ADOBE® FLASH® PLAYER
Administration Guide for Microsoft® Windows® 8

Legal notices

Contents

Chapter 1: Introduction

About this guide

This guide provides information that can be used to manage large-scale deployments of Adobe® Flash® Player on Windows 8, typically in a corporate environment. It is intended for the following audiences:

- IT administrators who need to deploy Flash Player on their network computers and who need to understand techniques for controlling Flash Player behavior, capabilities, and functionality.
- Developers (including programmers and other authors) designing and publishing SWF applications who want to understand the implications of SWF content deployment in their network environment.
- IT managers interested in the security of SWF applications in their network environment.

This document assumes that you are at least partially familiar with Flash Player and with Adobe® ActionScript®.

To deploy the player, you must first acquire a license to do so. Distribution licenses are free of charge and can be acquired through the online licensing application at www.adobe.com/licensing/distribution. For answers to questions regarding Flash Player licensing and deployment, see the Adobe Player Distribution FAQ at www.adobe.com/licensing/distribution/faq.

Flash Player and Windows 8

Adobe® Flash® Player ships as part of Windows 8, and is used by IE in Modern Mode and Desktop Mode. This distribution mechanism has the following administrative implications:

- Flash Player is updated through Windows Update.
- Because it ships as part of the browser, you can't uninstall Flash Player, although you can disable it on user computers.
- If you already have a Flash Player update mechanism (for example, using SMS, or CAB files), it will no longer work in Windows 8.
- If your users have Windows 7, the Windows 8 upgrade install updates Flash Player, and maintains existing user data (shared objects).

Additional resources

The following sites provide information about some general topics related to the Flash Platform, Flash Player, and design and development tools. For information about sites related specifically to issues covered in this document, see the chapter that covers that issue. For an extensive list of resources specific to the topic of security, see “Additional security resources” on page 25 in “Security considerations” on page 21.

Flash Player and deployment

The following sites contain information and links to help you understand how to deploy Flash Player and work with SWF files.

- The Adobe Flash Player page at http://www.adobe.com/software/flash/about/ provides the version number of the viewing computer's Flash Player. It also lists the latest Flash Player version, including the latest version for Windows 8. Remember, however, that in Windows 8, you update the Flash Player using Windows Update. Alternatively, while a SWF file is playing, you can right-click on the SWF content and select "About Flash Player" from the context menu.
- The Flash Player Developer Center at www.adobe.com/devnet/flashplayer provides extensive information about Flash Player, including development and deployment of applications. The content includes Tech Notes, articles, and tutorials.
- The SWF File Format Specification at www.adobe.com/go/swf_file_format documents the SWF file format and describes how to write SWF files.
- The Flash Player Release notes at www.adobe.com/support/documentation/en/flashplayer/releasenotes.html contain information about features, fixes and improvements, and known issues for each version of the player.

Chapter 2: Flash Player environment

Player files and locations

The Adobe Flash Player is deployed as an ActiveX control. The Flash Player ActiveX control is used by Microsoft Internet Explorer and is an OCX file. To know the version number of the OCX file, right-click the file and select Properties. The Details tab provides the version.

Note: *There is also a stand-alone player, but it's usually installed by the development tools, not deployed by administrators.*

The installer places these OCX files in directories that differ by OS version, as follows:

- 32-bit Windows - %WINDIR%\System32\Macromed\Flash
- 64-bit Windows, 32-bit mode - %WINDIR%\SysWow64\Macromed\Flash
- 64-bit Windows - %WINDIR%\System32\Macromed\Flash
- Windows ARM - %WINDIR%\System32\Macromed\Flash

Note: *The %WINDIR% location represents the Windows system directory, such as C:\WINDOWS.*

Additional files

A utility file named FlashUtil_ActiveX.exe is installed with Flash Player.

The FlashUtil.exe file is associated with brokering the interaction between the ActiveX control and Internet Explorer. There is also a file named FlashUtil_ActiveX.dll.

Data formats used

Several file types are created or read by Flash Player. These file types are summarized in the following list.

- SWF: The SWF file format is an efficient delivery format that contains vector graphics, text, video, and sound. Flash Player executes SWF files. SWF files can be loaded into Flash Player dynamically by instructions in other SWF files.
- CFG: These are configuration files that network administrators and developers can deploy along with Flash Player to customize Flash Player settings and address certain security issues for all users. For more information, see "Administration" on page 6. End users can also create CFG files to address certain security issues for that specific user; see "The User FlashPlayerTrust directory" on page 20.
- SWC: These are SWF files that developers deliver as components for use when working in the Flash authoring environment.

- SO: Shared object files are used by Flash Player to store data locally. For example, a developer may create a game application that stores information on high scores. This data may be stored either for the duration of a Flash Player session, or persistently across sessions. In addition, Flash Player creates a persistent shared object that stores player settings, such as the amount of disk space a web site can use, if any, when creating shared objects. On Windows 8, shared object files are stored under C:\Users\username\AppData\Macromedia\Flash Player\#SharedObjects\randomDirectoryName, however, this area is partitioned, as follows:

  Local shared object The Flash Player default for shared data. When running in this mode, local shared objects are shared among other desktop browsers.

  Enhanced protected mode When users run IE with Enhanced Protected Mode, Flash Player stores local shared objects in this special, high-integrity data store. When running in enhanced protected mode, IE in the Desktop Mode and Modern Mode share the same data store.

  Shared objects are stored in a directory with a randomly generated name for security purposes. Flash Player remembers how to direct a SWF file to the appropriate location, but users of other applications outside Flash Player, such as a web browser, cannot use those applications to access the data. This limitation ensures that the data is used only for its intended purpose.
- MP3 - The compressed audio file format.
- JPG, PNG, and GIF- Image file formats. The TIF and BMP formats are not directly supported for use in SWF files.
- FLV - Flash Player compressed video format.
- FXG - Flash XML graphics format. An XML-based graphics interchange format for the Flash Platform.
- XML (eXtensible Markup Language) - Used for sending and receiving larger amounts of data with structured text.
- MXML - The XML-based language that developers use to lay out components in Flex applications.

Note: *If you block access to any of these file types, certain functionality of Flash Player may be disabled.*

Network protocols used

Flash Player can use the following network protocols:

- HTTP
- HTTPS
- RTMP (Real Time Messaging Protocol) - a proprietary protocol used with Flash Media Server to stream audio and video over the web. The default connection port is 1935.
- RTMPT - RTMP tunneling via HTTP. The default connection port is 80.
- RTMPS - RTMP tunneling via HTTPS. The default connection port is 443.
- SOAP - Simple Object Access Protocol
- UNC - Universal Naming Convention
- TCP/IP - Transmission Control Protocol/Internet Protocol
- FTP - File Transfer Protocol
- SMB - Server Message Block. SMB is a message format used by DOS and Windows to share files, directories, and devices. Flash Player can load animations and SWF files from remote SMB shares. Flash has restrictions on what Flash SWF files loaded from SMB shares are allowed to do.
- SSL - Secure Sockets Layer

- AMF - ActionScript Message Format
- RTMFP (Real-Time Media Flow Protocol): UDP-based proprietary protocol used with Flash Media Server to stream audio, video, etc.

