

The SunPCi[™] File System Driver (Sun FSD)

By SunPCi Product Development Team

<http://www.sun.com/desktop/products/sunpci/>

Copyright 2003 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 94054 U.S.A. All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, SunPCi, PCNetLink and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

RESTRICTED RIGHTS: Use, duplication, or disclosure by the U.S. Government is subject to restrictions of FAR 52.227-14(g)(2)(6/87) and FAR 52.227-19(6/87), or DFAR 252.227-7015(b)(6/95) and DFAR 227.7202-3(a).

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2003 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, CA 94054 Etats-Unis. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, SunPCi, et Solaris sont des marques de fabrique, ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REpondre A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

Introduction

One of the major interoperability requirements for the SunPCi™ package was access to files resident on the host Solaris™ Operating System from the SunPCi based Windows Operating System. This is accomplished by developing a SunPCi network file system driver, known as the Sun File System Driver, or Sun FSD.

This document covers the functionality of Sun FSD for Windows NT (and server variants), Windows 2000 (and server variants), Windows XP and Windows .NET/2003 only and does not contain information specific to Sun FSD on Windows 95 or Windows 98 (and its variants).

What is Sun FSD?

Before we delve into the wherefores and whys of Sun FSD, it might be helpful to go into some background about Sun FSD.

Microsoft Windows Operating Systems allow for a PC to export (or “share”) drives and folders over the network so that they may be accessed by other PCs. The most common method a user employs to access these exported shares is by associating that share with a drive letter (e.g., aliasing drive **Z:** with the exported shared directory **ralph** on the group’s file server PC). In Microsoft parlance, a user will *map* a Windows drive letter to a shared drive, this drive being defined as an “extended” drive on the mapping PC. This mapping is usually done through Windows Explorer, but there are other mechanisms (such as the command **net use**) that can be employed to map drive letters as well.

In the same way, the SunPCi environment maps “shares”, but in this case these shares consist of directories on the host Solaris Operating System file system. However, for the SunPCi environment there are actually 2 types of extended drives: *network* drives and *file system* drives.

Network Drives

A network drive is an extended drive that is mounted from a PC to a networked PC's shared drive. The SunPCi environment can act either as the client (the PC which mounts the drive) or can be configured to act as the host (the networked PC which exports a shared drive). This mapping is primarily accomplished thorough Microsoft's network protocol known as **NETBEUI**. Additionally, there are some other services that must be configured (e.g., "Client for Microsoft Sharing" and "File and Printer Sharing") to fully engage this sharing ability. For more details and information, the user should consult the Microsoft network documentation.

While **NETBEUI** enables the sharing of exported drives between real PCs running some version of a Microsoft Windows Operating System (from Windows 95 to Windows .NET/2003), there exist 3rd party software solutions which will enable the Solaris Operating System to emulate **NETBEUI** and thus export Microsoft-type shares on the network. These 3rd party software solutions include the public domain package **Samba** as well as a Sun product called **PCNetLink**. For more details on these facilities, see www.samba.org and www.sun.com respectively.

File System Drives

A primary functional issue for the SunPCi environment is that Windows-like file system drive support places the burden upon the user (and the Solaris Operating Systems administrator) to require that the Solaris Operating System have a “PC share” enabler, such as **Samba** or **PCNetLink**, installed on the host system. Instead, the SunPCi development team created a Windows facility which would allow the SunPCi session to access Solaris Operating System

directories and files without any additional software. This facility (which actually consists of a Windows device driver and a set of Windows services) is known as the Sun File System Driver, or Sun FSD.

Thus, with the Sun FSD facility (or just Sun FSD), the SunPCi user can create an extended drive, known as a *file system* drive, which maps a Windows drive letter to a specified Solaris Operating System directory. Because both the Windows Operating System and the SunPCi environment reserves some drive letters (e.g., **A:**, **C:**), there are only 22 letters available for mapping either network or file system drives.

