

# Re: [Full-disclosure] Microsoft Windows Vista/2003/XP/2000 file management security issues

---

*Source:* <http://www.derkeiler.com/Mailing-Lists/Full-Disclosure/2007-03/msg00116.html>

---

- *From:* 3APA3A <[3APA3A@xxxxxxxxxxxxxxxxxxx](mailto:3APA3A@xxxxxxxxxxxxxxxxxxx)>
  - *Date:* Fri, 9 Mar 2007 15:40:04 +0300
- 

Dear M. Burnett,

—Friday, March 9, 2007, 7:12:31 AM, you wrote to [3APA3A@xxxxxxxxxxxxxxxxxxx](mailto:3APA3A@xxxxxxxxxxxxxxxxxxx):

MB> 3APA3A, I just wanted to say that is very clever research you have done.  
MB> It's true that this does require some re-thinking of security practices, but  
MB> I don't think it's accurate to say it's impossible to secure a private  
MB> folder in a public one—I believe there is a way to do it securely.

Of course, there is always a way to solve one specific problem. A solution for problem #1 has problem #2. Solution for problem #2 has problem #3.

MB> There are three basic attacks you described:  
MB> 1. Attacker deletes a users' new folder then immediately re-creates it,  
MB> establishing ownership of the folder  
MB> 2. Attacker predicts filename, creates the file, then keeps it open for all  
MB> access, retaining rights on the file  
MB> 3. Attacker creates and deletes a file but keeps it open, denying any other  
MB> access to that file.

And

4. By forcing irresolvable replication collision on DFS replicated folder in conjunction with [3] attacker can delete newly created files (and probably folders, I did no test) created by another user even in case of permissions like this:

Users: Add & read  
Creator Owner: full control (or something, doesn't matter).

MB> The last two attacks could be prevented by creating new files in a private  
MB> folder that prevents others from creating files. However, item 1 could  
MB> compromise the security of that folder when it is initially created. To  
MB> prevent that situation you would need to make sure that, within a public  
MB> directory, only the CREATOR OWNER can delete a folder, which I believe is

Re: [Full-disclosure] Microsoft Windows Vista/2003/XP/2000 file management security issues

MB> the default setting.

MB> But, as you noticed, others can still delete that folder. There is a quirky  
MB> thing in NTFS that allows users to delete subfolders and files without even  
MB> having delete permissions on those files (see  
MB> <http://xato.net/bl/2007/01/04/pointless-permissions/>). I think that if you  
MB> set permissions on a folder that prevented users from deleting children, and  
MB> only allowed CREATOR OWNER to delete new folders, when a user creates a new  
MB> folder it will be secure, therefore protecting you from 2 and 3.

The problem with replicated folder has no relation to NTFS permissions.  
It's not replication service itself, not user who deletes the file of  
folder, because of collision. Attacker only forces collision situation.  
This can not be fixed by NTFS permissions. "Add" NTFS permission is  
only required to remove somebody's newly created file.

For replication service attacks is:

1. Victim creates Folder
2. Attacker creates Folder with same name on different replication mirror and locks it.
3. Replication service detects collision and removes Folder
4. Attacker creates Folder again
5. Folder is replicated. Attacker is now folder owner.

MB> I haven't fully tested this to verify it, but I believe this would prevent  
MB> all the scenarios you described, although a user could still prevent the  
MB> initial folder creation if they could predict the filename.

As you can see, there is at least one situation where your assumption  
is wrong, a case of DFS replication.

Of course, it still can be solved somehow, but who can guarantee it's  
impossible to find problem #5 in solution for problem #4?

MB> Nevertheless, these are still important issues that illustrate some of the  
MB> confusion that the NTFS quirkiness leads to.

MB> Mark Burnett

MB> <http://xato.net>

MB> -----Original Message-----

MB> From: 3APA3A [<mailto:3APA3A@xxxxxxxxxxxxxxxxxxxx>]

MB> Sent: Thursday, March 08, 2007 12:59 PM

MB> Subject: Microsoft Windows Vista/2003/XP/2000 file management security  
MB> issues

MB> To: bugtraq@xxxxxxxxxxxxxxxxxxxx; full-disclosure@xxxxxxxxxxxxxxxxxxxx

Re: [Full-disclosure] Microsoft Windows Vista/2003/XP/2000 file management security issues

MB> This is an article I promised to publish after Windows  
MB> ReadDirectoryChangesW (CVE-2007-0843) [1] issue. It should explain why  
MB> you must never place secure data inside insecure directory.

