Use of Access Control to Minimise Ransomware Impact

Simon Parkinson, University of Huddersfield

Abstract: A ransomware attack is potentially highly damaging for businesses of all size, and due to being financially motivated, is increasing in occurrence. Although continuous innovations in ransomware complexity make it hard to defend against, simple safeguards are often ignored. The safeguard discussed in this article is that of correctly implementing and auditing file system access control to limit locations where ransomware can encrypt data should it execute under the user's credentials. A discussion of the mechanisms within Microsoft's New Technology File System (NTFS) that create the potential to over allocate permissions is provided. Following this, steps are presented to derive a clear policy to assist in ransomware protection.

Introduction

The potential for financial gain has resulted in the establishment of a multi-billion Dollar ransomware industry, which is founded on exploitation [1]. Those with weak, unprotected systems, as well as those with little security-specific knowledge are more likely to fall victim. Such users may not be able to adequately assess the potential risks and maybe caught unaware [2]. Those with the most to lose, for example a user, whom is heavily reliant on their IT system and its data for undertaking their daily business, heighten the potential for exploitation.

Ransomware, as with all businesses aiming to maximise profit, is continuously increasing in innovation, making it harder to detect and in some cases virtually impossible to rectify without paying the ransom fee [3]. There are a wealth of Ransomware systems in operation, each with a slightly different way of acquiring execution on the host PC, as well as different mechanisms of encryption, but a commonality amongst all variations is that they aim to encrypt data stored on both local and remote directory structures. Some Ransomware performs the encryption process in an intensive short duration, whereas some mechanisms encrypt data slowly with the aim of ensuring the encrypted files are propagated through any back-up mechanisms, maximising damage and aiming to make recovery even more challenging.

There are many potential aftermarket solutions aiming to help protect a system from ransomware; however, in this article we discuss how simply maximising the use of built-in system security controls can have a significant impact on preventing ransomware from acquiring execution on the host computer and encrypting data. It is worth noting that implementing strong access controls is not the definitive mechanism to prevent against a ransomware attack, but correctly restricting file system access can provide a strong first line of defence, and additionally contribute to improving the overall system security.

How damage is done

Let's consider the encryption phase within the ransomware cycle [4]. In its primitive form, ransomware works in the same was as any other application interacting with the underlying file system. The traditional file use cycle, similar to that a user goes through when editing a text document, is that files are opened, modified, and stored.

However, the fundamental difference is that ransomware is malicious in that it aims to create an encrypted version of the file and overwrite the original.

The host operating system employs mediation mechanisms (known as the reference monitor [5]) to control who can access and modify files. A fundamental principle of reference monitor is that it is enforced and that it is not possible to interact with a file without the reference monitoring granting permission. Figure 1 illustrates how the reference monitoring works, as well as introducing some of the common technical phrases. The subject is the user and/or process interacting with the system, the object is the resource which the subject is requesting access to, and the configuration is the level of access they require.

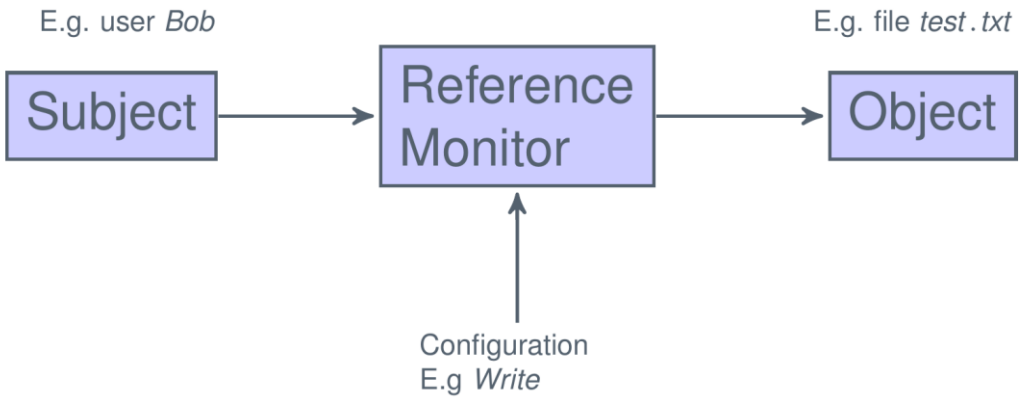

Figure 1: Reference Monitor

Here lies a dilemma in that ransomware is a malicious piece of software, but the reference monitor is authorising permission for it to access, modify and delete files. Why? Because that ransomware has determined a way to execute and could be achieved through one of the three following ways: The first is by modifying the reference monitor (and operating system) through installing malicious software, the second is by achieving privilege escalation and operating under the maximum permission level, and the third is by simply executing under the user's credentials and relies on the user having a high level of permission. The first two mechanisms can be hard to protection against and largely require strong security mechanisms embedded in operating system, as well as a culture of awareness amongst the users to prevent executing malicious software. In this paper, we consider the assignment and evaluation of file system permissions to minimise damage should a ransomware attack arise from executing under the user's credentials and therefore acquiring their permission.

