Global Information Assurance Certification Paper

Alternate Data Streams, the Hidden Threat

Lynn Price
Sept. 15, 2000

Introduction

Alternate Data Streams (ADS) have been a part of NTFS ever since it was introduced with Windows NT 3.1. Since these streams are little known and not often utilized, security issues revolving around this "feature" of NTFS have only been recently brought to the attention of the public. The major antivirus companies do not see ADS as a threat so as of now do not scan for them, creating many possibilities for malicious use of this feature. This document will explain ADS and how they could possibly be used to wreak havoc on a Windows NT system.

I.) What are Alternate Data Streams and how do they work?

ADS were incorporated into the NT File System as a feature to allow for "flexibility." The official explanation from Microsoft is:

> **One of the major design goals of Windows NT at every level is to provide a platform that can be added to and built upon, and NTFS is no exception. NTFS provides a rich and flexible platform for other file systems to be able to use. In addition, NTFS fully supports the Windows NT security model and supports multiple data streams. No longer is a data file a single stream of data. Finally, under NTFS, a user can add his or her own user-defined attributes to a file.**[1]

As far as is known, Microsoft uses streams for compatibility with the Macintosh operating system, which uses a type of ADS to store information about what application was used to create a file. Unlike Windows which bases application to file associations on the extension of the file, the Macintosh OS bases the association on information in this ADS, which is referred to as a "resource fork." Microsoft also uses ADS for Internet Information Server (IIS) scripts, which created a problem that will be briefly discussed in a later section.

Now we will demonstrate in a simple manner how ADS work. First, we'll need a computer running Windows NT on an NTFS partition, with a test directory to work in that can be deleted later. For the purposes of this document, let's assume this directory is c:\test. First we can create a normal text file called "data.txt":

Echo "this is some text in the data stream" > data.txt

Now the data.txt file can be viewed in all the normal manners for viewing text files, i.e. Type, the DOS editor, Windows Notepad, Word for Windows, etc. Now to create an ADS associated with data.txt:

---

[1] Microsoft Corp. "Windows NT 4.0 FAQS: Setup and Upgrading Questions". Microsoft Technet. Sept. 1998. Url: http://www.microsoft.com/technet/winnt/winntas/technote/troubleshooting/topntqa.asp. (Sept. 15, 2000)

Dir c:.* /s > data.txt:stream.txt*

: is the format for an ADS. If you list the contents of the c:\test directory now, you'll notice that data.txt is still the same size as it was before we created the stream, but also notice that there is a bit of drive space missing that was there before we created the stream. Now, to prove the stream exists:

Notepad data.txt:stream.txt

You'll notice notepad displays the contents of the c: drive, which we dumped to the stream. Also note that attempting to use type, edit, Word, etc, will fail attempting to open data.txt:stream.txt, although we know the data is there. The ADS data is effectively hidden unless we use Notepad, since notepad is the only Microsoft editor supplied with NT that understands ADS. Also of note is that the only way we can see the stream is by knowing its exact name, even with Notepad we could not browse for ADS.

Now you'll probably want to delete our test files. You'll notice that if you try:

Del data.txt:stream.txt

Del will exit with an error and data.txt still exists as the only file in the directory. Simply deleting data.txt also gets rid of the stream. You'll also notice that by deleting this file you freed up more than the 40 or so bytes that data.txt occupied.

II.) How can ADS be used against us?

Just from the brief introduction above, several ways that ADS can be used against system administrators come to mind.

The first and most obvious "attack" that can utilize ADS is a Denial of Service (DOS) attack on Windows NT servers or workstations. Windows NT has a tendency to crash when it runs out of drive space. If enough data is written to one or more ADS on a Windows NT computer, the disks can be filled with no easy way of finding where the "extra" data is. This will not only crash the operating system, but will make recovery quite difficult since there is no easy way to track down the ADS without knowing its exact name and location. This type of attack would bypass even servers where users had quotas on shares, since no known quota managing software detects ADS. This is an extremely high risk attack, since any user with write access to any part of an NTFS partition could carry it out with only minimal knowledge of ADS.

The makers of the major antivirus scanners have not as of yet addressed detecting malicious code in ADS. The reasoning from these companies is that only "inert" code can be contained in alternate streams. This was shown to be untrue from the following example:[2]

Type c:\winnt\notepad.exe > myfile.txt:hidden.exe

Start myfile.txt:hidden.exe

This example demonstrates hidden.exe (notepad.exe) being executed from a stream. Although trying to launch the application "myfile.txt:hidden.exe" in the normal manners (typing the name of the executable, Start/Run executable name) will fail, launching with the "start" command will execute the code.