Mapping a file system drive following the standard Microsoft convention for mapping network drives, with the exception that the *network_name* which must be the first parameter specified (after a leading double backslash - "\\"), must be the keyword **localhost**. Thus, a command line example of mapping drive letter **I:** to the Solaris Operating System path **/export/home/katy** would be as follows:

Net use I: [\\localhost\export\home\katy](#)

Note: The Sun FSD mapping syntax and limitations for a Win98 SunPCi session is slightly different. See the SunPCi Users Guide for more details.

What's Printer Got To Do With It?

A totally non-obvious functional concern that accompanies Sun FSD is that this service is also responsible for network printer support. This linkage is a little more obvious to those users who have attached a printer to a SunPCi session.

When connecting a network printer, the user is presented with a dialog box (from the *Add Printer* wizard) which enumerates all network connections. Typically, the 2 networks to choose from are the *Microsoft Windows Network*, which is **NETBEUI**, and the *Sun Filesystem* (and the subsequent *SunPCi Host* connection), which is handled by the Sun FSD services.

This is merely pointed out as an "interesting implementation fact", and most network file access issues do not effect network printing.

But, Let's Be Careful Out There

While this powerful facility allows a natural access to files on the Solaris Operating System file system (either local or NFS mounted), there are a few cautionary points which need to be highlighted.

What's for Launch?

On the host Solaris Operating System, the SunPCi session is merely a user mode process, started by the user when they issue the **sunpci** command. Since the SunPCi session is like any other process, it inherits the same security credentials as any other non-setuid process. Thus, the SunPCi process cannot access any files that the user could not access otherwise. By extension, the user cannot access any files **through** the SunPCi process that they could not access otherwise as well. This is true irrespective of the security credentials that may have been granted with the Windows Operating System account.

More to the point, if the Solaris Operating System **root** account launches a SunPCi session, that SunPCi session has all of the security credentials and privileges of the **root** account. The non-obvious consequences of this security credential inheritance is that any access of a file on a file system drive that is mapped by the SunPCi session will have the **same** Solaris Operating System security credentials as the Solaris Operating System Session that launched the SunPCi

session. Consider this Solaris/SunPCi administrator's "nightmare" scenario: the system administrator creates a SunPCi environment (say using Windows XP) and generates a number of non-privileged Windows XP accounts which are to be used by the general user population. Before leaving for his vacation to South Padre Island, the administrator logs into the Solaris Operating System as he normally does, as **root**, and then launches the SunPCi session so that the Windows user community can log into Windows XP. The first Windows XP user logs in and maps the path **/usr/bin** by mistake (they really wanted to mount **/usr/katy/bin**) and proceeds to delete the contents of that directory. Since the SunPCi session was launched under the **root** account, the mapping is allowed and, unfortunately, the command to delete the contents of that directory is also allowed. Of course, after the deletion is complete, the host Solaris Operating System is rather unhappy.

Windows Server Deployments

Since the process which launches the SunPCi process determines the access rights, privileges and file security semantics for Sun FSD, the Sun FSD service is **disabled** by default for Windows Server installations/deployments. Thus, by default, no drive mappings of directories on the Solaris Operating System can be performed through Sun FSD, even if the Windows Administrator account is used. However, the Windows Administrator can explicitly enable the Sun FSD service by invoking the file:

C:\sun\Sun FSD\fsdon.reg

However, once Sun FSD has been enabled, all users will be able to map and unmap Sun FSD drives. Furthermore, since the typical SunPCi launch process for SunPCi Windows Server deployments is **root**, all Sun FSD file accesses, from **any** Windows account, typically will have the file access rights, privileges and file security credentials of **root**. Thus, it is cautioned that the Windows Administrator take great care when enabling Sun FSD and mapping Sun FSD drives (which is necessary for the update of the SunPCi Windows drivers if a the base SunPCi package has been updated). Additionally, it is always recommended that the Windows Administrator unmap all Sun FSD drives after any administrative activity is completed **and** to disable the Sun FSD service by invoking the file:

C:\sun\Sun FSD\fsdoff.reg

What's NFS Have To Do With It?

