



## **Macromedia Flash MX Security**

by Mike Chambers

March 2002

Copyright © 2002 Macromedia, Inc. All rights reserved.

The information contained in this document represents the current view of Macromedia on the issue discussed as of the date of publication. Because Macromedia must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Macromedia, and Macromedia cannot guarantee the accuracy of any information presented after the date of publication.

This white paper is for information purposes only. **MACROMEDIA MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.**

Macromedia may have patents, patent applications, trademark, copyright or other intellectual property rights covering the subject matter of this document. Except as expressly provided in any written license agreement from Macromedia, the furnishing of this document does not give you any license to these patents, trademarks, copyrights or other intellectual property.

The Macromedia logo and Macromedia Flash are either trademarks or registered trademarks of Macromedia, Inc. in the United States and/or other countries. The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Macromedia, Inc.  
600 Townsend Street  
San Francisco, CA 94103  
415-252-2000

## Contents

<b>Introduction to Macromedia Flash MX .....</b>	<b>1</b>
What is Macromedia Flash MX? .....	1
Why is Macromedia Flash MX secure? .....	2
How does Macromedia Flash MX protect users' privacy?.....	2
<b>Macromedia Flash Player 6 sandbox .....</b>	<b>3</b>
What is the Sandbox?.....	3
Domain-based authentication .....	3
How is the sandbox implemented? .....	4
Local file I/O access .....	4
Cross-movie communication .....	5
Flash-JavaScript communication .....	8
LiveConnect API.....	8
ActiveX Control API.....	10
<b>Security of data transport .....</b>	<b>12</b>
Data encryption with SSL .....	12
One-way data encryption within Macromedia Flash MX using md5.....	13
Security of data transfer from Macromedia Flash MX projectors.....	13
Security of data and algorithms within a Macromedia Flash MX movie.....	14
Security concerns of an open format technology .....	14
Best practices for securing data within a Macromedia Flash movie .....	14
<b>Virus or trojan security concerns .....</b>	<b>15</b>
Macromedia Flash playback through projectors.....	15
Macromedia Flash projectors as a carrier of viruses.....	15
Malicious files disguised as Macromedia Flash files .....	15
Macromedia Flash movies that contain malicious code .....	16
Macromedia Flash movies as e-mail attachments .....	16
<b>Resources .....</b>	<b>17</b>
<b>Acknowledgments .....</b>	<b>17</b>



Macromedia takes security very seriously and has done everything possible to ensure that Macromedia Flash is a secure technology that protects the user's security and privacy. This document discusses security and privacy concerns with Macromedia Flash content including a number of potential security issues associated with playing content from the local file system that originates from untrustworthy or unknown sources.

## Introduction to Macromedia Flash MX

### What is Macromedia Flash MX?

The full potential of the Internet is untapped. Its potential has been limited by the constraints of the user experience of the web, which is characterized by a lack of user-centric design and technologies.

The Internet experience is becoming less about *browsing*. Instead, it is transforming into a broader Internet of *doing*. Whether booking travel, managing a business, or communicating with friends, users are interacting with the web. For this to be effective, the content and usability of Internet applications must significantly improve to make users more successful.

Change is here. The Internet is being enabled by standard, rich-client technology, and it's transforming what organizations and people can do with rich content and applications that offer radically more effective user experiences across a broad range of devices and platforms.

Macromedia Flash Player, the Macromedia rich client, is the most widely distributed software in the history of the Internet and is bundled with Internet Explorer, AOL, Netscape Navigator, Opera, and Windows XP. Over 414 million web users can see Macromedia Flash content immediately, without having to download a player. Macromedia Flash Player is also available on a growing number of Internet connected devices such as wireless handhelds, iTV and game consoles.

The Macromedia Flash MX solution couples the ubiquitous client technology of Macromedia Flash Player with the Macromedia Flash MX development environment and optimized server-side connectivity that allows rich Internet application development to be accomplished quickly. This enables a more immediate, consistent and familiar experience for end users and maximizes the customer experience online.

