

App-V Application Publishing and Client Interaction

White Paper Descriptor

This document provides information about how the client operates and where it stores data to support virtual applications.

Copyright © 2008 MICROSOFT CORPORATION

[Overview 4](#_Toc210557839)

[Intended Audience 4](#_Toc210557840)

[Prerequisite Knowledge 4](#_Toc210557841)

[Application Publishing on the Client 5](#_Toc210557842)

[App-V Client Data Storage Locations 5](#_Toc210557843)

[Client Cache 5](#_Toc210557844)

[OSD Cache and Icon Cache Directories 5](#_Toc210557845)

[Shortcut\_ex.dat File 6](#_Toc210557846)

[Per-Package File System Container Volumes 6](#_Toc210557847)

[User Location 7](#_Toc210557848)

[System Locations 7](#_Toc210557849)

[App-V Cache Behavior 8](#_Toc210557850)

[Cache Size Settings 8](#_Toc210557851)

[Cache Monitoring 9](#_Toc210557852)

[Cache Pre-allocation 9](#_Toc210557853)

[Cache Clean-up 9](#_Toc210557854)

[Registry Keys 10](#_Toc210557855)

[Application Publishing and Streaming 12](#_Toc210557856)

[Publishing Refresh 12](#_Toc210557857)

[Application Launch/Load 13](#_Toc210557858)

[AutoLoad Packages 14](#_Toc210557859)

[Application Streaming with Streaming Servers 16](#_Toc210557860)

[Application Streaming with File Servers 16](#_Toc210557861)

[Application Streaming with IIS 16](#_Toc210557862)

[Stand-Alone Mode with MSI 17](#_Toc210557863)

[Streaming Mode with MSI 17](#_Toc210557864)

[Application Upgrade 19](#_Toc210557865)

[Active Upgrade 19](#_Toc210557866)

[Differential Streaming 20](#_Toc210557867)

[Package and Data Management 23](#_Toc210557868)

[Initial Launch, Shutdown, and Subsequent Launch 24](#_Toc210557869)

[Package Upgrade 25](#_Toc210557870)

[Repair 26](#_Toc210557871)

[Unload/Delete 26](#_Toc210557872)

[Load 26](#_Toc210557873)

[Client Logging 28](#_Toc210557874)

[File Log 28](#_Toc210557875)

[System Event Log Level 29](#_Toc210557876)

[Conclusion 31](#_Toc210557877)

# Overview

The App-V client performs many tasks to present virtual applications that appear to be locally installed to a user. Often, these tasks are not looked at with the detail that the server components receive. In order to effectively support virtual applications, an understanding of the processes performed by the client is imperative.

This document will explain the following topics:

* Important file locations for storing App-V information and data
* Client cache settings and behavior
* Publishing and delivery scenarios
* Active upgrade
* Package and data management
* Client logging

## Intended Audience

This document is intended for App-V administrators to provide a greater understanding of the client operations to assist in supporting App-V.

## Prerequisite Knowledge

This document assumes an understanding of App-V infrastructures and concepts. In order to gain a greater command of App-V knowledge and better understanding of this document, please refer to the App-V Document Library located at: <http://go.microsoft.com/fwlink/?LinkId=127120> and the App-V Planning and Deployment Guide at: <http://go.microsoft.com/fwlink/?LinkId=122063>.

# Application Publishing on the Client

# App-V Client Data Storage Locations

The App-V client is a very important piece of the overall infrastructure. It uses many locations to store data and is responsible for putting that data together to give the appearance of a locally installed application to the user. This section will describe the location of these files and explanation of the data stored in them. This information will provide a foundation for the topics covered in the remainder of the document.

## Client Cache

The App-V client creates a cache file that is used to store the application packages for usage called **sftfs.fsd**. At application runtime, the cache file contents are mounted to the application virtualization drive letter in the virtual environment. The mount gives access to the file system and files in the package. When packages are inserted into the cache via streaming or MSI-based virtual application installs, the client stores them persistently in cache for subsequent launches. The **sftfs.fsd** file is located in the **Public** profile on Vista and the **All Users** profile on Windows XP. Both share the same path in their respective profile which is **Documents\SoftGrid Client**. It is possible to change the location of App-V cache during setup or post-installation. Changing its path after installation will require that the machine be rebooted to take effect.

## OSD Cache and Icon Cache Directories

The icon cache and OSD cache directories store the icons and OSD files for virtual applications. The icons in the icon cache are used to generate the program shortcuts, used for file type associations, and displayed in the App-V Client management console. The OSD cache stores the OSD files necessary for launching virtual applications. In a traditional connected App-V infrastructure, the publishing refresh process informs the clients of the location of the icon and OSD files for the virtual applications for the user performing the publishing refresh. Next, the client copies the icons and OSD files and places them into their respective directory. While in stand-alone mode, the client directories are populated by the MSI (which contains the icons and OSD files) or by using SFTMIME commands (used with SCCM R2).

An **OSD cache** and an **icon cache** directory are created on App-V client computers for the whole machine. The directories are located in the **Public** profile on **Vista** and the **All Users** Profile on **Windows XP** with the path of **Documents\SoftGrid Client**.