Player processes

Although Flash Player runs as part of the browser and does not launch any new processes on the end user's computer, on Windows 8, you will see the following in process list: *Adobe (R) Flash (R) Player utility*. FlashUtil_ActiveX.exe is the image name of the process.

Player versions

If you want to learn which version of Flash Player is installed on an end user's machine without going to each machine individually, you or a developer at your site can create and distribute a SWF file that implements the `System.Capabilities.version` API and reports the results to a database using a command such as `HTTP GET` or `POST`. This technique is useful for activities such as collecting statistics on how many users have which version of Flash Player.

Chapter 3: Administration

You can create and place files on the end user's machine to manage features related to security, privacy, use of disk space, and so on.

Privacy and security settings (mms.cfg)

As a network administrator, you can enforce enterprise-wide security and privacy settings by installing a file named mms.cfg on each client machine.

The mms.cfg file is a text file. When Flash Player starts, it reads its settings from this file, and uses them to manage functionality as described in the following sections.

mms.cfg file location

Assuming a default Windows installation, Flash Player looks for the mms.cfg file in the following system directories:

- 32-bit Windows - %WINDIR%\System32\Macromed\Flash
- 64-bit Windows - %WINDIR%\SysWow64\Macromed\Flash
- Windows ARM - %WINDIR%\System32\Macromed\Flash

Note: *If your site uses the default Windows 8 configuration (32-bit Windows for IE in Desktop mode; 64-bit Windows for IE in Modern mode), you will need to maintain an mms.cfg file in both locations.*

You might use third-party administration tools , to replicate the configuration file to the user's computer. Additionally, you can use standard Windows administration techniques to hide or otherwise prevent end users from seeing or modifying the mms.cfg file on their systems.

Setting options in the mms.cfg file

This section discusses how to format and set options in the mms.cfg file. The value of some mms.cfg options can be queried through the use of ActionScript. When this is possible, the ActionScript API is noted in the option's description.

File format

The format of the mms.cfg file is a series of `name = value` pairs separated by carriage returns. If a parameter is not set in the file, Flash Player either assumes a default value or lets the user specify the setting by responding to pop-up questions, or by using Settings dialog boxes or the Settings Manager. (For more information on how the user can specify values for certain options, see "User-configured settings" on page 17.)

The options in the mms.cfg file use the following syntax:

```
ParameterName = ParameterValue
```

Only one option per line is supported. Specify Boolean parameters either as `"true"` or `"false"`, or as `1` or `0`, or as `"yes"` or `"no"`.

Comments are allowed. They start with a # symbol and go to the end of the line. This symbol can be used to insert comments or to temporarily disable directives.

White space is allowed, including blank lines or spaces around equal signs (=).

Character encoding

Some mms.cfg directives may have values that include non-ASCII characters, so the character encoding of the file is significant in those cases. We support a standard text file convention: the file may use either UTF-8 or UTF-16 Unicode encoding, either of which must be indicated by including a "byte order mark" (BOM) character at the beginning of the file; if no BOM is found, Flash Player assumes that the file is encoded using the current system default code page. Many popular text editors are capable of writing UTF-8 or UTF-16 files with BOMs, although you may need to specify that as an option when saving.

Summary of mms.cfg options

The following table summarizes the options available in mms.cfg, in alphabetical order.

Option	Description
"AllowUserLocalTrust" on page 13	Lets you prevent users from designating any files on local file systems as trusted.
"AssetCacheSize" on page 12	Lets you specify a hard limit, in MB, on the amount of local storage that Flash Player uses for the storage of common Flash components.
"AVHardwareDisable" on page 8	Lets you prevent SWF files from accessing webcams or microphones.
"AVHardwareEnabledDomain" on page 9	Allows SWF files from a specific domain or IP address to access webcams or microphones.
"DisableDeviceFontEnumeration" on page 9	Lets you prevent information on installed fonts from being displayed.
"DisableNetworkAndFilesystemInHostApp" on page 14	Lets you prevent networking or file system access of any kind.
"DisableSockets" on page 14	Lets you enable or disable the use of the `Socket.connect()` and `XMLSocket.connect()` methods.
"EnableSocketsTo" on page 14	Lets you create a whitelist of servers to which socket connections are allowed.
"EnforceLocalSecurityInActiveXHostApp" on page 13	Lets you enforce local security rules for a specified application.
"FileDownloadDisable" on page 10	Lets you prevent the ActionScript FileReference API from performing file downloads.
"FileDownloadEnabledDomain" on page 10	Allows the ActionScript FileReference API to perform file downloads from a specific domain or IP address.
"FileUploadDisable" on page 11	Lets you prevent the ActionScript FileReference API from performing file uploads.
"FileUploadEnabledDomain" on page 11	Allows the ActionScript FileReference API to upload files to a specific domain or IP address.
"FullScreenDisable" on page 9	Lets you disable SWF files playing in a browser from being displayed in full-screen mode.
"LegacyDomainMatching" on page 12	Lets you specify whether SWF files produced for Flash Player 6 and earlier can execute an operation that has been restricted in a newer version of Flash Player.

Option	Description
"LocalFileLegacyAction" on page 13	Lets you specify how Flash Player determines whether to execute certain local SWF files that were originally produced for Flash Player 7 and earlier.
"LocalFileReadDisable" on page 10	Lets you prevent local SWF files from having read access to files on local hard drives.
"LocalStorageLimit" on page 11	Lets you specify a hard limit on the amount of local storage that Flash Player uses (per domain) for persistent shared objects.
"OverrideGPUValidation" on page 15	Overrides validation of the requirements needed to implement GPU compositing.
"RTMFPP2PDisable" on page 15	Specifies how the NetStream constructor connects to a server when a value is specified for peerID, the second parameter passed to the constructor.
"RTMFPTURNProxy" on page 15	Lets Flash Player make RTMFP connections through the specified TURN server in addition to normal UDP sockets.
"ThirdPartyStorage" on page 12	Lets you specify whether third-party SWF files can read and write locally persistent shared objects.

This document describes mms.cfg options that let you do the following:

- Control access to camera, microphone, and system font information (see "Privacy options" on page 8).
- Specify whether SWF files playing in a browser can be displayed in full-screen mode (see "User interface option" on page 9).
- Control access to the local file system (see "Data loading and storage options" on page 10).
- Specify adjustments to Flash Player's default security model (see "Security options" on page 12).
- Specify whether low-level socket connections are allowed (see "Socket connection options" on page 14).
- Override settings related to GPU compositing (see "GPU Compositing" on page 15).
- Specify settings related to Peer-to-Peer connections using the RTMFP protocol (see "RTMFP options" on page 15).

