Adobe® Flash® Player 32.0 Administration Guide

December 9, 2020

Contents

Introduction

Why install Flash Player?

Adobe® Flash® Player is the software that allows computers to play multimedia content contained in SWF (pronounced “swiff”) files, which are the main type of file used by Flash Player. This content can be created by Adobe® Animate, Adobe® Flash® Builder™, or other tools that output the SWF file format.

SWF content can range from simple animations to online advertisements to complete applications that communicate over the Internet.

Adobe Flash Player is available in multiple forms. In its most popular form, it is embedded in a web browser as a plug-in or an ActiveX control.

You may have been asked to deploy Adobe Flash Player in your network environment because someone in your company has built a SWF application for business use, or because there is external SWF content that employees want to have access to.

To deploy Adobe Flash 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.

NOTE: You must use your company or organization email address when requesting a distribution license. Public email addresses (such as gmail.com, yahoo.com, hotmail.com, and so on) are not allowed.

For answers to questions regarding Adobe Flash Player licensing and deployment, see the Adobe Player Distribution FAQ at www.adobe.com/licensing/distribution/faq.

Additional resources

The following sites provide information about the Adobe Flash Platform, Adobe Flash Player, and related design and development tools. For information about sites related specifically to issues covered in this document, see the chapter that covers that issue.

For example, for an extensive list of resources specific to the topic of security, see *Additional security resources* in *Security considerations*.

For the latest version of this guide, see the Adobe Flash Player Administration Guide section of the Flash Player Developer Center at www.adobe.com/devnet/flashplayer/articles/flash_player_admin-_guide.html.

Flash Player and deployment

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

- The Adobe Flash Player product page at www.adobe.com/products/flashplayer.html provides information on a number of topics relating to installing, using, and deploying Flash Player. It also contains links to documents that can answer just about any question you might have about Flash Player, locations for downloading the player, user forums, and so on. Much of the information in this document is excerpted from documents available from the Support Center.
- The Flash Player Developer Center Archive at helpx.adobe.com/air/archived-docs-download.html provides extensive information about Adobe 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 Adobe Flash Player Release notes at www.adobe.com/support/documentation/en/flash-player/releasenotes.html contain information about features, fixes and improvements, and known issues for each version of the player.

Design and development tools

Adobe provides the following tools for developing SWF files (the file format that executes in Flash Player):

- Animate (www.adobe.com/products/animate/)

  In Animate (formerly Flash Professional), designers and developers create FLA files that contain graphical elements, a timeline, and ActionScript code. Both ActionScript 2.0 and ActionScript 3.0 are supported. FLA files are compiled into SWF files.

- Adobe® Flash® Builder®™ (www.adobe.com/products/flash-builder.html/)

  In Adobe® Flash® Builder™ 4 (formerly Adobe® Flex® Builder™), developers and designers create MXML files and FLA files using the open source Flex framework. They can also use ActionScript 3.0. Both MXML and ActionScript compile into SWF files.

- Adobe® Flex® (www.adobe.com/products/flex/)

  In Flex, developers create MXML files that describe the visual and code elements of their applications. They can also use ActionScript 3.0. Both MXML and ActionScript compile into SWF files.

Flash Player environment

Player files and locations

Adobe Flash Player is normally deployed as a browser plug-in or ActiveX control. For each player environment, two versions of Flash Player are available—a "Content Debugger" version for developers, and a "Release" version for end users. The Content Debugger player implements the same feature set as the Release player, but also displays run-time errors. Each of these implementations is described in this section.

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

Firefox/Mozilla NPAPI plug-in architecture

Browsers that implement the Netscape Plug-In API (NPAPI), including Mozilla Firefox and Apple's Safari browser use this plug-in.

Windows NPAPI plug-in filenames and locations

On Windows, files named NPSWF32.dll (NPSWF64.dll for 64-bit Windows) and flashplayer.xpt are installed.

NOTE: For Adobe Flash Player 11.2 and later, the DLL file name also includes the build number. For example, NPSWF32_11_2_202_228.dll (32-bit Windows) and NPSWF64_11_2_202_228.dll (64-bit Windows).

The installer places these 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, 64-bit mode - %WINDIR%\System32\Macromed\Flash

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

The Windows plug-in installer also places a broker application called FlashUtil*nnn*_Plugin.exe in the same directory as the Flash Player Plug-in DLL. The *nnn* represents the version number and changes with each release. FlashUtil*nnn*_Plugin.exe includes functionality required by Windows Vista and above, and as an upgrade and uninstall mechanism.

NOTE: For Flash Player 11.2 and later, the broker file name also includes the build number. For example, FlashUtil32_11_2_202_228_Plugin.exe (32-bit Windows) and FlashUtil64_11_2_202_228_Plugin.exe (64-bit Windows).

macOS NPAPI plug-in filenames and locations

On macOS, files named Flash Player.plugin and flashplayer.xpt are installed. These files are placed in the /Library/Internet Plug-Ins folder.

Linux plug-in filenames and locations

On Linux, files named libflashplayer.so and flashplayer.xpt are installed. The install location is dependent upon the browser, Linux distro, and distro version.

Chromium PPAPI plug-in architecture

Chromium-based browsers (such as Opera) on Windows and macOS use this plug-in.

Windows PPAPI plug-in filenames and locations

On Windows, files named pepflashplayer32.dll (pepflashplayer64.dll for 64-bit Windows) and manifest.json are installed.

NOTE: The dll file name also includes the build number. For example, pepflashplayer32_22_0_0_157.dll (32-bit Windows) and pepflashplayer64_22_0_0_157.dll (64-bit Windows).

The installer places these 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, 64-bit mode - %WINDIR%\System32\Macromed\Flash

NOTE: The %WINDIR% location represents the Windows system directory, such as C:\WINDOWS. The Windows PPAPI plug-in installer also places a broker application called FlashUtil*nnn*_pepper.exe in the same directory as the Flash Player PPAPI Plug-in DLL. The nnn represents the version number and changes with each release. FlashUtil*nnn*_pepper.exe includes functionality required by Windows Vista and above, and as an upgrade and uninstall mechanism.

macOS PPAPI plug-in filenames and locations

On macOS, files named PepperFlashPlayer.plugin and manifest.json are installed. These files are placed in the /Library/Internet Plug-Ins/PepperFlashPlayer folder.

Linux PPAPI plug-in filenames and locations

On Linux, files named libpepflashplayer.so and manifest.json are installed. The install location is dependent upon the browser, Linux distribution, and version.

ActiveX Control on Windows

The ActiveX control is used by Microsoft Internet Explorer as well as some legacy Windows applications. The player is an OCX file whose name reflects the version number.

NOTE: For Flash Player 11.2 and later, the .ocx file name also includes the build number. For example, Flash32_11_2_202_228.ocx (32-bit) and Flash64_11_2_202_228.ocx (64-bit Windows).

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, 64-bit mode - %WINDIR%\System32\Macromed\Flash

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

NOTE: The Adobe Flash Player ActiveX control on Windows 8.1 and above is a built-in component of Internet Explorer and Edge, managed directly by Microsoft via Windows Update. Adobe Flash Player installations managed directly by Windows Update can only be modified by the operating system. These installations cannot be modified by Adobe-distributed installers or uninstallers.

NOTE: Windows 8.0 is no longer supported. Users are strongly encouraged to upgrade to Windows 8.1 or Windows 10 to continue to receive Flash Player updates.

Additional files

On Windows, Adobe Flash Player includes helper utilities that facilitate automatic updates and broker requests from sandboxed processes on Windows Vista and higher.

FlashUtil.exe

A utility file named FlashUtil*nnn*_ActiveX.exe is installed with Adobe Flash Player. The utility is versioned with the control; for example, FlashUtil10h_ActiveX.exe is installed with the control Flash10h.ocx.

NOTE: For Flash Player 11.2 and later, the FlashUtil file name includes the entire build number. For example, FlashUtil32_11_2_202_228_ActiveX.exe (for 32-bit) and FlashUtil64_11_2_202_228_ActiveX.exe (for 64-bit).

The FlashUtil*nnn*.exe file is associated with the notification auto-update functionality, uninstallation, and brokering the interaction between the ActiveX control and Internet Explorer (brokering only occurs on Windows Vista and above). There is also a file named FlashUtil*nnn*_ActiveX.dll.

When the browser plug-in is installed, a similar application named FlashUtil*nnn*_Plugin.exe or FlashUtil*nnn*_Pepper.exe is installed.

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. Adobe 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 Adobe Flash Player to customize settings and address certain security issues for all users. For more information, see *Administration*. End users can also create CFG files to address certain security issues for that specific user; see *The User FlashPlayerTrust directory*.
- SWC (pronounced “swik”): 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 Adobe 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.

  Shared object files are stored in the following locations:

Windows Vista and above

C:\Users\username\AppData\Roaming\Macromedia\Flash Player\#SharedObjects\randomDirectoryName

Windows 2000 and Windows XP

C:\Documents and Settings\username\Application Data\Macromedia\Flash Player\#SharedObjects\randomDirectoryName

macOS

/Users/username/Library/Preferences/Macromedia/Flash Player/#SharedObjects/randomDirectoryName

Linux

GNU-Linux ~/.macromedia#SharedObjects/randomDirectoryName

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

Player processes

Most often, Adobe Flash Player runs as a browser plug-in. When run as a stand-alone player, it launches a process named FlashPlayer.exe. The one exception to this statement is when content is played back using Internet Explorer on Windows Vista or above. In this case FlashUtil*nnn*_ActiveX.exe will be in the process list.

Flash and Flex developers can package their SWF files into stand-alone EXE files, called projectors. When a projector is run, it launches a single process, named for the projector executable filename.

Other processes are created when an Adobe Flash Player auto update occurs. GetFlash.exe, FlashUtil*nnn*_ActiveX.exe, FlashUtil*nnn*_Plugin.exe, FlashUtil*nnn*_Pepper.exe, or FlashPlayerUpdateService.exe will be running during an auto update request and subsequent downloading and installing of the updated player. FlashUtil*nnn*_ActiveX.exe, FlashUtil*nnn*_Plugin.exe, , or FlashUtil*nnn*_Pepper.exe processes will be visible when the Flash Player is uninstalled on Windows via Add/Remove Programs.

Player versions

Before deploying the player, you might want to know what version is already installed on an end user's machine. An easy way to determine the version of Adobe Flash Player installed is to navigate to www.adobe.com/products/flash/about; this page displays a message stating which version is installed. Alternatively, while a SWF file is playing, right-click (Windows or Linux) or Command-click (macOS) on the SWF content and then choose "About Flash Player" from the context menu.

A Master Version XML file that lists all Adobe Flash Player versions for the various supported platforms and browsers is available at https://fpdownload.macromedia.com/pub/flashplayer/masterversion/masterversion.xml.

Customers who use automation scripts to check for updates can use this file in their automation scripts.

On macOS, the file Flash Player.plugin is located in the /Library/Internet Plug-Ins folder for NPAPI browsers (Safari), or PepperFlashPlayer.plugin in the /Library/Internet Plug-Ins/PepperFlashPlayer folder for PPAPI browsers (Chromium, Opera). To determine the version number, Command-click and choose Get Info. The version number is available on the General menu.

On Windows, you can determine which version of the ActiveX control is installed by navigating to the directory where the OCX file is located (see *ActiveX Control on Windows* for the default location). Right-click on the OCX file and choose Properties, then inspect the value in the Version tab. If the OCX file isn't installed in the default location, you can determine its location and name by inspecting the following registry key, which is created when the OCX control is registered:

```
HKEY_CLASSES_ROOT\CLSID\{D27CDB6E-AE6D-11cf-96B8-444553540000}\InprocServer32
```

Similarly, you can determine the NPAPI or PPAPI Plug-in version by examining the version tab of the NPSWF32.dll or pepflashplayer32.dll file, in the same folder as the ActiveX control.

Player installation

Installers

When you license Flash Player, you will receive an email containing the license agreement and a link to the Adobe Flash Player Distribution Page to download the installers from. Save this email and use the link whenever you need to download the installation files.

The licensed installers for Flash Player are available in a number of forms. For the ActiveX control (Microsoft Internet Explorer), NPAPI Plug-In (Mozilla Firefox) and PPAPI (Chromium, Opera) plug-ins, you can download either an executable (EXE) or MSI installer.

NOTE: Flash Player ActiveX control installers are only for Windows 7 and below. For Windows 8.1 and higher, Flash Player is embedded in Internet Explorer and Edge. All updates to the embedded Flash Player ActiveX for Internet Explorer/Edge are distributed by Microsoft via Windows Updates.

If you are using Microsoft System Center Updates Publisher 4.5, you can import the Adobe Flash Player Catalog for deployment via WSUS 3.0 SP2. The Adobe Flash Player Catalog for System Center Updates Publisher supports the delivery of the ActiveX control, NPAPI and PPAPI plug-ins.

If you are using Microsoft Systems Management Server (SMS) 2003 R2, you can also import the Adobe Flash Player Catalog with the Inventory Tool for Custom Updates. The Adobe Flash Player Catalog only supports the delivery of the ActiveX control.