Another non-obvious Windows/Solaris Operating System interaction "feature" occurs whenever a user has mapped a Solaris Operating System directory which is not local to the host Solaris Operating System system. That is, if the target directory is located on an NFS device. As with any NFS mounted directory, the first access will cause the directory and file information to be cached on the local NFS client. Subsequent accesses will be very fast, until the NFS cache must be flushed or becomes "stale". Since the SunPCi session is merely a client of the host environment, any Sun FSD file system drive access will have the same characteristics as any other NFS client. Thus, there will be times when a user, say using Windows Explorer, may experience a delay when enumerating a directory or accessing a file. This delay may be further exacerbated (as would access from the Solaris Operating System itself) if there exist any issues in the network path (e.g., delays imposed by routers, repeaters, switches, etc. - the normal encumbrances for any typical network interconnect).

Functionality Restrictions

As with any complex software system, there are some inherent restrictions in functionality. In some cases, there may be a workaround that the user can employ. In other cases, the restrictions are imposed by the architecture and design of the system and a solution may be under investigation. In the following sections, the known set of Sun FSD functionality restrictions up to and including the release of SunPCi 3.0.1 are enumerated, and workarounds (if any) are detailed.

However, it should be stated that there exist some restrictions that are labeled as “will not be fixed”. The main reason why these functional restrictions may not be addressed is that it is functionally equivalent to those restrictions that exist with the 3rd party network file system **Samba**. In fact, one of the major design goals for Sun FSD is that it be functionally “bug for bug” compatible with **Samba**.

Certain NFS Directories Cannot Be Mapped

In the process of mounting Solaris Operating System NFS volumes and directories, there are special NFS “nodes” which are generated on the host (local) Solaris Operating System. Specifically, there are some file nodes, known as **NFS autofs mount points**, which cannot be mapped by a network file system. This restriction applies to Sun FSD as well as **Samba**. These “invalid” NFS autofs mount points are the non-exported parent directories of exported directories).

As an example, suppose there exists a Solaris Operating System computer **fred** and it has shared a number of directories, say **files**, **files1**, **files2**, and **files3**. Thus, the following directories are available over the Solaris Operating System network:

/net/fred/files

/net/fred/files1

/net/fred/files2

/net/fred/files3

A Windows user might assume that there are a number of ways to map these as Sun FSD file system drives. One way would be to map each full path to a separate drive letter. For example, map drive letter **Q:** to **//localhost/net/fred/files**, map drive letter **R:** to **//localhost/net/fred/files1**, and so forth. However, another Windows user might think that if they map a single drive letter to the path **//localhost/net/fred**, then this could preserve drive letters for other mappings. However, this mapping will (and must) fail.

The reason the Windows user cannot map the path **//localhost/net/fred** is because **fred** is a non-exported NFS autofs mount point.

A quick way to tell if a NFS autofs mount point is “valid” so that Sun FSD can map it is to use the Solaris Operating System **df** command. It turns out that one of the critical criteria that allow Windows to mount network drives is that there must be space allocation information associated with the mount point. This “space allocation information” is essentially the number of used or available blocks, as displayed by the **df** command. Sun FSD (and **Samba** as well) will not map a mount point that has zero used and zero available space. The internal technical reason is that the Windows Application Programming Interface (Win32 API) parses a specified file name path and performs special file system queries on each section of that parsed path. This is done as a feature of the Win32 API and the applications programmer has no control over these semantics. The semantics for this special query require that if a such query returns a zero value for the used or available blocks values, the query is considered a failure, and

therefore the specific API request will return an error code, and, typically, the application will fail.