The reference monitor is simply following the system policy, which is either set by the user, administrator or is the system default. A policy is the security restrictions enforced within the system, and in the particular context of this article, refers to both user and file system permissions.

Your access control policy

In terms of file system access control, the policy is the set of permissions granted to a specific user. If we consider the Microsoft environment, a policy could be that user

“Bob” has “Full Control” on a file system resource such as “proposal.docx” that has been set through the Microsoft “Security” tab located in the “Properties” of a file or directory. Figure 2 shows the standard Microsoft interface for managing these permissions, and it is here where Bob’s permission can be set and evaluated.

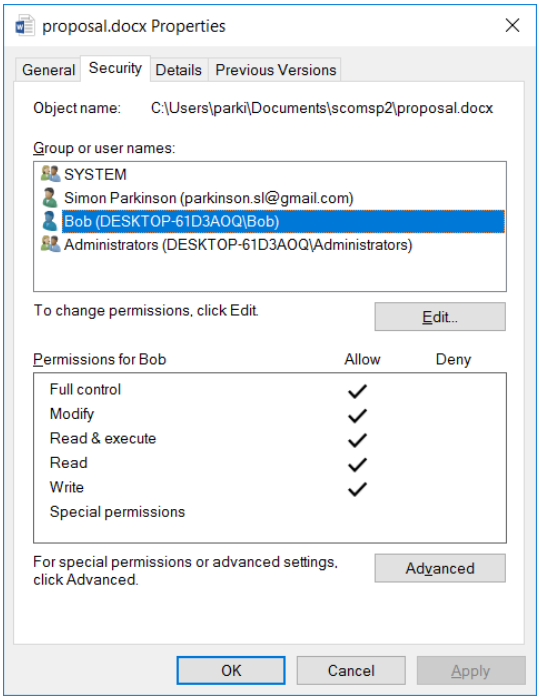

Figure 2: “Bob” has “FullControl” on proposal.docx

Although in the presented example identifying Bob’s permission is a trivial process, there are many complicating factors that can inadvertently result in the over allocation of permissions, as well as difficulty in permission evaluation. In this section, we provide an overview of these complexities, demystifying access control policies for both standalone single-user PCs to networked multi-user environments. In latter sections we then discuss some ‘easy steps’ that can be taken to ensure that user permissions are correctly set to minimise the potential impact of such ransomware if an attack should occur.

1) Group Membership

A fundamental aspect of access control within the file system access control is that of group membership. A subject (group, user or process) that interacts with the file system can be a member of any group. This means that permissions can be inherited from any of the associated groups if they are entered within any Access Control List (ACL). Figure 2 illustrates an example ACL. Subjects, in this case users, will often be grouped together (separation of duty) to make management easier. However, understanding implemented permissions can become significantly more complex by group association [6]. To correctly evaluate a user's effective permissions you would have to know which groups they are a member of. Group allocation is set and examined through the “Computer Management” tool in Windows based systems.

2) Propagation and Inheritance
It is necessary to discuss the different mechanisms behind the way that NTFS permissions can propagate throughout the directory structure. Within the ACL there are two types of entries, which are formally named Access Control Entries (ACE). These types are: (1) Explicit and (2) Inherited. Explicit entries are those that are applied directly to the directory or file's ACL, whereas inherited are those that are propagated from their parent directory. This makes implementation permissions easier as they can be set a high-level in the directory hierarchy and automatically propagate. This is of significant as permissions of different proposition types are processed differently and this leads us on to consider permissions accumulation.

3) Accumulation
Accumulation is the possibility for the subject to receive the effective permission of multiple different policies, resulting in the possibility for a subject to receive permissions from multiple different ACEs within the same ACL. Furthermore, any subject that interacts with the NTFS can be assigned to any number of groups, which can be entered into the ACE. This means that the user does not have to be directly entered into the ACE, and they could simply be a member of the group that is entered.

The policy combination is handled within the operating system by the reference monitor, which combines the permissions together to effectively create the union of all the policies; however, there are few complexities within permission accumulation due to the structured way in which ACEs are processed. These are:

- Explicit permissions take precedence over inherited permissions.
- Explicit deny permissions always take precedence over apply permissions.
- Permissions inherited from closer relatives take precedence over relatives.