For macOS, a PKG installer for the NPAPI or PPAPI plug-in is provided.

On openSUSE and Red Hat Linux, use an appropriate RPM or YUM package manager. For Ubuntu, use Ubuntu Software Updater or APT delivery.

Adobe strongly recommends that you implement network installation strategies in a testing environment prior to implementation in a live environment. Adobe support cannot provide troubleshooting assistance for customized installations.

On Windows and macOS, Adobe Flash Player enables system administrators to push updates to the client systems they manage. The update mechanism supports background updates, which do not require actions from the end-user. For more information, see *Performing a background update*.

For Windows 8.1 and higher, updates to Adobe Flash Player for Internet Explorer and Edge (ActiveX) are distributed by Microsoft, via Windows Update. Adobe Flash Player Installers and Uninstallers cannot modify instances of the ActiveX Flash Player installed via Windows Update.

Uninstalling Flash Player

To minimize the potential for installation issues, administrators should consider uninstalling any existing copies of Adobe Flash Player and rebooting target systems before installing new versions.

NOTE: Beginning with Flash Player 11.5, uninstalling Flash Player resets the AutoUpdateDisable and SilentAutoUpdateEnable settings in mms.cfg to their default values of AutoUpdateDisable=0 and SilentAutoUpdateEnable=0 (Notification Updates enabled, Background Updates disabled).

NOTE: For administrators running the Flash Player uninstaller as part of their deployment process, who also configure update settings via mms.cfg, any custom changes that you have made to either AutoUpdateDisable and/or SilentAutoUpdateEnable must be re-deployed with mms.cfg as part of the upgrade installation.

Uninstalling on Windows

Before uninstalling Adobe Flash Player, all applications using Adobe Flash Player must be closed. Historically, it was common to find copies of the Adobe Flash Player plug-in active in popular instant messaging applications (AOL instant Messenger, Yahoo Messenger, etc). Instances of applications hosted in Google Chrome, like the Google Hangouts client, also can keep Chrome instances running in the background.

Use the uninstaller available at www.adobe.com/go/tn_14157 to uninstall any version of the player.

Silent mode

Beginning with the Adobe Creative Suite 5 and Adobe Flash Player (10.1.r52 and 10.1.r53), the /silent method of uninstalling the player is deprecated in favor of "-uninstall".

To uninstall only one Adobe Flash Player type, include the player type (active-x, plugin, or pepper-plugin) as an argument when uninstalling silently, as follows:

```
uninstall_flash_player.exe -uninstall
```

To uninstall only one particular Flash Player type include the player type (active-x plugin, or pepper-plugin) as an argument when uninstalling silently, as follows:

- ActiveX Control: `uninstall_flash_player.exe -uninstall activex`
  - Windows 7 and prior. Microsofts embeds Flash Player for IE/Edge on Windows 8 and above and Adobe's Flash Player uninstaller will NOT remove the embedded Flash Player ActiveX Control.
- NPAPI Plugin: `uninstall_flash_player.exe -uninstall plugin`
- PPAPI Plugin: `uninstall_flash_player.exe -uninstall pepperplugin`

For more information, see Install earlier Flash Player version | Internet Explorer.

Uninstalling on Linux

To uninstall Flash Player on Linux, log in as root and use one of the following commands, depending on the method used to install the plug-in originally (via rpm, yum, or APT):

NPAPI Plugin:

```
rpm -e flash-plugin
```

PPAPI Plugin:

```
rpm -e flash-player-ppapi
```

NPAPI Plugin:

```
yum remove flash-plugin
```

PPAPI Plugin:

```
yum remove flash-player-ppapi
```

NPAPI and PPAPI Plugin:

```
apt-get remove adobe-flashplugin
```

RPM and YUM are for Red Hat and openSUSE. You can use YUM for Red Hat.

Uninstalling on macOS

To uninstall Adobe Flash Player on macOS, make sure all browsers are closed, along with any programs that might be running SWF content, such as the Dashboard. Then use the Mac's standalone uninstaller to completely uninstall the Flash Player. You can download the appropriate uninstaller at www.adobe.com/go/tn_14157.

As of macOS 11.6, silent uninstall is available using the standalone uninstaller, as follows:

1) Extract the Adobe Flash Player uninstaller bundle (Adobe Flash Player Uninstaller.app) from the .DMG file.
2) Open a terminal window and change to the directory where the .app file is saved. For example, if the .app file is saved on the Desktop of the current user, type: cd ~/Desktop.
3) Run the uninstaller contained in the .app file using the following command:

   ```
   sudo /Adobe Flash Player.app/Contents/MacOS/Adobe Flash Player Install Manager -uninstall.
   ```

4) Type the root password to proceed with the uninstallation.

NOTE: Uninstalling Flash Player on Mac will uninstall all Player types installed (such as NPAPI and PPAPI).

Manually Uninstalling Flash Player on Macintosh

1) Reset the *Update Notification* option and unload the *SAU* daemon:
   a) Set the *Update Notification* options to default values in `mms.cfg`:

      ```
      AutoUpdateDisable=0
      SilentAutoUpdateEnable=0
      ```

   b) Run *launchctl unload* to unload the *SAU* daemon. At the prompt type:

      ```
      sudo /bin/launchctl unload
      /Library/LaunchDaemons/com.adobe.fpsaud.plist
      ```

2) Delete the following files, if found:
   a) SYSTEM NPAPI PLUGIN:

      ```
      /Library/Internet Plug-Ins/Flash Player.plugin
      /Library/Internet Plug-Ins/Flash Player Enabler.plugin
      /Library/Internet Plug-Ins/flashplayer.xpt
      ```

   b) SYSTEM PPAPI PLUGIN

```
/Library/Internet
Plug-Ins/PepperFlashPlayer/PepperFlashPlayer.plugin
/Library/Internet Plug-Ins/PepperFlashPlayer/manifest.json
```

c) SAU:

```
/Library/LaunchDaemons/com.adobe.fpsaud.plist
/Library/Application Support/Adobe/Flash Player Install
Manager/fpsaud
/Library/Application Support/Adobe/Flash Player Install
Manager/FPSAUConfig.xml
```

3) Delete install receipts:

- Delete any bundles that have the `com.adobe.pkg.FlashPlayer` bundle identifier in `/Library/Receipts`. (The `CFBundleIdentifier` entry in the `Info.plist` inside the bundle) .
- If `pkgutil` is present, run the following command:

  `sudo pkgutil --force --forget com.adobe.pkg.FlashPlayer.`

4) Remove the Flash Player PreferencePane:

- Delete `/Library/PreferencePanes/Flash Player.prefPane`.
- Remove the `com.adobe.preferences.flashplayer` entry from inside `~/Library/Preferences/com.apple.systempreferences.plist`.

5) Remove the *Install Manager* app:

If the file exists at `/Applications/Utilities/Adobe Flash Player Install Manager.app`, remove it.

EXE installation

The EXE installer can be run in either of two modes, interactive or silent. The interactive mode presents a full user interface and displays error dialogs if necessary. The silent mode does not present a user interface, and returns error codes if necessary.

Warnings and errors are written to the FlashInstall log file located at the following locations:

- 32-bit Windows: C:\\Windows\System32\Macromed\Flash\FlashInstall32.log
- 64-bit Windows: C:\\Windows\System32\Macromed\Flash\FlashInstall64.log and C:\\Windows\SysWow64\Marcomed\Flash\FlashInstall32.log

To run the EXE in silent mode, use the "-install" command line parameter:

`path to installer``\install_flash_player_active_x.exe -install`

The following exit codes are returned by the Windows EXE installers for Flash Player 10.1 and above:

Error code	Meaning
0	No errors detected
1003	Invalid argument passed to installer

Error code	Meaning
1011	Install already in progress
1012	Does not have admin permissions (W2K, XP)
1013	Trying to install older revision
1022	Does not have admin permissions (Vista, Windows 7)
1024	Unable to write files to directory
1025	Existing player in use
1032	ActiveX registration failed
1041	An application that uses the Flash Player is open. Quit the application and try again. The following exit codes are returned by the Windows EXE installers for Flash Player 9:

The following exit codes are returned by the Windows EXE installers for Flash Player 9.

Exit code	Meaning
3	Does not have admin permissions
4	Unsupported OS
5	Previously installed with elevated permissions
6	Insufficient disk space
7	Trying to install older revision
8	Browser is open

Active Directory installation

To deploy the Flash Player MSI through the Active Directory, you use group policies. Also, the MSI for Flash Player must exist within a network share on which everyone has read permissions.

Flash Player can be deployed to either computers or users.

- Publish Flash Player to users.

  Publishing is a group policy action.Therefore, when you publish Flash Player it doesn't install the MSI, but it does make it available to users the next time they log in. This implementation gives the user the choice to install Flash Player through the Add/Remove Programs option in the Control Panel.
- Assign Flash Player to users.

Assigning Flash Player to users is like publishing, in that it is also a group policy action; the assignment does not take effect until the next time the user logs in. However, unlike publishing, when the user logs in, Flash Player will be installed and an icon added to the desktop.

- Assign Flash Player to computers.

  Assigning Flash Player to a computer works similarly to assigning it to a user, with two major differences. First, the assignment is linked to the computer and not to the user. The change takes effect the next time that the computer is restarted. The second difference is that the deployment process installs Flash Player without prompting the user.

To perform the deployment, open the Group Policy Editor.

Publish or assign an application to a user:

1) Navigate through the group policy console.
2) Select User Configuration > Software Settings > Software Installation.
3) Right-click on the Software Installation container
4) Select the New > Package commands from the context menu.
5) Select the Flash Player MSI and select Open.
6) Choose if you want to publish or assign Flash Player.
7) Select OK.

Assign Flash Player to a computer

1) Navigate through the group policy console.
2) Select Computer Configuration > Software Settings > Software Installation.
3) Right-click on the Software Installation container.
4) Select the New > Package commands from the context menu.
5) Select the Flash Player MSI and select Open.
6) Choose to assign Flash Player.
7) Select OK.

You can see that the instructions to assign Flash Player to a user or to a computer are similar. The main difference is selecting the user or computer configuration in step two.

Flash Player Catalog for Microsoft System Center Updates Publisher

If you are using Microsoft System Center Updates Publisher (SCUP) 4.5, you can import the Adobe Flash Player Catalog to deploy the Adobe Flash Player ActiveX control and Plug-in via WSUS 3.0 SP2.

Perform the following steps:

1) Start the Microsoft System Center Updates Publisher 4.5.
2) Right-click System Center Updates Publisher and select Settings.
3) Click Add.
4) In Add Catalog, provide location of the CAB file and complete the other fields as outlined in the remainder of this procedure:

http://fpdownload.adobe.com/get/flashplayer/distribution/win/AdobeFlashPlayerCatalog_SCUP.cab

5) Right click System Center Updates Publisher and select import update(s).
6) Select Bulk catalog import.
7) Click Next.
8) Select Accept on the next dialog box; this imports the catalog.
9) Click Close. Now all updates available in the catalog can be viewed in the SCUP console.
10) Right click on each update to set the publish flag.
11) After setting up the publish flags, right-click on System Center Updates Publisher and select publish update(s), to publish all flagged updates to WSUS 3.0 SP2 Server.
12) Follow the wizard to publish the updates. Then click Next.
13) Click Close on the confirmation dialog to complete the wizard.

These updates will be available under the SCCM console at the next sync cycle and are ready to be deployed.

Configuring SMS

If you plan to use SMS to deploy Adobe Flash Player using either the Adobe Catalog or the MSI file, follow these instructions before starting the deployment process.

1) Start the SMS Administrator Console.
2) Expand the Site Hierarchy, select Site System, and double-click on the SMS site server. (In this example the site server is \\MCNALLY)
3) Confirm that “Use this site system as a management point” is enabled.
4) If you have not yet selected the default management point, the following error message is displayed.

   Select Yes to continue, then select Component Configuration, and then select Management Point. This server is now set to be the default Management Point for your site.
5) If necessary, reopen the Site System Properties. Then, on the Server Locator Point tab, enable “Use this site system as a server locator point”. This setting helps the client find the site server.
6) Select Start, All Programs, Administrative Tools, Internet Information Services (IIS) Manager.

   Notice that your website was added to IIS Manager.
7) As a final step, administrators may configure Discovery Methods in the SMS Administrative Console, so that the site will generate collections (machines or user ID’s) automatically.

SMS and Adobe Catalog installation

SMS 2003 R2 includes two tools for software deployment—the Inventory Tool for Custom Updates (ITCU) and the Custom Updates Publishing Tool (CUPT). This section briefly describes these tools and explains how to use them to deploy Flash Player.

NOTE: Installation using SMS can fail if the player is being installed on a machine where the logged-in user does not have administrative privileges. For information on resolving this issue, see the TechNote entitled “Flash Player MSI installation will fail on machines that don't have administrative privileges” at www.adobe.com/go/df875c9e.