A user can detect such NFS autofs mount points by use of the **df** command in a Solaris Operating System window as shown below:

```
ekb 694 =>df -h /net/fred
```

Filesystem	size	used	avail	capacity	Mounted on
-hosts	0K	0K	0K	0%	/net

```
ekb 695 =>df -k /net/fred/files1
```

Filesystem	kbytes	used	avail	capacity	Mounted on
fred:/files1	66514798	51161878	8701441	86%	/net/fred/files1

```
ekb 696 =>ls -ls /net/fred
```

```
total 6
 2 drwxr-xr-x  26 root    root          1024 Jan 27 10:10 files/
 2 drwxr-xr-x   6 root    root           512 Feb 19  2002 files1/
 1 dr-xr-xr-x   1 root    root            1 Feb 14 07:47 files2/
 1 dr-xr-xr-x   1 root    root            1 Feb 14 07:47 files3/
```

Note that the directories **files2** and **files3** do not show any space. In fact, if the **df** command is used on this directory, the following would appear:

```
ekb 698 =>df -h /net/fred/files3
```

Filesystem	size	used	avail	capacity	Mounted on
-hosts	0K	0K	0K	0%	/net/fred/files3

While this may seem like **files3** is another NFS autofs mount point, in this case it happens that this mount point has yet to be accessed. If an **ls** command is used, this will cause NFS to mount the file system. After that, a **df** will show the NFS cached information:

```
ekb 699 =>ls -ls /net/fred/files3
```

```
total 16
 2 drwxr-xr-x  4 root    other          512 Jul  9  2002 FredServerDrives/
 2 dr-xr-xr-x  2 ama     dos            512 Dec 17  2001 OsBuildIncludes/
 2 dr-xr-xr-x  3 root    dos            512 Feb 19  2002 auto_test_drives/
 2 -rwxr-xr-x  1 root    other          991 Jun  3  2002 clean.doug*
 2 dr-xr-xr-x  2 10043  staff          512 Feb 19  2002 roboserver/
 6 drwxrwxrwx  9 10041  staff         2560 Feb 11 16:34 template_drives/
```

```
ekb 700 =>df -h /net/fred/files3
```

Filesystem	size	used	avail	capacity	Mounted on
fred:/files3	134G	100G	32G	76%	/net/fred/files3

Note that Sun FSD will detect such NFS conditions and will attempt to force NFS to mount the file system. Note that this may cause some delay in the response of such an access because of the request to NFS to mount the network drive and is dependent upon the response of the underlying network.

This is a known issue and will not be fixed.

Short File names (8.3 Format) Are Not Supported

In the early, early days of Windows (say, in the times of Windows 3.0/3.1, the era of Win16 programs) the file systems were in their early stages of "maturity". So, there were functional "features" (otherwise known as "constraints") of these file systems that were just accepted,

probably out of pure joy that a PC would have a file system that was fairly reliable. One of those “features” was that fact that the file name specification was restricted to a specific format. That is, the file name consisted of a maximum of 8 characters (only alphanumeric, so special characters or embedded spaces can not be used), followed by a period, followed by an optional extension specification that could consist of a maximum of 3 characters. This is known as the **8.3 format** for file names, also known as “short” file names.

However, with the evolution of Windows to more mature (and more functional) operating systems and file systems, the 8.3 file name format constraint became less tolerable. Eventually (about Windows 95/98 timeframe) this restriction was lifted for most file systems. However, to maintain compatibility (because Windows wanted to guarantee that legacy Windows programs would run on later versions of the Windows Operating Systems, and, more importantly, application transition away from 8.3 file name format was slow), there had to be a way to represent files that had file name longer than the 8.3 format on Windows Operating Systems that only supported the 8.3 file name convention. Thus, for files that had file names that were longer than 8 characters and/or extension names that were greater than 3 characters, users might see these file names displayed with a truncated portion of the original file name that would contain the tilde (“~”) character and a number. The most common way for these “translated” file names to be displayed would be through the Windows **dir** command.

As an example, a file with the file name **DataForEmmy.dat** may be displayed as **DataFo~1.dat**. The unfortunate problem with this type of scheme is if there is another file in the same directory that might have the same first 6 characters, say **DataForKaty.dat**. This would have the 8.3 file name of **DataFo~2.dat**. The problem is that the legacy program (or the user) will not really know which file is which, unless they have been built more recently, or employ applications that can utilize full file names and not just 8.3 file names.

