First Look at the File Based Write Filter

By Sean D. Liming and John R. Malin
SJJ Embedded Micro Solutions

First Printing: September 2006

Published in the United States by

SJJ Embedded Micro Solutions, LLC.
11921 Tivoli Park Row #5
San Diego, CA 92128 USA

www.sjjmicro.com

Table of Contents

1 A New Feature to Protect Individual Files

The Enhanced Write Filter (EWF) has been the cornerstone of XP Embedded. Feature Pack 2007 continues to add new capability to EWF, but it also adds a new feature that allows the user to protect individual files instead of whole volumes. The File Base Write Filter (FBWF) adds a new feature to protect media, but still allow for some files, like databases, to be written to the drive. This simplifies some of the persistent information storage solution and still protects the drive from unwanted writes. This article provides a first look at this new Embedded Enabling Feature.

2 How It Works

EWF protects whole volumes or partitions from any writes. All writes to a EWF protected partition are sent to an overlay. The overlay can either be Disk or RAM / RAM-REG. The only way to update files or registry settings was to disable EWF, perform the change, and then re-enable EWF. If data needed to be saved to something like a database file, a second unprotect partition was required.

2.1 FBWF Architecture

FBWF takes a different approach by protecting on a file level instead of the whole volume or partition. The developer can set specific files or directories to be unprotected. Any writes made to protected files are sent to RAM overlay. Any writes to unprotected files are passed through to the disk. Figure 1 shows how FBWF interacts with the protected disk.

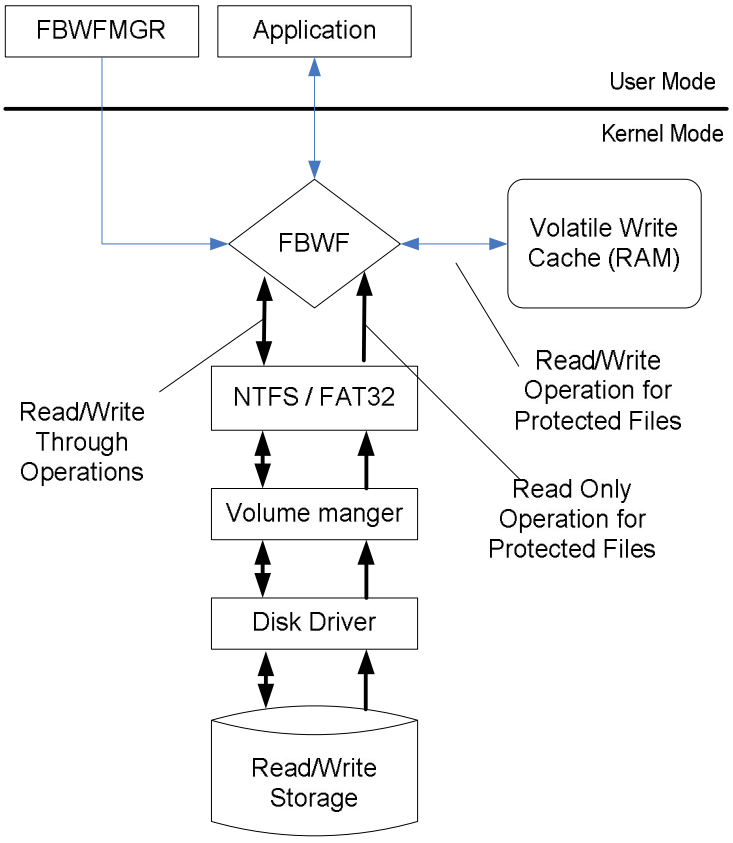

Fig 1 – FBWF Architecture

2.2 FBWF Features:

FBWF working at a file level provides some features that EWF doesn't provide, like the ability to add or remove protected volumes, to allow file writes to pass through to the protected volume, and to control the amount of RAM that FBWF will use. Table 1 lists the features for FBWF:

Feature	Description
File and Directory Management Transparency	Applications don't need to be modified for FBWF. All read write access to media is transparent to the application. There are a few things to be aware of: 1. Errors can occur if the following are performed:: • Moving files across protection boundaries, for example, from a protected volume to an unprotected volume. • Committing new or deleted files. 2. FBWF supports NTFS and FAT32 file systems. FAT12, FAT16 and Flash File System are not supported. 3. FBWF can protect formatted volumes only. FBWF cannot recognize unformatted volumes. 4. Files in FBWF overlay cache are unencrypted.
Selective Write-Through	Selective Write-Through allows writes to a predefined set of files and directories to reach the underlying protected volume. Files and directories are designated for Selective Write-Through in the following ways: • Using the Target Designer during the OS build process. • Using FWBF APIs at run time. • Using the FWBF Manager at run time in the development environment
Selective Commits and Restores	Selective Commit moves a file (or file changes) from the overlay to the protected volume. Selective Commit is immediate and persists through reboots. Selective Restore discards an overlay file and restores the view of the underlying volume. Both of these functions are useful for devices that are not frequently rebooted or devices that are shared across users.
Dynamic Protection	Volumes and files can be added and removed from FBWF protection, but a reboot is required for any changes to take effect.
Improved Overlay Memory Use	Unlike EWF with RAM overlay, you can preset the amount of RAM to be used by the Overlay cache. 1GB is the maximum FBWF conserves and reclaims memory in overlay the cache. For instance, FBWF frees overlay memory when files are deleted or reduced in size.

Table 1 – FBWF Features

Of course, there are some limitations. There are some operations at the file level that FBWF cannot handle, but EWF can:

- File locking and unlocking
- File ID in NTFS
- Reparse points
- Quota
- Hard links

- Opportunistic lock
- File compression and encryption

With these limitations for FBWF, you are left to determine which filter is best for your application. Both filters can be in the same image, but they cannot protect the same volume.

Also, like EWF, if the target system is going to participate in a Domain or Terminal Service then the Registry Filter must be added to the configuration if FBWF is protecting the OS files. The Registry Filter will preserve the two critical registry keys that get updated behind the scenes. Loss of Domain participation or running out of TSCAL licenses will result if the Registry Filter is not in place.

2.3 Controling FBWF with FBWFMGR.EXE

EWF has the EWFMGR.EXE utility to control the state of EWF. FBWF has FBWFMGFR.

Syntax:
fbwfmgr [/? | /help /[switch] | /displayconfig | /overlaydetail | /enable | /disable | /addvolume [volumename] | /removevolume [volumename] [1|0] |
/addexclusion [path] | /removeexclusion [path] |
/setthreshold [threshold] | /setcompression [1|0] | /setpreallocation [1|0]]

Table 3 lists the option commands:

Switch	Description
displayconfig	Displays all configuration information for the write filter including protected volumes list, overlay configuration and write-through paths. The command returns: State—Indicating current filter state (enable or disable) and state for next boot. Protected Volumes—List of protected volumes including the current and next boot state. Compression—Current and next boot state for cache compression. Threshold—Current and next boot values for the overlay cache threshold. Write-Through Paths—Displays a complete list of active and next boot Write-Through paths. Pre-allocation Status—Displays current and next boot status for cache pre-allocation.
overlaydetail	Displays detail on the current overlay contents for all protected volumes. The command returns: Contents—Files and folders currently in the overlay for all protected volumes including sizes (size of data in overlay) and open file handles. Memory Usage—Total amount of memory being consumed by the overlay.
enable	Enables the write filter on the next restart.
disable	Disables the write filter on the next restart.
addvolume	Adds a volume to the protected volume list for next boot.
removevolume	Removes a volume from the protected volume list for next boot.
addexclusion	Adds a Write-Through path to the exclusion list for next boot.

Switch	Description
removeexclusion	Removes a write-through path from the exclusion list for next boot.
setthreshold	Sets the overlay threshold value for next boot.
setcompression	Sets overlay compression as enabled (1) or disabled (0) for next boot.
setpreallocation	Sets cache pre-allocation as enabled (1) or disabled (0) for next boot.
commitfile	Commits a specified file.
restorefile	Restores a specified file.
?	Displays usage and help.
help / [switch]	Displays help information for a specific FBWF Manager switch.

Table 2 – FBWFMGR Command Options

The following are the Field descriptions:

Field	Meaning
volumename	Full path to a volume
1	Remove exclusion list
0	Preserve exclusion list
path	Full file or directory path
threshold	Overlay threshold in MB

Table 3 – FBWFMGR Field Descriptions

2.4 FWBF in Action

Let's see how FWBF works in a real image. FBWF will protect the Operating System files and level one directory open for access. Some of the steps assume that you are familiar with creating components and configurations. If you are not, there are other articles that can help in this area.