System requirements for SMS deployment

To use SMS 2003 R2, the hierarchy, including clients, must be updated to SMS 2003 Service Pack 2 (SP2). In addition, in order to use the CUPT, Microsoft Management Console (MMC) 3.0 or higher is required. CUPT is not required on the SMS Site Server, but it must be installed on at least one Windows XP machine. The CUPT requires either SQL Server 2005 or SQL Server Express Edition for hosting its database. The CUPT tool allows administrators to manage custom updates in the SMS system and to validate catalogs before publishing them in SMS.

SMS tools for deploying custom updates

The ITCU is an inventory tool that works with custom update catalogs such as the Adobe Catalog. ITCU creates custom collections, packages, and advertisements that are used for deploying the scan tools to SMS clients in the enterprise. ITCU retrieves the catalog, in this case the custom updates catalog, from an accessible SMS distribution point, performs he scan based on catalog data, insert the results of that scan into Windows Management Instrumentation (WMI), and reports the results via hardware inventory.

Custom updates using the CUPT can take two forms—updates that are provided by third-party vendors for software they produce, such as Adobe, and updates created internally that are unique to an environment. These updates are distributed as catalogs. Using third-party updates is a simple matter of downloading the catalogs and adding them to SMS.

Downloading the Flash Player catalog

Adobe provides the Flash Player Catalog, AdobeFlashPlayerCatalog.cab, for licensing and use with SMS 2003R2. Download the catalog from your licensed download page. After downloading the catalog, import it into the CUPT and publish it to SMS. The rest of this section explains how to perform these tasks.

Importing the Flash Player catalog

Follow these steps to import the Adobe Flash Player Catalog into SMS:

1) Select Start, All Programs and choose Systems Management Server.
2) Select Custom Updates, then choose Publishing Tool to launch the Custom Updates Publishing Tool console.
3) In the Actions pane, click Import Update(s).
4) Select Next to accept the default Single Catalog Import option.
   A wizard asks for the location of the Adobe .cab files you downloaded.
5) Select Browse to locate and select the latest Adobe Catalog for SMS.

CUPT validates the catalog and displays the Security Warning to confirm that you would like to accept this catalog signed and published by Adobe.

6) Click Accept.

When the import is done, the Import Software Catalog Wizard confirmation dialog box shows the number of updates imported.

7) Select Close.

8) To display Adobe software updates, click the Adobe node under Custom Updates Publishing Tool.

Publishing the Flash Player catalog

Follow these steps to publish the Adobe Flash Player Catalog:

1) In the tree pane of the CUPT console, select a software name (for example, Adobe Flash Player 10) under the Adobe node.

The result pane shows the custom update software.

2) Select the desired software version in the result pane and then select Set Publish Flag in the Actions pane. The flag should turn green.

NOTE: Initially, custom updates are not flagged in the Publish column. Each update must be flagged for publication in order to be deployed. If an update is not flagged, it will not be included when the request to publish is made.

To see details about a software version, double-click it in the Result pane.

3) Select the Adobe node on the tree pane.

4) In the Actions pane, select Publish Updates.

5) Check Synchronize with Site Database of Systems Management Server and select Next.

The Publish Wizard summary dialog box indicates the update is ready to be published.

6) Select Next to publish the update to SMS.

When it completes, the Publish Wizard confirmation dialog box appears indicating the synchronization is successful.

7) Select Close.

The Custom Updates Publishing Tool closes.

8) Run the SMS Administrator Console. In the console tree, select the Software Updates, select the Action menu, and click Refresh.

The list of software updates in the details pane should contain the custom updates you published.

Confirming successful publication

To confirm that the catalog was successfully published:

1) In the SMS Administrator Console, navigate to the Software Updates Tree and highlight software.

The right pane should show the same update that was published using the CUPT tool, under the type "Custom Update."

2) In the Software Updates Tree, highlight Software Updates.

3) Navigate to the Advertisements Tree and highlight Custom Updates Tool. Right click and select Re-Run Advertisement. Select OK on the mandatory assignment pop-up note.

   Advertisement is manually initiated and Scan for Custom Updates occurs on all clients. This scan takes a period of time to complete. Forcing makes it occur immediately.

   You can view scan progress by going to System Status, Advertisement Status, Custom Updates Tool and Highlight Site in right pane. Right-click on "Show Messages", and select "All". This displays the current status of the Custom Update scan and install.

4) Navigate to the Reporting Tree and select Reports. Sort reports in right pane by category. Scroll down to Software Update Compliance category.

5) Select Compliance by Product Report. Leave the Product field blank and select Custom Update for the Type value.

   The Software Compliance report generated in this step provides the number of machines where the update is either successfully installed or missing.

Deploying the update

To distribute the update across your network using SMS:

1) In the SMS Administrator Console, navigate to the Software Updates tree and highlight Software Updates. Right-click and select "Distribute Software Updates".
2) When the wizard opens, select "Custom Update" for update type. For SMS package, choose New and enter a Package Name of your choice (e.g. "Adobe Flash Player Update 2").
3) Accept the default Program Name and enter "Adobe Inc." as the Organization.
4) Change Program Name to Custom Updates Tool (expedited).
5) Check all Adobe Updates that are listed. Press the Information Button to go to the Adobe website.
6) Select "I will download source files myself."
7) Select Properties and choose Import. Select the appropriate MSI file from your local hard drive for the update and click OK.
8) Check SMS Distribution Point, Collect Inventory, and Advertise. Click Browse and Select the collection to distribute to.

A program, package, and advertisement for the Update that you created should now be visible. Clients poll on an hourly interval, and it take up to an hour for the change to fully propagate. To expedite this process, go to Control Panel, Systems Management, and Actions on the clients. Highlight each action, and click "Initiate Action" to force the client to poll the server.

Verify that the update was successfully installed:

1) Navigate to the Reporting Tree and select Reports. Scroll down to Software Update Compliance category.
2) Select Compliance by Product Report. Leave the Product field blank and select Custom Update for the Type value.

In the generated report, you should see that all systems where the update was applicable are now compliant (have installed the update).

To see which systems were not able to install the update, check the software updates node of the generated report to determine Requested Systems (systems that are eligible for update) versus Compliant Systems (systems that were able to install the update).

Additional resources

The following sites provide additional information about deploying custom updates with SMS.

- Systems Management Server 2003 Concepts, Planning, and Deployment Guide at www.microsoft.com/technet/prodtechnol/sms/sms2003/cpdg
- Deploying Custom Software Updates with SMS 2003 R2 at technet.microsoft.com/en-us/magazine/cc162463.aspx

Interactive MSI installation using SMS

This section describes how to install Flash Player using the MSI installer and the Microsoft Systems Management Server (SMS) 3.0 Console. If you prefer to do a command line installation, see *Command line MSI installations*.

The following instructions assume the following system requirements:

- Windows 2003 Server (r2)
- SQL Server 2000 (SP4)
- SMS 2003 (SMS 3.0)
- Active Directory
- IIS (Microsoft Internet Information Server)
- BITS (Background Information Transfer Service)
- Flash Player MSI

These instructions also assume that you have already installed and configured SMS 3.

NOTE: Installation using SMS can fail if the player is being installed on a machine where the logged-in user does not have administrative privileges. For information on resolving this issue, see the TechNote entitled “Flash Player MSI installation will fail on machines that don't have administrative privileges” at www.adobe.com/go/df875c9e.

1) Start the SMS Administrator Console.
2) Expand the Site Database.
3) Right-click on Packages and select New > Package.
4) On the Package Properties General tab, name your package. You can also include additional data, such as the version number, publisher, language, and comments.
5) On the Data Source tab, enable “This package contains source files”. Click Set and browse to the network location where your source files reside. For this example, the Flash Player MSI was saved on the local C:\ drive.
6) On the Data Access tab, select “Access distribution folder through common SMS package share” and click OK.

7) To make your Distribution Points (locations where SMS packages are stored), expand Packages, right-click on Distribution Points and select New > Distribution Points.
8) Select Next to start the Distribution Point wizard. Select the servers to which you want to copy the package and then click Finish.
9) Right-click on Programs and select New > Program. This creates the program that will execute your deployment commands.
10) In the General tab, name your program and type in the command line information. In this example, we named the program “install” and then used the following command:

```
msiexec /i install_flash_player_active_x.msi /qn
```

11) To designate the conditions under which the application will be installed, select the Environment tab. In this example, the conditions are, “Only when a user is logged on,” “Run with administrative rights,” and “Runs with UNC name”.
12) To make an advertisement that will apply the package program to the collection at a set time, right-click on the package and select All Tasks > Distribute Software.
13) Select your Distribution Points and click Next.
14) When asked “Do you want to advertise from this package?” choose Yes, then click Next.
15) Select the program to advertise, then click Next. For this example, we named the program “install”.
16) At this point, you can select the Collection (designated group of machines that you want to target). In the Advertisement Target pane, select, “Advertise this program to an existing collection” and select Browse. For this example, we selected “All Windows XP Systems.”
17) Select the default for the Advertisement Name, or change the name, then click Next.
18) Specify whether the advertisement should apply to subcollections, then click Next.
19) Specify when the program will be advertised, then click Next. This allows you to advertise a program after hours when users are not on their computers.
20) You are now ready to assign your program to your collection. Select “Yes. Assign the program,” then click Next
21) Look at the Details before clicking Finish.

If your deployment is successful, you will see a message that says, “Program About to Run”.

Command line MSI installations

The MSI installer is provided for administrative installations using software such as Microsoft Systems Management Server (SMS). An administrative installation is the first step in preparing an MSI installer for deployment over a network. This section discusses how to deploy Flash Player over a Windows network using msiexec and the MSI installer. If you prefer to do an interactive installation using the SMS Console, see *Interactive MSI installation using SMS*.

NOTE: Installation using SMS can fail if the player is being installed on a machine where the logged-in user does not have administrative privileges. For information on resolving this issue, see the TechNote entitled “Flash Player MSI installation will fail on machines that don't have administrative privileges” at www.adobe.com/go/df875c9e.

To run an administrative installation, use the /a command line switch. For example, to run the Flash Player ActiveX control installer in interactive administrator mode, you would use this syntax:

```
msiexec /a "install_flash_player_11_activeX.msi"
```

NOTE: The examples in the rest of this chapter use the ActiveX control filename. If you are installing the browser plug-in, simply substitute the correct filename in your installation.

On some machine configurations, spaces in the MSI filename interfere with running the installer from the command line, even with quotes around it. If you rename the MSI file for any reason, do not use any spaces in the filename.

When started as shown above, the installer runs through its admin UI sequence, involving a series of dialog boxes. The first dialog box is a simple welcome screen, and the next dialog prompts for the network location that you want to install to.

Click Next in the Welcome dialog to open the Network Location dialog, then click Install in the dialog box to deploy the admin tree to a network share.

NOTE: The admin install includes only those files contained within the MSI file itself. Other support files required by the installation such as bootstrap files, MSI runtime installers, or patches, should be copied to the shared folder by some other means of your choice (manually, with a script, batch file, and so on).

Once the admin install is deployed to the shared folder, there are multiple options available to install the product onto a workstation. These are discussed in the rest of this section.

Manually launch the installer on the client

One easy way to pull the installation from an administrative image is to run it manually, by sitting at the client machine and launching it interactively from the site on which it is being shared. You could do this either by double-clicking the bootstrap file, or by double-clicking the MSI file. The bootstrap file is the recommended one to use, as it automatically installs the required version of the MSI runtime first, if needed, before launching the MSI file in turn.

NOTE: If you've renamed the MSI file to avoid command line problems with spaces in the filename, the bootstrap file will no longer work, because the bootstrap file is looking for a specific hard-coded file-name. In this case, run the MSI file directly instead.

Launch the installer on the client using quiet mode

If you don't need to customize the installation options, then you can run the installation non-interactively. This method requires with a command line switch, as shown below. When run in this mode, the default options are used for all items that would be presented as choices in the interactive install.

```
msiexec /i "install_flash_player_11_activeX.msi" /qn
```

The simple command line syntax shown above works in most cases, but other command line elements and switches are available. A more comprehensive version of the syntax looks like this (to be entered all on one line):

```
%Comspec% /c msiexec /i "\\network
path\install_flash_player_11_activeX.msi" /qn
```

In both cases, the final `/qn` switch must be on the same line as the rest of the command.

The arguments used in the command line example above are described below.

- `%Comspec%` is an environment variable provided by Windows. It points to the command interpreter, cmd.exe.
- `/c` is a switch passed to cmd.exe telling the shell to wait until the msiexec.exe command completes before proceeding. Without this switch, the shell will execute subsequent commands before the current command finishes.
- `msiexec.exe` is the Windows installer runtime. When you double-click an MSI file (for example, foo.msi) you are implicitly running msiexec /i foo.msi.
- `/i` instructs MSIEXEC to install the MSI file listed after the switch. There is also an `/x` switch that uninstalls the MSI file specified after the `/x` switch.
- `/qn` specifies a user interface level for the action. The `/qn` switch suppresses all prompts and is therefore useful for silent installations. When attempting to debug, you can switch to `/qb`, which displays basic modal dialogs.