Another “feature” of Windows file systems is that they will reuse 8.3 file names as files get deleted and/or renamed. Thus, if there is sufficient file manipulation in a directory where there are a number of files that can be mapped to the same first 6 character file names, depending upon the creation/deletion/rename activities, the same 8.3 name (that is, the same ~N portion) will be reused for a different physical file. For those just dying to know tidbits like this, the Windows file system technique is known as “8.3 file name tunneling”.

So, what has this 8.3 file name convention have to do with Sun FSD? Well, all Microsoft file systems must support the 8.3 file name convention. Therefore, all 3rd party and network file systems should as well. For Sun FSD, a design decision was made **not** to support 8.3 the file name convention. Since the majority of recent (as of 2000) applications do not assume an 8.3 file name restriction, this decision makes some sense. However, this may cause problems with some legacy (older) Windows applications. In addition, a small number of more recent applications, while developed on newer Windows Operating Systems (e.g., Win98, Windows NT, etc.), may use application libraries and/or application helpers/services which are still based on legacy (8.3) conventions. This is particularly true of installation/setup programs and some less-recent versions of virus scanners. A fairly complete list of the applications/services which have 8.3 convention dependencies is listed in the SunPCi User’s Guide, Appendix A. Some of these problems are mentioned below.

This is a known problem and will not be fixed.

Installation Woes

If an application installation program/service is built using libraries and/or services that assume or support 8.3 file names, and if the installation is targeted to a Sun FSD drive, the typical Windows error message of **Path Not Found** or **File Not Found** will occur during installation.

An example of just such an installation problem, which is a result of the 8.3 file name convention, is Corel Word Perfect Office 2000. When the destination directory for Corel Word Perfect Office 2000 is specified as a Sun FSD mapped directory, all portions of the path must adhere to the 8.3 naming convention. This is because a portion of the Corel installation

program attempts to use the 8.3 file name convention when accessing files that it has previously installed. The default root directory for installation is **WordPerfect Office 2000** (note the length and the embedded spaces). While the most of the Corel installation program will copy files from the installation media into the correct directory on the Sun FSD mapped drive, there are portions of the installation program which will use the 8.3 convention to access files that it had previously installed, and it is this type of access which results in the **Path Not Found** and **File Not Found** errors.

The solution to this problem is to not use the default root directory for installation (**WordPerfect Office 2000**), but to specify a different root directory that conforms to the 8.3 naming convention, such as **WP2000**). Making this change during the installation wizard results in a successful application installation.

Application Execution Woes

Some older applications may also have hidden 8.3 file name format dependencies which result in **Path Not Found** and **File Not Found** errors during execution, even though installation was successful.

An example of this is Norton Antivirus™ 2000. This version of the popular antivirus scanner assumed that all file names conformed to the 8.3 convention, most likely because this would work on all Windows file systems, from Windows 95 to Windows .NET/2003. When scanning a Sun FSD mapped drive, the scanner would get confused because files, which it assumed were in 8.3 file name format, could not be found.

The only workaround for this type of execution error is to install the application to an emulated drive (C: or D:), rather than a network drive.

Virtual CDRom Installations

At some sites, the administrators may copy the entire contents of a CD onto a Solaris Operating System disk to facilitate networked users installing an application without having to use a physical CDRom. For most applications, this technique works well, since all that has to be done is to map a drive letter to the Solaris Operating System directory through Sun FSD and then execute the installation program/script (typically something called **setup.exe**).

However, with some installation applications there are assumptions made about the path to the root directory. The failure would manifest itself with the installation program, or sections of the installation process seeming to become inactive. That is, the installation program will launch, but will not install anything or show any activity, and Windows will still be running and active. If this occurs, this probably means that the installation program is looking for files assuming the absolute path of \ rather than the mapped path.