There is also an **icon cache** directory created for each individual user that stores per user icons. This is the default storage location for icons as App-V assigns applications per user. On Windows Vista it is stored under the user’s profile at **\AppData\Roaming\SoftGrid Client\**, and on Windows XP under the user’s profile at \**Application Data\SoftGrid Client.**

## Shortcut\_ex.dat File

The shortcut\_ex.dat file lists the application shortcuts that are present to support the virtual applications on the client. This file stores the location of the shortcuts that are created during the publishing refresh. There are two shortcut\_ex.dat files; one is for the user and one is for the machine. The per user file is updated with data from publishing refresh operations when the user logs off the machine with data from publishing refresh. The machine-based file is updated when adding a package using SFTMIME ADD PACKAGE with the /Global switch or adding an MSI-based package. These files are located in the profiles:

**Per User:** \UserProfile\AppData\Roaming\SoftGrid Client on Windows Vista and \UserProfile\Application Data\SoftGrid Client on Windows XP.

**Per Computer:** \Public\Documents\SoftGrid Client on Windows Vista and \All Users\Documents\SoftGrid Client on Windows XP

## Per-Package File System Container Volumes

The per-package file system container volumes store changes that are made to packages made by users or system processes. These changes allow for users and the machine to make setting and configuration changes to the base package, without affecting it. These changes are stored in several PKG files that are described below. The files are individual for each package and stored in unique directories that are created by combining the Package Root directory name where the package was installed on the sequencer and the first portion of the package GUID. An example for **Microsoft Office 2007** where the Package Root is **OFF2K7.V1** and the package GUID for office is **5C99B562-F61F-4009-AB16-B38E16093AE4** the resulting directory would be **OFF2K7.V1-5C99B562-F61F-4009.** Two directories for each package will be created, one for the user’s profile and one for the machine at the following locations:

**Windows XP**

* Per-user at: %USERPROFILE%\Application Data\SoftGrid Client
* Per-machine at: All Users\Documents\SoftGrid Client\AppFS Storage

**Windows Vista**

* Per-user at: %USERPROFILE%\AppData\Roaming\SoftGrid Client\ and %userprofile%\AppData\Local\SoftGrid Client\
* Per-machine at: Public\Documents\SoftGrid Client\AppFS Storage

NOTE: Due to changes in profiles in Windows Vista a third directory is created for the temporary version of the PKG file while the application is in use.

The following description of the files describes how data is populated into these files and will be covered in further detail in a Package and Data Management section of this document.

### User Location

The **usrvol\_sftfs\_v1.pkg** file contains user-specific files that are modified or new files that are created by any user process in the virtual environment. This volume also contains the virtual environment configuration as modified by the user.

### System Locations

**UsrVol\_sftfs\_v1.pkg** contains new or modified user-specific data from a system process that is not associated with a specific user context but is associated with a specific package.

**GlblVol\_sftfs\_v1\_<SID>.pkg** contains application-specific files that are modified by any user process in the virtual environment. The SID of the user is appended to the volume name to uniquely identify it.

**GlblVol\_sftfs\_v1\_S-1-5-20.pkg** contains any application-specific data that is modified by a system process. The well-known SID for system is appended to the volume. In SoftGrid 4.0 and 4.1, this volume was used for all modified application data; in 4.2 and 4.5 modifications are separated into those made by system processes such as the Listener, and those made by user application processes. User modifications go instead to the Application Data Isolation Volume. The global package volume also contains the virtual environment configuration for system processes.

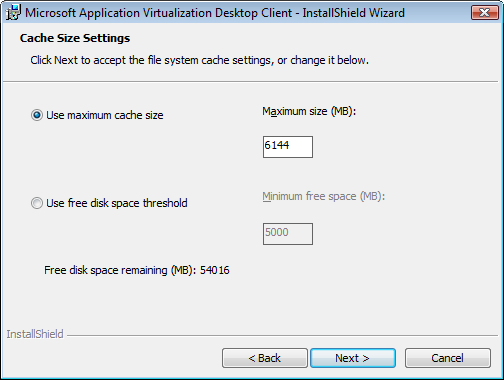
# App-V Cache Behavior

The App-V client cache is important for designing and supporting an App-V infrastructure. Understanding the settings and the way the cache behaves will allow administrators to properly setup and support their App-V client community. The following topic explains how to configure the cache size, and how the App-V client manages the cache as more applications are added.

## Cache Size Settings

The client cache has two options for configuration: maximum cache size and minimum available free disk space. The minimum available free disk space setting is new and provides administrators the flexibility of allowing the cache to grow based on available disk space. The absolute maximum cache size is 1TB even if the client has additional disk space available. The cache size settings can be configured during the setup of the App-V client or post-installation using the registry. When configuring the settings during setup the default value for maximum cache size is 6GB and can be set to a maximum of 1TB (1,000,000MB). The default value for minimum available disk space is 5GB and, if the free disk space remaining is less than 5GB the client will calculate the value to ½ of the remaining free disk space as the maximum cache. This setting will not allow for growth as it will be set in the FileSize registry value.

Figure 1: Setup Cache Size Settings



NOTE: If using an automated setup of the App-V client use the SWICACHCESIZE for setting a maximum cache file size and MINFREESPACEMB to allow the cache to grow until the minimum free space threshold is larger than available disk space.

The following registry keys are used to set the cache size:

**HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\SoftGridClient\CurrentVersion\AppFS\**

**FileSize** is a DWORD value that is the maximum size in megabytes of the file system cache file (sftfs.fsd). If this value is changed in the registry, you must also change the **State** to **0** and reboot the machine. The FileSize setting can be adjusted to a larger value in the App-V Client Management Console and does not require a restart of the computer.

Note: Setting the state to 0 and rebooting the machine will flush the file system cache and all packages that were in the cache will be removed.

Note: The following SFTMIME command can be used to flush the client cache:

**SFTMIME remove obj:app /global /complete**

**MinFreeSpaceMB** is a DWORD value that specifies the amount of free space in megabytes that needs to be available on the host before the cache size increase.