## **Why is Macromedia Flash MX secure?**

Macromedia Flash player has an extensive list of checks, restrictions and features in order to ensure that Macromedia Flash content is secure and safe. These include:

- The ability to use the encryption capabilities of the browser, including SSL, to encrypt all communications between a Macromedia Flash movie and the server
- An extensive sandbox security system that limits transfer of information that might pose a risk to security or privacy
- Macromedia Flash player does not allow web content to read data from the local drive except for SharedObjects that were created by that domain.
- Macromedia Flash player cannot write any data to the disk except for data that is encapsulated in SharedObjects.
- Macromedia Flash player does not allow web content to read any data from a server that is not from the same domain unless that content explicitly allows access.
- Macromedia Flash player does not allow web content to place more than 100k of data on the local disk from a single domain.
- Macromedia Flash player enables the user to disable the storage of information for any domain.
- Macromedia Flash player does not allow data to be sent from a camera or microphone unless the user gives permission for a domain.

## **How does Macromedia Flash MX protect users' privacy?**

Macromedia understands the importance of user privacy and has worked to ensure that Macromedia Flash content does not invade that privacy.

This is accomplished in the following ways:

- Users must explicitly approve access to local web cameras and microphones.
- A domain cannot access data stored from another domain. This is similar to the way web browser cookies work.
- Macromedia Flash does not have access to any personal information, unless explicitly provided by the user.

## Macromedia Flash Player 6 sandbox

The Macromedia Flash Player 6 implements a browser-like security sandbox scheme in order to ensure the security and privacy of both Macromedia Flash movie, as well as the client machine.

### What is the Sandbox?

The sandbox defines a limited space in which a Macromedia Flash movie running within the Macromedia Flash Player is allowed to operate. Its primary purpose is to ensure the integrity and security of the client's machine, and as well as security of any Macromedia Flash movies running in the player.

The concept of the sandbox is simple. A Macromedia Flash movie executes inside a sandbox. Any information inside the sandbox can be communicated only to the domain from which the movie came. Access to information within the sandbox from outside of the sandbox is severely limited.

A Macromedia Flash movie's sandbox consists of:

- Everything contained in the SWF file
- User actions directed at the Flash movie
- Servers in the domain from which the Macromedia Flash movie originated (see *Domain-based authentication* below)
- Local SharedObjects written by Macromedia Flash movies from the same domain (see *Local file I/O access* below)
- Limited configuration information about the computer on which the Macromedia Flash movie is running

In order to support local development and testing of Macromedia Flash movies by developers, movies accessed as local files (either on the end user's local disk or on LAN servers) have no sandbox limitations. This is consistent with the added caution users should always take when running any type of file locally.

### Domain-based authentication

As mentioned above, the sandbox includes all servers in the domain, or more correctly the "nth-level sub-domain", from which the movie originated. Membership in the domain is checked by comparing server names.

Two servers are in the same domain if the following are true:

- 1 The server names have the same number of tokens.
- 2 There are at least 3 tokens.
- 3 All but the first token are the same except for differences in case.

If a server's name is expressed in only one or two tokens (for example, "foo" for foo.macromedia.com), the server is in a domain by itself. If a server is identified by its IP address, the server is also in a domain by itself.

Examples:

- A.B.macromedia.com and C.b.macromedia.com are in the same domain.
- A.b.macromedia.com and www.macromedia.com are not in the same domain (different number of tokens).
- A.b.macromedia.com and a.c.macromedia.com are not in the same domain (second token differs).

The idea is that a large domain should be allowed to split into isolated sub-domains by using additional tokens in server names. This algorithm is also applied to URLs that include country abbreviations (for example, <http://www.hrp.org.uk/>).

The limitation on one- and two-token server names and servers identified by IP address avoids hard-to-compute cases, such as when a user has multiple search domains in their TCP/IP configuration. Two examples that are a result of this are:

- foo and bar are not in the same domain, but foo.macromedia.com and bar.macromedia.com are in the same domain.
- www.macromedia.com and macromedia.com are not in the same domain, despite the fact that they address the same server.

