Global Information Assurance Certification Paper

Kayron C. Valentine
Version 1.2e
Encrypting File System Primer: Basics and Best Practices

Introduction

With the introduction of Windows 2000, Microsoft implemented a number of security-based improvements aimed at making their flagship operating system (OS) more robust and attractive for enterprise deployment. They've built on existing technology from NT 4.0 and added several features, which take advantage of standards that for years have been proven effective in the industry. The result is an OS less dependent of flawed proprietary standards, such as NTLM and WINS, which have been plagued with numerous vulnerabilities and over the years have made the job of system security an even more difficult task.

Among the many improvements in Windows 2000, one of the more notable ones is the addition of the Encrypting File System (EFS). EFS is a means of protecting user data which takes advantage of two well known industry standards: Data Encryption Standard X (DESX) and RSA public key exchange. It is supported in all versions of Windows 2000, and works transparently (without user input) to encrypt and decrypt files for user access. In the past, a user would have to encrypt and decrypt files using a third party application which had to be purchased and installed separately on their system, and which most often depended on the user entering a correct password. With EFS, this is no longer a necessity. As always however, most security measures are only effective if your users are properly informed, and EFS is no exception. As such, we'll take a look at EFS, what it is, how it works, and more importantly, what you need to know to make it work effectively for you.

EFS Basics

EFS is a Windows 2000 file encryption technology which is based on the public-key encryption standard. It works in conjunction with NTFS permissions to grant and deny users access to files and folders. EFS can be particularly beneficial to users most vulnerable to data theft through the physical removal of their system or its hard drive. An example of this would be laptop users. It is also attractive because there are no added cost, overhead, or possible compatibility complications that are usually associated with third party encryption software, and the encryption and decryption procedures are greatly simplified yet very secure.

In it's current implementation EFS uses Data Encryption Standard X (DESX) to encrypt the actual data. However, future releases of the OS are expected to allow alternate encryption schemes to be substituted for stronger security. For it's initial release, DESX was selected because it is a symmetrical encryption scheme, meaning that the same key is used to encrypt and decrypt the data. This in itself is weak but is faster than most other encryption techniques, which makes it perfect for encrypting larger data files with reduced

overhead.

Encryption and decryption can be initiated by two different means. The first method is to encrypt and decrypt files using the Windows Explorer GUI by right mouse clicking on a file or folder and performing a **file -> properties** then selecting the **advanced** button, and selecting the appropriate checkbox. The second method is by running the cipher.exe utility at the command line. To understand the encryption process, take a look at the table below, which gives a detailed account of what takes place when either of these tasks is performed.

EFS Encryption Process Summary

Step in Sequence	Process
1.	The user profile loads to the Registry, if necessary
2.	EFS creates a log file named efsX.log in the System Volume Information subdirectory. X is a unique number in the filename (e.g.,, efs0.log). EFS writes to the log file when performing subsequent steps in the encryption process so that EFS can recover the file in case of system failure during the encryption process.
3.	Microsoft Base Cryptographic Provider generates a random 128-bit File Encryption Key (FEK) for the file (128 bit in the US and Canada only).
4.	EFS reads the KEY_CURRENT_USER\Software\Microsoft\WindowsNT\CurrentVersion\EFS\CurrentKeys\CertificateHash Registry value to identify the user's public key/private key pair.
5.	EFS creates a Data Decryption Field (DDF) key ring with an entry for the user and associates the key ring with the file. The entry contains a copy of the FEK that the user's EFS public key encrypted.
6.	EFS creates a Data Recovery Field (DRF) key ring for the file with an entry for each Recovery Agent on the system. Each entry contains a copy of the FEK that the Recovery Agent's EFS public key encrypted.
7.	EFS creates a backup file, efsX.tmp, in the directory in which the file undergoing encryption resides. X is a unique number in the filename (e.g., efs0.tmp).
8.	EFS places the DDF and DRF key rings in a header and adds the header to the file as the file's EFS attribute.
9.	EFS marks the backup file as encrypted and copies the original file to the backup file.
10.	EFS destroys the original file's contents and copies the backup to the original file. The copy operation results in the data's encryption, because the backup file is marked as encrypted.
11.	EFS deletes the backup file.
12.	EFS deletes the log file.

13. The user profile unloads from the Registry if it loaded in step 1.

A copy of the table above is located at the following URL: http://www.win2000mag.com/Articles/Print.cfm?ArticleID=5592.

Strengths and Weaknesses

EFS will automatically encrypt files copied or moved to directories that have been marked as encrypted. This is beneficial because encrypting a folder allows users to save files to a specific location on their hard drives and know that they will be encrypted, without having to perform the process manually each time. This is one of the many features that could be considered a pro or strength of EFS. In fact, EFS has many strengths, and also some inherent weakness as well, a few of which are listed here.

