of cybersecurity, a knowledge of filesystems is of utmost importance, and what better way to learn about it that undertaking a project on timestamps in NTFS. NFTS is at the root of the most popular operating system in the world – Microsoft Windows. My thinking is that in order to further my skills in cybersecurity and digital forensics, starting at the very bottom, in this case inside the computer’s hard disk, then inside the clusters of the drive, and then again inside the sectors [7]. Inside these sectors I am looking at the tiny pieces that make up the filesystem in the MFT entries [7]. Then I dived into these MFT entries and study the attributes within them. At this level I am dealing with bytes, one byte is 8 bits. In my opinion taking this route to building my knowledge on digital forensics is one of the best routes, as I am literally starting at the very bottom of the computer’s architecture. As the project progresses, I would like to see how the individual bits have an impact on the NTFS timestamps, however I have not gotten to that level yet.

Another aspect of the project that interests me is that in court timestamps are immutable witnesses. They record the exact sequence of events that has occurred. The power of these tiny pieces of information really amazes me. My job as a digital forensics’ investigator is to prove that these timestamps are legitimate. Through my experiments I will outline which timestamps are legitimate by determining the expected behaviour of legitimate and illegitimate timestamps. I find this really cool as when these timestamps are proven, they serve as undeniable evidence as to the sequence of events that occurred on the device or in this case the Windows computer.

Knowledge of these timestamps does not only trace criminals’ tracks but it can also find patterns in use in a corporate environment, allowing us to plot graphs and make productivity optimisations to files that are accessed regularly. This can also help us to identify threats early, if based on these patterns we detect an abnormal sequence of timestamp altering events.

These tiny pieces of information are incredibly powerful. The fact that they can hold their precision down to 100 nanoseconds in incredible in itself and allows for astonishing accuracy.

Every timestamp captures specific events and from the creation of a file to the present moment, timestamps offer a defined map to trace back these steps. The use of anti-forensics tools serves as a spanner in the works almost, however this adds yet another challenge to overcome which makes it all the more interesting. The fact that when presented in court they serve as reliable witnesses just showcases the use even more, and as a digital forensics investigator to prove that they are genuine before presenting them in court. NTFS timestamps are key to unlocking the true sequence of events on a Windows computer, they transform data into data that humans can understand into a plottable real life story of events.

**Chapter 3: The History and Future of NTFS**

The New Technologies File System (NTFS) was launched in 1993 alongside the release of an update to the Windows NT (New Technology) operating system [7]. This was known as Windows NT 3.1. [7] NTFS replaced the previous files system used in Windows, known as the File Allocation Table (FAT). NTFS is a much more complex file system than FAT, as it has many more features and is also very scalable for larger storage devices [2]. This was a much-needed improvement as FAT was created in 1977 and so much had changed between 1977 and the advent of NTFS. FAT is more suited to devices with smaller storage capacity such as removable storage devices, cameras and smart TVs among other portable devices. NTFS of course can handle larger storage devices as it has a maximum partition of around 2TB [2]. This is essential in modern computers, as they have more storage than was ever imaginable on personal computers in the 1970s.

Some of the mystery around NTFS stems from the fact that Microsoft has only published very limited information regarding its on-disk layout. They have published some high-level details of the filesystem however low-level details have not been published [2]. This is essential to maintain a high level of security within the filesystem. To this day NTFS is the standard operating system used in Windows operating systems, although it has received various updates over the years, which added features and enhanced security.

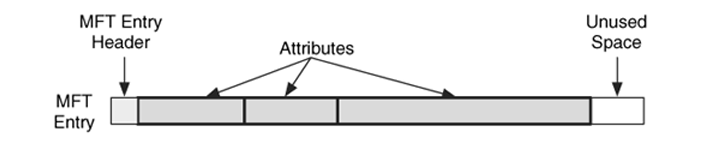
In 1995 Windows updated NTFS to support compressed files, named streams and they also added access control lists. In October 2001, another key update came, this allowed for the expansion of the Master File Tables entries with redundant MFT file record numbers. This was essential to allow for the recovery of damaged MFT files. Based on my research so far this seems to be the last time NTFS has received a large update [7] [2].