For more information about command line options available for msiexec, see “Command-Line Options” in the MSDN Library at msdn.microsoft.com/en-us/library/aa367988.aspx.

Reinstalling a Flash Player using a batch routine

If you need to uninstall and reinstall the Flash Player, you can use a batch file like this one:

```
REM Begin quietInstall.bat
REM Uninstall Flash Player ActiveX
%Comspec% /c msiexec /x "\\network
path\install_flash_player_9_activeX.msi" /qn
REM Install Flash Player ActiveX
%Comspec% /c msiexec /i "\\network
path\install_flash_player_9_activeX.msi" /qn
REM End quietInstall.bat
```

Performing a background update

During a standard Flash Player update, a dialog box announces the availability of the update to the user to let the user either accept, postpone, or reject the update. If the user accepts the update, the user's default browser is launched to Adobe's site to download the latest version. Once downloaded the user can install the update immediately or at a later date. This type of update is called a notification update.

On Microsoft Windows and macOS, a Flash Player background update installs the update silently in the background, without any user interaction. A background update installs the ActiveX control (IE), NPAPI plug-in (Firefox, Safari) and PPAPI plug-in (Chromium-based browsers) players when appropriate.

For some browser types, if the user has a browser open at the time of an update, the browser does not use the updated player until a new browser instance launches. Browser instances open during the update process continue to use the previous player version until they are restarted.

Background update is disabled by default. Based on the install type, the background update varies:

MSI and PKG installers do not provide update options and therefore do not set the update options in the mms.cfg file. To set the update option when installing Flash Player using the MSI or PKG installer, deploy a custom mms.cfg file with the desired update options to the following locations:

- 32-bit Windows: C:\Windows\System32\Macromed\Flash
- 64-bit Windows: C:\Windows\SysWOW64\Macromed\Flash
- macOS: /Library/Application Support

All other installer types:

During installation, you can select the update option (silent, notification, or do not update). If you previously opted in to background updates, and have not uninstalled the player (see note in the uninstall section about update options being reset when the player is uninstalled), the update options will not be displayed.

An installation performed by the MSI or PKG installer does not create or update these entries in the mms.cfg file.

When the Flash Player is installed, it also installs a Windows 32-bit service application and task or, for a Mac, a LaunchDaemon. When all player types are removed, the Windows service and task, or Mac LaunchDaemon, are also removed.

If background updates are enabled, the task or LaunchDaemon check for an update once every 24 hours. However, if no network or internet connection is available at the time of the check, the check occurs again every hour until a connection is detected. After the next successful check, another check does not occur for 24 hours.

The update task runs as the SYSTEM user, not as the current user. The check runs regardless of who is logged on, and runs even if no one is logged on. The only requirement is that the system has an internet connection. It is the responsibility of the system administrator to ensure that processes running as the SYSTEM user account are correctly configured to use any appropriate corporate proxies.

Background updates from an internal server

You can use the background update mechanism to host and deploy updates on internal networks. Deploying Flash Player from an internal server requires obtaining the Adobe Runtimes / Reader Distribution License if you don’t have a distribution license.

Prerequisites

- A server with the following configuration:
  - Open port 443 for HTTPS requests.
  - A valid SSL certificate, issued by a trusted third-party certificate authority, for HTTPS access on port 443.
- The ability to store files on the server in an Adobe-specified folder structure (outlined later in this section).
- The ability to deploy mms.cfg configuration files to clients on the network.

Configure the server

1) In your server root, create the following structure: /pub/flashplayer/update/current/sau
2) Download the Background Update Resources archive from the Adobe Flash Player Distribution page using the link in the email you received when licensing Flash Player.

   A link to the Background Update Resources archive is also posted on the https://www.adobe.com/licensing/distribution/strategies/sms.html page.
3) Unpack the downloaded .cab archive. The archive contains the required files in the appropriate format and directory structure as required by Flash Player.
4) Copy the contents of the unpacked archive to the /sau directory created in step 1.
5) When finished, you should see something similar to the following:

   Current release:

```
https://your.server.com/pub/flashplayer/update/current/sau/currentmajor.xml
https://your.server.com/pub/flashplayer/update/current/sau/11/xml/version.xml
https://your.server.com/pub/flashplayer/update/current/sau/11/install/install_all_win_ax_sgn.z
https://your.server.com/pub/flashplayer/update/current/sau/11/install/install_all_win_pl_sgn.z
https://your.server.com/pub/flashplayer/update/current/sau/11/install/install_all_mac_pl_sgn.z
https://your.server.com/pub/flashplayer/update/current/sau/20/xml/version.xml
https://your.server.com/pub/flashplayer/update/current/sau/20/install/install_all_win_ax_sgn.z
https://your.server.com/pub/flashplayer/update/current/sau/20/install_all_win_pep_sgn.z
https://your.server.com/pub/flashplayer/update/current/sau/20/install/install_all_win_pl_sgn.z
https://your.server.com/pub/flashplayer/update/current/sau/21/xml/version.xml
https://your.server.com/pub/flashplayer/update/current/sau/21/install/install_all_win_ax_sgn.z
https://your.server.com/pub/flashplayer/update/current/sau/21/install_all_win_pep_sgn.z
https://your.server.com/pub/flashplayer/update/current/sau/21/install/install_all_win_pl_sgn.z
```

Configure clients

- Create an mms.cfg file with the following entries (replacing your.server.com with the name of your server):

```
AutoUpdateDisable=0
SilentAutoUpdateEnable=1
SilentAutoUpdateServerDomain=your.server.com
```

- Deploy Flash Player 11.3 or higher.

- Deploy the mms.cfg to all clients for which you want the Background Updater redirected to your internal server.

The SilentAutoUpdateServerDomain server name and the CN name on the SSL cert must match (for production and test servers).

When hosting the Background Update resources locally, Flash Player will only update in the background. Users will not see an update notification informing them an update is available. If the Background Update resources are not hosted locally and the client machines are configured for Background Updates, they may occasionally receive notifications that an update is available instead of being updated through the Background Updates.

Windows registry keys

In addition to the registry keys you can use to determine the installed version of a player (see *Player versions*), Flash Player creates other registry keys when it is installed or registered. These keys are summarized in the Flash Player TechNote entitled "Flash Player | Windows registry permissions".

PKG Installer for macOS

To distribute Flash Player across the enterprise, use the PKG installer in conjunction with your package management tool of choice to install Flash Player to the current volume, a non-boot volume, or a disk image to be replicated across your enterprise.

1) Extract the Adobe Flash Player package installer (Install Adobe Flash Player.pkg) from the .DMG file.
2) Import the .PKG file into your package management tool of choice and distribute Flash Player across your enterprise.

Silent installation of Flash Player (using .pkg installer package)

Use the .pkg installer package to install the Flash Player silently, using the installer utility, to the current volume, a non-boot volume, or a disk image to be replicated across your enterprise.

App installer for macOS

Double-click the DMG image file to extract the .app installer bundle and follow the guided installation instructions.

NOTE: Flash Player 11 or later is not supported on Power PCs.

Silent installation of Flash Player (using .app installer bundle)

Do the following to silently install Flash Player 11.3 or later on macOS:

1) Extract the Adobe Flash Player installer bundle (`Install Adobe Flash Player.app`) from the `.DMG` file.
2) Open a terminal window and change to the directory where the `.app` file is saved.
   For example, if the `.app` file is saved on the Desktop of the current user, type: `cd ~/Desktop`
3) Run the installer contained in the `.app` file using the following command:
   ```
   sudo ./Install Adobe Flash Player.app/Contents/MacOS/Adobe Flash Player Install Manager -install
   ```
4) Type the password to proceed with the installation.
   NOTE: You need to be a super user to proceed with the installation.

Customizing player behavior

After you deploy the player, you can install a privacy and security configuration file (mms.cfg) to specify rules about Adobe Flash Player security. The file controls security-related behavior of the player after installation.

The primary purpose for the mms.cfg file is to support the corporate and enterprise environments where the IT department would like to install Flash Player across the enterprise, while enforcing some common global security and privacy settings (supported with installation-time configuration choices). The mms.cfg file can be used to control data loading operations, user privacy, auto-update behavior, background update behavior, and local file security.

For detailed information about customizing player behavior, see *Administration*.

Troubleshooting installation problems

The following TechNotes address installation problems you may encounter.

- Troubleshoot Adobe Flash Player installation for Windows (www.adobe.com/go/tn_19166)
- Troubleshoot Adobe Flash Player for Intel-based Macs (www.adobe.com/go/2dda3d81)
- Safe versions security restrictions when installing Flash Player (Internet Explorer on Windows) (http://kb2.adobe.com/cps/402/kb402435.html)

Additional resources

For answers to questions regarding Flash Player licensing and deployment, see Adobe Player Licensing at www.adobe.com/licensing/distribution and the player Distribution FAQ at www.adobe.com/licensing/distribution/faq.

To receive notification of when a new version of Flash Player is available, register for the Security Bulletin and Advisories email notification at helpx.adobe.com/security.html.

Notifications are also posted on the Flash Player user forums. See https://forums.adobe.com/thread/890491 for more information.

The following sites outside Adobe provide general information on deploying software on Windows systems:

- Windows Installer Resources for System Administrators at www.install-site.org/pages/en/msi/admins.htm.
- Applying Small Updates by Patching an Administrative Image in the MSDN library at msdn.microsoft.com/en-us/library/aa367573.aspx.
- Applying Small Updates by Reinstalling the Product in the MSDN library at msdn.microsoft.com/en-us/library/aa367575.aspx.
- For information on detecting the version of Adobe Flash Player from a website, see the “Detection and Installation” section at the Flash Player Developer Center (www.adobe.com/devnet/flashplayer/detection_installation.html).

Administration

Enterprise Enablement

To help enterprises prepare for Flash Player's end of life (EOL), Adobe is making several changes to Flash Player's behavior and functionality.

Starting with the Flash Player June 2020 release, enterprises can configure Flash Player to only load content from a list of allowed URLs. Doing so has multiple uses:

- Explicitly allow Flash Player content that you trust and block all other Flash content. While many browsers provide the option to block Flash Player on a per site basis, some do not. With this change, it won't matter which browser is being used because Flash Player itself will keep a list of allowed URLs.
- Provides logging capability to identify Flash content being used by client systems. We recommend reaching out to our distribution and support partner, HARMAN, to help identify comprehensive solutions for enterprises with large client bases.

Any use of the domain-level allow list after the EOL Date is strongly discouraged, will not be supported by Adobe, and is entirely at the user's own risk.

Beginning January 12, 2021, Adobe will block Flash content from running by default. Any allowed Flash content will continue to run on browsers that have not disabled Flash Player.

Enterprise Enablement includes the preferences AllowListPreview, TraceOutputEcho, EnableAllowList, and AllowListRootMovieOnly. In addition, we recommend you review the section "Suppressing EOL Uninstall Prompts" for information on optional preferences that may affect your users in 2020.

Suppressing EOL Uninstall Prompts

In the latter half of 2020, as part of Flash Player's end of life (EOL) process, Adobe will start prompting customers to uninstall Flash Player. This prompt is optional and can be dismissed by the end user.

To reduce friction in a managed installation environment, administrators can set either of the following properties in the client's mms.cfg to disable the prompt from appearing:

```
AutoUpdateDisable = 1
Or
EOLUninstallDisable = 1
```

Many administrators have already disabled automatic updates for Flash Player by setting AutoUpdate-Disable=1. If this is true in your environment, your users will not see the uninstall prompt. If automatic updates are enabled (the default setting) and you would like to suppress the uninstall prompts from appearing for your clients, set EOLUninstallDisable = 1.

EOLUninstallDisable

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

Disabled (0) by default. EOLUninstallDisable =1 allows system administrators to disable uninstall prompts by Flash Player scheduled to start appearing in the second half of 2020. When enabled (1), users can still uninstall Flash Player, however unsolicited prompts by Adobe to uninstall Flash Player will be suppressed.

AllowListPreview

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

AllowListPreview is disabled (0) by default. When EnableAllowList = 1, requests matching patterns are allowed, and all others are blocked. After Flash Player EOL, this setting will be ignored.