The reason for the domain limitation is to protect servers behind firewalls from being attacked by machines outside the firewall. A movie from outside.hacker.org must not be able to read data from any files stored on inside.macromedia.com when the movie is running on a machine behind the Macromedia firewall.

## How is the sandbox implemented?

### ***Local file I/O access***

Macromedia Flash Player 6 allows limited local file storage through the use of SharedObjects. SharedObjects, which can be thought of in terms of web browser cookies, allow developers to store and retrieve information to the users local file system.

SharedObjects have the following restrictions:

- Data can only be written to a specific directory. The developer cannot control that directory.
- The user has control of how much data can be stored including the ability to disable storage on a per domain basis.
- Data stored on local file system is binary, serialized data controlled by Macromedia Flash player. The files all have a standard header and a .SO file extension that prevents placement of executable code or application data that might inadvertently be launched by the user.
- Data access is restricted by the domain-based authentication rules of Macromedia Flash Player (see *Domain-based authentication* above).

All file exchange through Macromedia Flash player is limited to a specific directory on the client's machine that is created by Macromedia Flash Player 6. The following limitations apply to the directory when being accessed from a Macromedia Flash movie playing in a web browser:

- The user controls how much information can be stored for a given domain. The default amount of data that each domain can store on the user's machine is 100K. Each shared object is stored in its own file. Each file will always be counted as using at least 1000 bytes of data so that the default 100K limit will allow up to 100 different SharedObjects for a single domain.
- The data for each domain is stored in an individual directory in the "application data" directory. This directory is user accessible.

These restrictions ensure that:

- Non-Flash information on the user's machine cannot be overwritten by a Macromedia Flash movie running in a browser.
- The threat of Denial of Service attacks, caused by filling the user's hard drive with data, is minimized.

The player controls the format of the information stored on the file system by Macromedia Flash Player. Developers do not have the ability to control the format of the data. The information is a binary serialization of the data being stored, and thus cannot be used for malicious attacks on the client machine.

Finally, data contained within SharedObjects may only be accessed under the domain-based rules discussed above. For example, this means that data in a SharedObject set by a movie from `www.domain1.com` cannot be accessed by a movie from `www.domain2.com`.

### ***Cross-movie communication***

Macromedia Flash Player sandbox also applies when loading a SWF into an existing Macromedia Flash movie. However, movies loaded from separate domains exist within their own sandboxes. This isolates them from the other movies currently playing in the player. Content within a movie's sandbox cannot penetrate outside the sandbox, and content outside the sandbox cannot see into the sandbox.

It is also possible for movies to communicate with each other through the LocalConnection object. The sandbox security model also applies to cross-movie communication using this method. The key rule to remember is that a movie in domain A cannot extract any information from a movie in domain B unless the movie from domain B has given explicit permission for domain A to have access to it. As mentioned above, this rule must be strictly kept to prevent a SWF on the public Internet from loading a SWF from behind a firewall and extracting data. As a result of this, each sandbox receives its own copy of the `_global` scope. Code within a sandbox can only access the global object for that sandbox.

However, it is sometimes necessary for two movies to reside in separate domains, yet access each other's data. For example, the Macromedia Answers Panel is loaded from the local disk by the Macromedia Flash authoring tool, but may wish to update itself by accessing the Macromedia website. In this case, it is necessary to allow a movie loaded from `www.macromedia.com` to exchange data with the movie loaded from the local disk.

We meet this need using the “tunneling” feature of the sandbox. Tunneling is implemented by the ActionScript method `System.security.allowDomain` and has the following syntax:

```
System.security.allowDomain(domain1, ..., domainN);
```

This command grants `domain1` through `domainN` access to the sandbox of the SWF that executes the command.