However, with the release of Windows 11 in October 2021, it seems that Microsoft has added support for a new filesystem called the Resilient File System (ReFS) [8]. This new filesystem is currently used in Windows Servers, however unlike NTFS, ReFS is much better at data availability and scalability. This may be a hint that Microsoft plans to ditch NTFS in the future, but who knows. NTFS supports a maximum file size of 256TB; however, ReFS supports up to 35PetaBytes (1PB is 1024TB!) [8]. This is probably one of the reasons why they are currently using ReFS on Windows servers, as they can have much larger storage sizes in comparison to personal computers.

In my opinion NTFS is here to stay for at least another couple of years as ReFS is still in the development stage. It does not yet have support for compression, encryption, disk quotas or removable media [8] – NTFS on the other hand is well equipped to deal with these challenges.

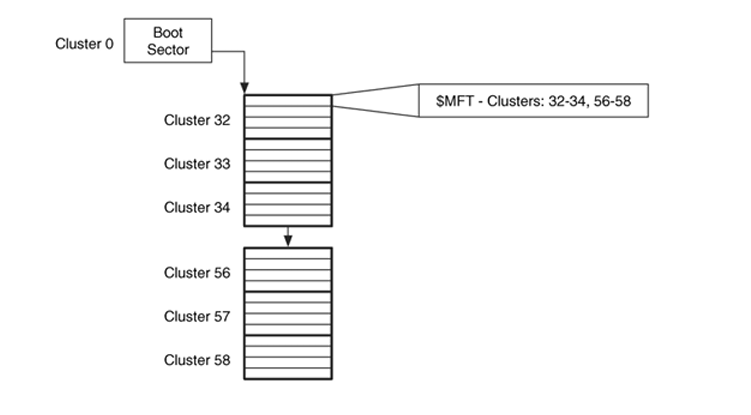
**Chapter 4:** **The Master File Table in NTFS**

At the heart of NTFS is the Master File Table (MFT). The MFT contains information regarding all files and directories, therefore it is the most important file in an NFTS file system [7]. Each file and directory have at least one entry in the MFT. The entries into this MFT are very simple. Microsoft calls each entry a “file record” [7]. These file records or entries are 1KB (1024 bytes) in size, however only the first 42 bytes have a defined purpose [7]. The remaining bytes are used to store attributes [2]. These attributes all have specific essential purposes, such as storing the files name, storing the files content and so on [7].



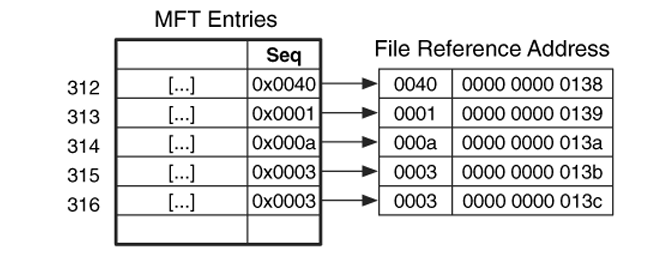
**Figure 1. Structure of an MFT entry (file record) with a small header, 3 attributes and some unused space at the end.** [7]

Like everything in NTFS, the MFT is a file and what is really interesting is that the first entry in the MFT is named $MFT which is a record of the location of itself on the disk [2]. The next 15 entries in the MFT are reserved for other NTFS system files such as $MFT Mirror, $Volume, $Bitmap and $LogFile. These files are known as the file system metadata files [2]. The starting location of the MFT is defined in the boot sector. The boot sector is always located in the first sector of the filesystem [7]. As seen below the boot sector is used to find the first MFT entry ($MFT). Here we can see that the MFT is fragmented and goes from clusters 32 to 34 and 56 to 58 [7]. Similar to FAT, NTFS uses clusters too. Clusters are groups of consecutive sectors.