Important: If both of the registry settings are configured the client will prefer to use the **FileSize** setting over the **MinFreeSpaceMB** setting.

### Cache Monitoring

The cache can be monitored using performance counters on the client. In Windows Performance Monitor the App-V client counter is called **App Virt Client Cache** and has the following counters.

* % free space: The percentage of unused cache.
* Cache free space (MB): The size in MB of unused cache.
* Cache size (MB): The size in MB of the space reserved for virtual applications.

### Cache Pre-allocation

During a launch, load, or import of a package, the client first verifies that there is enough space available in the cache based on its settings before proceeding with the operation. This ensures there is enough space available in the cache before completing the operation.  If there is not enough space in the cache for the entire package, including primary and secondary feature blocks, the operation will fail unless the App-V client can clean up the cache or expand the cache based on the minimum free space setting on the client. This cache clean-up process will be covered in the next section of this document.

### Cache Clean-up

During normal operations on the client for launch, load, or import, the App-V client could run into a situation where there is not enough space in the cache for a package or the disk does not have enough space to allow the cache to grow. In these situations the App-V client can remove unused packages from the cache to free up space for a new package.

The App-V client can dynamically unload the least recently used package or packages from cache to make room for new packages. The App-V client can remove any package that is not locked or currently in use. The client will select the packages to remove based on the last launch time or last accessed time if the package has never been launched. If two packages have the same last launch or accessed time, the larger package will be unloaded first. In the event of a tie between multiple packages the client will randomly remove one of the packages.

During the background streaming process (auto loading), if there is not enough space in the cache, a package will not be unloaded from cache. An error is logged in the event log. The following registry keys configure the settings for cache clean-up:

### Registry Keys

HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\SoftGrid\CurrentVersion\AppFS\

| Value | Type | Description |
| --- | --- | --- |
| UnloadLeastRecentlyUsed | DWORD | If set to 1 (default), then when a user attempts to launch or load an application, if there’s not enough space in the cache to pre-allocate the package, the client will attempt to free space in the cache by unloading the packages already cached, based on a Least Recently Used algorithm. Values are 0 and 1. |
| MinPkgAge | DWORD | Specifies the minimum amount of time, in days, since a package was last accessed before it can be included in the Least Recently Used candidates for unload. |

Figure 2: App-V Cache Clean-up Flow Chart



# Application Publishing and Streaming

Being able to publish and deliver applications to users and computers is critical to managing the virtual applications in an environment. With the introduction of App-V 4.5, new components and features exist that were not present in previous versions. These changes were made to further enhance the App-V capabilities and answer customer needs. The components and features that influence the publishing and delivery of applications are the following:

* App-V Streaming Servers
* Application Source Root (ASR)
* Icon Source Root (ISR)
* OSD Source Root (OSD)
* Override URL
* File and IIS-based package streaming
* MSI-based application delivery
* AutoLoad of packages

The following detailed look at the publishing and delivery process will enable the reader to identify the available options for these processes. Additional information about topics in this section can be found at:

<http://go.microsoft.com/fwlink/?LinkId=122939>

<http://go.microsoft.com/fwlink/?LinkId=122063>

## Publishing Refresh

The publishing process that has been available in previous versions of App-V is still available with the release of App-V 4.5. The Management Server is the only App-V component that can perform the publishing refresh process for App-V client computers. The following steps detail what occurs during a publishing refresh operation.

The publishing refresh can be triggered by user login (default) or as a background timed operation. These settings can be configured on the client or specified on the Management Server using provider policies.

1. The client computer sends the user’s Kerberos ticket to the Management Server for authentication.
2. After successful authentication, the client computer establishes a connection to the Management Server. The App-V client supports connections between the client and the server to commence using RTSP/RTSPS or a web based provisioning service.

NOTE: The App-V Management Server does not include a web-based provisioning service. In order to use this option for publishing refresh, a custom web-based provisioning service would have to be created.

1. Next, the client will send a request for publishing information to the Management Server over RTSP or RTSPS depending on the client configuration.
2. The Management Server will contact the Data Store and retrieve a list of the application records and build an XML file, named applist.xml, containing the applications that have been published for the requesting user.
3. The client computer will process the XML file from the Management Server that contains the location of the ICO and OSD files. These files will be copied to the client computer and populated into the OSD cache and icon cache directories. The communication protocol being used to download ICO and OSD files will be specified in the publishing information and would include SMB or HTTP/HTTPS.

NOTE: The use of OSDSourceRoot and IconSourceRoot registry values can allow a client computer to receive its ICO and OSD files from an alternate location than the one provided in the publishing information received from the Management Server. These values can be configured via Group Policy using the App-V ADM Template or directly in the registry on an App-V client.

1. The client computer creates the appropriate shortcuts and registers file type associations based on the publishing information from the server.
2. Finally, any offline reporting information will be sent to the Management Server to be placed in the Data Store for reporting.

This process happens each time the publishing refresh is performed by the client computer.

## Application Launch/Load

After applications have been published to a user there will be a time when an application is initially launched. During this initial launch process for an application the package that contains the application will have to be loaded, unless the auto-load features have been enabled to preload packages for published applications. In App-V 4.5, several options are available for loading an application. The following steps discuss how the operation would occur using a Management Server as the location for the application packages, followed with additional options that are available for making the application packages available in different environments.