Strengths

- ✓ EFS is embedded in the Windows 2000 kernel.
- ✓ EFS protects a user's encryption key from ever being written to a system pagefile.
- ✓ Because the EFS drivers are integrated into the kernel they are protected from direct user access.
- ✓ Currently, there are no known methods of circumventing EFS by direct access to the hard drive using NTFSDOS, an NTFS driver for Linux, or similar methods.
- ✓ EFS works seamlessly in the background, encrypting and decrypting files without user input, thus a user needs only know your login password to decrypt your files. Users will never forget to encrypt a file that they've decrypted for access, leaving it unprotected and vulnerable.
- ✓ EFS (more specifically, the OS) is flexible and allows for upgrades to stronger encryption processes and the use of smart cards for private key storage in the future.

Weaknesses

- ✓ EFS only works on Windows 2000 systems with the new NTFS 5.0 partition. Currently, there is no support for NTFS in Windows NT 4.0, FAT16, FAT32, ext2fs, or other types of partitions found on other operating systems either by Microsoft or other vendors.
- ✓ EFS does not support the encryption of system files. Encrypted system files may cause your system to become unusable.
- ✓ Third party backup utilities are not guaranteed to work on EFS encrypted files.
- ✓ EFS relies heavily on the premise that a strong authentication practice is already in place in order be most effective.
- ✓ Antivirus software must run with master keys which enable it to decrypt files while scanning and updating virus definition files
- ✓ EFS works seamlessly in the background, encrypting and decrypting files without

user input, thus a user needs only know your login password to decrypt your files. Access to your login means access to your encrypted files with no further effort.

EFS Best Practices and Recommendations

These recommendations are intended for both the user and the systems administrator and could go a long way toward maximizing the effectiveness of EFS on your systems. A list of Microsoft's recommended best practices can be found on their website at http://support.microsoft.com/support/kb/articles/Q223/3/16.asp; these have been incorporated into this list as well.

1. Laptop and standalone users are not encouraged to try changing the default recovery agent. Changing the default recovery agent is somewhat simplified when your system is a domain member and when Active Directory (AD) is setup and running. However, because EFS needs to generate a private/public key pair it requires a Certificate Authority, and as such, at least one certificate server to change the default recovery agent.

2. Download the ***efsinfo*** utility from Sysinternals and familiarize yourself with its usage. This utility has the ability to tell you who encrypted a file as well as which users have the ability to decrypt it as well. This is available for download at http://www.sysinternals.com/ntw2k/source/misc.shtml. Keep in mind that by default all a user needs are read/write NT permissions to encrypt a file. It is then possible for a user to inadvertently or maliciously encrypt key documents or other files on your organization's system. This utility will help you determine who the culprit is, and can also help you recover a file should a user's FEK be lost or inaccessible.

3. Encrypt the "My Documents" folder for all users, which is the default storage location for most user-generated documents.

4. Encrypt the "Temp" folder to protect your user's documents when they are temporarily stored during editing.

5. Encrypt folders instead of the individual files. Create a folder and designate it as encrypted. This prevents inadvertent decryption especially during editing and with the "Save As" feature in most software. "Save As" will only save the new document as encrypted if it is saved to a folder that has been marked as encrypted. Otherwise, using "Save As" while working on an encrypted document removes encryption from the new file that is created.

6. Generate your private keys on a physically secure system.

7. Outline a backup procedure for your private keys, especially for your local administrator account on your standalone systems and laptops. One way of accomplishing this is by exporting these to a PFX file and storing them on a floppy disk in a secure location. Instructions on exporting your private keys can

be found on the Microsoft website at http://support.microsoft.com/support/kb/articles/Q241/2/01.asp.

8. Designate two or more recovery agents per Organizational Unit (OU).

9. Be careful not to designate too many users as recovery agents. EFS will generate as many DRFs as there are recovery agents. This information is then stored with the file (generating larger files) and could cause a significant loss in system performance. Furthermore, each addition recovery agent is one more user who may not be following proper security precautions and whose password may be compromised, opening your files and other users' for easy access by an intruder. Remember that designated recovery agents do not need to know ***your*** private key to gain access to your encrypted data. This also limits the number of users that you can know for sure may gain access to your files. Remember that not all domain administrators are designated recovery agents, so even though NTFS permissions give them permission to view your file or maybe even take ownership and change permissions, they do not have access to a key which can decrypt your data and are still unable to gain full access.

10. Disable the usage of print spool files. These files are not encrypted and can easily be compromised by a would-be intruder.

11. Develop a recovery agent archive program to ensure that encrypted files can be recovered using obsolete recovery keys. Again, recovery certificates and private keys must be stored in a secure location (not your desk draw or on a floppy disk in your system).