For example, the Answers Panel in the Macromedia Flash MX authoring tool has a shim SWF that is loaded as a local file. The Answers Panel SWF that is loaded from `macromedia.com` needs to have access to the shim's variables. So the shim calls:

```
System.security.allowDomain("macromedia.com");
```

This command adds `macromedia.com` to the shim's “friends” list. Any SWF loaded from `macromedia.com` or sub domains, such as `sub.macromedia.com`, can now access variables in the shim SWF.

Access cannot be revoked once it has been granted and there is no way to retrieve a list of allowed domains.

Limited communication between sandboxes is possible. ActionScript code in one sandbox may obtain a reference to the top-level object of another sandbox and modify the movie clip properties. For instance, if `movie1.swf` loads `movie2.swf` into the movie clip `_level0.mcHolder`, the following apply:

- `movie1.swf` code can access `_level0.mcHolder` and modify the movie clip properties.
- `movie1.swf` cannot access any other properties of `_level0.mcHolder`.

The movie clip properties that are available are listed in the properties folder of the Action Panel's toolbox and include the following:

- `_alpha`
- `_currentframe`
- `_droptarget`
- `_focusrect`
- `_framesloaded`
- `_height`
- `_name`
- `_quality`
- `_rotation`
- `_soundbuftime`
- `_target`
- `_totalframes`
- `_url`
- `_visible`
- `_width`

- `_x`
- `_xmouse`
- `_xscale`
- `_y`
- `_ymouse`
- `_yscale`

The following table lists the basic ActionScript functions and the security restrictions placed upon them.

**Table 1:** *Security restrictions on basic actions*

Action	Security restrictions
gotoAndPlay/gotoAndStop	No security restriction unless a target path is specified, for example: <code>gotoAndPlay("mc:1");</code> If a target path is specified the caller must be in the same sandbox as the target movie clip.
play	No security restriction
stop	No security restriction
toggleHighQuality	No security restriction
stopAllSounds	No security restriction
getURL	No security restriction
FSCommand	Caller must be in the same sandbox as HTML page
loadMovie	No security restriction
loadMovieNum	No security restriction
unloadMovie	No security restriction
unloadMovieNum	No security restriction
loadVariables	Caller must be in the same sandbox as the target movie clip
loadVariablesNum	Caller must be in the same sandbox as the target movie clip
tellTarget	Caller must be in the same sandbox as the target movie clip
ifFrameLoaded	No security restriction
print	No security restriction
printNum	No security restriction
printAsBitmap	No security restriction
printAsBitmapNum	No security restriction

The sandbox restrictions do not apply to Macromedia Flash movies and projectors loaded from the local file system, with the following exception:

A sandbox from a Flash movie that is not a local file, may not access a local file's sandbox. For example, if movie1.swf on a user's local disk load movie2.swf from an HTTP server, the following is true:

- movie1.swf and movie2.swf are granted separate sandboxes.
- movie1.swf may access the contents of movie2.swf.
- movie2.swf is barred from accessing the contents of movie1.swf.

### **Flash-JavaScript communication**

Macromedia Flash Player supports a JavaScript API for controlling movie properties, getting and setting variables, and invoking functions. These capabilities are also restricted by the Macromedia Flash player sandbox security model.

## **LiveConnect API**

The Netscape plug-in version of Macromedia Flash Player exposes an API via Netscape Navigator's LiveConnect interface. This API may be accessed by Java and JavaScript running inside Netscape Navigator.

The security restrictions on the LiveConnect API parallel the sandbox restrictions for ActionScript. A similar sandbox is created for the HTML page, using the HTML page's URL. For this sandbox, the normal rules are enforced. Variables from one sandbox cannot be accessed from another sandbox. Movie clips in another sandbox cannot be controlled. The movie clip properties of a top-level movie clip in another sandbox may be read, but not written to.

The following table lists all of the properties and methods in the LiveConnect API and their associated security restrictions.