Where a setting has a default value, it is displayed in bold type.

Note: *Some of these capabilities are not available on all platforms. For example, Camera, Microphone, and RTMFP are not available as features on Windows 8 Modern IE.*

Privacy options

Settings in this category let you: disable the use of camera and microphone devices to capture video and audio streams; and disable the ability to view the list of system fonts installed on a user's computer.

AVHardwareDisable

```
AVHardwareDisable = [ 0, 1 ] (0 = false, 1 = true)
```

If this value is set to 1, SWF files cannot access webcams or microphones. If this value is 0 (the default), the Settings Manager or Settings tabs let the user specify settings for access to webcams and microphones. (See "Privacy options" on page 17.)

If this value is set to 1, the privacy pop-up dialog never appears. However, the user can still access the Privacy tab and the Settings Manager, as well as tabs to let them designate which camera or microphone an application can use. These settings appear functional, but any choices the user makes are ignored. Also the recording level meter on the Microphone tab is disabled, and the Camera tab does not bring up a thumbnail of what the camera is seeing.

Note: *In ActionScript, an author can query the* `System.capabilities.avHardwareDisable` *property to determine the value of this setting.*

AVHardwareEnabledDomain

```
AVHardwareEnabledDomain = domain name or IP address
```

If the `AVHardwareDisable` value is set to 1, it prohibits SWF files from accessing webcams or microphones. The `AVHardwareEnabledDomain` settings provide exceptions to that rule. They create a "white list" of approved domain names or IP addresses to which data can be transmitted using a webcam or microphone. If the active security context is in the list of domains and IP addresses then camera and microphone access will be allowed. Otherwise it will default to the behavior specified by the `AVHardwareDisable` setting.

This value must be set to a string containing a full domain name or IP address. The string value must exactly match the domain name or IP address to be enabled. Strings with wildcards such as *.adobe.com or 10.1.1.* are not supported. The mms.cfg file can contain multiple `AVHardwareEnabledDomain` settings to allow access to multiple domains and IP addresses.

For example the following settings only allow access to cameras or microphones when connected to servers with the domain name test.mydomain.com or the IP address 10.1.1.10:

```
AVHardwareDisable=1
AVHardwareEnabledDomain=test.mydomain.com
AVHardwareEnabledDomain=10.1.1.10
```

DisableDeviceFontEnumeration

```
DisableDeviceFontEnumeration = [ 0, 1 ] (0 = false, 1 = true)
```

This setting controls whether the `Font.enumerateFonts()` method in ActionScript 3.0 and the `TextField.getFontList()` method in ActionScript 1.0 and 2.0 return the list of fonts installed on a user's system. If this value is 1, information on installed fonts cannot be returned. If this value is 0 (the default), information on installed fonts can be returned.

User interface option

The setting in this category determines whether SWF files playing in a browser can be displayed in full-screen mode.

FullScreenDisable

```
FullScreenDisable = [ 0, 1 ] (0 = false, 1 = true)
```

This setting controls whether a SWF file playing in a browser can be displayed in full-screen mode; that is, taking up the entire screen and thus obscuring all application windows and system controls. If you set this value to 1, SWF files that attempt to play in full-screen mode fail silently. The default value is 0.

In IE in Desktop Mode, users exit full-screen mode with the Esc key, swipe left, or swipe right. In IE in Modern Mode, they exit full-screen mode with the Esc key, or a swipe from the top, bottom, left, or right.

Full-screen mode is implemented with a number of security options already built in, so you might choose to disable it only in specific circumstances. To learn more about full-screen mode, see www.adobe.com/go/fullscreen.

Data loading and storage options

Settings in this category let you do the following:

- prevent local SWF files from reading local files
- prevent uploading and downloading of files between remote servers and local file systems
- limit (optionally to zero) the amount of local storage web sites can use for persistent shared objects
- limit (optionally to zero) the size of the asset cache (also called the cross-domain cache)
- prevent third-party SWF files from reading and writing locally persistent shared objects

Note: *Disabling features may cause certain web sites and applications to work incorrectly. If these features are needed for applications running in your environment, do not disable them.*

LocalFileReadDisable

```
LocalFileReadDisable = [ 0, 1 ] (0 = false, 1 = true)
```

Setting this option to 1 prevents local SWF files from having read access to files on local hard drives; that is, local SWF files can't even run. In addition, remote SWF files are unable to upload or download files. The default value is 0.

If this value is set to 1, ActionScript cannot read any files referenced by a path (including the first SWF file that Flash Player opens) on the user's hard disk. Any ActionScript API that loads files from the local file system is blocked. File upload/download via methods of the FileReference and FileReferenceList ActionScript APIs are also blocked if this flag is set. In addition, any values set for FileDownloadDisable and FileUploadDisable are ignored.

It is important to remember that, except for uploading and downloading files, the only SWF files that can read local files are SWF files that are themselves local. Therefore, you do not need to use this option to prevent remote SWFs from reading local data; that is always prevented anyway.

If this option is disabled, the ActionScript methods `FileReference.browse()` and `FileReferenceList.browse()` are also disabled.

Note: *In ActionScript 1.0 and 2.0, an author can use the `System.capabilities.localFileReadDisable` API to query the value of this setting. The corresponding ActionScript 3.0 API is `Capabilities.localFileReadDisable`.*

FileDownloadDisable

```
FileDownloadDisable = [ 0, 1 ] (0 = false, 1 = true)
```

If this value is set to 1, the ActionScript `FileReference.download()` method is disabled; the user is not prompted to allow a download, and no downloads using the FileReference API are allowed. If this value is set to 0 (the default), Flash Player allows the ActionScript `FileReference.download()` method to ask the user where a file can be downloaded to, and then Flash Player downloads the file after the user approves the file save location. Files are never downloaded without user approval.

FileDownloadEnabledDomain

```
FileDownloadEnabledDomain = domain name or IP address
```

If the `FileDownloadDisable` value is set to 1, it prevents SWF files from downloading files using the FileReference API. The `FileDownloadEnabledDomain` settings provide exceptions to that rule. They create a "white list" of approved domain names or IP addresses from which files can be downloaded. If the active security context is in the list of domains and IP addresses then file downloads will be allowed. Otherwise it will default to the behavior specified by the `FileDownloadDisable` setting.

This value must be set to a string containing a full domain name or IP address. The string value must exactly match the domain name or IP address to be enabled. Strings with wildcards such as *.adobe.com or 10.1.1.* are not supported. The mms.cfg file can contain multiple `FileDownloadEnabledDomain` settings to allow downloading from multiple domains and IP addresses.

