COMP7024- Operating Systems Security and Development

Coursework- Part 1: Security of an OS- File System Security

WindowsOS is operating on global level which means that from local laptop system to large corporation networks take advantage of the power that it brings. This establishes the need of organising and controlling the access to different users when it comes to a networkable usage. Defining operational access to commands such as: Read, Write, Delete, Append, is a must in order for the administrator to keep track on how the files are used by the users and not to have a user with access that is not needed for them. On later stages, when an issue occurs, the administrator could pull audit findings to track and monitor the behavior of given user or in general. This is especially important when it comes to legal documents and financial transactions. Furthermore, backing up and secure removal of data should be ensured as if the user deletes by mistake documents, the file system should be able to recover the documents. However, if the user really wants to remove the file from the software, then the system should ensure that the data is erased and cannot be recovered. In case of an attack on the system or the erasing process, the file system should have put in place an encryption mechanism to prevent malicious activities to all files and folders written on the software.

To demonstrate how WindowsOS implements security features listed above, I will begin with the core, thus organising the files. Move up to structuring the access controls and controlling users’ activity. Will further deep dive into encryption of the system and data. Auditing the logs, would be the final touch where we can extract data for the users and the actions taken for audit purpose and analysis.

1. Organising files

Originally files in the system are orchanised in a hierarchical way with the help of NTFS which first was implemented back in 1993 with the Windows NT 3.1 [1]. However, the newest technology that Microsoft has introduced almost 10 years ago (ReFS), is offering greater data integrity and resilience, designed to handle large sum of files and work with them efficiently. Using B+ trees algorithm to structure all metadata and file data, which is further pushed to a database table. All components are limited to 64-bit number therefore the maximum file size is 35 petabytes in comparison to NTFS’s 256 terabytes. [2] The Resilient File System was first added to Windows Server 2012 [3] and the performance is taken out of the roof. Reason being the core concept of dividing the volume into tiers where each tier has own drive and resilience types, further allowing each tier to make the decision to optimize either the capacity or the performance of the operations. [4] This is pre-configured so for example given operations such as ‘write’ is operated by one tier (performance tier) and the left-overs will be moved to the other tier in real time (capacity tier).

1. Structuring and Controlling accesses

Once the structure of the files and the network of devices/accounts is done, next step is to consider accesses and having the ability of control over. WindowsOS uses ACL (access-control-list) which outlines the accesses given to the user by defining specific rules. [5] This technique is tested and implemented firstly back in 1984 by Multics OS. Nowadays, almost all operating systems take advantage of the idea. WindowsOS uses the list by creating entries inside based on their category. Two types of lists exist DACL (Discretionary ACL) and SACL (System ACL) , one describes the access policy and the other the auditing policy of the security descriptors respectively [6][7]. The list is created and begins with zero entries, meaning that the targeted object will not let any user as the user is not presented in the ACL. Most common ACE is to grant world access to the object by using SID (SeWorldSid). For each new entry:   
call BuildExplicitAccessWithName function, grant the relevant EXPLICIT\_ACCESS that describes the needed access, select the ACL and select the array of EXPLICIT\_ACCESS created in the previous step to add those accesses to the list. [8]

Furthermore, once the access are granted WindowsOS implements UAC (User Account Control) which is a feature that warns not authorised operations to the system and files. The software presents to the user or the administrator of the system that a permission is needed before allowing to perform commands. [9] Firstly introduced in the Windows Vista and Server 2008, just like any other software, it did get updates for better performance and flexibility. [10]

1. Encryption

The purpose of the report is to analyses the security features of the WindowsOS and where to go without talking about encryption. Encryption in the operating system is controlled and monitored by the default software BitLocker firstly implemented in 2007 with the Windows Vista and continues used in the newest Windows 2011.[11] In both file systems NTFS and ReFS. However, BitLocker is a general mechanism for the entire operating system to prevent from offline attacks on the drive. In order to provide extra level of security on individual components of the system, for example the file system and individual files and directories, Microsoft introduced EFS (Encrypting File System) together with the NTFS file system back in 1993 where encryption of components is done using a public-key stream. EFS is available in all versions of Windows from Windows 2000 onwards. On the other hand, the system is currently only available with the NTFS and not implemented in ReFS. [12]

1. Audit

Current NTFS provides auditing capabilities which allow admins to track and monitor logs for file and folder operations performed by the user. The admin should be present in the SACL as a entry with the relevant EXPLICIT\_ACCESS specified for the relevant object. The logs provide simple information such as if the file was requested/closed/deleted/changed based on event codes 4656/4658/4660/4985 respectively. [13][14]

To compare the features that WindowsOS implements against Linux, there will be many common facts. Both programs implement ACLs to control the access to files and directories, but on Windows is required deeper configuration to use them effectively, whereas on Linux the ACL is more flexible due to ability of kernel modifications. Both systems use deeper encryption mechanisms, Linux’s dm-crypt subsystem build in the kernel provides strong and effective encryption on individual files as the entire file system, whereas Windows uses BitLocker which needs to be installed and configured accurately to perform checks and actions.