**Table 2:** *Security restrictions on Macromedia Flash Player LiveConnect API*

<b>Java/JavaScript method</b>	<b>Security restrictions</b>
boolean IsPlaying();	No security restriction
void Play();	HTML page must be in the same sandbox as _level0
void StopPlay();	HTML page must be in the same sandbox as _level0
int TotalFrames();	No security restriction
int CurrentFrame();	No security restriction
void GotoFrame(int position);	HTML page must be in the same sandbox as _level0
void Rewind();	HTML page must be in the same sandbox as _level0
void Back();	HTML page must be in the same sandbox as _level0
void Forward();	HTML page must be in the same sandbox as _level0
int PercentLoaded();	No security restriction

<b>Java/JavaScript method</b>	<b>Security restrictions</b>
boolean FrameLoaded(int frameNum);	No security restriction
int FlashVersion();	No security restriction
void Pan(int x, int y, int mode);	No security restriction
void Zoom(int percent);	No security restriction
void SetZoomRect(int left, int top, int right, int bottom);	No security restriction
void LoadMovie(int layer, String url);	HTML page must be in the same sandbox as _level0
void TGotoFrame(String target, int frameNum);	HTML page must be in the same sandbox as the targeted movie clip
void TGotoLabel(String target, String label);	HTML page must be in the same sandbox as the targeted movie clip
int TCurrentFrame(String target);	HTML page must be in the same sandbox as the targeted movie clip
String TCurrentLabel(String target);	HTML page must be in the same sandbox as the targeted movie clip
void TPlay(String target);	HTML page must be in the same sandbox as the targeted movie clip
void TStopPlay(String target);	HTML page must be in the same sandbox as the targeted movie clip
void SetVariable(String name, String value);	HTML page must be in the same sandbox as the variable's container object
String GetVariable(String name);	HTML page must be in the same sandbox as the variable's container object
void TSetProperty(String target, int property, String value);	HTML page must be in the same sandbox as the targeted movie clip
String TGetProperty(String target, int property);	HTML page must be in the same sandbox as the targeted movie clip
void TCallFrame(String target, int frameNum);	HTML page must be in the same sandbox as the targeted movie clip
void TCallLabel(String target, String label);	HTML page must be in the same sandbox as the targeted movie clip
double TGetPropertyAsNumber(String target, int property);	HTML page must be in the same sandbox as the targeted movie clip
void TSetProperty(String target, int property, double value);	HTML page must be in the same sandbox as the targeted movie clip

**Note:** If LiveConnect is unavailable, these commands cannot be used and these security restrictions do not apply.

## ActiveX Control API

Macromedia Flash Player for Internet Explorer on the Windows platform is an ActiveX Control. This ActiveX Control supports a COM API for querying properties of and manipulating Macromedia Flash movies.

This API is most frequently used by JavaScript code on the same HTML page as a Macromedia Flash movie. The ActiveX API closely resembles the LiveConnect API provided to JavaScript in Netscape Navigator.

The security restrictions on the ActiveX Control API parallel the sandbox restrictions for ActionScript. A similar sandbox is created for the HTML page, using the HTML page's URL. For this sandbox, the normal rules are enforced. Variables from one sandbox cannot be accessed from another sandbox. Movie clips in another sandbox cannot be controlled. The movie clip properties of a top-level movie clip in another sandbox may be read, but not written to.

The following table lists all of the properties and methods in the ActiveX Control API and their associated security restrictions.

**Table 3:** *Security restrictions on Macromedia Flash Player ActiveX Control API*

Property/method	Security restrictions
SWRemote property	No security restriction
TGetPropertyNum method	HTML page must be in the same sandbox as the targeted movie clip
TSetPropertyNum method	HTML page must be in the same sandbox as the targeted movie clip
TCallLabel method	HTML page must be in the same sandbox as the targeted movie clip
TCallFrame method	HTML page must be in the same sandbox as the targeted movie clip
TGetProperty method	HTML page must be in the same sandbox as the targeted movie clip
TsetProperty method	HTML page must be in the same sandbox as the targeted movie clip
GetVariable method	HTML page must be in the same sandbox as the variable's container object
SetVariable method	HTML page must be in the same sandbox as the variable's container object
TStopPlay method	HTML page must be in the same sandbox as the variable's container object
TPlay method	HTML page must be in the same sandbox as the variable's container object
TCurrentLabel method	HTML page must be in the same sandbox as the targeted movie clip
TCurrentFrame method	HTML page must be in the same sandbox as the targeted movie clip
TGotoLabel method	HTML page must be in the same sandbox as the targeted movie clip