For example the following settings only allow files to downloaded from servers at test.mydomain.com and 10.1.1.10:

```
FileDownloadDisable=1
FileDownloadEnabledDomain=test.mydomain.com
FileDownloadEnabledDomain=10.1.1.10
```

FileUploadDisable

```
FileUploadDisable = [ 0, 1 ] (0 = false, 1 = true)
```

If this value is set to 1, all `FileReference.upload()`, `FileReference.browse()`, and `FileReferenceList.browse()` activity is disabled; the user is not prompted to upload files, and no uploads using the FileReference API are allowed. If this value is set to 0 (the default), Flash Player allows files to be uploaded using the FileReference API. The user is prompted to select a file to upload and to approve the selection. Files are never uploaded without user approval.

FileUploadEnabledDomain

```
FileUploadEnabledDomain = domain name or IP address
```

If the `FileUploadDisable` value is set to 1, it prevents SWF files from uploading files using the FileReference API. The `FileUploadEnabledDomain` settings provide exceptions to that rule. They create a "white list" of approved domain names or IP addresses to which files can be uploaded. If the active security context is in the list of domains and IP addresses then file uploads will be allowed. Otherwise it will default to the behavior specified by the `FileUploadDisable` setting.

This value must be set to a string containing a full domain name or IP address. The string value must exactly match the domain name or IP address to be enabled. Strings with wildcards such as *.adobe.com or 10.1.1.* are not supported. The mms.cfg file can contain multiple `FileDownloadEnabledDomain` settings to allow uploading to multiple domains and IP addresses.

For example the following settings only allow files to be uploaded to servers at test.mydomain.com and 10.1.1.10:

```
FileDownloadDisable=1
FileDownloadEnabledDomain=test.mydomain.com
FileDownloadEnabledDomain=10.1.1.10
```

LocalStorageLimit

```
LocalStorageLimit = [ 1, 2, 3, 4, 5, 6 ] (1 = no storage, 2 = 10 KB, 3 = 100 KB, 4 = 1 MB, 5
= 10 MB, 6 = user specifies upper limit)
```

This value specifies a hard limit on the amount of local storage that Flash Player uses (per domain) for persistent shared objects. The user can use the Settings Manager or Local Storage Settings dialog box to specify local storage limits (see "Local storage options" on page 18). If no value is set here and the user doesn't specify storage limits, the default limit is 100 KB per domain. If this value is set to 6 (the default), the user specifies the storage limits for each domain.

If LocalStorageLimit is set, the Local Storage tab shows the limit specified. and the user can use this tab as if the limit does not exist. If the user sets more restrictive settings than the value set by LocalStorageLimit, they are honored (and displayed the next time the Settings dialog box is loaded). However, if the user selects settings higher than the limit set by LocalStorageLimit, the user's settings are ignored.

The local file storage limit is best obtained from the Settings dialog box, because this security setting is just a maximum value, and the user may have set a lower limit.

ThirdPartyStorage

```
ThirdPartyStorage = [ 0, 1 ] (0 = false, 1 = true)
```

Third party refers to SWF files that are executing within a browser and have an originating domain that does not match the URL displayed in the browser window.

If this value is set to 1, third-party SWF files can read and write locally persistent shared objects. If this value is set to 0, third-party SWF files cannot read or write locally persistent shared objects.

This setting does not have a default value. If it is not included in the mms.cfg file, the Settings Manager or Local Storage Settings dialog box lets the user specify whether to permit locally persistent shared objects. If the user doesn't make any changes, the default is to permit shared objects.

AssetCacheSize

```
AssetCacheSize = [ 0, number of megabytes ]
```

This value specifies a hard limit, in MB, on the amount of local storage that Flash Player uses for the storage of common Flash components. If this option is not included in the mms.cfg file, the Settings Manager lets the user specify whether to permit component storage. However, the user can't specify how much local storage space to use. The default limit is 20 MB.

Setting this value to 0 disables component storage, and any components that have already been downloaded are purged the next time Flash Player runs.

Security options

These options let you modify the default Flash Player security model. For more information on the security model, see "Security considerations" on page 21

LegacyDomainMatching

```
LegacyDomainMatching = [ 0, 1 ] (0 = false, 1 = true)
```

This setting controls whether to allow a SWF file produced for Flash Player 6 and earlier to execute an operation that has been restricted in a newer version of Flash Player.

Flash Player 6 made security sandbox distinctions based on superdomains. For example, SWF files from www.example.com and store.example.com were placed in the same sandbox. Flash Player 7 and later have made security sandbox distinctions based on exact domains, so, for example, a SWF file from www.example.com is placed in a different sandbox than a SWF file from store.example.com. The exact-domain behavior is more secure, but occasionally users may encounter a set of cooperating SWF files that were created when the older superdomain rules were in effect, and require the superdomain rules to work correctly.

When this occurs, by default, Flash Player shows a dialog box asking users whether to allow or deny access between the two domains. Users may configure a permanent answer to this question by selecting Never Ask Again in the dialog, or by visiting the Settings Manager. The LegacyDomainMatching setting lets you override users' decisions about this situation.

This setting does not have a default value. If it is not included in the mms.cfg file, the user can determine whether to allow the operation in a global manner (using the Settings Manager), or on a case-by-case basis (using an interactive dialog box). The values the user can choose among are "Ask," "Allow," and "Deny." The default value is "Ask".

If this value is set to 1, Flash Player behaves as though the user answers "allow" whenever they make this decision. If it is set to 0, Flash Player behaves as though the user answers "deny" whenever they make this decision.

LocalFileLegacyAction

```
LocalFileLegacyAction = [ 0, 1 ] (0=false, 1=true)
```

This setting controls how Flash Player determines whether to execute certain local SWF files that were originally produced for Flash Player 7 and earlier.

Flash Player 7 and earlier placed all local SWF files in the local-trusted sandbox. Flash Player 8 and later have, by default, placed local SWF files in either the local-with-filesystem or local-with-networking sandbox. In order for a SWF file to be placed in the local-trusted sandbox in Flash Player 8 or later, that SWF file must be designated trusted, using either the Settings Manager or a trust configuration file. This latter behavior is more secure, but occasionally users may encounter an older local SWF file that was created when the older local-trusted behavior was in effect, and must be in the local-trusted sandbox in order to work correctly. Users are notified of such situations by a dialog box, but the dialog is only a failure notification, not a means to trust the SWF file in question.

Users can restore the functionality of such SWF files on a case-by-case basis by designating them trusted in the Settings Manager, but if users encounter a large number of such files, they may also elect in the Settings Manager to place all local SWF files published for Flash Player 7 or earlier into the local-trusted sandbox. The LocalFileLegacyAction setting lets you override users' decisions about this situation.

This setting does not have a default value. If it is not included in the mms.cfg file, the user can use the Settings Manager to specify whether to place all older local SWF files into the local-trusted sandbox.

If this value is set to 1 (the most permissive setting), Flash Player behaves as though users had elected to place all older local SWF files into the local-trusted sandbox. If this value is set to 0 (the most restrictive setting), Flash Player behaves as though users had elected never to automatically place older local SWF files into the local-trusted sandbox, and also suppresses the failure notification dialog.