Key differences between the two operating systems are the anti-virus focus and security improvements. Well known fact is that Linux installs programs safer which limits the probability of getting a malware. On Windows and Mac OS mainly the user downloads files such .exe which require administrational privileges whereas in Linux the files are from trusted repositories, therefore the chance is much lower. On top of that, Linux is limiting the malware to basic functionality due to the fundamental structure of Linux. Furthermore, Linux is public operating system which means that it is open-source and anyone from the community has the change to investigate and review vulnerabilities, if any. This brings extra level of security for Linux and its performance.

Developers need to compile with regulations and policy of the operating system and also the computability of the software on other file systems as different security features and requirements exists. The way the application is written is important because most applications use more than one mid-tier and backend which require accesses to different resources, therefore the application developer must ensure that no access breaches are present when the user interacts with the application. Added functionality is needed to be coded when the user doesn’t have sufficient privilege. On top of that, developer must ensure that they write code that generates logs for the auditing aspect. Generate log when certain action is taken by the user and so on. Furthermore, a developer should ensure that there is a functionality that performs checks when the resources and output is being tempered. This requires further checks and testing on the final product for the test acceptance.

In conclusion the security of the file system itself and individual files is extremely important to prevent fraudulent activities. I outlined major security features that a system should have and evaluated how WindowsOS is coping with the target. Furthermore, compared the operating system to an alternative but quite different in core system and distinguished possible vulnerabilities. In my opinion, the security on the file system could be improved when it comes down to single file encryption and changing the methods and algorithms from time to time. I think that the public information is quite a bit and in order to prevent malicious attacks, the company should enforce internal policy. Furthermore, in terms of managing and granting accesses to users in a networkable environment, I think that implementing the idea of roles of users could be beneficial in the process of granting/auditing/controlling. For example, administrative executive joins the network, then he/she have separate disk space in the network where all employees on the same position have only access to the same projects and resources, therefore when there is a new joiner, the administration should only add the role to this user in the ACL and the role itself contains the need array of accesses.

References

[1] [NTFS - Wikipedia](https://en.wikipedia.org/wiki/NTFS#:~:text=New%20Technology%20File%20System%20(NTFS,Linux%20and%20BSD%20as%20well.) – NTFS was introduced back in 1993 with Windows NT 3.1

[2] [Resilient File System (ReFS) overview | Microsoft Learn](https://learn.microsoft.com/en-us/windows-server/storage/refs/refs-overview) – Limits

[3] [Resilient File System (ReFS) overview | Microsoft Learn](https://learn.microsoft.com/en-us/windows-server/storage/refs/refs-overview) – ReFS first implemented with Windows Server 2012

[4] [Resilient File System (ReFS) overview | Microsoft Learn](https://learn.microsoft.com/en-us/windows-server/storage/refs/refs-overview) – Performance

[5] [Security descriptors in file systems - Windows drivers | Microsoft Learn](https://learn.microsoft.com/en-us/windows-hardware/drivers/ifs/security-descriptors) – Use of ACLs and ACEs

[6] [Security descriptors in file systems - Windows drivers | Microsoft Learn](https://learn.microsoft.com/en-us/windows-hardware/drivers/ifs/security-descriptors) – List Types: DACL and CACL

[7] [Access Control List - Windows drivers | Microsoft Learn](https://learn.microsoft.com/en-us/windows-hardware/drivers/ifs/access-control-list) – Deep dive in methodology

[8] [Creating or Modifying an ACL - Win32 apps | Microsoft Learn](https://learn.microsoft.com/en-us/windows/win32/secauthz/creating-or-modifying-an-acl) – ACL steps

[9] [User Account Control (Windows) | Microsoft Learn](https://learn.microsoft.com/en-us/windows/security/identity-protection/user-account-control/user-account-control-overview) – UAC concept

[10] [User Account Control - Wikipedia](https://en.wikipedia.org/wiki/User_Account_Control) – User Access Control – dates

[11] [BitLocker - Wikipedia](https://en.wikipedia.org/wiki/BitLocker) – BitLocker – dates

[12] [Resilient File System (ReFS) overview | Microsoft Learn](https://learn.microsoft.com/en-us/windows-server/storage/refs/refs-overview) – The following features are unavailable on ReFS at this time

[13] [Basic audit directory service access (Windows 10) | Microsoft Learn](https://learn.microsoft.com/en-us/windows/security/threat-protection/auditing/basic-audit-directory-service-access) – Auditing file and folders

[14] [Audit File System (Windows 10) | Microsoft Learn](https://learn.microsoft.com/en-us/windows/security/threat-protection/auditing/audit-file-system) – File audit logs

Appendix

Version Control : <https://github.com/x17ed64life/COMP7024>