As an example, suppose the administrator had set up an NFS available directory that contains a number of directories that contain the entire contents of CDRoms. A typical user may map a Sun FSD drive to that directory:

Net use E: \\localhost\net\fred

Suppose that the user wishes to install a version of Adobe Photoshop™ which is located in the subdirectory **./Utilities/Graphics/Pshop**. Typically, the user will navigate to the **E:** drive and down the **./Utilities/Graphics/Pshop** directory until the installation program (**setup.exe**) is found, and then launches it.

However, this version of Adobe Photoshop assumes that a physical CDRom is being employed, and therefore all files that it requires are located at the “assumed” root directory of the CDRom. That is, the root path for all file access is assumed to be **./**. However, since we are a network drive, the actual root path is **\Utilities\Graphics\Pshop**. Thus, the Adobe Photoshop installation program will get “confused” and stop execution. The solution is to map the Sun

FSD drive directly to the directory that contains the installation image. That is, the mapping to fix the installation problem would be:

net use F: [\\localhost\net\fred\Utilities\Graphics\Pshop](#)

With this mapping, when the user navigates to the installation program, that installation program is at the “root” directory of drive **F**, and this is enough to convince the installation program to properly execute.

This is a known issue and will not be fixed.

Embedded UNC Paths Within Office Files Are Not Supported

In Windows, there are a number of ways to identify a file. The most common way is by its drive letter path. This is the path that contains a drive letter, directory path and the target file name (e.g., **R:\Emmy\graduation.jpg**). Since drive letters may be used to map network file system drives, and the fact that drive letters can be reused and reassigned, this drive letter type of file location specification is transient. Thus, there exists a file location specification which is “drive letter” independent. This is known as the **Universal Naming Convention**, or **UNC**. While most of the path for the file remains the same, the “drive letter” portion is replaced by a specification that describes the network path. Thus, if a user had mapped the drive letter **M**: to the path **\public\data** on the PC **bethy** and wanted to access the file **katy\simdata.dat**, the drive letter path would be:

M:\katy\simdata.dat

However, the UNC path would be:

[\\bethy\public\data\katy\simdata.dat](#)

In terms of Sun FSD, the string **localhost** serves as the PC node identifier (in the above example, the PC node identifier is **bethy**). Thus, if the Solaris Operating System directory **/export/datasets** on the machine **emmy** was to be mapped via Sun FSD to drive letter **S**:, and the user wanted to access the file **scamper\pcinfo.txt**, the drive letter path would be:

S:\scamper\pcinfo.txt

While the UNC path would be:

[\\localhost\export\datasets\scamper\pcinfo.txt](#)

Thus, file specification, even for files resident on Solaris Operating System file systems can be made without being drive letter dependent.

However, there are some applications, most notably the Microsoft Office suite of applications, for which Sun FSD UNC paths do not work properly. This is particularly true when embedding files in any of the Office applications, as when files, artwork or images are embedded or “inserted” into a document, and those files are “linked” back to the original sources. If these source files are resident on a Sun FSD file system, the Microsoft Office application may not recognize the insertion command and will not properly “link” the files. Thus, insertion of files resident on Sun FSD file systems should be avoided. The suggested workaround is to copy the files to the local emulated drive and then insert the document.

This is a known problem and is under investigation by the SunPCi Engineering Team.

Shortcuts Are Not Updated

Microsoft Windows provides a facility for creating “shortcuts” to files. This enables a user to select an icon which is a “link” to a file. If the target of the shortcut is an executable, then double-clicking on this shortcut will start target execution. As a feature, if the target of a shortcut is moved, the information contained by the shortcut is also updated without any other user intervention.

The Sun FSD file system supports shortcuts. However, since the target of a Sun FSD based shortcut is located on a Solaris Operating System file system, if that target is moved/ deleted, this information is **not** reflected in the shortcut.

This is a known issue and will not be fixed.