AllowUserLocalTrust

This setting lets you prevent users from designating any files on local file systems as trusted (that is, placing them into the local-trusted sandbox). This setting applies to SWF files published for any version of Flash.

```
AllowUserLocalTrust = [ 0, 1 ] (0=false, 1=true)
```

If this value is set to 1 (the default), Flash Player allows the user to specify whether local files can be placed into the local-trusted sandbox, through the use of the Settings Manager Global Security Settings panel and user trust files. If this value is set to 0, the user cannot place files into the local-trusted sandbox. That is, the Settings Manager Global Security Settings panel and user trust files are ignored.

EnforceLocalSecurityInActiveXHostApp

```
EnforceLocalSecurityInActiveXHostApp = "executable filename"
```

By default, local security is disabled whenever the ActiveX control is running in a non-browser host application. In rare cases when this causes a problem, you can use this setting to enforce local security rules for the specified application. You can enforce local security for multiple applications by entering a separate `EnforceLocalSecurityInActiveXHostApp` entry for each application.

The filename string must specify the executable filename only, not the full path to the executable; if you specify a full path, the setting is ignored. You can optionally include the EXE file extension.

The text encoding of mms.cfg is significant when specified filenames include non-ASCII characters; see "Character encoding" on page 7.

FullScreenInteractiveDisable

```
FullScreenInteractiveDisable = [ 0, 1 ] (0 = false, 1 = true)
```

If this value is set to 0 (the default), applications can enable full-screen with text input mode (known as full-screen interactive mode). To use full-screen interactive mode, an application must prompt the user for a key-press or mouse-click to enter the mode. Once in full-screen interactive mode, Flash Player displays an overlay that indicates it is in full-screen interactive mode, the domain of the current page, and an Allow button. The overlay continuously displays until the user presses Allow. Full-screen interactive mode is intended for use by full-screen games that require text and keyboard input.

Note: *IE in Modern Mode always uses full screen, so no overlay displays.*

In IE in Desktop mode, users exit full-screen interactive mode with the Esc key, swipe left, or swipe right. In IE in Modern mode, they exit full-screen interactive mode with the Esc key, or a swipe from the top, bottom, left, or right.

In past releases, this feature was available in AIR applications only.

DisableNetworkAndFilesystemInHostApp

```
DisableNetworkAndFilesystemInHostApp = "executable filename"
```

When an ActiveX control is running within an application specified, it will be as though the HTML parameter `allowNetworking="none"` had been specified. That is, no networking or file system access of any kind will be permitted, and the SWF running in the Flash Player will run without the ability to load any additional media or communicate with any servers. You can enforce local security for multiple applications by entering a separate `DisableNetworkAndFilesystemInHostApp` entry for each application.

The filename string must specify the executable filename only, not the full path to the executable; if you specify a full path, the setting is ignored. You can optionally include the EXE extension.

The text encoding of mms.cfg is significant when specified filenames include non-ASCII characters; see "Character encoding" on page 7.

Socket connection options

These settings determine whether socket connections using the ActionScript Socket and XMLSocket classes are permitted. Socket connections also require the presence of a socket policy file on the target server; for more information, see "Data loading through different domains" on page 24.

DisableSockets

```
DisableSockets = [ 0, 1 ] (0 = false, 1 = true)
```

This option enables or disables the use of the `Socket.connect()` and `XMLSocket.connect()` methods. If you don't include this option in the mms.cfg file, or if its value is set to 0, socket connections are permitted to any server. If this value is set to 1, no socket connections are allowed. However, if you want to disable some but not all socket connections, set this value to 1 and then use EnableSocketsTo to specify one or more servers to which socket connections can be made.

EnableSocketsTo

```
EnableSocketsTo = [ host name, IP address ]
```

This option is effective only when `DisableSockets` has a value of 1; it creates a whitelist of servers to which socket connections are allowed. Unlike most other mms.cfg options, you can use this option as many times as is appropriate for your environment. Note that the servers specified are target servers, to which socket connections are made; they are not origin servers, from which the connecting SWF files are served.

The values specified here must exactly match the values specified in the ActionScript `connect()` methods. If you specify an IP address here, but the `connect()` method specifies a host name, the method fails even if that host name resolves to the specified IP address. Similarly, if you specify a host name here but the `connect()` method specifies an IP address, the method fails.

Using this option does not take the place of a socket policy file on the target server. That is, this option has no effect if the specified server does not have a socket policy file.

GPU Compositing

Flash Player rendering can use the graphics processor unit (GPU) on the video card to accelerate image compositing. In certain circumstances, Flash Player disables GPU compositing. The option in this section lets you override this action and enable GPU compositing.

OverrideGPUValidation

```
OverrideGPUValidation  = [ 0, 1 ] (0 = false, 1 = true)
```

The GPU compositing feature is gated by the driver version for video cards. If a card and driver combination does not match the requirements needed to implement compositing, set OverrideGPUValidation to 1 to override validation of the driver requirements. For example, you might want GPU compositing enabled during a specific test suite, even if the video driver in the test machine doesn't meet compositing requirements. This setting overrides driver version gating but still checks for VRAM requirements.

Adobe recommends that you use this setting with care. Overriding GPU validation can result in rendering problems or system crashes due to driver issues. After completing the tests or programming tasks that require the use of this setting, consider setting it back to 0 (or removing it from the mms.cfg file) for normal operations.

RTMFP options

The mms.cfg options described in this section let you specify settings related to peer-to-peer (P2P) connections and the Real Time Media Flow Protocol (RTMFP). For more information about RTMFP, see the FAQ at www.adobe.com/go/rtmfp_faq.

RTMFPP2PDisable

```
RTMFPP2PDisable  = [ 0, 1 ] (0 = false, 1 = true)
```

This option specifies how the NetStream constructor connects to a server when a value is specified for peerID, the second parameter passed to the constructor. If RTMFPP2PDisable has a value of 0 or is not present in the mms.cfg file, a peer-to-peer (P2P) connection can be used. If this value is 1, any value specified for peerID is ignored and P2P connections are disabled; NetStream objects can connect only to Flash Media Server.

RTMFPTURNProxy

```
RTMFPTURNProxy = URL of TURN proxy server
```

If this option is present, Flash Player attempts to make RTMFP connections through the specified TURN server in addition to normal UDP sockets. TURN Servers are useful for conveying RTMFP network traffic through firewalls that otherwise block UDP packets.

The Global FlashPlayerTrust directory

Application installers can specify that certain files or directories of files that are stored on the user's computer should be *trusted* for all users, and be placed in a local-trusted sandbox. (For a discussion of sandboxes, see "Security sandboxes for local content" on page 22.) If you are deploying applications with content that should be trusted for all users on a computer, you can place trust information for that application in a directory that you specify as a trusted directory. Because information in this directory applies to all users, the directory requires administrative access.