2.4.1 Part 1 Create and Build the Image

1. Use TAP.EXE under Windows XP to capture your target systems hardware components.
2. Import the PMQ file into Component Designer and create a platform macro component.
3. Import the SLD with the platform macro component into the database using Component Database Manager.
4. Open Target Designer and create a new configuration called FBWF test.
5. Add the platform macro component
6. Add the following components:
    a. Runtime Quick Start Helper Macro
    b. CMD - Windows Command Processor
    c. User Interface Core
       Modify User Interface Core Settings – check the following:
        - Show Desktop icons:
        - Show Control Panel on Start Menu:
        - Show Run on Start Menu:
        - Show Log Off on Start Menu:
        - Show Shut Down on Start Menu:
        - Show All Programs list on Start menu:
        - Show context menu on Shell folders:
        - Show context menu on Task bar:
        - Show Notifications on Task bar:
    d. Windows Accessories
    e. File Based Write Filter
7. Under the Extra Files, add a new folder called Data. The Folder will be used as our write-through test.

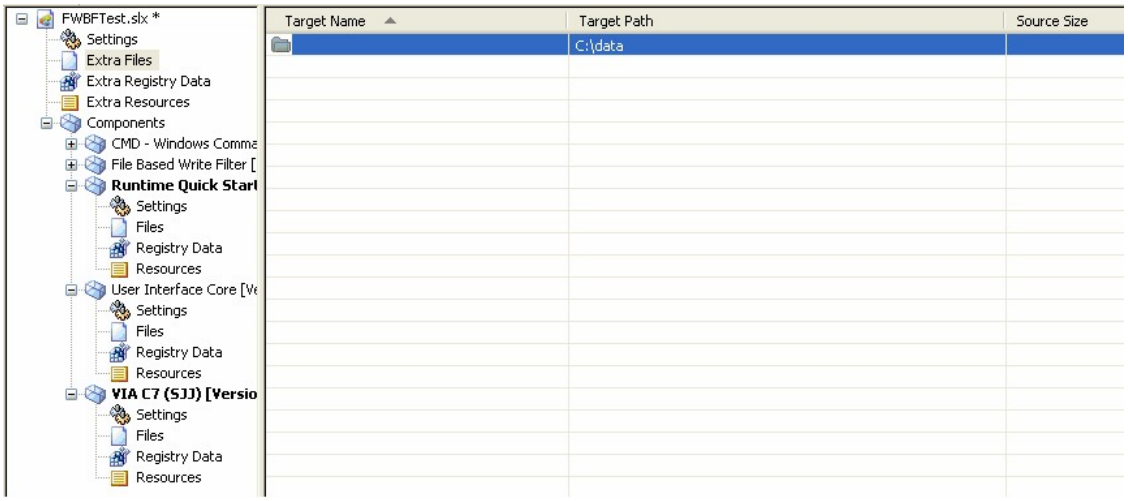

Fig 2 – Adding a Directory

Now that we have a folder to write to, let's modify the File Based Write Filter settings. Like EWF, FBWF's Settings page is split into an upper half (File Based Write Filter Configuration) and a lower half (settings for each Protected Volume). The settings for the File Based Write Filter Configuration are described in Table 4.

8. Keep the File Based Write Filter Configuration settings as the defaults. We will only be protecting the C drive so add "C:" to the Protect Volume #1.
9. Add the \Data directory to the Write-Through Files and Folders list. This means all other files and folders under C:, including the OS files, will be protected from writes.