MB> Title: Microsoft Windows Vista/2003/XP/2000 file management security issues  
MB> Author: 3APA3A, <http://securityvulns.com/>  
MB> Vendor: Microsoft (and potentially another vendors)  
MB> Products: Microsoft Windows Vista/2003/XP/2000, Microsoft resource kit  
MB> for Windows 2000 and different utilities.  
MB> Access Vector: Local  
MB> Type: multiple/complex (weak design, insecure file operations, etc)  
MB> Original advisory: <http://securityvulns.com/advisories/winfiles.asp>  
MB> Securityvulns.com news:  
MB> <http://security.nnov.ru/news/Microsoft/Windows/files.html>

MB> 0. Intro

MB> This article contains a set of attack scenarios to demonstrate security  
MB> weakness in few very common Windows management practices. Neither of the  
MB> problem explained is critical, yet combined together they should force  
MB> you to review your security practices. I can't even say  
MB> "vulnerabilities" because there is no something you can call  
MB> "vulnerability". It's just something you believe is secure and it's not.

MB> 1.1 Problem: inability to create secured file / folder in public one.  
MB> Attack: folder hijack attack

MB> First, it's simply impossible with standard Windows interface to create  
MB> something secured in insecure folder.

MB> Scenario 1.1:

MB> Bob wishes to create "Bob private data" folder in "Public" folder to  
MB> place few private files. "Public" has at least "Write" permissions for  
MB> "User" group. Bob:

MB> I Creates "Bob private data" folder  
MB> II Sets permission for folder to only allow access to folder himself  
MB> III Copies private files into folder

MB> Alice wants to get access to folder Bob created. She

MB> Ia Immediately after folder is created, deletes "Bob private  
MB> data" folder and creates "Bob private data" folder again (or  
MB> simply takes ownership under "Bob private data" folder if  
MB> permissions allow). It makes Alice folder owner.  
MB> IIa Immediately after Bob sets permissions, she grants herself

Re: [Full-disclosure] Microsoft Windows Vista/2003/XP/2000 file management security issues

MB> full control under folder. She can do it as a folder owner.  
MB> IIIa Reads Bob's private files, because files permissions are  
MB> inherited from folder

MB> Alice can use "Spydir" (<http://securityvulns.com/soft/>) tool to  
MB> monitor files access and automate this process. As you can see, [1]  
MB> elevates this problem significantly.

MB> This is not new attack. Unix has "umask" command to protect  
MB> administrators and users. Currently, Windows has nothing similar.

MB> CreateFile() API supports setting file ACL on file creation (just like  
MB> open() allows to set mode on POSIX systems). ACL can be securely set  
MB> only on newly created files. This raises a problem of secure file  
MB> creation.

MB> 1.2 Problem: Inability to lock / securely change permissions of already  
MB> created file  
MB> Attack: pre-open file/directory attack.

MB> There are few classes of insecure file creation attack (attempt to  
MB> open existing file), exploitable under Unix with hardlinks or  
MB> symlinks. It's believed Windows is not vulnerable to this attacks  
MB> because

MB> I. There is no symlinks under Windows. Symlink attacks are not  
MB> possible.  
MB> II. Security information in NTFS is not stored as a part of  
MB> directory entry, it's a part of file data. Hard link attacks are  
MB> not possible.  
MB> III. File locks in Windows are mandatory. It means, if one  
MB> application locks the file, another application can not open  
MB> this file, if user doesn't have backup privileges. It mitigate  
MB> different file-based attacks.

MB> There is at least one scenario, attacker can succeed without symbolic  
MB> link: to steal data written to file created without check for file  
MB> existence regardless of file locks and permissions.

MB> Attack description: if attacker can predict filename to be written, he  
MB> can create file, open it and share this file for all types of access.  
MB> Because locking and permissions are only checked on file open,  
MB> attacker retain access to the file even if it's locked and it's  
MB> permissions are changed to deny file access to attacker.

MB> Exploit (or useful tool): <http://securityvulns.com/files/spyfile.c>

MB> Opens file, shares it for different types of access and logs changes,  
MB> keeping the file open.

MB> Compiled version is available from <http://securityvulns.com/soft/>

Re: [Full-disclosure] Microsoft Windows Vista/2003/XP/2000 file management security issues

MB> Scenario 1.2.1:

MB> Bob is now aware about folder hijack attack. He use xcopy /O /U /S to  
MB> synchronize his files to newly created folder. xcopy /O copies  
MB> security information (ownership and permissions) before writing data  
MB> to file.

MB> Alice use "Spydir" to monitor newly created folders and files in  
MB> Bob's directory. She use Spyfile to create spoofed files in target  
MB> directory and waits for Bob to run xcopy. Now, she has full control  
MB> under content of Bob's files despite the fact she has no permissions  
MB> to access these files.

MB> In a same way directory content may be monitored by pre-opening  
MB> directory.