<b>Property/method</b>	<b>Security restrictions</b>
TGotoFrame method	HTML page must be in the same sandbox as the targeted movie clip
Quality2 property	No security restriction
BGColor property	No security restriction
EmbedMovie property	No security restriction
DeviceFont property	No security restriction
Scale property	No security restriction
Base property	HTML page must be in the same sandbox as _level0
Menu property	No security restriction
SAlign property	No security restriction
WMode property	No security restriction
PercentLoaded property	No security restriction
Pan method	No security restriction
Zoom method	No security restriction
SetZoomRect method	No security restriction
FrameNum property	READ: No security restriction; WRITE: HTML page must be in the same sandbox as _level0
Movie property	No security restriction
Loop property	READ: No security restriction; WRITE: HTML page must be in the same sandbox as _level0
BackgroundColor property	No security restriction
AlignMode property	No security restriction
ScaleMode property	No security restriction
Quality property	No security restriction
Playing property	READ: No security restriction; WRITE: HTML page must be in the same sandbox as _level0
TotalFrames property	No security restriction
ReadyState property	No security restriction
FlashVersion method	No security restriction
FrameLoaded method	No security restriction
CurrentFrame method	No security restriction
GotoFrame method	HTML page must be in the same sandbox as _level0
Rewind	HTML page must be in the same sandbox as _level0
Forward	HTML page must be in the same sandbox as _level0
Back	HTML page must be in the same sandbox as _level0
IsPlaying	No security restriction
Stop	HTML page must be in the same sandbox as _level0

<b>Property/method</b>	<b>Security restrictions</b>
StopPlay	HTML page must be in the same sandbox as _level0
Play	HTML page must be in the same sandbox as _level0
LoadMovie	HTML page must be in the same sandbox as _level0
FlashVars property	HTML page must be in the same sandbox as _level0

## Security of data transport

### Data encryption with SSL

A Macromedia Flash movie playing in a browser has many of the same security concerns as an HTML page being displayed in a browser. This includes the security of the Macromedia Flash movie while it is being loaded into the browser, as well as the security of communication between Macromedia Flash and the server after the movie has loaded and is playing in the browser. In particular, data communication between the browser and the server is susceptible to being intercepted by third parties. The solution to this issue in HTML is to encrypt the communication between the client and server in order to make any data captured by third parties undecipherable and thus unusable. This is done by using an SSL enabled browser and server.

Since Macromedia Flash movies running within a browser use the browser for almost all of its communication with the server, it can take advantage of the browser's built-in SSL support. This allows communication between the Macromedia Flash movie and the server to be encrypted. Furthermore, the actual bytes of the Macromedia Flash movie are encrypted while they are being loaded into the browser.

Thus, by playing a Macromedia Flash movie within an SSL enabled browser through an HTTPS connection with the server, you can ensure that the communication between the Macromedia Flash Player and the server is encrypted and secure.

The one exception to this is the way Macromedia Flash uses persistent sockets (through the ActionScript XMLSocket object), which does not use the browser to communicate with the server. Because of this, it cannot take advantage of the built-in encryption capabilities of the browser. However, it is possible to use one-way encryption algorithms written in ActionScript to encrypt the data being communicated.

## One-way data encryption within Macromedia Flash MX using md5

Md5 is a one-way encryption algorithm described in *rfc1321*. This algorithm has been ported to ActionScript, which enables developers to secure one-way data using the md5 algorithm before it is sent from Macromedia Flash movie to the server. More information about rfc1321 can be found at <http://www.faqs.org/rfcs/rfc1321.html> or <http://www.rsasecurity.com/rsalabs/faq/3-6-6.html>.