**Figure 2. How the boot sector is used to find $MFT and therefor determines the layout of the MFT** [7]

Each MFT entry is defined by a sequential 48-bit value where the first entry has an address of 0 [7]. Microsoft calls this address the “file number” [5]. The maximum MFT address updates as the table grows. NTFS uses file reference addresses to refer the MFT entry’s location. All of the MFT entries have a 16-bit sequence number that is incremented when the entry is allocated [7]. We can say that MFT entry 123 has a sequence number of 1, if file 123 is deleted the MFT entry is then allocated to a new file. However, the new file will have a sequence number of 2, as it is the second file to exist at this address. This MFT entry and sequence number are combined, holding the sequence number in the upper 16 bits which then forms the 64-bit file reference address [7] [4].



**Figure 3. MFT entry address and sequence number combining to form the file reference address** [7]

NTFS uses the file reference address to refer to MFT entries as the sequence number allows it to better determine when the file system is corrupted such as, if the system crashes when certain data structures for a file are in the allocation phase, the sequence number can tell NTFS whether a data structure contains an MFT entry address because the previous file used it or because it is part of the newly created file. This is interesting as we can also use it to recover deleted content by checking if there is an unallocated data structure with a file reference number inside it. We can then check if the MFT entry has been moved, since the data structure we are looking at used it [7]. We can view the MFT using an application called “The Sleuth Kit” which runs in the command line or we can use its GUI version called “Autopsy” [7].

Now that you have knowledge about the background of NTFS and its MFT components such as attributes, we will tie this into timestamps in NTFS. The attributes located in the MFT entries associated with NTFS timestamps are found in the $STANDARD\_INFORMATION and $FILENAME attributes. [1] [7] When we use an application such as The Sleuth Kit or autopsy to display the “hidden” information regarding the filesystem such as viewing the Master File Table, we can see the entries and within these entries we can view the attributes. For my investigation I am most interested in the timestamp attributes of course. The 3 attributes mentioned above each store 4 timestamps which are modified, accessed, inode changed and born [7]. However, the $FILE\_NAME attribute can also have an extra four timestamps as it stores the short filename and also the long filename [2]. This depends on if the file has a long name or short name. If it is long then it will store both sets of timestamps [7]. We can compare these attributes to determine whether timestamps are genuine or have been tampered with. This is possible as certain operations will alter certain timestamps and not others. For example copying a file in File Explorer will update the $STANDARD\_INFORMATION and $FILENAME attributes, however if we are modifying timestamps artificially then we can only modify the $STANDARD\_INFORMATION attribute as $FILENAME can only be modified by the Windows kernel [2].

**Chapter 5:** **Timestamp Information Storage in NTFS**

Now that you understand the Master File Tables entries and attributes, from the previous chapter, we can look at the attributes that are specific to timestamps in more detail. The two main attributes in an MFT entry that store timestamp information are $STANDARD\_INFORMATION ($SI) and $FILENAME ($FN). The process that Microsoft chose to record the passage of time in the timestamps is by counting the number of nanoseconds that have passed since the 1st of January 1601 [2]. It counts in 100 nanosecond intervals and stores the result as a 64-bit value within the attribute. The reason the 1st of January 1601 was chosen was because it was the first day of the 400 year “leap year” cycle in the Gregorian calendar, which was in effect when windows NT was released [7], accompanied by NTFS. Now we know that timestamps are calculated to very high accuracy. They are always represented in Coordinated Universal Time (UTC), so I will ensure to apply the correct conversion to GMT when undertaking the experiments in this project.