This directory is named FlashPlayerTrust, and is called the Global FlashPlayerTrust directory. It is located alongside the directory that contains the mms.cfg file (see "mms.cfg file location" on page 6). For example, if the mms.cfg file is in C:\Windows\System32\Macromed\Flash, the location of the Global FlashPlayerTrust directory is C:\Windows\System32\Macromed\FlashPlayerTrust. (For information on specifying content as trusted only for the current user, see "The User FlashPlayerTrust directory" on page 20.)

The Global FlashPlayerTrust directory can contain any number of trust configuration files. At startup, Flash Player reads all files in this directory. The names of these files are unimportant; you can choose any filenames you want for your trust configuration files. Generally, each file contains information on a single application, but you can put information on several applications in a single file if you prefer. The configuration file is a text file; each line contains the name of a file or directory, to be trusted. If you specify a directory, all files at or below that directory level are trusted.

Create a configuration file to trust a file or directory

1. Create a new file in the Global FlashPlayerTrust directory using a text editor, and save it with a unique name.

   Choose a name for your trust configuration file that is unlikely to collide with the names of any other trust configuration files that might be installed. One good way to do this is to name the file after the particular product you are trusting. For example, if you are trusting an employee vacation application, you might call the trust configuration file EmployeeVacation.cfg.

2. Type or paste each directory path (any directory path on the user's hard disk) or file name on a new line in the file. You can paste multiple directory paths on separate lines. When you finish, your file might look similar to the following:

   ```
   # Trust all files in the Employee online calendar app
   C:\Program Files\Personnel\Employees\OnlineCalendar
   # Trust the file that checks remaining vacation days for an employee
   C:\Program Files\Personnel\Employees\VacationDaysRemaining.swf
   ```

   In this example, the SWF file is not in the same directory as the online calendar app, so it must be trusted separately.

3. Save your changes.

4. To test whether the files have been trusted correctly, you can do one of the following:

   - Run the SWF file named in the configuration file.
   - Create a SWF file in the trusted directory that displays the value returned by the ActionScript API `System.security.sandboxType` (ActionScript 1.0 or 2.0) or `Security.sandboxType` (ActionScript 3.0). Run the SWF file in a browser, not through the use of the Test Movie command in Flash. (When SWF files run via Test Movie, local security is not implemented.) The value should be `"localTrusted"`.

Chapter 4: User-configured settings

End users can set a variety of options for managing privacy and security settings when running Adobe Flash Player on their computers.

Accessing user settings (IE in Desktop Mode)

Flash Player lets users make a number of decisions regarding privacy, local storage, and so on. In IE in Desktop Mode, these settings are available to the user in three primary ways:

- Pop-up dialogs that appear when Flash Player tries to perform an activity that requires user consent, such as accessing a camera or saving data to disk.
- A tabbed set of dialogs that the user can display by right-clicking and choosing Settings from the context menu.
- The Flash Player Settings Manager, which the user can display by right-clicking and choosing Global Settings from the context menu.

Users can also display the Flash Player Settings Manager using one of the following:

- Control Panel\All Control Panel Items > Flash Player
- Start menu > type Flash > Settings > Display

In many cases, you can use the mms.cfg file to override user-specified settings, and implement more stringent or more accessible settings. For more information, see “Administration” on page 6.

Privacy options

Privacy options let the user specify whether an application can have access to the camera or microphone. Users specify these options in one of several ways, summarized below. You can use the “AVHardwareDisable” on page 8 option in the mms.cfg file to override user privacy settings.

- The first time a site tries to access the camera or microphone, a pop-up dialog appears. This dialog lets the user specify a one-time preference to allow or deny access.
- The Privacy tab lets the user allow or deny access to the camera and microphone for all applications from the current website without asking for permission each time.
- The Website Privacy Settings Panel at www.adobe.com/go/website_privacy_settings lets the user specify settings for any of the web sites that have already requested permission to use the camera or microphone.
- The Global Privacy Settings Panel at www.adobe.com/go/global_privacy_settings lets the user reset privacy options for all web sites.

Local storage options

Local storage options let the user specify whether an application can place a shared object on their computer, and the maximum size that object can attain. Applications use shared objects to store data such as user names, game scores, shopping preferences, and so on. Users specify these options in one of several ways, summarized below. You can use a number of options in the mms.cfg file to override user local storage settings; see "Data loading and storage options" on page 10.

- The first time a site tries to store information on the user's computer, a pop-up dialog appears. This dialog lets the user specify a one-time preference to allow or deny access.
- The Local Storage tab lets the user allow or deny access for local storage for all applications from the current website without asking for permission each time.
- The Website Storage Settings Panel at www.adobe.com/go/website_storage_settings lets the user specify storage settings for any of the web sites that have already requested permission to store data locally.
- The Global Storage Settings Panel at www.adobe.com/go/global_storage_settings lets the user specify storage settings for any web sites that have not yet requested permission to store data locally. This panel also lets the user choose whether to store data for a third-party local shared objects (objects being stored by a website whose originating domain does not match the URL displayed in the browser window) and whether to store common Flash components to reduce download times.

Security options

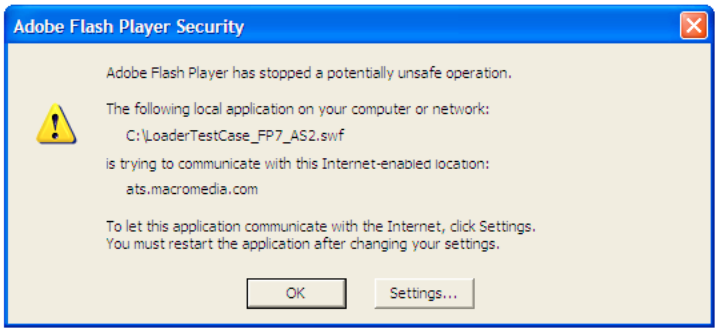
This section describes the security options available to end-users. For more information on Flash Player security in general, see "Security considerations" on page 21 You can use a number of options in the mms.cfg file to override user security options; see "Security options" on page 12.

End users should rarely need to intervene in Flash Player security decisions. However, because the Flash security model evolves over time, occasionally Flash Player encounters a situation in which Flash content attempts to perform an operation that was permitted in a previous version of Flash Player, but is no longer permitted by default. In these situations, it is impossible for Flash Player to tell whether the Flash content in question is legitimate older content that was authored before the change in rules, or malicious content that is attempting to break the newer rules. Flash Player handles these situations conservatively, guiding users toward secure choices, but offering users the ability to restore functionality of older content that has been inadvertently affected.

When Flash content attempts to use older domain matching rules, Flash Player presents a Security dialog box:

Users may interactively allow or prevent the attempted operation. If they choose "Never ask again", their allow or deny choice is remembered and used for all future instances where this dialog would be presented. Users can later see or change their remembered choice in the Settings Manager at www.adobe.com/go/global_security_settings. Their remembered choice is shown there as "Always ask", "Always allow", or "Always deny".