The md5 algorithm works by creating a secure one-way hash from a string. The hash cannot be unencrypted into its original string, but it can be compared against other data that has been encrypted using the md5 algorithm. While this does not offer the ability to encrypt and decrypt data like with SSL, it can be useful if you need to transfer sensitive data and cannot use the SSL capabilities of a browser. These cases might include:

- A movie needs to be able to run in a browser that is not capable of SSL.
- You do not have access to an SSL-capable web server.
- You need to communicate with the server using an XML socket.
- You are using a Macromedia Flash projector to run the movie.

The process of encrypting data is as follows:

- 1 Macromedia Flash encrypts the data with a one-way hash.
- 2 Macromedia Flash sends the data to the server.
- 3 The server receives the data.
- 4 The server validates the one-way hash against a pre-existing hash.

An md5 algorithm written in ActionScript can be downloaded from <http://flashexperiments.insh-allah.com/#MD5>.

## Security of data transfer from Macromedia Flash MX projectors

Since Macromedia Flash projectors are executables that run outside a browser, they cannot take advantage of the SSL capabilities of a browser. Because of this, if sensitive information is transferred between the projector and a server, you must either:

- Encrypt the data yourself with an ActionScript algorithm such as the md5 routine discussed above.
- Require that your users have a secure network connection to the server such as a Virtual Private Network (VPN) connection.

## **Security of data and algorithms within a Macromedia Flash MX movie.**

This section discusses the security and integrity of data and algorithms contained within a compiled Macromedia Flash movie file (SWF). It examines the ability to extract data from a SWF file, and discusses best practices for ensuring the security and integrity of data and algorithms in a SWF file.

### **Security concerns of an open format technology**

As previously mentioned, Macromedia Flash movies share many of the same concerns and issues as web pages when it comes to protecting the security of data. Since the SWF file format is an open format, it is possible to extract data and algorithms contained within a Macromedia Flash movie. This is similar to how HTML and JavaScript code can be easily view by users. However, Macromedia Flash movies make viewing the code more difficult. A SWF file is compiled and is not human-readable like HTML or JavaScript.

But security is not obtained through obscurity. A number of third party tools have been written to extract data from compiled SWF files. The most popular and fully featured is called ActionScript Viewer (ASV) and is available at <http://www.buraks.com/>. While these tools may not include support for all versions of SWF files, it should be assumed that they might soon add such support.

In essence, any data, variables, or ActionScript code compiled into a Macromedia Flash movie or projector should not be considered secure.

### **Best practices for securing data within a Macromedia Flash movie**

Just because data and algorithms compiled into a Macromedia Flash movie can be extracted, does not mean that sensitive information cannot be protected. There are a number of techniques that can be utilized to secure sensitive information and still make it available for use in your Macromedia Flash movie.

- 1** Do not hard-code sensitive information, such as usernames, passwords or SQL statements into Macromedia Flash movies.
- 2** If your Macromedia Flash movie needs access to sensitive information, load the information into the movie from the server at runtime. The data will not be part of the compiled SWF file and thus cannot be extracted by users. Be sure to use a secure transfer mechanism, such as SSL, when loading the data.
- 3** Implement sensitive algorithms on the server instead of in ActionScript.
- 4** Only deploy your web applications from a trusted server. Otherwise, the server-side aspect of your application could be compromised.

## **Virus or trojan security concerns**

Macromedia Flash movies running within a web browser are safe. To date there have been no reported viruses or trojans that are distributed using Macromedia Flash movies playing through a web browser. Macromedia Flash Player has stringent security policies and limited access to local system resources that make this highly unlikely.

### **Macromedia Flash playback through projectors**

As with any executable file, there are potential issues with running Macromedia Flash projectors from untrustworthy sources. However, the issues have more to do with running untrustworthy executables than with the Macromedia Flash projector itself. In general, users should never run a Macromedia Flash executable, or any other executable file, unless it was received from a trusted source.