Media Description Not Updated in Windows Explorer

In Windows Explorer, for each drive letter, there is an associated descriptive tag, such as “*Local Disk*” for C:, or “*SUNWspci3 on 'localhost\opt\'*” for Sun FSD drive X:. There are certain conditions where this description becomes “confused”, specifically after a new physical media device has been attached.

When a new physical media device is attached (like another emulated drive D:, or an external CDROM, or a USB scanner, etc.), Windows will pick an available drive letter and associate that drive letter with the device. However, for some media devices (like another emulated drive, or a new partition), Windows may try and use drive letters that may have already been allocated, especially those drive letters earlier in the alphabet, even if those have already been allocated to other devices. An example would be if an external CDROM was assigned drive letter D:, and the user decides to attach another emulated drive. Windows would then decide to use drive letter D: for this drive and reassign the CDROM to drive letter E:. On a regular PC, this occurs when the user decides to add another physical disk, and, if applications had been installed that had hard coded the drive letter of the CDROM that did the installation, confusion reigns.

For the SunPCi session, if this “drive letter shift” occurs, especially for Sun FSD drives, Windows explorer might not update the media description. Thus, the new media may retain the descriptive tag that was associated with the displaced Sun FSD drive. If the user reboots the SunPCi session after the new media has been allocated its drive letter, the descriptive tag should be updated.

As a side note, this is a lot of the reason why, when assigning drive letters in Windows XP, the user is presented with drive letters starting at the other end of the alphabet (e.g., Z:, Y:, etc.).

This is a known problem and is under investigation by the SunPCi Engineering Team.

File Attributes May Not Work Properly

Because of the nature of network file systems, and due to the functional limitations of Sun FSD, file attributes, such as Read-Only, Write-Only, Exclusive access may not work properly.

This is a known problem and is under investigation by the SunPCi Engineering Team.

Unguarded Simultaneous File Access Between Different Machines Is Not Supported

An obvious outgrowth of a networked computer is the ability to share files across that network. Associated with the notion of sharing files, is the notion that one could have multiple machines

access the same file and maintain proper data integrity for that file. There are rigorous rules for maintaining data integrity in such situations, and involves the application program overly employing file sharing and byte range locking techniques. This is typical of the functions which distributed applications use to insure file content integrity. This file sharing and byte range locking functionality is fully supported in the SunPCi Sun FSD services.

However, there is a general Windows network file system feature that allows network file systems to cache file information locally without fear that the data may become "stale". This functionality is beyond that of explicit file sharing and byte range locking. The Sun FSD services do not support this "network file system cache coherency" functionality. (For those still dying to know these types of tidbits, this functionality is known as "Oplocks" or "Opportunistic Locking for maintenance of distributed cache coherency").

Now, what this means is that if 2 different machines (SunPCi sessions, Solaris hosts, real PCs) access and modify the same file near the same time without employing explicit file sharing or byte range locking, the contents of the file may be unpredictable. The example would be if 2 different users on 2 different machines were editing the same file. Another example would be if a user on SunPCi session (machine **A**) was executing a program while a user on real PC (machine **B**) was recompiling the same program. The net result is indeterminate.

Again, this only applies to situations where:

- The users are on different machines. That is
 - Different Solaris Operating System hosts
 - Different PCs on the network
 - Different SunPCi sessions
- File sharing and byte range locking is **not** employed

However, a very important distinction is if multiple processes from **the same** SunPCi session make simultaneous accesses to a single file (again with out explicit use of file sharing and byte range locking), data integrity **is** maintained. That is, the simultaneous accesses of a file from a **single** source machine maintain data integrity.

This is a known issue and is under investigation by the SunPCi Engineering Team.

Links Are Not Supported (Windows XP only)

Windows XP has the notion of a "link", which closely mimics the link functionally found in the Solaris Operating System. This functionality is not currently supported by the Sun FSD file system.

This is a known issue and is under investigation by the SunPCi Engineering Team.

Add Network Printer Dialog Hangs (or Stalls)

As an ancillary feature (and a Microsoft kernel requirement), the Sun FSD services also support the network printer facilities.