1. A user attempts to launch an application by double-clicking a shortcut or by opening an associated file type association.
2. The App-V client sends the initiating user’s ticket to the Management Server for authorization to launch the application. The client will also send the GUID of the application to verify that the most current version of the package is delivered. Licensing will be checked if enforced by the assigned provider policy.
3. After successful authorization and licensing, the client will verify the cache space required for primary and secondary feature blocks of the package is present. If there is not enough space in the cache the App-V client will need to remove packages to make space, increase the size of the cache, or fail to load the application.
4. After ensuring disk space is available, the client will stream the primary feature block of the package (SFT file) from a server with the appropriate package. App-V supports streaming package files from the following locations:
   1. App-V Management Server using RTSP or RTSPS
   2. App-V Streaming Server using RTSP or RTSPS
   3. File Server using SMB/CIFS
   4. IIS Server using HTTP/HTTPS

NOTE: The registry value ApplicationSourceRoot can be configured during setup, using the App-V ADM Template, or via the registry. This value will override all HREF elements in all OSD files on an App-V client to source from a different server and/or protocol.

NOTE: Using SFTMIME ADD or CONFIGURE PACKAGE with the /OVERRIDEURL switch allows an administrator to change the streaming server on an individual package.

1. When the primary feature block is streamed to the App-V client, the application’s virtual environment will be built and the client will attempt to launch the application.
2. The secondary feature block will be streamed to the App-V client when a user requests information from it on a block-by-block basis called an Out of Sequence Operation, unless autoload settings (described below) are enabled.

### AutoLoad Packages

One new feature in App-V 4.5 that can affect the streaming of packages is the autoload settings in the registry. These settings are presented during the App-V client installation and can be adjusted in the registry. By default, when a user logs onto an App-V client or after launching an application the client will attempt to load all previously used applications in their entirety in the cache. This will ensure that all assets required for an application will be available locally on the computer. For mobile users, this option will enable them to leave the network and be able to use any application that they have previously launched without needing to contact a streaming server. However, this will not give them all of the assets required for packages that they have not launched before. The autoload options can be changed to download all applications published for a user and can also be additionally triggered by any background publishing refresh.

NOTE: If authorization is required for cached applications, at least one application from each package downloaded in the background would need to be launched successfully, in order to cache authorization of the package.

Note: To prevent excessive network traffic in the morning after provisioning a new application to many users, it is recommended to avoid auto loading ALL applications that are provisioned to a user On Login.

The following registry values control the auto load behavior for App-V clients.

**HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\SoftGrid\4.5\Configuration**

| Value | Default | Description |
| --- | --- | --- |
| AutoLoadTriggers | 5 (On Launch and on Login) | AutoLoad is a client runtime policy configuration parameter that enables the secondary feature block of a virtualized application to be streamed to the client automatically in the background. The AutoLoad Triggers are flags to indicate events that initiate AutoLoading of applications. AutoLoad implicitly uses background streaming to enable the app to be fully loaded into cache. Feature block 1 will be loaded first and the remaining feature blocks will be loaded in the background to allow for foreground operations (i.e., user interaction with applications) to take place and provide optimal perceived performance. Bit mask values:  (0) Never: No bits are set (value is 0), no auto loading will be performed, as there are no triggers set.  (1) OnLaunch: Perform background loading when user launches an application.  (2) OnRefresh: Perform background loading anytime a publishing refresh occurs.  (4) OnLogin: Perform background loading for any application when a user logs in.  Default: 0x5 (OnLaunch|OnLogin) |
| AutoLoadTarget | 1 (Previously Used) | This parameter indicates what will be autoloaded when any given AutoLoadTriggers occur. Bit mask values:  (0) None - No autoloading, regardless of what triggers may be set.  (1) PreviouslyUsed (default) - If any AutoLoad Trigger is enabled, load only the packages where at least one app in the package has been previously used by a user (i.e., launched or precached,). This targets ‘important’ apps, meaning apps that have been used before are likely to be more important to a user than apps that have never been launched.  (2) All - If any enabled AutoLoad trigger is enabled, all applications in the package (per package) or for all packages (set for client) will be automatically loaded, regardless of whether they have ever been launched. |

### Application Streaming with Streaming Servers

Another new component in App-V 4.5 is the Streaming Server. This server does not provide the publishing refresh process and will rely on another publishing resource to properly perform this operation for the clients. This can be achieved by implementing a Management Server to handle the publishing refresh and then using the Application Source Root (ASR) setting on the client to point to the Streaming Server. Another option would be to use the manifest.xml file created during sequencing and script the package publishing using SFTMIME. These options can give flexibility in branch office scenarios where the expense of additional servers would outweigh the business value to the user population. This is based on the fact that a Management Server would require a SQL Server. Streaming Servers implement authorization via NTFS permissions.

### Application Streaming with File Servers

A file server can be used to deliver packages in an App-V infrastructure. However, it cannot provide the publishing refresh process to the clients. As was previously documented before, the file server would rely on a Management Server for the publishing refresh process or some other manual publishing to the client. This file server could be an actual server or in a very small office a powerful workstation machine. File servers implement authorization by using NTFS permissions.

### Application Streaming with IIS

IIS can be used to deliver packages in an App-V infrastructure. However, it cannot provide the publishing refresh process to the clients. The client will support a publishing refresh over an HTTP/HTTPS connection but currently App-V has no web-based publishing refresh service. As previously documented, the IIS streaming option would require a Management Server or some manual process to publish applications. IIS Servers implement authorization by using NTFS permissions.