Blocked requests (but not allowed requests) are logged using trace().

```
*** EnableAllowList blocks 'http://www.example.com/blocked.swf'. ***
```

When AllowListPreview = 1, all requests are allowed, and trace() is used to log whether each request would be allowed or blocked by the current AllowList:

```
*** AllowListPreview: AllowList allows
'http://www.example.com/allow.swf'. ***
*** AllowListPreview: AllowList blocks
'http://www.example.com/blocked.swf'. ***
```

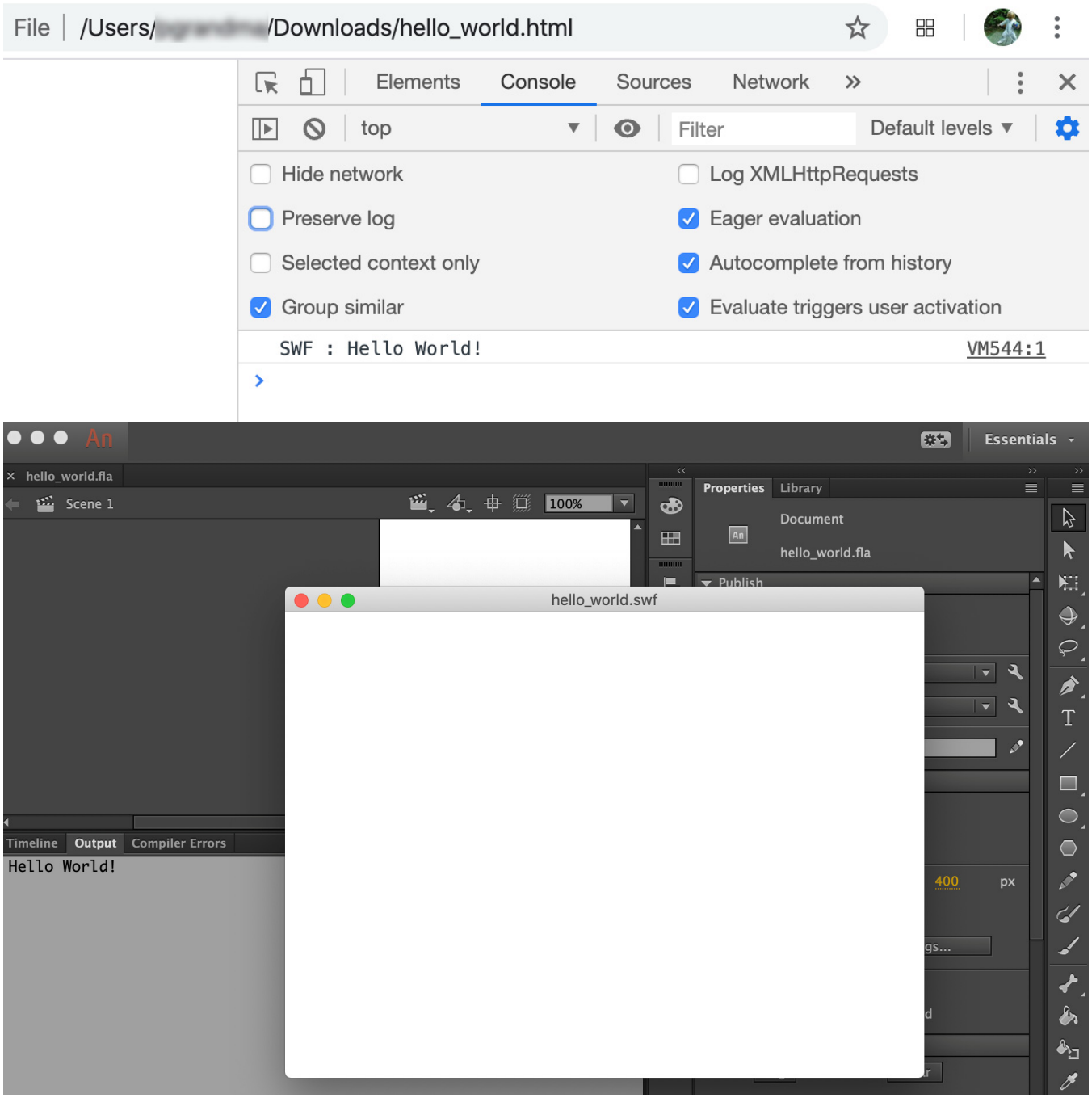
NOTE: AllowListPreview is ignored unless EnableAllowList=1 is specified.

Example:

An admin adds the following to MMS.CFG:

```
# duplicate actionscript console output
# in browser's console for javascript
TraceOutputEcho=1

# Enable the AllowList feature
EnableAllowList=1

# Normally, the allow list blocks URL requests
# unless the url matches a pattern in the list.

# In preview mode, all requests go unblocked,
# but console output is written for each request
# indicating which pattern it matched or that
# no match was found.

AllowListPreview=1
```

trace() statements from the player now appear in the browser's JavaScript console (each trace() is prefaced with SWF: to distinguish it from other console output):



TraceOutputEcho

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

Disabled by default. TraceOutputEcho=1 will cause ActionScript trace (msg:String) statements to be duplicated in the browser's JavaScript console using console.log ("SWF: " + msg).

Example:

```
# trace() statements are echoed in the browser's javascript console
# with console.log().
TraceOutputEcho=1
# Enable the AllowList feature EnableAllowList=1
# In preview mode, the allow list won't block url requests,
# but for each url request, it will trace() whether
# the current whitelist would block or allow the request
AllowListPreview=1
Would produce trace() output in the browser :
SWF: *** AllowListPreview: allow list allows
'http://www.example.com/allow.swf'.
*** SWF: *** AllowListPreview: allow list blocks
'http://www.example.com/blocked.swf'. ***
```

EnableAllowList

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

Disabled by default. Allows system administrators to allow Flash Player to only load content from a set of allowed URLs. After Flash Player EOL, EnableAllowList will default to true and the MMS.CFG setting will be ignored.

Troubleshooting

Flash Player typically does not provide visible runtime errors to end-users, and users will not see an error message when a request is blocked by the EnableAllowList flag while using the generally available Flash Player version.

Errors related to the allow list feature can be logged to flashlog.txt by using Debug version of Flash Player with logging configured, by setting ErrorReportingEnable=1 in mm.cfg. As of the time of this writing, we recommend Firefox and Internet Explorer as the easiest browsers to configure for debugging and logging to the filesystem.

See the following guide on configuring the debugger version of Flash Player for details:
https://helpx.adobe.com/flash-player/kb/configure-debugger-version-flash-player.html

Once enabled, error messages will be logged to flashlog.txt, and will look like this:

```
Example – Root Movie Blocked
*** EnableAllowList Activated - Root Movie Blocked ***
No AllowListURLPattern matches 'https://www.example.com/movie.swf
```

In this instance, the parent webpage attempted to embed a SWF, but no AllowListUrlPattern entry allows https://www.example.com/movie.swf to be loaded.

Example – Root Movie Loaded, Subsequent Request Blocked

```
*** EnableAllowList Activated - Request Blocked ***
No AllowListURLPattern matches 'http://www.example.com/intro.html'
```

In this instance, the parent SWF was permitted to load, but at runtime, the SWF attempted to navigate to http://www.example.com/intro.html. The navigation was not permitted by a matching AllowListUrl-Pattern entry, and was blocked.

AllowListRootMovieOnly

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

Disabled by default. Only apply allow list restrictions to the parent SWF. Subsequent requests made from the loaded SWF to arbitrary URLs are allowed.

In some instances, it may be desirable to restrict Flash Player to loading only a trusted set of parent movies, but to then allow those movies to make arbitrary requests to other resources. This minimizes the number of AllowListUrlPattern entries required, and provides administrators with a simple, flexible option for narrowing what Flash content can be loaded in their environment.

AllowListUrlPattern

```
AllowListUrlPattern = <scheme> ://<host>:<port>/<path>
<scheme> = '*' | 'http' | 'https'
<host> = <any char except '.' and '*'>
<port> (optional) = <any valid port number>
<path> = '/' <any chars>
```

With EnableAllowList=1 set, administrators can then specify a discrete URL or pattern to allow.

Pattern	Description
http://www.example.com/folder/flash.swf	You can use an url as a match pattern,for instance, this pattern will match http://www.example.com/folder/flash.swf but not http://www.example.com/folder/file.txt.
http://www.example.com/folder/	You can omit path components to match a set of urls with a common prefix, for instance, this pattern will match http://www.example.com/folder/flash.swf and http://www.example.com/folder/file.txt.
http://www.example.com/	You can omit the path entirely, for instance, this pattern will match any url with the origin http://www.example.com:80/. NOTE: The default port for HTTP is 80, so http://www.example.com:80/ is equivalent to http://www.example.com/.

Pattern	Description
https://www.example.com/	This pattern will match any url with the origin https://www.example.com:443/. NOTE: The default port for HTTPS is 443, so https://www.example.com:443/ is equivalent to https://www.example.com/.
*://www.example.com/	The wildcard scheme (*:) will match HTTP or HTTPS, for instance, this pattern will match either http://www.example.com:80/ or https://www.example.com:443/.
file://www.example.com/	This pattern will match FILE: requests to www.example.com.
http://*.example.com/	You can use a leading wildcard to match all subdomains of a domain, for instance, http://*.example.com/ matches the subdomains of example.com. NOTE: The wildcard must be followed by at least two labels, so http://*/ is not valid, nor is http://*.com/.
blob:*	A wildcard can be used to allow all requests using the specified scheme. NOTE: http:*, https:* and *:* are not permitted as they are overly general.
http://192.168.1.20/	The host can be an IPv4 Address instead of a domain name. Wildcards are not supported with IPv4 Addresses.
http://[::1]/	The host can be an IPv6 Address instead of a domain name. Wildcards are not supported with IPv6 Address. NOTE: More than one string can represent the same IPv6 address, for instance, “[::1]” and “[0::1]” are equivalent, but patterns will be matched using string comparison, so (“[::1]” != “[0::1]”).
file:///folder/file.txt	Local file requests can omit the host.
http://user:pass@www.example.com/	User info is ignored, so this pattern is the same as http://www.example.com/.
http://www.example.com/flash.swf?query	Query strings are ignored, so this pattern is the same as http://www.example.com/flash.swf?query.
http://www.example.com/flash.swf#fragment	Fragments are ignored, so this pattern is the same as http://www.example.com/flash.swf.

Examples:

Strict - Load Only Single Specified File, Require HTTPS

Restricts Flash Player to loading only a trusted set of parent movies, but allows those movies to make arbitrary requests to other resources. This minimizes the number of AllowListUrlPattern entries required, while providing compatibility for the target application.

```
mms.cfg
EnableAllowList = 1
AllowListUrlPattern=https://my.intranet.com/legacyApp/application.swf
```

How requests would be handled:

Allowed

```
https://my.intranet.com/legacyApp/application.swf
```

Denied

```
http://my.intranet.com/legacyApp/application.swf
https://my.intranet.com/legacyApp/application_child.swf
https://my.intranet.com/legacyApp/other_resource.jpg
http://example.com/randomContent.xml
```

Strict – Load Only Content from Specified Folder, Require HTTPS

Allow Flash Player to load any content from a specific directory, over HTTPS only. Any child content loaded by the parent SWF outside this directory must be explicitly allowed in order to load.

```
mms.cfg
EnableAllowList=1
AllowListUrlPattern=https://my.intranet.com/legacyApp/
```

Strict Folder Access, HTTP and HTTPS

Allow Flash Player to load any content from a specific directory, over HTTPS only. Any child content loaded by the parent SWF outside this directory must be explicitly whitelisted in order to load.

```
mms.cfg
EnableAllowList=1
AllowListUrlPattern=https://my.intranet.com/legacyApp/
AllowListUrlPattern=http://my.intranet.com/legacyApp/
```

How requests would be handled:

Allowed

```
https://my.intranet.com/legacyApp/application.swf
https://my.intranet.com/legacyApp/application_child.swf
https://my.intranet.com/legacyApp/other_resource.jpg
```

Denied

```
https://my.intranet.com/otherApp/other_resource.txt
http://my.intranet.com/legacyApp/application.swf
http://example.com/randomContent.xml
```

Intermediate – Parent content must be loaded from trusted directory, HTTPS and HTTP

Allow Flash Player to load content from a specific trusted directory. Any subsequent content requested by the trusted application will not be subject to allow list restrictions.

```
mms.cfg
```

```
EnableAllowList=1
AllowListRootMovieOnly=1
AllowListUrlPattern=https://my.intranet.com/legacyApp/
AllowListUrlPattern=http://my.intranet.com/legacyApp/
```

How the initial request for Flash content would be handled:

Allowed

```
https://my.intranet.com/legacyApp/application.swf
http://my.intranet.com/legacyApp/application.swf
```

Denied

```
http://my.intranet.com/legacyApp/application.swf
https://my.intranet.com/otherApp/other_resource.txt
http://example.com/randomContent.xml
```

Once loaded, if the SWF at https://my.intranet.com/legacyApp/application.swf (or its children) made subsequent requests for resources available on the network, they would be permitted regardless of allow list restrictions.

Allowed (when requested by the permitted Flash instance):

```
http://my.intranet.com/legacyApp/application.swf
https://my.intranet.com/otherApp/other_resource.txt
http://example.com/randomContent.xml
```

EnableInsecureAllowListLocalPathMatching

There are a number of legacy desktop applications that incorporate Flash Player by embedding native browser windows into their UI. While this practice is not technically supported, we do our best to keep those applications working while prioritizing security over compatibility.

In order to match file references to the file and blob AllowListUrlPattern directives, Flash Player requires that paths to local files be provided as RFC 3986 conformant URIs. Modern web browsers do this conversion automatically; however, some desktop applications embedding Flash Player provide a number of unique variations that do not work with the default pattern matching logic.