12. Always adhere to strong password practices. EFS is by no means a substitute for password authentication or NTFS permissions. Encrypted files are safe as long as your private keys are not compromised. In some instances it would be far easier to compromise a user's weak password, or that of the default administrator account, and gain access to your encrypted files simply by logging on with these credentials.

13. For standalone systems, which are not members of a domain, change the default Syskey setting to use the system password option instead of storing the key locally in the system registry. In Windows 2000 the OS uses Syskey to encrypt the password hash, which (speaking of the hash) can easily be compromised using utilities such as L0phtcrack. However, the possibility still exists for an intruder to remove the encrypted hash and replace it with an un-encrypted one, to which he knows the password. By default, Windows will boot and encrypt this un-encrypted hash using Syskey, allowing the intruder to access your system using a password, which he has chosen. Storing the password on a floppy disk is especially not recommended for laptop users who may me tempted to leave this

floppy disk in the drive thus defeating the purpose.

14. Download and update your copy of cipher.exe from the Microsoft website. Whenever a file is encrypted or moved to an encrypted folder, a plain text copy of the file is made for the purposes of recovery should an error occur during this process. Although these plain text copies are deleted, fragments of the file are still accessible until the location where that files was stored is overwritten either by another file or by wiping this disk space. Wiping the disk would be good practice, however, most third party disk-wiping utilities fail to wipe disks formatted with Windows 2000's NTFS file system. The new version of cipher.exe now includes the ability to wipe deleted data from your hard disk. This is available for download at http://support.microsoft.com/support/kb/articles/Q298/0/09.asp. Be sure to follow the instructions for installation.

15. Do not compress your encrypted files and/or folders. Compression and EFS are what you might call mutually exclusive, that is, it's either one or the other. This is apparent if you are encrypting or compressing files using the Window Explorer GUI as selecting one option automatically deselects the other. However, this is not obvious to users who may choose to use the command line utilities. Therefore, it is possible that users may inadvertently remove encryption from files. If compression is applied to a folder marked as encrypted, the system prompts the user and asks if they would like to compress and decrypt the folder, and whether or not to apply this to all files and subfolders as well. If however, the user is working within an encrypted folder, this warning will not appear if compression is applied to a single file or an empty (newly created) folder. The parent folder is not affected, however, these files and folders are no longer protected when being accessed and moved even though they reside within a folder marked as encrypted. Encrypting a compressed file will remove compression.

Conclusion

EFS is new technology built on proven technology, built with the expectation of considerable improvements as the standards and technology evolves. It's present shortcomings may be attributed to it's implementation which causes unexpected side effects, vulnerabilities, and compatibility woes with third party software packages such as virus scanners, backup utilities, and disk wiping software. But EFS is one more step toward preventing intruders from gaining access to your important data, one more hurdle that an intruder must overcome to compromise the integrity of your system, and coupled with other security measures, is one more example of the advantages of defense in depth.

References

Mark Russinovich, "Inside Encrypting File System, Part 1", June 1999, Windows 2000 Magazine
URL: http://www.win2000mag.com/Articles/Print.cfm?ArticleID=5387

Mark Russinovich, "Inside Encrypting File System, Part 2", July 1999, Windows 2000 Magazine
URL: http://www.win2000mag.com/Articles/Print.cfm?ArticleID=5592

Michael Seamans, "Encrypting File System", February 21, 2000
URL: http://home.rochester.rr.com/seamans/Technical%20reviews/EFS.htm

Mark Minasi, "Decrypting EFS", Winter 2000, Windows 2000 Magazine
URL: http://www.win2000mag.com/Articles/Print.cfm?ArticleID=15907

Microsoft Corporation, "Encrypting File System: Your Secrets are Safe"
URL: http://www.microsoft.com/windows2000/techenthusiast/features/EFS.asp

Microsoft Corporation, "Step-by-Step Guide to Encrypting File System (EFS)"
URL:
http://www.microsoft.com/windows2000/techinfo/planning/security/efssteps.asp

Microsoft Corporation, "Best Practices for Encrypting File System"
URL: http://support.microsoft.com/support/kb/articles/Q223/3/16.asp

Microsoft Corporation, "Cipher.exe Security Tool for the Encrypting File System"
URL: http://support.microsoft.com/support/kb/articles/Q298/0/09.asp

Coleman Communications Consulting, "Windows 2000 Encrypted File System Weaknesses"
URL: http://www.colemancomm.com/news/20010612efs.htm

Eric Brock, "Windows 2000 Encrypting File System", July 27, 2000
URL: http://www.sans.org/infosecFAQ/win2000/w2k_efs.htm