### Stand-Alone Mode with MSI

The App-V 4.5 Sequencer supports creating an MSI during the sequencing process. This process does not have a formal publishing or streaming procedure. This MSI will contain the ICO, OSD, and Manifest.xml files that are necessary for publishing the application on the machine it is run from and how to import the SFT file into the App-V client cache. The package file (SFT) is not part of the MSI that is generated during sequencing and will need to be in the same directory as the MSI to successfully complete by default. In some scenarios it may be necessary to place the SFT files at an alternate location like a file server. In this scenario the administrator could use the SFTPATH parameter to specify an alternate location of the SFT file. This would not require the SFT file to be in the same directory as the MSI. In Standalone mode the package will be published and the SFT file contents will be loaded into the client cache completely. The following command shows how to install an MSI based package when the SFTFILE is located on a file server.

Msiexec.exe /i \\PathToMsi\packagename.msi SFTPATH=\\server\share\package.sft /q

The following registry values must be set on the client to enable Standalone Mode to support MSI deployment:

HKLM\SOFTWARE\Microsoft\Softgrid\4.5\Client

* Network\Online = 0
* Configuration\RequireAuthorizationIfCached = 0
* Configuration\AllowIndependentFileStreaming = 0
* Permissions\ToggleOfflineMode = 0 (Recommended)
* Network\LimitDisconnectedOperation = 0 (Recommended)

### Streaming Mode with MSI

Another option is to use MSI based packages but choose to stream the SFT file contents from an App-V Server, file server, or local file location. In this scenario the package will be published, but the SFT file will be streamed to the App-V client cache. The process of streaming the SFT will be done by default as part of the installation of the MSI, but could optionally be configured to happen when the user launches the application the first time. Streaming mode enables features of streaming mode, like Active Upgrade to still be available. In order to stream the SFT file contents when using an MSI based install, the MODE and OVERRIDEURL options listed below will need to be used:

Msiexec.exe /i \\PathToMsi\packagename.msi MODE=STREAMING OVERRIDEURL=\\\\server\share\package.sft LOAD=FALSE /q

The following registry value must be set on the client to enable Streaming Mode to support MSI deployment:

HKLM\SOFTWARE\Microsoft\Softgrid\4.5\Client

* Configuration\AllowIndependentFileStreaming = 1

**NOTE: To support streaming mode with MSIs do not configure the options for Stand-Alone Mode. The App-V client must be operating Online in order for streaming to work properly.**

**Table of MSI based package switches**

| MSI Switch | Values | Description |
| --- | --- | --- |
| MODE | **Stand-alone** or streaming | **Stand-Alone (Default):** Specifies packages are published, loaded, and serviced by MSI distribution. The App-V Client has no interaction with any App-V Server  **STREAMING:** Specifies packages are published and optionally loaded by MSI distribution. The App-V Client is configured with Application Source Root (ASR) or OVERRIDEURL is set thru MSI switch for the location (App-V Server, File Server, or local file) of the package file for streaming |
| SFTPATH | **NULL** or Qualified path to SFT | In Stand-Alone mode only, it used to specify the location of the SFT package file if it is not located in the same directory as the MSI file executed for publishing |
| OVERRIDEURL | **NULL** or Qualified path to SFT | In Streaming mode only, it is used to specify the location of the SFT package file for streaming. This can be an App-V Server (RTSP/RTSPS URL), UNC, or local file location. |
| LOAD | **TRUE** or FALSE | Specifies loading the entire package into the App-V Client cache.  NOTE: If this switch isn’t used the default is TRUE, so even in Streaming mode it will initially load the package during the execution of the MSI. |

# Application Upgrade

Upgrading applications is a time-consuming and expensive operation for any organization. App-V can help reduce the administrative overhead by reducing the amount of time to deploy updated applications and result in less user impact. App-V has two features that enhance a normal upgrade procedure. These two features are Active Upgrade and Differential Streaming. The following section will explain these two features and the support for different App-V infrastructures.

## Active Upgrade

The Active Upgrade feature became available in the 4.0 version of App-V to help support upgrading applications. Prior to the 4.0 release, a sequencer could take a package and upgrade it, but when it was to be deployed all users had to be disconnected from the existing package before a new version could be deployed for upgrade. With the Active Upgrade feature, administrators can seamlessly upgrade a package without requiring App-V servers to be shut down or users to disconnect. This was enabled by allowing a single package in the Data Store to be associated with multiple versions of SFT files.

When an administrator adds a new SFT version to an existing package, the App-V client will retrieve the new SFT file the next time the application is launched. This is done when the user launches the application as it will send the package GUID to the Management or Streaming Server and the client’s current version of the package. The server will send the latest version associated with the package GUID and, if newer than the one in the cache on the client, the client will stream the later version before launching the application. This doesn’t require disconnecting users from the server or bringing the server down to ensure there are no active connections on the Management or Streaming Server.

Note: the client is also able to perform an Active Downgrade of the cached application if the cached version is newer that the latest version that is currently published.

When using App-V 4.5 there are multiple options for delivery of packages which are described earlier in this document. Using a traditional Management or Streaming Server, the Active Upgrade feature will support the client sending the package GUID and current client cached version. This will allow for the App-V client to learn of new versions when the package GUID is checked in the Data Store and a newer version is present.

App-V 4.5 supports streaming packages from an IIS or File Server, however, the same rules do not apply. If using an IIS or File Server for streaming packages, the package GUID and version number are not sent to the IIS or File Server as they do not understand the request for the latest version for a package GUID. When using IIS and File Servers for streaming, the administrators would have to explicitly change the HREF path in the OSD to point to the location of the new version, as shown below,

**First Version of HREF path**

HREF="https://appvms.contoso.com:443/content/Microsoft\_Office\_2003\_MNT/**Microsoft\_Office\_2003\_MNT.sft**

**New Version of HREF path**

HREF="https://appvms.contoso.com:443/content/Microsoft\_Office\_2003\_MNT/**Microsoft\_Office\_2003\_MNT\_v2.sft**

This update to the OSD file is copied to the client during a publishing refresh. In a traditional model with a Management or Streaming Server, updates are applied on the next launch of the application package, where with an IIS or File Server it will be on the next publishing refresh. In certain scenarios this process could be advantageous, as it would give the administrators a way of staging the upgrade. This could be done to allow for a back-end component to be upgraded, and then release the upgrade to the client application after introducing the package.