When Flash content attempts to use older local security rules, Flash Player presents a different dialog box:



This dialog box is only a failure notification - it does not provide an interactive allow option. However, the Settings button in this dialog box brings users to the same Settings Manager link given above. In the Settings Manager, users can affect local security rules in two ways:

- The “Always ask”, “Always allow”, or “Always deny” choice affects not only domain matching, as previously mentioned; it also governs Flash Player's behavior when content attempts to use older local security rules. However, the Ask/Allow/Deny choice affects only content that is apparently older; that is, content that specifies an older version number.
- Users can add local file system paths that are to be placed in the local-trusted sandbox (see “Security sandboxes for local content” on page 22). This enables finer-grained control than the Ask/Allow/Deny choice, and also works for Flash content of any version. Only local paths have any effect in this list; Web domains and URLs have no effect, as remote content may never be placed in a local sandbox. Also, this list, unlike the Ask/Allow/Deny choice, affects only local security rules, not domain matching rules.

Flash Player administrators can use several options in the mms.cfg configuration file to restrict users' ability to make these security choices.

- The LegacyDomainMatching and LocalFileLegacyAction options control Flash Player's behavior in the situations where, respectively, the domain matching or local security dialogs would be displayed. There is only a single user control (Ask/Allow/Deny) for both of these situations, but you can specify different options for each of them using these two mms.cfg options.
- The AllowUserLocalTrust option controls users' ability to add individual paths to the local-trusted sandbox.

For more information on these options, see “Security options” on page 12 or “Administration” on page 6.

Display options

Display options let the user specify whether to enable hardware acceleration.

The User FlashPlayerTrust directory

Application installers or end users can specify that certain files or directories of files that are stored on the user's computer should be *trusted*, and be placed in the user's local-trusted sandbox. (For a discussion of sandboxes, see "Security sandboxes for local content" on page 22.) Information on these trusted files is stored in a directory called the User FlashPlayerTrust directory. This directory registers files or directories as trusted only for the current user. (For information on registering files as trusted for all users, see "The Global FlashPlayerTrust directory" on page 16.) You can specify whether users can permit applications to be trusted; see "Security options" on page 12.

Information about trusted files can be placed in this directory in two ways:

- An administrator or end-user can create a config file and store it in the User FlashPlayerTrust directory.
- A user without administrative rights can install an application that registers itself as locally trusted.

The User FlashPlayerTrust directory is located under C:\Users\username\AppData\Roaming\Macromedia\Flash Player\#Security\FlashPlayerTrust.

For information on how to create and format these configuration files, see "The Global FlashPlayerTrust directory" on page 16.

Chapter 5: Security considerations

Clearly, it is critical to maintain the security and integrity of your users' computers when you install Adobe Flash Player. This section provides an overview of security, focusing on those aspects of particular interest to administrators deploying Flash Player. Adobe has developed a number of web pages, white papers, chapters in other books, and tech notes that address these security issues, as well as others, in more detail. For a list of these resources, see "Additional security resources" on page 25.

Security overview

As a computer system administrator, one of your primary responsibilities is to ensure the security and integrity of the data on the systems you manage. Adobe addresses Flash Player security in a number of ways, ranging from settings users can control individually to files that must be placed on servers to allow advanced applications to pass information between different domains.

Because of security issues that arise with relation to Internet access, Adobe (and formerly Macromedia) has implemented more stringent security measures with each release of Flash Player. Through improvements in the security model, Flash Player 10 by default provides much stricter limitations on potentially malicious activities than earlier versions of Flash Player. (In fact, some of these improvements can require you, application authors, or end users to specifically permit actions that were permitted by default in earlier players; see "About compatibility with previous Flash Player security models" on page 23.) Additionally, you can control a number of security-related settings through the use of a config file that you deploy on a user's system when you deploy the player.

Depending on how security settings are permitted or prohibited by the application author, the end user, or you (the administrator), Flash Player may or may not be able to download files to the local disk, upload files from the disk, write shared objects to disk (sometimes referred to as "Flash cookies"), access and run other SWF files on the local disk, or communicate between the local disk and the Internet.

In addition, there are certain activities that Flash Player can never perform, such as reading the path of a local file. For example, even if an application (SWF file) tries to upload or download a file, the application can't set the default file location for the file; the default location shown in the dialog box is the most recently browsed folder, if that location can be determined, or the desktop. Also, the application can't read from or write to the transferred file. In fact, the SWF file that initiated the upload or download can't access the uploaded or downloaded file or even the file's location on the user's disk. Another example is that a SWF file can never determine the contents of a local directory.

With regard to ensuring security of users' computers, the areas of primary interest to administrators are the following:

- How Flash uses security sandboxes to determine whether and how a SWF file on the local disk can communicate with SWF files on the network (see "Security sandboxes for local content" on page 22)
- How users can interactively allow or prohibit certain potentially malicious activities (see "User-configured settings" on page 17)
- How you can deploy a configuration file to override choices users might make with regards to security and privacy issues (see "Administration" on page 6)

The area of cross-domain security might also be of interest, although it is usually addressed by application authors. However, authors of applications you plan to deploy might request that you implement a server-side policy file, for example, to permit certain types of cross-domain file access. For more information, see "Data loading through different domains" on page 24.

Note: *Users who are working in the Flash authoring environment to create applications have access to a number of ways to implement certain security features. These techniques are described in the documentation that accompanies the authoring tool, and are not discussed in this document. If some of your users are developing Flash content, ensure that security measures that you implement are compatible with the features of the applications they are developing, and vice versa.*

Security sandboxes for local content

Client computers can obtain individual SWF files from a number of sources, such as by downloading them from external web sites or by copying them from a network server. Flash Player individually assigns local SWF files (those stored on the end-user's computer) and other resources, such as shared objects, bitmaps, sounds, videos, and data files, to security sandboxes based on their origin when they are loaded into Flash Player.

Interaction between files in different sandboxes is limited; these limitations prevent SWF files from performing operations that could introduce security breaches. Restricting how a file can interact with the local file system or the network helps keep users' computers and files safe. By default, local SWF files can communicate within the local file system or with the Internet, but not both.

Note: *The restrictions that are discussed in this section do not affect SWF files that are served from a web site on the Internet.*

Local SWF files can have the following levels of permission:

Access the local file system only (default) A local SWF file can read from the local file system and universal naming convention (UNC) network paths but cannot communicate with the Internet. These files are placed into the local-with-filesystem sandbox.

Access the network only A Flash author can specify that a SWF file be able to communicate between the local system and the network, but not have access to the local file system where it is installed. These files are placed into the local-with-networking sandbox.