What the antivirus software vendors fail to realize is that the code in an alternate stream does not even need to be executable to be dangerous, however. A virus writer could store the code for his malicious program in an ADS and use this to his advantage in a few different ways. One use of this that comes to mind is that the hacker could "deploy" his code to an ADS on many computers, using a method similar to the ILOVEYOU virus, although no effects would be noticed immediately. Once the hacker is fairly sure that many machines have the ADS code written to their drives, he could deploy a very small piece of code that simply calls the virus code hiding in the stream, effectively launching a massive attack that antivirus scanners couldn't detect.

The more sinister use of writing virus code to an alternate stream is the possibility of turning the system's virus scanning software against the system itself. If a well known virus's code was written to an ADS attached to a system file, for example explorer.exe, ntoskrnl.exe or ntfs.sys, or any of the other Windows NT kernel or service/device files, then an attack could be put into motion by simply letting this code execute. Most antivirus scanners will perform one of several pre-defined actions when virus code is detected, almost all of which could be devastating to the operating system. Most virus scanners are set to "clean" the infected file by default, and if this fails (as it almost certainly would if the virus was executing from an alternate stream) , the scanner will either assign "no access" to the file, or delete the file, depending on the scanner and how it was set up. As you can see from this example, denying access or deleting certain files would basically force the virus scanner to create a "no boot" situation for a Windows system quite easily, or at the very least create havoc by rendering a server unusable by eliminating certain devices or services.

Recently Microsoft came under fire for opening a hole in their Internet Information Server product because of alternate streams. Because of the way IIS uses and processes scripts, a user could request a url such as :

http://server/script.asp:$DATA

which would reveal the source code of the script instead of processing the script itself. Basically by knowing the name of the script, a user could add the documented name of the alternate stream that IIS uses to display the contents of the script, potentially displaying server names, user names, and passwords.[3]

[2] Carvey, H. "NTFS Alternate Data Streams". URL: http://patriot.net/~carvdawg/ads.html. (Sept.14, 2000)

III.) What can we do?

As we have seen in the few examples above, there are many possibilities of security risks with Alternate Data Streams. There are a few things we can do to try to limit the risk involved.

a.) Check for alternate streams using Sfind.exe from the NT Objectives Forensic Toolkit (available at www.ntobjectives.com). This tool can be run periodically to determine if there are any new datastreams on the system. If any show up it is a good idea to try to track down the source and determine if it should be deleted.
b.) It should be possible to delete streams by deleting the "visible" file. If for some reason this fails, the data can be moved to a non-NTFS partition and then copied back to the original source. Since NTFS is the only file system that understands streams, moving the data in this manner will cause the streams to be lost. The non-NTFS partition can be a mapped network drive to a FAT, Linux Native, FAT 32, or NFS drive.
c.) Request that antivirus vendors add support for scanning alternate data streams. If enough people complain, eventually the major virus scan vendors will supply this support.
d.) Ensure that all permissions are set correctly on Windows NT system files and directories, so that only administrators and the system may write to these directories.

Acknowledgements

1.) Microsoft Corp. "Windows NT 4.0 FAQS: Setup and Upgrading Questions". Microsoft Technet. Sept. 1998. Url: http://www.microsoft.com/technet/winnt/winntas/technote/troubleshooting/topntqa.asp. (Sept. 15, 2000)
2.) Carvey, H. "NTFS Alternate Data Streams". URL: http://patriot.net/~carvdawg/ads.html. (Sept.14, 2000)
3.) Brenton, Chris. "Sans Flash Alert –Virus Scanner Inadequacies with NTFS" Sept. 5, 2000. URL: http://archives.neohapsis.com/archives/sans/2000/0082.html (Sept. 15, 2000)
4.) Sawicki, Ed. "NTFS Streams". 1998. *Networking Solutions Magazine*. URL: http://www.ntique.org/reports/ads.htm (Sept. 15, 2000)
5.) Russinovich, Mark. "Inside NTFS". Jan. 1, 1998. URL: http://www.winntmag.com/Articles/Print.cfm?Action=Print&ArticleID=3455. (Sept. 13, 2000)
6.) Heyne, Frank. "FAQ: Windows NT's File System and Alternate Data

[3] Microsoft Corp. "$DATA Data Stream Name Returns Souce of a Remote File". Microsoft Technet. URL: http://support.microsoft.com/support/kb/articles/Q193/7/93.asp (Sept. 17, 2000)

Streams". March 21, 2000. URL: http://www.heysoft.de/nt/ntfs-ads.htm (Sept. 15, 2000)
7.) Microsoft Corp. "'$DATA' Data Stream Name Returns Source of a Remote File". Microsoft Technet. Sept. 2000. URL: http://support.microsoft.com/support/kb/articles/Q193/7/93.asp