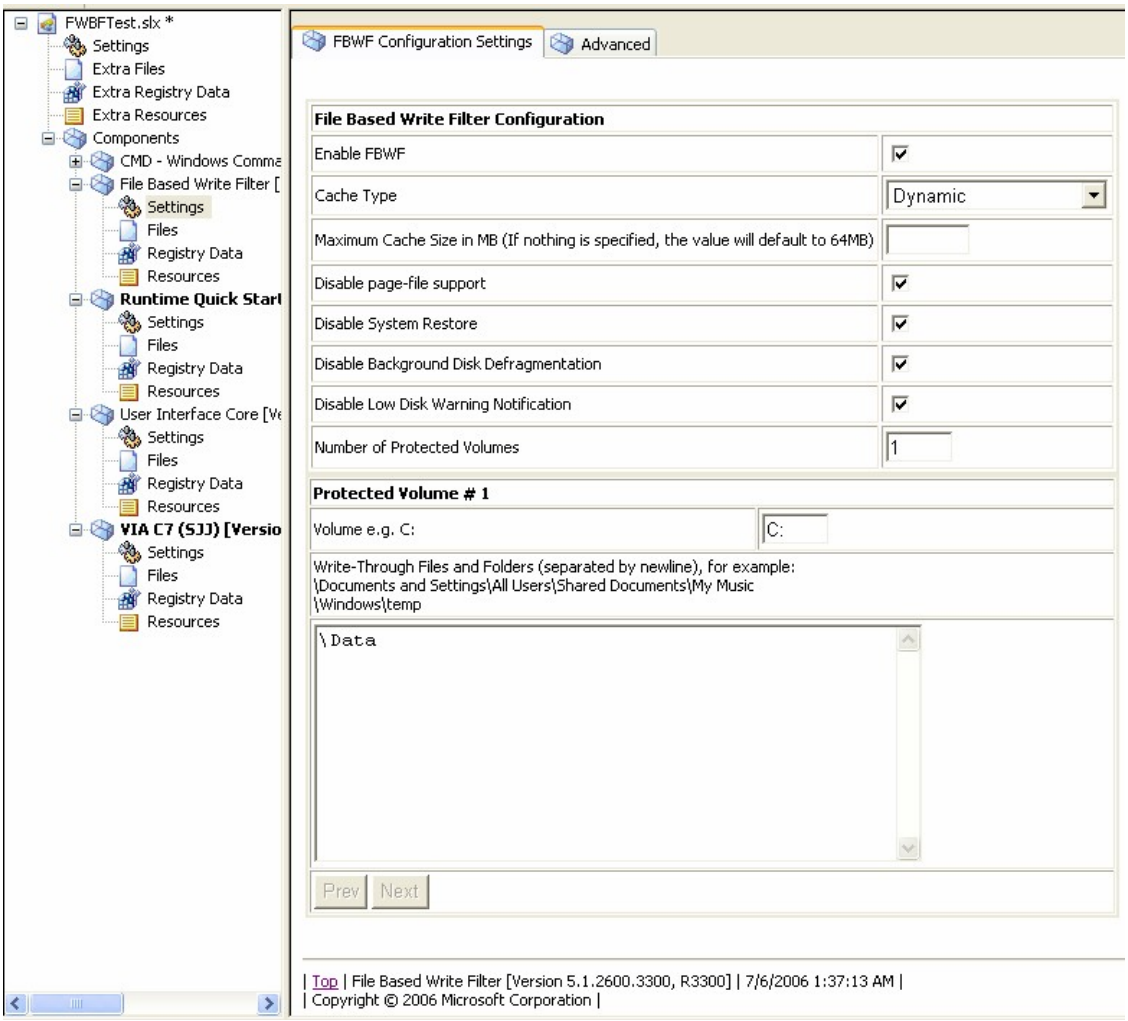

Fig 3 – FBWF Settings

Item	Description
Enable FBWF	Check to enable.
Cache Type	Select **Dynamic**, **Dynamic Compressed**, or **Pre-allocated**. **Pre-allocated** is the default.
Maximum Cache Size in MB	Specify overlay cache size; defaults to 64 MB. FWBF overlay cache is limited to a maximum size of 1GB.
Disable page-file support	Check to disable the creation of a page file. You can have a page file in the system to address an application's memory resource requirements. Uncheck if a page file is needed
Disable System Restore	Check to disable system restore. It is recommended that this option always be checked, or you could run into some unexpected errors.
Disable Background Disk Defragmentation	Check to disable background disk defragmentation. You don't want the system defragmenting the drive and eating up cache, but since this is a file filter, you want to have the flexibility if you are only protecting a few files.
Disable Low Disk Warning Notification	Check to disable low disk warning.
Number of protected volumes	Enter the number of volumes to protect. Each volume is configured in turn by clicking on the **Prev** and **Next** buttons.

Item	Description
Protected Volume #	Displays the current volume number.
Volume	Displays the drive letter of the current volume to be protected.
Write-Through Files and Folders	List those files and folders for Selective Write-Through, separated by newlines. Note that the full paths should not contain the drive letter or environment variables. There is no mechanism to detect possible inconsistencies between the resolved value and the drive letter.