Although this feature was very powerful before App-V 4.5, the previous versions of Active Upgrade required a user to download the entire Primary Feature Block of the new version of the package during the Active Upgrade process, even if only small portions of the feature block have been changed. In the next section of the document a new feature of App-V 4.5 improves Active Upgrade with the addition of Differential Streaming.

## Differential Streaming

Before App-V 4.5, Active Upgrade loading by the client resulted in a full package being downloaded from the server to the client, even though the client only needs the blocks from the upgrade package that differ from the previous version of the package. The full download causes significant inefficiency, which for very large enterprises could have serious impact on bandwidth and performance capabilities of the enterprise network.

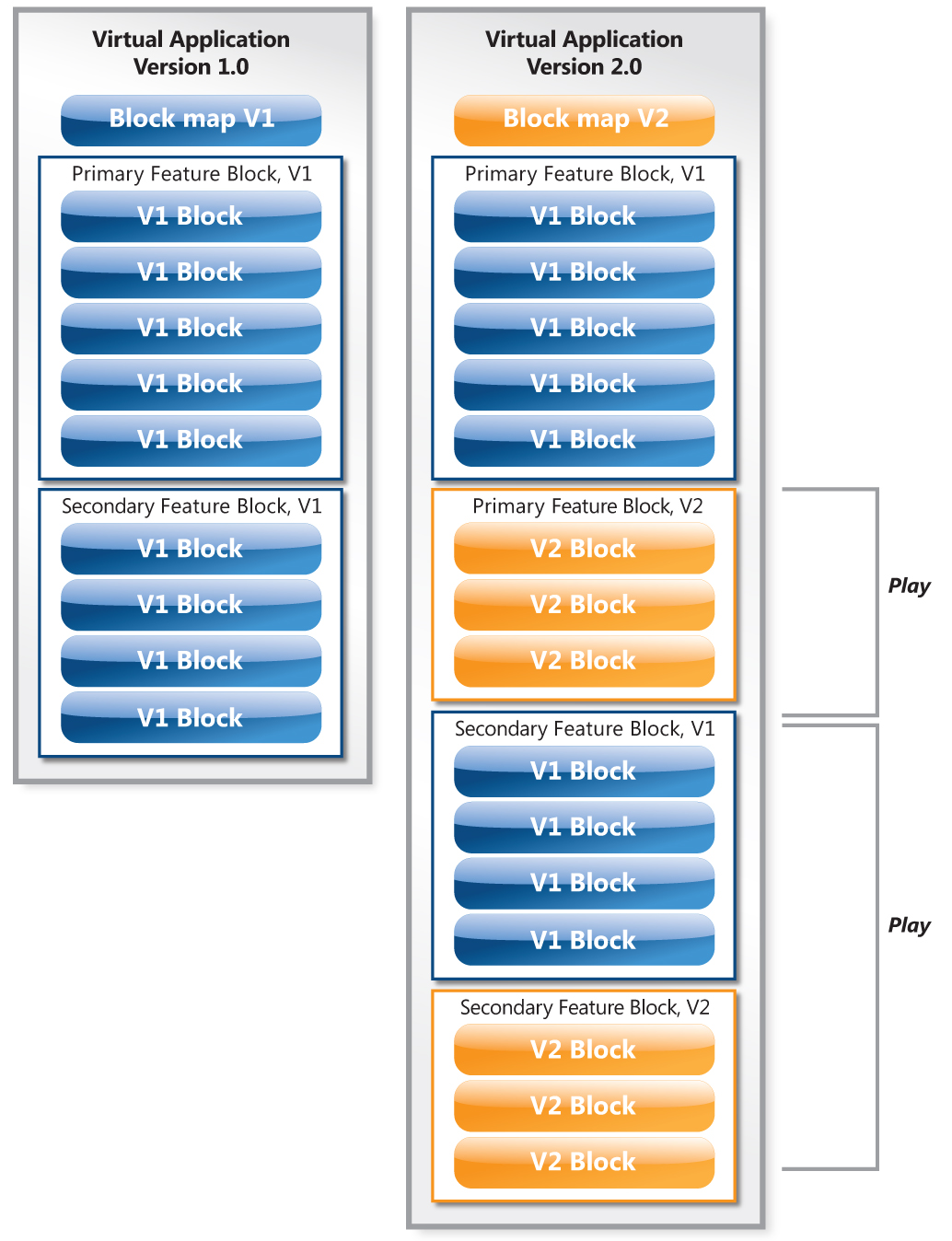
When sequencing packages using App-V 4.5, the sequencing process sets block version information within the SFT file to the same version number as the package. This allows for Differential Streaming, where the Active Upgrade process now has the information needed to only download the blocks of data that differ from one version of a package to the current package. The process of setting block version information is automatically done while sequencing but requires the App-V 4.5 Sequencer. If an organization already has sequenced applications with previous versions of the Sequencer they will need to reopen the packages and save a new SFT that sets block versions for subsequent upgrades to support Differential Streaming. Using the Sequencer to “Open Packages for Upgrade” on a package previously created by an App-V 4.5 Sequencer will automatically create the block version information required for Differential Streaming.