As a workaround, we have added the EnableInsecureAllowListLocalPathMatching option for administrators.

With this option enabled, our pattern matching for blob:* and file:* becomes more permissive.

This permissiveness allows for the use of ambiguous paths and UNC paths, which are otherwise not allowed. In addition, Flash Player is more forgiving in situations where blobs could not be parsed according to the default rules.

While this permissiveness may provide the flexibility necessary to allow a critical legacy application to operate without modification, it could potentially impact your overall security posture.

Privacy and security settings (mms.cfg)

Network administrators can install Adobe Flash Player across the enterprise while enforcing common global security and privacy settings (supported with installation-time configuration choices). To do this, you install 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

Windows

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

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

macOS

/Library/Application Support/Macromedia

Linux

/etc/adobe/

NOTE: Unlike Windows and macOS, the Linux player is in a directory named adobe, not in one named Macromed or Macromedia.

Google Chrome

Google Chrome uses its own version of the mms.cfg file, saved at:

- macOS: /Users/<username>/Library/Application Support/Google/Chrome/<Profile>/Pepper Data/Shockwave Flash/System
- Win: %USERNAME%/AppData/Local/Google/Chrome/User Data/<Profile>/Pepper Data/Shockwave Flash/System
- Linux: ~/.config/google-chrome/<Profile>/Pepper Data/Shockwave Flash/System/

  Note: Google Chrome supports the ability for each user on the system to create multiple user Profiles within Google Chrome. Each Profile has a separate instance of browser data and config files, including Flash Player data. To modify behavior across each Profile and user on the system using mms.cfg, an mms.cfg would need to be placed in each relevant folder. The primary profile on the system is named "Default", and subsequent profiles are enumerated with the name "Profile <n>".

  The System directory may not exist. If not, create it manually.

NOTE: Update related directives for Adobe Flash Player are not honored by Google Chrome. Google distributes Flash Player as a component of Chrome, via the Chrome automatic update mechanism. Administrators must use Chrome's administrative controls to modify the behavior of the Chrome updater.

Third-party administration tools such as Microsoft System Management Server may be used to replicate the configuration file to the user's computer.

Use the standard techniques provided by your operating system 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 via pop-up questions, settings dialog boxes or the settings manager. (For more information on how the user can specify values for certain options, see *User-configured settings*.)

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.

Whitespace 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, including Windows Notepad and Mac TextEdit, 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	Prevent users from designating any files on local file systems as trusted.
AssetCacheSize	Specify a hard limit (in MB) for the amount of local storage that Flash Player uses for common Flash components.
AutoUpdateDisable	Prevent Adobe Flash Player from automatically checking for and installing updated versions.
AutoUpdateInterval	Specify how often to check for an updated version of Adobe Flash Player. This setting is for notification updates. It is not for background updates. Do not use this setting if the intent is to use Background Updates to update the client systems.
AVHardwareDisable	Prevent SWF files from accessing webcams or microphones. Not applicable on Google Chrome or Microsoft Edge browsers.
AVHardwareEnabledDomain	Allow SWF files from a specific domain or IP address to access webcams or microphones. Not applicable on Google Chrome or Microsoft Edge browsers.
DisableDeviceFontEnumeration	Prevent information on installed fonts from being provided.
EnableInsecureActiveXNavigateToURL	In Flash Player 32 and above, override the strict enforcement of Same Origin Policy with requests made from NavigateToURL() in the ActiveX Flash Player for Microsoft Internet Explorer and Edge on Windows.
DisableHardwareAcceleration	Disable hardware acceleration.
DisableNetworkAndFilesystemInHostApp	Disable networking or file system access of any kind.
DisableProductDownload	Disable downloading of native code and applications, digitally signed and delivered by Adobe.
DisableSockets	Disable the use of the Socket.connect()and XMLSocket.connect()methods.

Option	Description
EnableInsecureActiveXMHTMLSupport	In Flash Player 32 and above, override the default behavior of restricting the ability to launch Flash Player from within an MTHML (.mhtml or .mht) document.
EnableInsecureAllowListLocalPathMatching	In Flash Player 32 and above, allows the file:* and blob:* AllowListPattern logic to match additional URIs sent to Flash Player that do not conform to RFC 3986. Enabling this flag may allow some legacy desktop applications to continue work when EnableAllowList is active; however, this choice also allows Flash Player to load content in more ambiguous contexts, potentially weakening your security posture.
EnableInsecureByteArrayShareable	In Flash Player 30 and above, override the default behavior of restricting the shareable property of the ActionScript ByteArray API class.
EnableInsecureByteArrayShareableDomain	In Flash Player 30 and above, override the default behavior of restricting the shareable property of the ActionScript ByteArray API class on a per-domain basis.
EnableSocketsTo	Create a list of servers to which socket connections are allowed.
EnforceLocalSecurityInActiveXHostApp	Enforce local security rules for a specified application.
FileDownloadDisable	Prevent the ActionScript FileReference API from performing file downloads.
FileDownloadEnabledDomain	Allow the ActionScript FileReference API to perform file downloads from a specific domain or IP address.
FileUploadDisable	Prevent the ActionScript FileReference API from performing file uploads.
FileUploadEnabledDomain	Allow the ActionScript FileReference API to upload files to a specific domain or IP address.
FullScreenDisable	Disable SWF files playing via a browser plug-in from being displayed in full-screen mode.

Option	Description
LegacyDomainMatching	Allow SWF files produced for Flash Player 6 and earlier to execute an operation that has been restricted in a newer versions of Flash Player.
LocalFileLegacyAction	Allow SWF files produced for Adobe Flash Player 7 and earlier to access local files in a way that has been restricted in newer versions of Flash Player.
LocalFileReadDisable	Prevent local SWF files from having read access to files on local hard drives.
EnableInsecureLocalWithFileSystem	Override the default behavior of disallowing files loaded from the local filesystem to read from the local filesystem.
LocalStorageLimit	Specify a hard limit (in MB) on the amount of local storage that Adobe Flash Player uses for persistent shared objects stored by an individual domain.
OverrideGPUValidation	Overrides validation of the requirements needed to implement GPU compositing.
ProductDisabled	Creates a list of ProductManager applications that users are not permitted to install or launch.
ProtectedMode	Enables the Protected mode feature.
ProtectedModeBrokerAllowListConfigFile	Bypasses the prevented actions by creating a white list of allowed actions (policies).
ProtectedModeBrokerLogfilePath	Specifies the path to the log file where policy violations are recorded.
RTMFPP2PDisable	Specifies how the NetStream constructor connects to a server when a value is specified for peerID, the second parameter passed to the constructor.
RTMFPTURNProxy	Allow Adobe Flash Player make RTMFP connections through a specified TURN proxy server, in addition to normal UDP sockets.
SilentAutoUpdateEnable	Allow Adobe Flash Player to install updates silently in the background, with no user interaction.

Option	Description
SilentAutoUpdateServerDomain	For administrators deploying silent updates to Adobe Flash Player from an internal server, specify the hostname of the update server to use.
SilentAutoUpdateVerboseLogging	Enable logging of warning and error codes during a background update.
ThirdPartyStorage	Configure ability of third-party SWF files to read and write locally persistent shared objects.
UseWAVPlayer	On Windows, use WAVAudio for playback instead of Windows Core Audio APIs.
NetworkRequestTimeout	On Windows, modify the default timeout for network socket requests (in seconds).
EnableInsecureJunctionBehavior	In Flash Player 14 and above, override the default behavior of restricting write access to paths that traverse junction files in Windows.
EnableLocalAppData	Configure Flash Player to write LSOs to the %LOCALAPPDATA% folder instead of %APPDATA%.
DefaultLanguage	Override Flash Player's default language.
IEClickToPlayBlocked	Provides domain block list functionality if EnableIEClickToPlay has been turned on.
EnableIEClickToPlay	Enable Flash Player click to play functionality in Internet Explorer on Windows 7 and below
IEClickToPlayBypass	Provides domain allow list functionality if EnableIEClickToPlay has been turned on.
EventJitterMicroseconds	Important mitigation for Spectre and Meltdown style attacks (CVE-2017-5753, CVE-2017-5715, CVE-2017-5754)
TimerJitterMicroseconds	Important mitigation for Spectre and Meltdown style attacks (CVE-2017-5753, CVE-2017-5715, CVE-2017-5754).
InsecureJitterDisabledDomain	Adding domains to this list disables important mitigations for Spectre and Meltdown style attacks (CVE-2017-5753, CVE-2017-5715, CVE-2017-5754), but may improve application performance in some limited circumstances.

Option	Description
EOLUninstallDisable	Allow system administrators to disable uninstall prompts by Flash Player scheduled to start appearing in the second half of 2020.
AllowListPreview	Requires EnableAllowList=1. Requests that would typically be blocked by the configured allow list rules are logged, but the requests are permitted.
TraceOutputEcho	TraceOutputEcho=1 causes ActionScript trace (msg:String) statements to be duplicated in the browser's JavaScript console using console.log
EnableAllowList	Only allow Adobe Flash Player to load content from URLs that match patterns defined by AllowListUrlPattern directives in mms.cfg.
AllowListRootMovieOnly	Only apply loading restrictions to the parent SWF. Subsequent requests made from the loaded SWF to arbitrary URLs are allowed.
AllowListUrlPattern	Requires EnableAllowList=1. Specify a discrete URL or pattern to allow Adobe Flash Player to load content from.

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

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

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

Privacy options

Settings in this category let administrators disable access from Adobe Flash Player to connected webcams and microphones; and disable the ability to access the list of fonts installed on a user's computer.

AVHardwareDisable

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

If this value is set to 1, Adobe Flash Player 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*.)

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, Adobe Flash Player is blocked from accessing webcams or microphones. The `AVHardwareEnabledDomain` settings provide exceptions to that rule. Administrators can specify an allow 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.

EnableInsecureActiveXNavigateToURL

EnableInsecureActiveXNavigateToURL = [0,1] (0=false, 1=true)

Allows Administrators to override the Flash Player 32 and above behavior of more strictly enforcing Same Origin Policy with requests made from NavigateToURL() in the ActiveX Flash Player for IE and Edge on Windows. For the vast majority of Flash content, this change should be transparent. Affected content will generally be making a request using NavigateToURL() where the protocol, host and port did not exactly match and will likely be leveraging UNC paths, file:/// or other Windows-specific schemes in the destination. For increased security, we recommend administrators leave this feature disabled.

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)
```

Availability: Flash Player 9.0.28.0 and higher

This setting controls whether a SWF file playing via a browser plug-in 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.

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.

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 Action-

Script 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 read access to the local filesystem 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`.

EnableInsecureLocalWithFileSystem

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

Beginning with Flash Player 23, local-with-network permissions will now be applied to all local SWF content, regardless of the preference chosen at compile time.

Background

When playing Adobe Flash (SWF) content from the local filesystem, developers have historically been able to configure content to exclusively read from the filesystem,or communicate to the network. When this functionality was introduced over a decade ago, it enabled an interesting array of use-cases ranging from simple games to interactive kiosks. In the context of modern web security, we believe that filesystem functionality in the browser plugin should be retired. At the same time, Adobe AIR has been established as a robust, mature solution for delivering ActionScript-based content as a standalone application.

The vast majority of Flash Player users and content will be unaffected by this change. This change only impacts Flash content played from the local filesystem, using the browser. Flash content hosted on the Internet and local web servers, as well as the Standalone Flash Player remain unaffected. Users who require this functionality can add local content to the list of Trusted Locations in Flash Player.

Workarounds for Legacy Content

We highly recommend that administrators only circumvent these controls to enable content from trusted sources.

For Individuals:

1) On the affected system, go to the Flash Player Settings Manager:
   – Mac: System Preferences > Flash Player
   – Windows: Control Panel > Flash Player
2) Select the Advanced tab.
3) In the Developer Tools section, click the Trusted Location Settings button.
4) Click the "Add..." button and add relevant files and folders to the list.

For Google Chrome (and other PPAPI browsers):

1) Navigate to the Settings Manager page.
2) Choose Edit Locations > Add Locations from the popup list.
3) In the text field that appears, type or paste the file/folder path that you'd like to trust.
4) Click the "Confirm" button.

NOTE: Please be aware that the "Browse for files" and "Browse for folder" buttons do not function properly. You must manually type or copy/paste your path into the text field above the buttons to add the file or folder to the trusted list.

For System Administrators:

The legacy behavior can be restored by applying the EnableInsecureLocalWithFileSystem=1 flag to mms.cfg.

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 list of allowed 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 list of allowed 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, Adobe Flash Player will follow 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 Localstorageoptions). 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

Availability: Adobe Flash Player 9.0.115.0

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

This value specifies a hard limit, in MB, on the amount of local storage that Adobe Flash Player uses for the storage of common Adobe 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.

Update options

Adobe Flash Player supports software updates by periodically checking for new versions of the player on Adobe.com. Settings in this category configure the auto-update mechanism used by Flash Player. Administrators can increase or decrease the frequency of checks for newer versions, enable background updates, or disable auto-update entirely.

Windows and macOS platforms support an automatic update method called a notification update. A notification update is an anonymous check that is only performed when the player is loaded to view Adobe Flash content, typically in the browser. By default, an update check only occurs if it has been at least seven days since Adobe Flash Player last checked for updates. Flash Player never runs in the background to perform the notification update check.

In a notification update, a dialog box announces the availability of the update to the user to let the user either accept, postpone, or reject the update. If the user accepts the update, the new installer is downloaded and run.

On Microsoft Windows and macOS, Adobe Flash Player supports a *background* update that installs the update silently in the background, without any user interaction. A background update installs both the ActiveX and plug-in players when appropriate.

Update settings can be configured by users with admin rights. Admin users can set the frequency of the checks, disable notification updates, or disable background updates by using the Adobe Flash Player Settings Manager. For more information, see *Update options*.

If you want to enforce standardized update settings for all users, you can use the mms.cfg options discussed in this section. Also, ensure that those users who should not be allowed to change these settings are configured as standard users and do not have admin rights.

AutoUpdateDisable

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

If this value is set to 0 (the default), Adobe Flash Player lets a user with admin rights enable or disable all updates for all accounts on the machine in the Settings Manager.

If this value is set to 1, Flash Player disables all updates.

NOTE: If this value is set to 1, the `AutoUpdateInterval`, `DisableProductDownload`, `ProductDisabled`, and `SilentAutoUpdateEnable` options in this section are ignored, disabling all non-manual updates on the system.

AutoUpdateInterval