It might be expected that deny permissions always take precedence over apply permissions to ensure that during the policy combination stage the user always operates as the least possible privilege elevation. However, the first point regarding explicit permissions taking precedence over inherited permissions can result in a situation where an inherited deny permission is never reached

A clear policy
The aforementioned aspects of file system permissions create the potential to inadvertently introduce weaknesses that could easily be exploited by ransomware. Here we consider the necessity of maintaining file system access control permissions to prevent ransomware damage through deriving and maintaining a clear access control policy. It is true that ransomware will continue to evolve and new, more sophisticated methods will be developed. However, following the below steps not only prevents the 'easy win' ransomware attack from successfully encrypting sensitive data, it also ensures a strong level of system security aiming to prevent against other threats, such as a malicious employee.

A clear policy is essential to ensure standardisation among allocated permissions and the functionality required by each role within the organisation. It is important to decide upon a policy for business of all sizes and it need not be an exhaustive and complicated task. Small business with only a few employees may neglect (through ignorance) the need to implement access controls beyond the defaults set by the

operating systems, whereas large organisations may have many administrators changing file system permissions on a daily basis, and each is making ad hoc modifications based on their experience and knowledge. Making ad hoc permission modifications has the potential to introduce error and over allocated permissions, which both could introduce a vulnerability.

The steps provided below are aimed to provide a starting point of criteria to consider for all businesses when creating their access control policy

1) Least privilege and Separation of Duty

In the contexts of file system control, the concept of least privilege is used to define the lowest possible level of access control required for a system user to undertake their daily tasks [7]. However, there will not be a single definition of least privilege for all users. A managerial user may need permission to view and modify certain employee records but does not need to see customer invoices, whereas secretarial staff should have permission to view, create and modify invoices but no permission to view employee records. A ransomware attack on employees in either of these roles would cause damage, but the separation of duty will help to compartmentalised the damage, resulting in only parts of the directory structure being encrypted.
An issue with implementing such role-based access control is that it is entirely feasible that new roles are also created as the company operations change and new employees are recruited. This can be problematic as if the new role is similar to one of the current roles, an administrator might make changes to the role by broadening the access control policy and allowing wide and potentially less restrictive access. This is problematic because any reduction in distance between the separations of duties reduces any potential compartmentalised and minimisation of ransomware damage.

2) Implementation

The position of the access control implementation in the directory structure hierarchy needs careful consideration to ensure that the data is restricted as far up the hierarchy as possible. There is a clear trade-off to be achieved here between administrative complexity and the level of restrictiveness. Access control can be implemented on a file level, but implementing restrictions based on each file would be tiresome, whereas setting permissions high up the directory structure to be propagated might make it challenging to effectively implement a middle section whereby two different roles both have permissions. For example, consider two distinctly separate hierarchical folder structures for both managerial and secretarial users. If both folder structures are secured on the root directory, it could become problematic if either user group needed to have access to a subdirectory within each role's hierarchy. At this point a new permission will need to be specified at the sufficient point within the hierarchy to only allow the user access to the locations required.

Considering the granularity of permissions is also of key importance. Microsoft's New Technology File System (NTFS) has six standard groups (read, write, modify, etc.) [8] and the possibility to create fine-grained permissions through using any combination of the fourteen individual attributes. There are specific instances whereby over allocation is performed through only using the standard groups. For example, only "Full Control" has the permission to delete directories. This can be problematic as assigning "Full Control" on a directory will allow a user to delete the directory itself and its entire contents. Using the "Modify" level disables the potential

to delete directories, but this would be too restrictive as a user would be able to create directories but not delete. The solution is to use the combination of the “Modify” on the root directory without any propagation, followed by “Full Control” reclusively propagating through all child directories. This is a clear example whereby careful consideration is required to determine how to translate a user’s role into a series of access control policies.

How the permissions are structured can have significant implications on administrative effort as well as introducing potential for making mistakes. Furthermore, all mechanisms to improve clarity in the allocation of permissions should be taken to reduce the potential for error. This means never implementing permissions on a user basis, rather using groups that represent roles within the system. As there is no mechanism to store comments within the allocation permissions, it is essential to ensure that the group name is something that adequately describes the permission. For example, something like “ManagerialFullControl” is more appropriate than “RoleOne”.

3) Auditing

Auditing is a very important task that should not be neglected. Old redundant permission are often left behind as employees leave, roles change, and new shared directories are introduced overtime. It is important to identify these permissions and remove them to avoid potential exploitation. It is also worth frequently revaluating how policies have been implemented to determine if any users are over privileged. A user will complain if they do not have the permission to undertake their job, but they have no incentive to report an over allocation.