Access to the local file system and the network SWF application installers, end users, and administrators can specify that a local SWF file (or multiple SWF files) be able to read from the local file system where it is installed, read and write to and from servers, and cross-script other SWF files on either the network or the local file system. These files are called trusted, and are placed into the local-trusted sandbox.

Each of these sandboxes is discussed in more detail in the following sections, and in even greater detail in white papers and other documents that are available online; see "Additional security resources" on page 25.

A Flash author can use the API `System.security.sandboxType` (ActionScript 1.0 or 2.0) or `Security.sandboxType` (ActionScript 3.0) to determine the sandbox in which a SWF file is placed. This API must be used while the SWF file is playing in a browser, not through the use of the Test Movie command in Flash. When SWF files run via Test Movie, local security is not implemented.

The local-with-file-system sandbox

By default, Flash Player places all local SWF files, including all legacy local SWF files (earlier than Flash Player 8), in the local-with-file-system sandbox. For some legacy SWF files, operations could be affected by prohibiting outside network access, but this default provides the most secure implementation. (For more information on potential issues with legacy SWF files, see "About compatibility with previous Flash Player security models" on page 23.)

From this sandbox, SWF files may read from files on local file systems or a UNC network path, but they may not communicate with the network in any way. This assures the user that local data cannot be leaked out to the network or otherwise inappropriately shared.

The local-with-networking sandbox

When a Flash author specifies that local SWF files should be assigned to the local-with-networking sandbox, the SWF files are allowed to access the network but forfeit their local file system access. However, a local-with-networking SWF file still is not allowed to read any network-derived data unless permissions are present for that action. That is, a local-with-networking SWF file has no local access, yet it has the ability to transmit data over the network and can read network data from those sites that designate site-specific access permissions.

Note: *Local HTML can't load local SWF using IE in Modern mode. It is prevented because of the security restrictions in 32-bit, 64-bit, and ARM machines.*

The local-trusted sandbox

As its name implies, placing files in this sandbox indicates that they can be trusted not to perform any malicious activities that would compromise the security of the local system or of the network. SWF files assigned to the local-trusted sandbox can interact with any other SWF files, and load data from anywhere (remote or local). Files (or entire directories) can be registered as trusted in a number of ways.

- An end user can respond to a pop-up dialog box or use the Flash Player Settings Manager to specify that a SWF file or set of files should be trusted for that user. For information on settings available to end-users, see "User-configured settings" on page 17. For information on how to control the end-users' ability to specify trusted files, see "AllowUserLocalTrust" on page 13.
- An administrator, an installer program, or an end-user can create configuration files and place them directly in the appropriate directories. The configuration files are placed in a directory named FlashPlayerTrust on the user's computer, in one of two locations. One location requires administrative access and applies to all users on a computer; see "The Global FlashPlayerTrust directory" on page 16. The other location doesn't require administrative access and applies only to the current user; see "The User FlashPlayerTrust directory" on page 20.

  When an installer installs local SWF files and HTML files, those files should be trusted, because the user consented to run an installer executable to create them. Likewise, when an installer installs an application that plays local SWF files by embedding a Flash Player, the application should be able to play local SWF files in a trusted mode, even if the embedded Flash Player would normally enforce local security. End users should exercise the same caution installing Flash applications as they would when installing any other applications on their computer.

About compatibility with previous Flash Player security models

As a result of the security feature changes over Flash Player's history, content that runs as expected in one Player version might not run as expected in later versions. In these cases, you (and end-users) can specify security settings that are less stringent than the Flash Player default settings. In other words, you can choose to run certain content in a less secure environment.

For example, local SWF files can't communicate with the Internet without a specific configuration on the user's computer. Suppose you have legacy content that was published before these restrictions were in effect. If that content tries to communicate with the network or local file system, or both, Flash Player stops the operation. By default, a Security pop-up question appears, and the user must explicitly provide permission for the application to work properly.

To prevent users from having to provide permission explicitly, Flash provides a number of options.

- An end-user can use the Global Security Settings Panel at www.adobe.com/go/global_security_settings to specify that a file or set of files should be trusted.
- An end-user, or an installer program run without administrative access, can place a local configuration file on the user's machine to specify that a file or set of files should be trusted (see "The User FlashPlayerTrust directory" on page 20).
- You, or an installer program run with administrative access, can place a global configuration file on the user's machine to specify that a file or set of files should be trusted (see "The Global FlashPlayerTrust directory" on page 16).
- You can set an option in a configuration file you deploy to users' machines, the mms.cfg file, to always allow or always deny such access (see "Security options" on page 12 in "Administration" on page 6).
- You can run a free, command-line utility called the Local Content Updater on the legacy SWF files. The Local Content Updater lets you change the security sandbox that the SWF file operates in when it is played as a local file in Flash Player 8 and above. It can add, remove, or check for local-with-networking privileges, operating on one or many SWF files. For more information or to download the utility, see Local Content Updater at www.adobe.com/support/flashplayer/downloads.html#lcu.

Data loading through different domains

To make data from a web server available to SWF files from other domains, you may be asked by a Flash author to create a policy file on your server. Policy files are XML files placed in a specific location on your server.

Policy files affect access to a number of assets, including the following:

- Data in bitmaps, sounds, and videos
- Loading XML and text files
- Importing SWF files from other security domains into the security domain of the loading SWF file
- Access to socket and XML socket connections

There are two types of policy files—URL policy files and socket policy files.

- URL policy files provide a way for the server to indicate that its data and documents are available to SWF files served from certain domains or from all domains.
- Socket policy files enable networking directly at the lower TCP socket level, using the Socket and XMLSocket classes.

Requirements for implementing policy files are more strict in Flash Player 10 than in earlier versions of Flash Player. For more information, see the Flash Player Developer Center at www.adobe.com/devnet/flashplayer, as well as the information listed below in "Additional security resources" on page 25.

Additional security resources

For quick reference, the following list summarizes various web pages and documents related to security, many of which are mentioned elsewhere in this chapter or in other chapters in this book.

- Flash Player Security and Privacy (www.adobe.com/products/flashplayer/security/). This document provides an overview of how Flash Player maintains users' privacy.
- Security Topic Center (www.adobe.com/devnet/security/). This document provides information on security and links to a number of other resources.
- Flash Player Developer Center (www.adobe.com/devnet/flashplayer). This site provides links to a number of security-related documents geared for developers.
- Flash Player Help for user setting panels (www.adobe.com/go/player_help_en). These pages explain security settings users can specify using the Settings Manager, settings dialog boxes, and questions that might pop up while a SWF is running.
- "How do I let local Flash content communicate with the Internet?"(www.adobe.com/go/4c093f20). This document describes the security issues involved in allowing (or preventing) local SWF files from accessing the Internet.
- The Flash Player Local Content Updater (www.adobe.com/support/flashplayer/downloads.html#lcu) lets you change the security sandbox in which SWF files written for Flash Player 7 and earlier operate.