When a user adds a network printer, the *Network Printer* wizard will allow the user to enumerate all printers on the *Sun Filesystem*. The printers that are enumerated on the *Sun Filesystem* are exactly all the printers that can be seen from the host Solaris Operating System. More specifically, the list of printers consists of all that can be seen when executing the command **lpstat -v**. In fact, this exact command is used by the SunPCi print enumeration service (a portion of Sun FSD). Since this command is a network based command, if the network is incorrectly configured, or there are a large number (more than 500) printers on the

network, this command may take anywhere from many seconds to several minutes to complete on the Solaris Operating System. If this is the case, the Sun FSD printer enumeration service will appear to hang, as this service waits for this command to complete. If this behavior is observed, it is recommended that the user contact the network administrator to determine the root cause of the delay in the **lpstat -v** command.

This is a known issue and is under investigation by the SunPCi Engineering Team.

Windows Account Conflicts (Windows 2000, Windows XP)

A feature of all Windows Operating systems, particularly Windows NT 4 and beyond, is the creation and use of different Windows accounts, each with a configurable set of privileges. If this functionality is employed with SunPCi sessions, there are some conflicts in creation and deletion of Sun FSD drives:

- The Administrator account **cannot** delete a Sun FSD drive that was originally created by a non-Administrator account.
- The Administrator account **can** delete a Sun FSD drive that was originally created by an Administrator account.
- Non-Administrator accounts **cannot** delete a Sun FSD drive that was originally created by an Administrator account.

This is a known problem and is under investigation by the SunPCi Engineering Team.

Antivirus Scanning of Sun FSD drives

While many antivirus scanners make no “up front” distinction as to which drive letters can be scanned, generally there is **no** support for the scanning of any type of network drives. Thus, antivirus scanning of Sun FSD drives is problematic. Experience with various scanners has show that some scanners will get into a scanning infinite loop, particularly if the target directory has subdirectories. This behavior has been seen with Norton AntiVirus 2002, 2003 and Norton Corporate Edition 7.6

This is a known restriction with 3rd party software, and will not be fixed.

Conclusion - Sun FSD Rules and Regulations

Hopefully, this paper has clarified the functionality and functional restrictions of Sun FSD in the SunPCi environment.

To help codify those Sun FSD restrictions, the following is a list of Sun FSD "Rules and Regulations" which should aid the user in maximizing the use of SunPCi Sun FSD.

1. The best performance will be obtained if the directory to be mapped is local to the Solaris Operating System host and not an NFS mounted directory.
2. In general, Sun FSD drives may be used as a **source** for installations. However, care must be taken on how files are transferred from the original installation media to the Solaris Operating System file system to avoid 8.3 file name conversion (performed by default by some Solaris Operating System services).
3. In general, Sun FSD drives may be used as a **target** for installation. However, some installation programs may not recognize a network drive (even a Samba network drive) as a valid installation target. This is typically a restriction in the installation program and not a Sun FSD restriction.
4. When used as a **target** for an installation, especially for applications released prior to 2002 (prior to the release of Windows XP):
 - a. Do not use embedded spaces in the target path
 - b. Keep the file names to 8 characters in length
5. Do not attempt to map non-exported parent directories
6. Expect some initial access delays if the Sun FSD drive is not local to the host Solaris Operating System
7. Expect delays in printer enumeration if there are many network printers on the network
8. Do not use embedded UNC pathnames when inserting files or objects in a Microsoft Office (Office 97, Office 2000, Office XP, etc.) application (e.g., Word, Excel, PowerPoint, etc.)
9. When running a version of Microsoft Server (e.g., Windows NT Terminal Server Edition, Windows 2003 Server, etc.), take great care in enabling Sun FSD functionality. Only enable Sun FSD by the Administrator, and then only for SunPCi driver updates. Insure that once activity for Sun FSD has been completed, the Administrator unmaps any Sun FSD drives and restores Sun FSD back to the disabled state.