There are many aftermarket tools available to assist in auditing permissions [9]; however, the Microsoft environment has a fundamental evaluation tool built-in. The “Effective Access” tab within the security properties allows you to query what a user or group’s effective permission is on a specific directory. This is somewhat cumbersome as it has to be performed on a directory by directory basis. Figure 3 shows an example of the effective access being displayed for the user Bob on the C:\ directory. The permission is detailed in terms of the individual permissions they are receiving on that specific location.

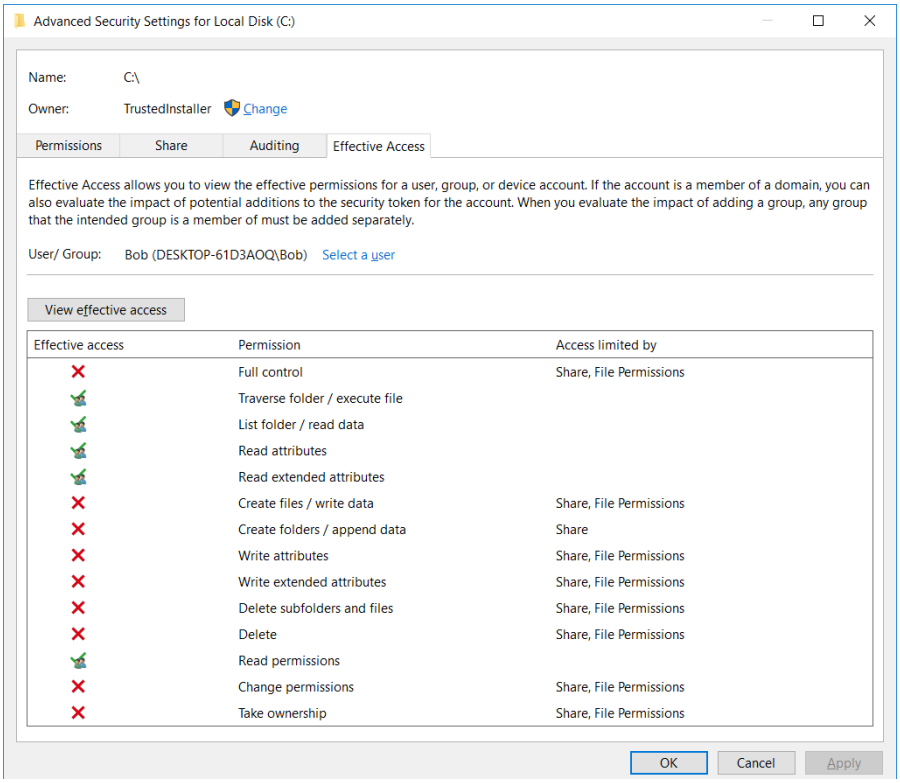

Figure 3: Effective Access

About the author

Dr Simon Parkinson is a Lecturer at the University of Huddersfield. His research interest is in developing intelligent systems for and cyber security. Simon is currently researching in the area of improving cyber security awareness and audits through developing novel, intelligent software tools. As part of his research Simon performs research into vulnerabilities of systems and to develop mitigation technologies.

References

1. 'Incidents of Ransomware on the Rise. Protect Yourself and Your Organization'. The Federal Bureau of Investigation, 2016 https://www.fbi.gov/news/stories/incidents-of-ransomware-on-the-rise
2. Mansfield-Devine, Steve. 'Ransomware: taking businesses hostage'. Network Security, Vol 10, 2016, pp 8-17.
3. Richardson, R., & North, M. 'Ransomware: Evolution, Mitigation and Prevention'. International Management Review, Vol 13, 2011, pp 10-21
4. Brewer, Ross. 'Ransomware attacks: detection, prevention and cure'. Network Security Vol 9, 2016, pp 5-9.
5. Pfleeger, C. P., & Pfleeger, S. L. 'Security in computing'. Prentice Hall Professional Technical Reference, 2002
6. Hanner, K., and Rudolf Hörmanseder. 'Managing Windows NT file system permissions—A security tool to master the complexity of Microsoft Windows NT file system permissions.' Journal of network and computer applications, Vol 22, 1999, pp 119-131.
7. Stiegler, M., Karp, A. H., Yee, K. P., Close, T., & Miller, M. S.'Polaris: virus-safe computing for Windows XP'. Communications of the ACM, Vol 49, 2006

8. Microsoft 2017, “File and Folder Permissions” https://msdn.microsoft.com/en-us/library/bb727008.aspx
9. Parkinson, S., Crampton, A. ‘A novel software tool for analysing NT file system permissions’. International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 4, 2013,