Table 4 - File Based Write Filter Configuration Settings

10. Run Dependency Check
11. There will be two errors that need to be resolved: NT Hardware Detect is the solution for both errors.
12. Run Dependency Check again
13. Build the image.
14. Deploy the image to the target system.
15. Boot the system and let the image run through FBA on the target. Towards the end of the FBA process at phase 65520, fbwfdll.dll configures FWBF:

```
20:07:16 PM -  [FBWF] ConfigureFbwf() Start.
20:07:16 PM -  [FBWF] Getting FBWF config parameters from registry.
20:07:16 PM -  [FBWF] EnablePostFBA will be 1.
20:07:16 PM -  [FBWF] Custom threshold 64MB.
20:07:16 PM -  [FBWF] CacheType 1.
20:07:16 PM -  [FBWF] Found volume to protect, C:
20:07:16 PM -  [FBWF] Exclusion to add, \Data
20:07:16 PM -  [FBWF] Number Protection Entries 1
20:07:16 PM -  [FBWF] Volume [C:]
20:07:16 PM -  [FBWF]  Exclusion [\Data]
20:07:16 PM -  [FBWF] ConfigureFbwf() End, finalStatus = 0x0.
```

Notice that unlike EWF, FWBF doesn't create a separate EWF Volume or partition.

2.4.2 Part 2 Testing FBWF

Let's test to see if FWBF really works. We will use the FBWFMGR utility that comes with the FBWF component to control the state of FBWF.

1. On the target system, open a Command Window.
2. Type the following to see the status of FBWF:

```
C:\>FBWFMGR

File-based write filter configuration for the current session:
   filter state: enabled.
   overlay cache data compression state: disabled.
   overlay cache threshold: 64 MB.
   overlay cache pre-allocation: disabled.
   protected volume list:
     \Device\HarddiskVolume1
   Write-Through list of each protected volume:
     \Device\HarddiskVolume1:

File-based write filter configuration for the next session:
   filter state: enabled.
   overlay cache data compression state: disabled.
```

```
overlay cache threshold: 64 MB.
overlay cache pre-allocation: disabled.
protected volume list:
  \Device\HarddiskVolume1
Write-Through list of each protected volume:
  \Device\HarddiskVolume1:
    \Data
```

3. Create a new folder called test1 under c:\data and another folder called test2 under c:\.
4. Reboot the system. After the system reboots. Test1 folder should still exist. Test2 should be gone.

5. Now let's use the FBWFMGR utility. Open a Command Window.
6. Enter the following at the command line to disable FBWF:

   C:\>**Fbwfmgr /disable**

7. Reboot the system.
8. Create a new folder called test2 under c:\
9. Reboot the system. Test2 should still be there.

10. Now let's re-enable FBWF, and check out some of the error conditions that we discussed earlier. Open a Command Window.
11. Enter the following at the command line to enable FBWF:

    C:\>**Fbwfmgr /enable**

12. Open File explorer.
13. Right click on the c:\Test2 directory and select delete from the context menu. Click OK when asked to send to the recycle bin. Test 2 should disappear.
14. Right click on the c:\data\Test1 directory and select delete from the context menu. Click OK when asked to send to the recycle bin. This time you will get an error.

Fig 4 Delete File Error

You cannot move a file from a Write-Through area to a protected area. You could use the keyboard and perform a SHIFT+DEL, which will bypass the recycle bin and delete the directory.

15. Try dragging a font file like Tahoma from c:\windows\fonts to the c:\data directory. Another error occurs as expected. You could copy and paste the file if needed.

Fig 5 Move File Error

16. Drag and drop the Test1 folder in c:\data to the c:\.
17. Reboot the system. The folder should still be there.
18. Create a Wordpad file under c:\test1 called mydoc.
19. Reboot the system. The file will not be there upon reboot. C:\test1 is not part of the exclusion list so any file written to the location will be lost.
20. Let's add c:\Test1 to the Write-Through list. Open a Command Window.
21. Enter the following to add c:\test1 to the write-through list:

    C:\>**Fbwfmgr /addexclusion c: \test1**

    Note: there is a space between c: and \test1

22. Reboot the system.
23. Create a WordPad file called Mydoc2 to the c:\test1 directory.
24. Reboot the system. This time the file should still be there. You could also add other volumes (partitions or drives) to the protected list. Of course files updated in the protected region could be individually preserved with a call to FBWFMGR commitfile.

Here is a summary of the operations that were performed:

Operation	Protect to Unprotected	Unprotected to Protected
Copy / Paste	Yes	Yes
Move	No	Folders only

Table 5: File/Folder operations between protect and unprotected FBWF areas

2.5 Summary

Now there are two filters to choose from: EWF and FBWF. EWF will protect an entire volume, where almost nothing can get through. FBWF will protect on a file level, where you can select the files you want to pass through. Both filters can be used to protect the OS on a flash disk, but FBWF will allow you to pass through write access so data can be stored. As we see this selective write-through works great. FBWF will be ideal for those using databases and still protect the OS. When it comes to controlling files, FBWF is going be a popular feature.