The timestamps stored in $SI and $FN are called MACB timestamps because they hold the time values for the following events: [7] [2]

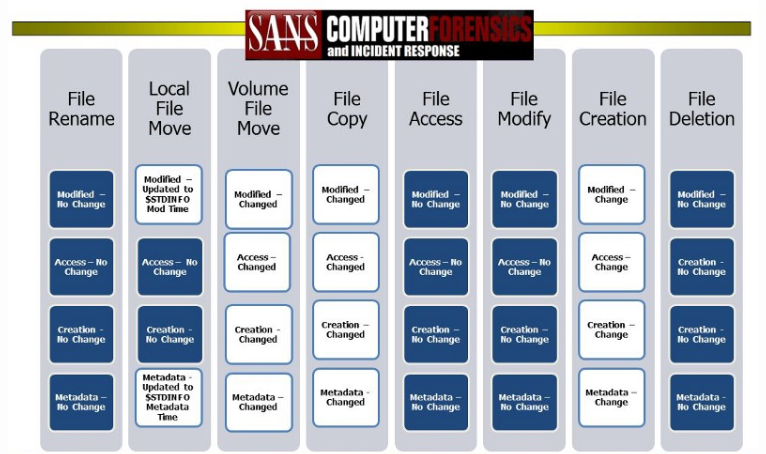
**Modified:** last update to content of the file

**Accessed:** last access to content of the file

**Changed:** $MFT Modified time (file metadata change)

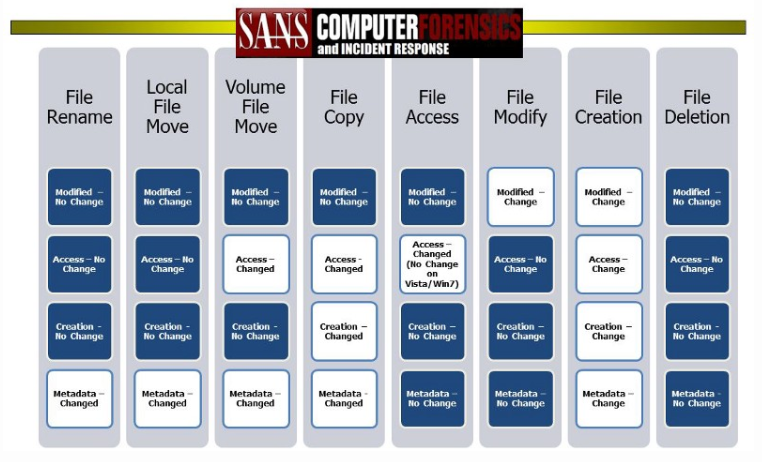
**Birth:** File Creation Time (MFT Entry Created)

As I mentioned earlier, $FILE\_NAME can sometimes have two sets of timestamps stored depending on the length of the filename. A short file name could be “nathan.txt” – this will only have one set of timestamps associated with it in $FN. However, a longer filename such as “nathanielpatrickteskeysfypintirimreport.txt” will have two sets of timestamps associated with it, both holding the same time records in $FN. The key takeaway from $FILE\_NAME timestamps is that they can only be modified by the Windows kernel. Therefore, there are no known anti-forensics utilities that can spoof this attributes timestamps [2]. I find this very interesting as this attribute will always serve as a comparison when comparing other timestamps to check their authenticity, such as comparing $FN timestamps with $SI timestamps.



**Figure 4. $FN Timestamp rules from SANS**

Each of these files mentioned above will also have a set of timestamps record stored in $STANDARD\_INFORMATION. On the other hand, $SI can be modified by any user level process such as the application called “Timestomp” which is a tool for modifying timestamps on Windows [1].



**Figure 5. $SI Timestamp rules from SANS**

I am sure you are familiar with the timestamps within the “properties” box on Windows when right clicking a file. The timestamps presented here come from the attribute $SI in the MFT Entry for the file. Referring to MACB timestamps, like above, the “C” – change in metadata in the $MFT attribute is not normally visible to the user [7]. The timestamps stored in $FN do not always store the same values that are visible in the $SI timestamps [2]. This makes our work as investigators more difficult, but not impossible.

The reason for this is that if a file say “nathan.txt” is created from scratch on my laptop then the timestamps present in $SI and $FN will store the same values. However, imagine for example I downloaded the “nathan.txt” file from the internet, therefore not creating it from scratch on my laptop, the timestamps stored in the $SI and $FN attributes will not correspond.