The client will be responsible for comparing the SFT file versions with the package version to determine when an Active Upgrade request is needed. Client version checking should be performed when an application is launched or loaded. During these operations, the client will compare the application’s block versions given in both the cached version of the app and the block map provided by the server. Newer version detection in the server-provided block map will trigger an update.

Differential Streaming support is based on the package being created with the App-V 4.5 Sequencer and being streamed to an App-V 4.5 client. All supported streaming methods support Differential Streaming, including Management and Streaming Servers, IIS Servers, and File Servers.

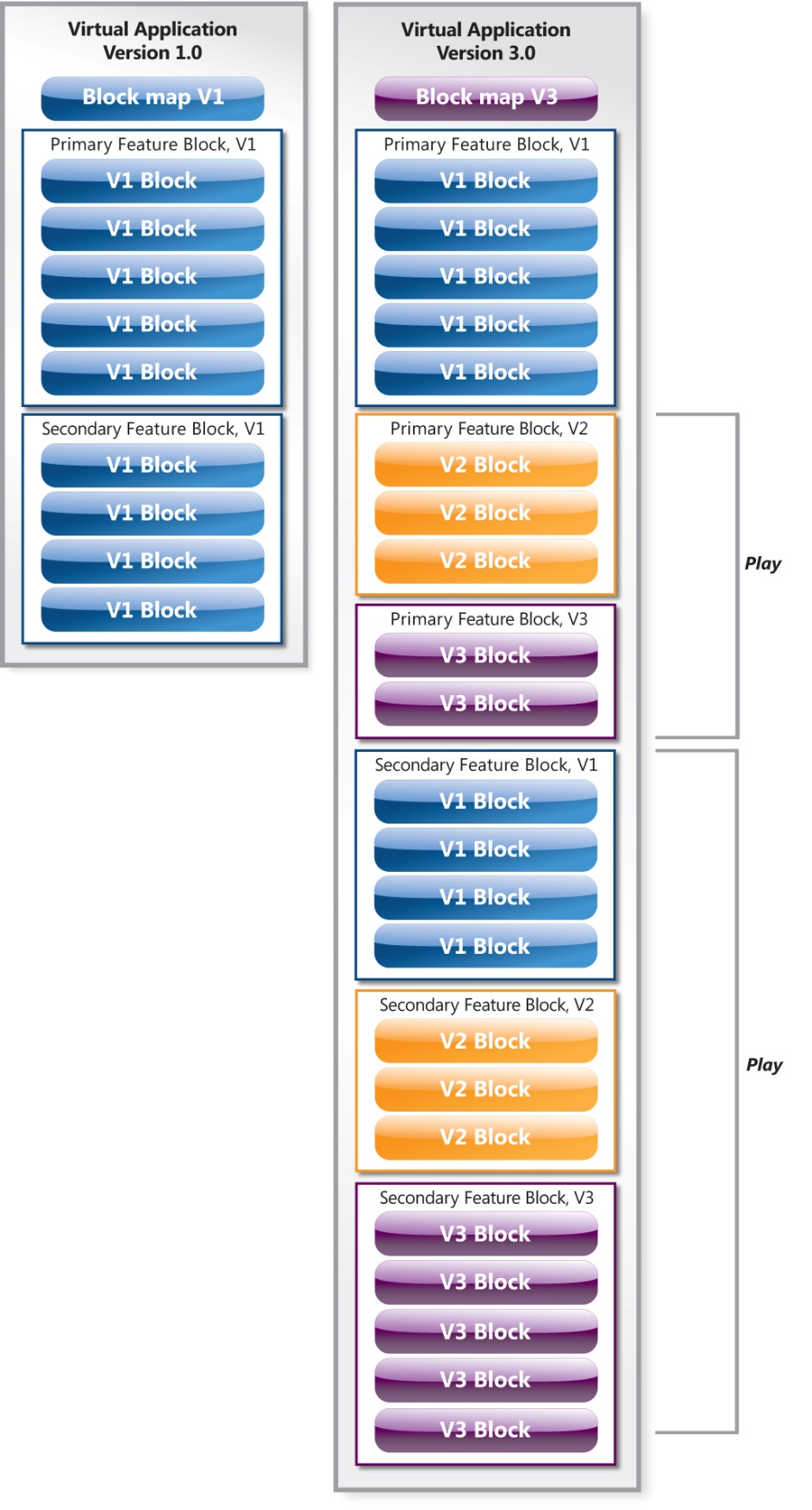
The following diagram shows an ordered approach and version 1 is assumed not to be fully cached. For an Active Upgrade, a Play request may be made by the client for the new version primary feature block and another play for v2 + v1 Secondary Feature Block.

Figure 3: Active Upgrade from Version 1 to Version 2



The following figure shows an update from V1 to V3. Note the V3 update SFT contains the new blocks from both V2 and V3 as well as the V1 blocks still needed. This allows any client who may have been disconnected to get all updates without having to do multiple updates.

Figure 4: Active Upgrade from Version 1 to Version 3



# Package and Data Management

Previously in this document, the per-package file system container volumes were defined and their locations were described. This section will focus on the purpose and impact of these files on virtual applications. Please reference the locations of the files referenced in this section with the information provided previously in this document and in the following diagrams.