```
AutoUpdateInterval = [ number of days]
```

If this is a negative value (the default), Flash Player uses the notification update interval value specified in the Settings Manager. (If users don't make any changes with the Settings Manager, the default is every 7 days.) If this value is set to 0, Flash Player checks for an update every time it starts. If this is a positive value, the value specifies the minimum number of days between update checks.

This applies to Windows ActiveX and NPAPI plug-in, and Mac NPAPI and PPAPI plug-ins. Windows PPAPI uses a Task Scheduler item to check for an update and does not utilize this setting in the mms.cfg file.

This setting modifies the notification update check frequency used to announce an update is available via a notification pop-up window. It is NOT used to modify the background update check frequency.

Do NOT use this setting if the intent is to use Background Updates to update the client systems.

DisableProductDownload

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

If this value is set to 0 (the default), Adobe Flash Player can install native code applications that are digitally signed and delivered by Adobe. Adobe uses this capability to deliver Adobe Flash Player updates through the developer-initiated Express Install process, and to deliver the Adobe Acrobat Connect screen-sharing functionality. If this value is set to 1, these capabilities are disabled.

However, if you want to allow only specific product downloads, set this value to 0 (or omit it) and use the ProductDisabled option to specify which product downloads are not permitted.

ProductDisabled

```
ProductDisabled = application name
```

Availability: Adobe Flash Player 10.0.2

This option is effective only when DisableProductDownload has a value of 0, or is not present in the mms.cfg file. Use this directive to create a list of ProductManager applications that are not permitted to install or launch. Unlike most other mms.cfg options, you can use this option as many times as is appropriate for your environment.

SilentAutoUpdateEnable

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

Adobe Flash Player 11.2 for Microsoft Windows, and Adobe Flash Player 11.3 for macOS.

Enables Adobe Flash Player updates to install silently in the background with no user interaction.

- On Windows: Installs the ActiveX control,NPAPI plugin, and PPAPI plugin where appropriate.
- On macOS: Installs NPAPI plugin and PPAPI plugin where appropriate.

This type of update is called an Adobe Flash Player background update.

Standard users cannot disable background updates if they are enabled by an administrator.

Enabling silent auto updates (background updates) does not disable notification updates and users may still receive notifications to update Flash Player, instead of the update occurring silently in the background.

Depending on the type of browser, if the user has a browser open at the time of an update, the browser might not use the updated player immediately. For more information, see *Performing a background update*.

The default value is 0 to disable background updates.

SilentAutoUpdateServerDomain

```
SilentAutoUpdateServerDomain = yourDomain
```

Availability: Adobe Flash Player 11.2 for Microsoft Windows.

Enables you to host and deploy Flash Player background updates from an internal server. For more information, see *Background updates from an internal server*. When hosting background updates internally, Notification Updates are disabled.

SilentAutoUpdateVerboseLogging

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

Availability: Adobe Flash Player 11.2 for Microsoft Windows, and Adobe Flash Player 11.3 for macOS.

Enables logging of warning and error codes to FlashInstall.log during a background update. The location of the FlashInstall.log file depends on your platform. For more information, see *Player files and locations*.

The default value is 0 to disable logging.

Security options

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

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 Adobe Flash Player.

Flash Player 6 made security sandbox distinctions based on super domains. 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. As an example, a SWF file from www.example.com is placed in a different sandbox than a SWF file fromstore.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”
```

Availability: Adobe Flash Player 9 and above

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 (Windows) or APP (macOS) file extension. On macOS, you can specify either the name of the actual executable or the name of an application bundle within which the executable is located.

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

FullScreenInteractiveDisable

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

Availability: Adobe Flash Player 11.3 and above

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, and includes 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.

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

DisableNetworkAndFilesystemInHostApp

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

Availability: Adobe Flash Player 9 and above

This option is similar to *EnforceLocalSecurityInActiveXHostApp*, but applies to plug-ins as well as the ActiveXcontrol, and imposes stricter security controls. When a plug-in or ActiveX control is running within a specified application, 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 (Windows) or APP (macOS) extension. On macOS, you can specify either the name of the actual executable or the name of an application bundle within which the executable is located.

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

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*.

DisableSockets

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

Availability: Adobe Flash Player 9.0.115.0 and above.

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.

EnableInsecureActiveXMHTMLSupport

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

This setting allows Administrators to override the Flash Player 32 and above behavior of restricting the ability for Flash Player to launch when loaded from an MHTML (.mhtml or .mhtm) document. We recommend that administrators leave this feature disabled.

EnableInsecureByteArrayShareable

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

In Adobe Flash Player 30 and above, this setting allows Administrators to override default restrictions the shareable property of the ActionScript ByteArray API class. Shared ByteArrays are used to share data between threads with ActionScript "Workers." Shared ByteArrays are an advanced feature of the Action-

Script API set and not commonly used in the vast majority of published Flash content. For increased security, we recommend administrators leave this feature disabled.

EnableInsecureByteArrayShareableDomain

```
EnableInsecureByteArrayShareableDomain = domain name or IP address
```

Availability: Adobe Flash Player 30 and above.

This option overrides the default behavior disabling the shareable property of the ActionScriptByteArray class. The EnableInsecureByteArrayShareableDomain settings provide exceptions to that rule. Administrators can create a list of allowed domain names or IP addresses to which the EnableInsecureByteArrayShareable setting will apply. If the active security context is in the list of domains and IP addresses, then access to the sharable ByteArray property will be allowed. Otherwise, access to the sharable ByteArray will be denied.

For domain names, prefixing a * wildcard is allowed. For example, *.adobe.com would allow all Flash content with the "shareable" property to run on www.adobe.com, get.adobe.com, helpx.adobe.com, and so on. Wildcards are not allowed when specifying IP addresses.

For example, the following settings only allow SWFs using the shareable ByteArray property to servers at test.mydomain.com and 10.1.1.10:

```
EnableInsecureByteArrayShareableDomain=test.mydomain.com
EnableInsecureByteArrayShareableDomain=10.1.1.10
```

EnableSocketsTo

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

Availability: Adobe Flash Player 9.0.115.0.

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)
```

Availability: Adobe Flash Player 10.0.2.

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).

RTMFPP2PDisable

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

Availability: Adobe Flash Player 10.0.2 and higher.

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
```

Availability: Flash Player 10.0.2

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.

Protected mode options

Adobe Flash Player Protected mode is a security enhancement designed to limit the impact of attacks launched from malicious SWF files against Flash Player. In the Protected mode, SWFs are rendered using a sandboxed Flash Player runtime.

NOTE: Protected mode is available with Adobe Flash Player in 32-bit Mozilla Firefox 4.0 or later on Windows Vista and higher.

On Windows Vista and Windows 7, the Protected mode is enabled by default. However, you can disable it using the appropriate option in the `mms.cfg`.

ProtectedMode

```
ProtectedMode = [0, 1] (0 = off, 1 = on)
```

Availability: Adobe Flash Player 11.3 and higher.

This option specifies whether the protected mode is enabled. If enabled, on Windows Vista and higher, SWFs are rendered in 32-bit Mozilla Firefox 4.0 or higher using a sandboxed Adobe Flash Player runtime.

ProtectedModeBrokerAllowListConfigFile

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

Availability: Adobe Flash Player 11.3 and higher.

Protected mode prevents a number of actions that can be bypassed by creating a list of allowed actions (policies). The component that performs the actions based on the policies is called a “broker.” If a properly configured policy file is provided, the broker can bypass the application’s default restrictions.

If this option is set to true, provide a policy file.

Ensure the following if you want to provide a policy file:

- Name the policy file as `ProtectedModeBrokerAllowlistConfig.txt`.
- Provide policy file in the Flash directory:
  - 32-bit Windows - `%WINDIR%\System32\Macromed\Flash`
  - 64-bit Windows - `%WINDIR%\SysWow64\Macromed\Flash`

ProtectedModeBrokerLogfilePath

```
ProtectedModeBrokerLogfilePath = path to the log file
```

Availability: Adobe Flash Player 11.3 and higher.

Specifies the path to the log file to record the policy file violations. If a path is not provided, no file is created. This option is applicable only if `ProtectedModeBrokerAllowlistConfigFile` is set to true.

Hardware Options

The options in this category let you select appropriate settings for your computer hardware.

DisableHardwareAcceleration

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

If this option is set to 1, hardware acceleration is disabled. You can use this option if you suspect that hardware acceleration is causing your system to become unstable.

Audio Options

The options in this category let you select audio settings for your computer.

UseWAVPlayer

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

If this option is set to 1, Flash Player will use WAV Audio for playback instead of the Windows Core Audio APIs. Use this option if you face audio playback problems in Adobe Flash Player on Windows 7 or higher.

NetworkRequestTimeout

```
NetworkRequestTimeout = [1-30] (configurable from 1 to 30 seconds, default
= 5)
```

Availability: Adobe Flash Player 14 and higher.

If you encounter delays loading web content due to slow or blocked network access, reducing this number allows Flash Player to shorten the time it waits for a network response and possibly improve page responsiveness.

If the Flash content requires additional time before the server responds, increasing this value will extend the period before Flash Player gives up on the network request.

EnableInsecureJunctionBehavior

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

Availability: Adobe Flash Player 14 and higher.

This directive overrides the default restriction on write access to paths that traverse junction files in Windows filesystems. This directive only applies to Internet Explorer with Protected Mode disabled.

We recommend that Administrators use this flag as a short term workaround and instead focus on a solution where the user's appdata folder remains in the local user profile folder.

EnableLocalAppData

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

If this value is set to 1, the location where Adobe Flash Player stores Local Shared Objects will be changed from %APPDATA% to %LOCALAPPDATA%. For environments where the %APPDATA% folders on end-user machines are stored on a network volume, enabling this option causes Adobe Flash Player to write user data to the local system, minimizing possible security and performance impacts.

DefaultLanguage

```
DefaultLanguage = language name from chart below
```

This property allows the user or admin to override Flash Player's default language by specifying one of the languages in the table below.

Language	Value	Win	Mac	PPAPI
Arabic	ar	Y	Y	N
Bulgarian	bg	Y	Y	N
Czech	cs	Y	Y	Y
Danish	da	Y	Y	N
German	de	Y	Y	Y
Greek	el	Y	Y	N
English	en	Y	Y	Y
English - United Kingdom	en_gb	Y	Y	N
Spanish	es	Y	Y	Y
Estonian	et	Y	Y	N
Finnish	fi	Y	Y	N
French	fr	Y	Y	Y
Hebrew	he	Y	Y	N
Croatian	hr	Y	Y	N
Hungarian	hu	Y	Y	N
Italian	it	Y	Y	Y
Japanese	ja	Y	Y	Y
Korean	ko	Y	Y	Y
Azeri	lt	Y	Y	N
Latvian	lv	Y	Y	N
Norwegian	nb	Y	Y	N
Dutch	nl	Y	Y	Y
Polish	pl	Y	Y	Y
Portuguese	pt	Y	Y	Y
Portuguese - Portugal	pt_pt	Y	Y	N
Romanian	ro	Y	Y	N

Language	Value	Win	Mac	PPAPI
Russian	ru	Y	Y	Y
Slovak	sk	Y	Y	N
Slovenian	sl	Y	Y	N
Serbian	sr	Y	Y	N
Swedish	sv	Y	Y	Y
Thai	th	Y	Y	N
Turkish	tr	Y	Y	Y
Ukrainian	uk	Y	Y	N
Chinese - China	zh-CN	Y	Y	Y
Chinese - Taiwan	zh-TW	Y	Y	Y

IEClickToPlayBlocked

```
IEClickToPlayBlocked = domain name or IP address
```

This option only applies when EnableIEClickToPlay is enabled. This directive creates a list of servers on which all hosted Flash content will be blocked. If a domain or IP is included in the block list, the user will not be presented with a play button and the content will not render. This directive can be specified multiple times, with one domain or IP per entry.