This is because the $FN timestamps (the ones that can only be modified by the Windows kernel) cannot be directly updated by the user, whereas $SI is updated by any operation performed on the file. $FN timestamps are updated by the operating system (Windows), with the values that it stores in $SI when there are changes in the $FN attribute in the MFT entry for the file. Changes occur in the MFT entries $FN attribute when creating, copying, moving or renaming a file or directory [7].

It is much harder to alter $FN timestamps due to the fact that they can only be changed by the Windows kernel, unlike the $SI timestamps, as there is a lack of support from the Windows API [2]. This is down to the fact that the Windows API has support for changing the $STANDARD\_INFORMATION timestamps built into it; however no such support exists for tampering with the $FILE\_NAME timestamps.

**Chapter 6: Plan for Future Work**

Over the next few months, I intend on undertaking the following tasks to add to my project:

**Create a testing environment VM** on the latest stable release of Windows 11 with a small disk size (possibly 30/40GB) to allow tools to run quickly as it takes about 12 hours on my personal laptop with a disk size of ~476GB. This will allow me to have a clean testing environment to ensure the clarity of my results.

**I intend on becoming proficient in filesystem tools** such as The Sleuth Kit, Autopsy, FTK Imager and Timestomp. I also want to be able to comfortably operate the SANS Sift Workstation, as it is packed with tools which help accelerate my learning on digital forensics. I already have the tools listed above installed, however I am just beginning to figure out how to use them.

**I will then begin the correct process of extracting the timestamps** along with their cryptographic hashes to ensure that they have not been modified by me since they were extracted. This is good practice as it proves that the timestamps are exactly what was extracted in the first place and will ensure that my tools have not modified them. If I have time I will then attempt live and dead analysis of timestamps. I am not sure how to do this yet or if it will produce different results, however I will still demonstrate it in my final report.

**I will use tools to alter the timestamps artificially** and compare them with legitimate timestamps in order to determine how the latest version of Windows 11 deals with timestamps and I will compare my findings with that of previous versions of Windows in case NTFS policies have changed recently. I will also form my own opinions on how different filetypes or different operations on the computer produce different timestamp outcomes. I will then present my findings and highlight what I think are the current rules in NTFS for recording and updating timestamps.

As I undertake the above action plan, it may increase based on what I discover or encounter over the course of my investigation. I will ensure that everything is accurate and explain the results in my final report to the best of my ability.

**References**

[1] R. Dumont, "Indicator Removal: Timestamp," Mitre, 29 March 2020. [Online]. Available: https://attack.mitre.org/techniques/T1070/006/. [Accessed 26 October 2023].

[2] "NTFS.com," LSoft Technologies, 17 January 2023. [Online]. Available: https://ntfs.com/ntfs\_basics.htm. [Accessed 26 October 2023].

[3] "Kroll," Kroll LLC, 16 January 2023. [Online]. Available: https://kroll.com/en/insights/publications/cyber/anti-forensic-tactics. [Accessed 26 October 2023].

[4] "Micronics Technologies India," MTIPL, 20 February 2023. [Online]. Available: https://micronicsindia.com/articles/an-in-depth-comparison-of-ntfs-across-all-windows-versions-what-has-changes-and-what-remains-the-same/. [Accessed 26 October 2023].

[5] "Microsoft Learn," Microsoft, 12 September 2023. [Online]. Available: https://learn.microsoft.com/en-us/windows-server/storage/file-server/ntfs-overview. [Accessed 26 October 2023].

[6] "HSE," HSE.ie, 16 May 2021. [Online]. Available: https://www2.hse.ie/services/cyber-attack/what-happened/. [Accessed 25 October 2023].

[7] B. Carrier, File System Forensic Analysis, Boston, MA: Addison-Wesley Education, 2005.

[8] "Microsoft Learn," Microsoft, 15 February 2023. [Online]. Available: https://learn.microsoft.com/en-us/windows-server/storage/refs/refs-overview. [Accessed 23 October 2023].

**Appendices**

**Appendix A:** Project Action Plan in Gantt Chart ………………………………………………………………… **1**

**Appendix A: Project Action Plan in Gantt Chart**