Figure 5: Per-package file system container volumes

NOTE: In this and all subsequent figures for package and data management, volumes in **Blue** are **Application Data** and volumes in **Purple** are **User Data.**

### Initial Launch, Shutdown, and Subsequent Launch

Packages are delivered to an App-V client with configurations and files that were captured during sequencing. When a user launches an application from a package the per-package file system container volumes are created as TMP files to store any changes that occur while the application is launched. Upon shutdown, the TMP files are saved permanently to their respective locations as PKG files. These files are stored for subsequent launches to preserve any changes that were made when the application was launched. On subsequent launches, the PKG files are copied to TMP files for changes that are made while the application is launched.

Figure 6: System container volume behavior during launch, shutdown, and subsequent launch

### Package Upgrade

When a package is upgraded, the App-V client will delete the GlblVol PKG files for the package and new GlblVol TMP files will be created for the upgraded package. The UserVol PKG files will be copied to TMP files for changes during the application launch. This means that an application upgrade will not remove any “user data” changes that have been made on the package. Upon shutdown of the newly upgraded package, the GlblVol TMP files will be saved as PKG files and the UserVol TMP files will be copied to PKG files for subsequent launches.

Figure 7: System container volume behavior during package upgrade

### Repair

Using the App-V Client Management Console gives administrative options to support virtual applications. The repair function deletes the usrvol\_sftfs\_v1.pkg file from the user profile for a specific package. In this scenario a call comes into the help desk. A user is experiencing problems with a virtual application that others are not experiencing. The help desk discovers that the user has made some changes to the problem application. The help desk walks the user through performing a repair of the application which will remove any user-specific changes to the package. The user then launches the applications and the problem is gone.

NOTE: The user will lose all changes that have been made for the entire package, not just the one application they were calling about.

NOTE: An alternate method would be to rename the usrvol\_sftfs\_v1.pkg and reopen the application. That would allow the administrator or user to rename it back if that did not solve the problem and not lose the individual user settings. The default repair operation permanently deletes the user’s changes.

**Clear**

The clear task can be performed to remove the shortcuts and file type associations for an application. None of the PKG files are removed during this administrative operation. This process could be done if a user is experiencing problems with file type associations and you need to remove all of the FTAs created by App-V.

### Unload/Delete

The Unload or Delete operation will delete all of the PKG files except the per user UsrVol\_sftfs\_v1.pkg file. This will retain any user changes if the package is then loaded or published again to the client. This process also deletes the package from the client cache. The difference between the two functions is the unload operation does not delete the shortcuts and file type associations created for the application in the package.

### Load

The load operation can manually populate the cache with the package. This process will also create the per-machine TMP files for the first launch of the application to capture any changes made to the application that are done by system processes or for non user-based files.

Figure 8: System container volume behavior during repair, load, unload, and delete

# Client Logging

The App-V client logs information to both the Windows Event log and to a local log file. Both of these logging options can be adjusted to change the type of information that is captured. The local file based log can only be accessed by a local administrator of the machine or the SYSTEM account on the machine.

## File Log

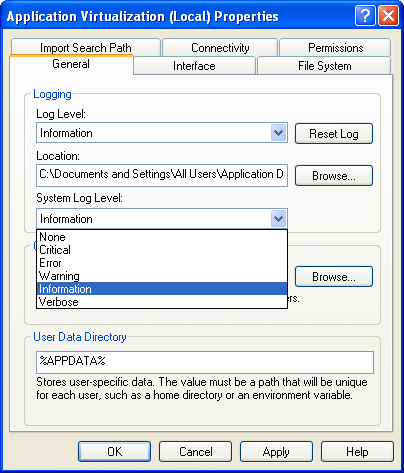
The local cache is located in profiles \All Users(Public on Vista)\Application Data\Microsoft\Application Virtualization Client\sftlog.txt. The settings for this file can be modified using the registry at the following location:

HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\SoftGrid\4.5\Configuration

| Value | Default | Description |
| --- | --- | --- |
| LogFileName | CSIDL\_COMMON\_APPDATA\Microsoft\Application Virtualization Client\ | Client log file. Can be modified to change the log file location. You must restart the sftlist service after changing this value. |
| LogMinSeverity | 4, Informational | Controls which messages are written to the log. The value indicates a threshold of what is logged – everything at and below that value is logged. For example, a value of 0x3 (Warning) indicates that Warnings (0x3), Errors (0x2), and Critical Errors (0x1) are logged.  Value Range: 0x0 == None, 0x1 == Critical, 0x2 == Error, 0x3 == Warning, 0x4 == Information (Default), 0x5 == Verbose |
| LogRolloverCount | 4 | Defines the number of backup copies that are kept of the log file, sftlog.txt when it is reset. The valid range is 0-9999. The default is 4. A value of 0 means no copies will be kept. |
| LogMaxSize | 256 | Defines the size in megabytes that the log file can reach before being reset. The default size is 256 MB. When this size is reached, a log reset will be forced on the next write attempt. |

## System Event Log Level

The system event logging level can be configured using the App-V Client Management Console by right clicking the root node and going to properties.



Managing the event logging that will be recorded can also be modified by using the following registry key:

**HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\SoftGrid\Client\CurrentVersion\Configuration\**

This registry value indicates the logging level at which App-V log messages will get written to the NT event log. The value indicates a threshold of what is logged – everything at and below that value is logged. For example, a value of 0x3 (Warning) indicates that Warnings (0x3), Errors (0x2), and Critical Errors (0x1) are logged.

| Value | Default | Recommend Management Server Configuration |
| --- | --- | --- |
| SystemEventLogLevel | 4 (Information) | 0 == None  1 == Critical  2 == Error  3 == Warning  4 == Information  5 == Verbose |

# Conclusion

This document provides a deep understanding of the App-V client and how it supports virtual applications. Readers should be able to more effectively make administrative and support decisions as they pertain to the client after reading this documentation.