MB> Scenario 1.2.2:

MB> Enterprise directory structure is replicated every day to another  
MB> user-writable location in order to allow users to recover suddenly  
MB> deleted or modified files. xcopy or robocopy (from resource kit) is  
MB> used for replication. Attacker can hijack content of newly created  
MB> files in newly created folders.

MB> Same problem may happen on archive extraction or backup restoration.

MB> Vulnerable applications:  
MB> xcopy (from all Windows versions),  
MB> robocopy (Windows 2000 Resource Kit),  
MB> different archivers  
MB> backup restoration utilities

MB> By default, xcopy warns user the file exists, unless /Y or /U key is  
MB> specified. But  
MB> I. /Y is always specified for replication  
MB> II. /Y can be specified via COPYCMD environment variable. COPYCMD  
MB> environment variable can be created in autoexec.bat file.  
MB> Different situations are possible, where autoexec.bat is writable by  
MB> attacker, if:  
MB> – Default Windows 2000 permissions are used or applied with domain  
MB> policy [2].  
MB> – One can try to re-create autoexec.bat using POSIX subsystem  
MB> III. Neither xcopy nor other utilities warn user on existing  
MB> directory. Pre-open directory attack will always succeed.

MB> As you can see, [1] again dramatically elevates this problem.

MB> 1.3 Problem: user can completely block access to the files  
MB> Attack: open file deletion  
MB> (including Windows file replication service DoS)

MB> If files is deleted while it's open, it still present in file system  
MB> under it's old name until close. Any operation on this file  
MB> (including attributes requests) fails, regardless of application  
MB> rights and permissions (including backup ones).

MB> Exploit: use spyfile, delete file while it's spied. Now, without  
MB> closing spyfile, attempt any operation on this file (e.g. try to  
MB> find it's ownership).

MB> Scenario 1.3.1

MB> Now Bob found an copy application to securely copy files. It deletes  
MB> old file before creating new one. But it fails if Alice tries to spy  
MB> on Bob files, because attempt to delete file succeeds, but file  
MB> still present and is unmanageable.

MB> Scenario 1.3.2

MB> Windows file replication service (FRS) is used to replicate data  
MB> between 2 public DFS folders to distribute load. Folder has  
MB> permissions:  
MB> Everyone: Add & read  
MB> Creator Owner: Full Control  
MB> Thouse, Alice has no permissions to delete files created by Bob.

MB> Replicated folder is available as a share on 2 different servers:  
MB> \\SERVER1\Share and \\SERVER2\Share. Bob is connected  
MB> to \\SERVER1\Share.

MB> Alice uses "Spydir" to monitor files creation by Bob. Every time Bob  
MB> creates new file on \\SERVER1\Share, Alice use spyfile to create  
MB> file with same name on \\SERVER2\Share. It effectively leads to FRS  
MB> collision. While trying to resolve collision, FRS fails to delete  
MB> file created by Alice and Bob file is deleted (original file is  
MB> moved to special hidden folder only accessible by administrator).

MB> Workaround: never try to use creator-owner based permissions in  
MB> replicated folders.

MB> Again, [1] seriously escalates this problem.

MB> 2. Conclusion:

MB> It's simply impossible to securely create something in public folder.  
MB> At least DoS conditions are always possible.  
MB> Developers should not consider mandatory file locking as a security  
MB> feature.  
MB> Developers should care about secure file creation to store sensitive  
MB> information. CREATE\_NEW should always be used and ACL should be set  
MB> with IpSecurityAttributes of CreateFile. No attempt to open existing

Re: [Full-disclosure] Microsoft Windows Vista/2003/XP/2000 file management security issues

MB> file should be made.

MB> Never try to create secure folder in public one. If you are forced,

MB> disconnect all users before this operation.

MB> Never use replication, archive extraction or backup restore to

MB> user-accessible folder.

MB> Bob and Alice should finally marry.

MB> 3. Vendor:

MB> All timelines are same with [1].

MB> [1]. Microsoft Windows ReadDirectoryChangesW information leak

MB> (CVE-2007-0843)

MB> <http://security.nnov.ru/news/Microsoft/Windows/ReadDirector.html>

MB> [2]. Windows 2000 system partition weak default permissions

MB> <http://securityvulns.ru/news2205.html>

MB> <http://winblogs.security-feeds.com>

---

~/ZARAZA <http://securityvulns.com/>

>GB5==K5 8A:>?05<K5! 'C >B 20A 40;L=59H8E ?8A5<. ("25=)

---

Full-Disclosure – We believe in it.

Charter: <http://lists.grok.org.uk/full-disclosure-charter.html>

Hosted and sponsored by Secunia – <http://secunia.com/>