To reemphasize, this section only applies to Macromedia Flash projectors that run locally, or Macromedia Flash movies running in the standalone Macromedia Flash player (which is generally only available to Macromedia Flash developers). It does not apply to Macromedia Flash movies running within a web browser.

There are three main issues to be concerned with when running Flash movies locally:

- Macromedia Flash movies as a carrier of viruses
- Malicious files disguised as Macromedia Flash files
- Macromedia Flash movies that contain malicious code

### **Macromedia Flash projectors as a carrier of viruses**

As with any executable file, it is possible for malicious developers to attach viruses to a Macromedia Flash player projector file. However, if all Macromedia Flash projector files that a user launches come from a trusted source and have been checked with an updated virus scanner, this issue will not pose a threat.

This is not a security issue with Macromedia Flash itself, but with executable files in general.

### **Malicious files disguised as Macromedia Flash files**

A trojan is a file that disguises itself as one thing, but is actually something completely different. The intent is to convince a user that an unsafe file is safe to run.

Due to the popularity of Macromedia Flash content, malicious individuals have tried to disguise their programs as Macromedia Flash movies with the expectation that more people will execute the file.

There are two main ways this is accomplished:

- The distributed file is named in some way that makes users think it is a Macromedia Flash movie, such as “cool\_flash\_movie.exe”.
- The icon for the executable is changed to that of Macromedia Flash Player.

Users should only execute files that come from trusted sources, and only with extreme caution. This is especially important with trojan files, as some virus scanners may not detect them.

This is not a security issue with Macromedia Flash itself, but with executable files in general.

### **Macromedia Flash movies that contain malicious code**

While never reported in the wild, it is theoretically possible for a Macromedia Flash projector or a Macromedia Flash movie played through the Macromedia Flash standalone player on a Windows operating system to perform malicious acts. This risk only occurs when malicious content is played back in a standalone Macromedia Flash Player and does not affect movies playing in a browser.

More information about this issue can be found at the Macromedia Security Zone (<http://www.macromedia.com/security/>).

As mentioned above, users should only run a Macromedia Flash projector or movie locally if the file comes from a trusted source and has been scanned with an updated virus scanner.

Again, this is only a theoretical issue, and has not been experienced in the wild.

### **Macromedia Flash movies as e-mail attachments**

There are two ways that Macromedia Flash movies can be transferred via e-mail. The first way is to simply send the movie as an e-mail attachment. The second way is to embed the movie in an HTML based e-mail with the intent of having the movie run when the e-mail is viewed.

In both cases, the Macromedia Flash movies are viewed as if they are loaded from the local file system, and thus the movies are not as secure as a Macromedia Flash movie was served from a web server and played in a web browser.

Users should only run a Macromedia Flash movie from the local file system or e-mail client if the file comes from a trusted source. In the case of Macromedia Flash movies embedded within e-mail, users should consult the documentation for the e-mail client for instructions on how to disable the automatic playback of ActiveX controls within e-mail.

## Resources

### **Macromedia Security Zone**

<http://www.macromedia.com/v1/developer/SecurityZone/>

Contains security bulletins and technical briefs about security issues.

### **Macromedia Flash Player**

<http://www.macromedia.com/software/flashplayer/>

Contains information and resources for Macromedia Flash Player.

### **Macromedia Flash MX**

<http://www.macromedia.com/software/flash/>

Contains information and resources for the Macromedia Flash MX authoring environment.

### **Macromedia Flash Support Center**

<http://www.macromedia.com/support/flash/>

Contains technical resources and information, including security related information, on Macromedia Flash development.

### **Macromedia Designer & Developer Center**

<http://www.macromedia.com/desdev/>

Contains articles, tutorials, and other information on Macromedia Flash development.

### **Macromedia Flash Player security e-mail address**

[flashplayer\\_security@macromedia.com](mailto:flashplayer_security@macromedia.com).

This email address can be used for questions or comments regarding Macromedia Flash player security.

## Acknowledgments

I would like to acknowledge Robert Hall (<http://www.impossibilities.com/>) for help with the discussion of the md5 hash algorithm.