For domain names, prefixing a * wild card is allowed. For example, *.adobe.com would allow block all Flash content hosted on www.adobe.com, get.adobe.com, helpx.adobe.com, etc. Wild cards are not allowed when specifying IP addresses.

IEClickToPlayBypass and IEClickToPlayBlocked directives can be used in conjunction with each other. For example, enterprises wishing to minimize Flash usage to only their company sub-domains can add the following to their user's MMS.CFG:

```
EnableIEClickToPlay = 1
IEClickToPlayBlocked = *
IEClickToPlayBypass = *.myenterprise.com
```

These two entries would disable all Flash playback except for that on any sub-domain of myenterprise.com, which would run without any user intervention.

EnableIEClickToPlay

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

Availability: Adobe Flash Player 27.0.0.130 and higher.

With Microsoft Internet Explorer on Windows 7 and below, administrators can change the default playback behavior to always prompt the user before playing SWF content.

Once enabled, visible Adobe Flash content within the page will be displayed with a “Play” button. When this play button is clicked, content playback will start immediately.

Please note that due to different methods used by content developers to instantiate Flash, clicking the play button may occasionally fail. If this occurs, we recommend that administrators use the IEClickToPlayBypass directive to add approved domains or URLs to a list of content allowed to automatically play on load. See *IEClickToPlayBypass* for more details.

IEClickToPlayBypass

```
IEClickToPlayBypass = domain name or IP address
```

This option is only applicable when EnableIEClickToPlay is enabled. This directive allows administrators to specify a list of servers on which all hosted Adobe Flash content will play back immediately, without user interaction. This directive can be specified multiple times, with one IP or domain per entry.

For domain names, prefixing a * wild card is allowed. For example, *.adobe.com would allow all Flash content to run on www.adobe.com, get.adobe.com, helpx.adobe.com, and so on. Wild cards are not allowed when specifying IP addresses.

IEClickToPlayBypass and IEClickToPlayBlocked directives can be used in conjunction with each other. For example, enterprises wishing to minimize Flash usage to only internal subdomains may add the following to their user's MMS.CFG:

```
EnableIEClickToPlay = 1
IEClickToPlayBlocked = *
IEClickToPlayBypass = *.myenterprise.com
```

These two entries would disable all Flash playback except for that on any sub-domain of myenterprise.com, which would run without any user intervention.

EventJitterMicroseconds

```
EventJitterMicroseconds = 0
(0 = disabled; entry not present = enabled)
```

Availability: Adobe Flash Player 30.0.0.113 and higher.

Setting this value to 0 disables an important mitigation for Spectre and Meltdown (CVE-2017-5753, CVE-2017-5715, CVE-2017-5754) style attacks, but may improve application performance in some limited circumstances. To enable the setting, delete the entry from the mms.cfg file.

TimerJitterMicroseconds

```
TimerJitterMicroseconds = 0
(0 = disabled; entry not present = enabled)
```

Availability: Adobe Flash Player 30.0.0.113 and higher.

Setting this value to 0 disables an important mitigation for Spectre and Meltdown (CVE-2017-5753, CVE-2017-5715, CVE-2017-5754) style attacks, but may improve application performance in some limited circumstances. To enable the setting, delete the entry from the mms.cfg file.

InsecureJitterDisabledDomain

```
InsecureJitterDisabledDomain = domain name or IP address
```

Adding domains to this whitelist disables important mitigations for Spectre and Meltdown (CVE-2017-5753, CVE-2017-5715, CVE-2017-5754) style attacks, but may improve application performance in some limited circumstances. This directive can be specified multiple times, with one IP or domain per entry. To re-enable the setting, delete the entries from the mms.cfg file.

For domain names, prefixing a * wildcard is allowed. For example, *.adobe.com would disable jitter mitigations on www.adobe.com, get.adobe.com, and helpx.adobe.com, and so on. Wildcards are not allowed when specifying IP addresses.

For example, the following settings would disable timer and event jitter at test.mydomain.com and 10.1.1.10:

```
InsecureJitterDisabledDomain=test.mydomain.comInsecureJitterDisabledDoma
in=10.1.1.10
```

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*.) 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*). 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*.)

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"`.

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

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

- Pop-up dialogs that appear when Adobe 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 (command-clicking on macOS) and choosing Settings from the context menu.
- The Flash Player Settings Manager, which the user can display by right-clicking (command-clicking on macOS) and choosing Global Settings from the context menu.

Users can also display the Flash Player Settings Manager from the native settings utility:

- **macOS**: System Preferences > Flash Player
- **Windows**:
  - **XP**: Control Panel > Flash Player
  - **Vista**: Control Panel > Classic View > Flash Player
  - **Windows 7 and above**: Control Panel\All Control Panel Items > Flash Player
- **Linux**: Although this varies slightly between distributions, it is usually Settings > Preferences > Flash Player.

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*.

NOTE: If you use the mms.cfg file to override user settings, the mms.cfg settings are unavailable or disabled to the end user. For example, when AutoUpdate is disabled via mms.cfg (AutoUpdateDisable=1), the Check for Updates section in the Settings Manager is disabled. If you think this might be confusing for your users, you might want to let them know that certain settings are unavailable to them.

Much of the information in this section is excerpted from the help page for Adobe Flash Player Settings (www.adobe.com/go/player_help_en). The documentation is geared towards end users and may help further clarify the available options.

NOTE: In the following sections, screen shots are provided to illustrate the pop-up dialog boxes and the tabbed Settings Panels. For Settings Manager pages, links are provided instead of screen shots, so you can navigate to that page and see the actual Settings Manager online.

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. Administrators can use the *AVHardwareDisable* 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*.

- 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.

Update options

Update options let the user specify whether Adobe Flash Player should display a notification when a new version is available, and how frequently to check for new versions. When installing the player on Windows and macOS, the user selects which option they want:

- Allow Adobe to install updates (recommended)
- Notify me to install updates
- Never check for updates (not recommended)

For Linux systems, users are automatically configured for notification updates. Use the Settings Manager or the mms.cfg file to change this setting.

You can use the *AutoUpdateDisable* and *AutoUpdateInterval* settings in the mms.cfg file to prevent the user from choosing auto-update, or to override the frequency of checking for new versions.

Note that any user can disable a notification update. However, background updates cannot be disabled. For more information on background updates, see the `SilentAutoUpdateEnable` option in *Update options*.

Use the Local Settings Manager to specify auto-update settings. On Microsoft Windows, access the Local Settings Manager from the Control Panel. On macOS, access it through the System Preferences. For Linux, access it by right-clicking on Flash content and selecting Global Settings from the context menu.

For more information on the Local Settings Manager, see http://www.adobe.com/go/global_privacy_-settings.

Security options

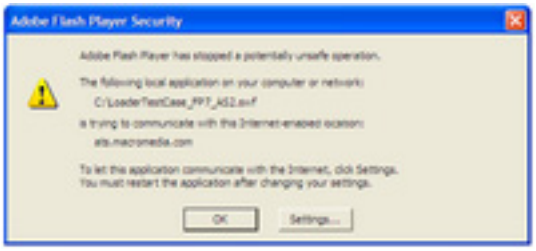
This section describes the security options available to end-users. For more information on Adobe Flash Player security in general, see *Security considerations* You can use a number of options in the mms.cfg file to override user security options; see *Security options*.

End users should rarely need to intervene in Adobe Flash Player security decisions. However, because the Flash security model evolves over time, occasionally Adobe Flash Player encounters a situation in which SWF content attempts to perform an operation that was permitted in a previous version of Adobe 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*). This enables more fine-grained control than the Ask/Allow/Deny choice, and works for SWF content of any version. Only local paths are applicable. Domain names and URLs have no effect, as remote content may never be placed in a local sandbox. Unlike the Ask/Allow/Deny choice, this list 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 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 the 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* in *Administration*.

Display options

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

The User FlashPlayerTrust directory

Application installers and 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*. Information on these trusted files is stored in a directory called the User FlashPlayerTrust directory. This directory registers files or directories as trusted for only the current user. For information on registering files as trusted for all users, see *The Global FlashPlayerTrust directory*. You can specify whether users can permit applications to be trusted; see *Security options*.

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 in the following location:

Windows Vista

C:\Users\username\AppData\Roaming\Macromedia\Flash Player\#Security\FlashPlayerTrust

Windows 2000 and Windows XP

C:\Documents and Settings\username\Application Data\Macromedia\Flash Player\#Security\FlashPlayerTrust

macOS

/Users/username/Library/Preferences/Macromedia/Flash Player/#Security/FlashPlayerTrust

Linux

GNU-Linux ~/.macromedia/#Security/FlashPlayerTrust

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

Security considerations

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 Adobe 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*.

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.

Newer versions of Adobe Flash Player sometimes include improvements to the security model of Adobe Flash Player and ActionScript that impact backward compatibility. In some instances, changes content developers to update their applications, and/or system administrators to specifically allow legacy behaviors as a workaround until content can be improved. See *About compatibility with previous Flash Player security models* for details.

A number of security-related settings can be controlled through the use of a config file (mms.cfg), which can be applied to a user's system when the player is deployed by an administrator.

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 Adobe Flash Player can never perform. As an example, SWF content running in Adobe Flash Player cannot determine or set the default location of a native File Picker dialog when attempting to upload or download a file from the user's filesystem. The default location shown in the native File Picker dialog is either the most recently browsed folder or the user's desktop, depending on the permissions in the environment. The running SWF content cannot read from or write to the transferred file. In fact, the SWF content that initiated the file transfer cannot access the transferred file or the file's location on the user's disk. Similarly, running SWF content 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*)
- How users can interactively allow or prohibit certain potentially malicious activities (see *User-configured settings*)

- How you can deploy a configuration file to override choices users might make with regards to security and privacy issues (see *Administration*)

The topic of cross-domain security may also be of interest to administrators. Some SWF content may require administrators to permit cross-domain requests by placing a cross-domain policy on the web server hosting the target content. An understanding of cross-domain mechanisms will help the administrator to craft policies that minimize the scope of granted permissions and associated attack surface. For more information, see *Data loading through different domains*.

NOTE: Users who are working in Adobe Animate to create applications have access to a number of ways to implement certain security features. These techniques are described in the Adobe Animate documentation. If developers in your organization are building SWF 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 by Adobe Flash Player.

Interaction between SWF content running 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 and 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: Restrictions 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

Application installers, end users, and administrators can specify that local SWFs are allowed to read from the local file system, determine the location where they are installed, read and write to and from servers, and cross-script with 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*.

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*.

The local-with-networking sandbox

When an ActionScript developer specifies that local SWF files should be assigned to the local-with-networking sandbox, the SWF files are allowed to access the network, but are blocked from accessing the local file system. Also, a SWF file running in the local-with-networking sandbox is only allowed to read network-derived data when permissions are present for that action.

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*. For information on how to control the end-users' ability to specify trusted files, see *AllowUserLocalTrust*.
- 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 Flash-PlayerTrust 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*. The other location doesn't require administrative access and applies only to the current user; see *The User FlashPlayerTrust directory*.

About compatibility with previous Flash Player security models

While great pains are taken to preserve the backward compatibility of SWF content when played in any version of Adobe Flash Player, security changes sometimes preclude the possibility of compatibility with

older versions. It is possible that SWF content that runs as expected in an old version of Adobe Flash Player may not run as expected in subsequent versions. In these cases, administrators can generally disable individual security sections in order to preserve backward compatibility until applications can be modified to accommodate new security requirements.

For example, local SWF files are blocked from communicating 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, local file system, or both, Adobe Flash Player would block those operations. In those instances, Adobe Flash Player would present a permission dialog to the end-user.

NOTE: In most security scenarios, restrictions are enforced silently with no opportunity for the end-user to intervene or loosen security constraints beyond modifying mms.cfg.

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*).
- Administrators or an installer program run with administrative access can place a global configuration file on the user's machine to specify that files should be trusted (see *The Global FlashPlayerTrust directory*).
- 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* in *Administration*).
- 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—Cross-domain policy files and socket policy files.

- Cross-domain 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.

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 9 Security white paper (www.adobe.com/devnet/flashplayer/articles/flash_player9_security_wp.html). This document focuses on how Flash Player 9.0.124.0 addresses a number of issues related to security, including features previously introduced in earlier versions of the product.
- Security changes in Flash Player 10 (http://www.adobe.com/devnet/flashplayer/articles/fplayer10_security_changes.html).
- 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.
- ActionScript 2.0 and Security (see the "Understanding Security" chapter in Learning ActionScript 2.0 in Adobe Flash).