secureSWF User's Manual

Version 3.1.2841

December, 2008

http://www.kindisoft.com/

Trademarks:

secureSWF, the Kindisoft logo, and the secureSWF logo are trademarks of Kindisoft LLC.
Flash, Flex, and ActionScript are trademarks of Adobe Systems Inc.
Windows is a trademark of Microsoft Inc.
Java is a trademark of Sun Microsystems Inc.
All other trademarks are property of their respective owners.

Warranties and Disclaimers :

THIS PUBLICATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

secureSWF is distributed under licenses restricting its use, copying, distribution, and decompilation.

Table of Contents

Introduction

Kindisoft secureSWF v3.1 has the following suite of features to help you protect your ActionScript in the most convenient way:

- **All Flash & Flex Versions Supported:**
  secureSWF now supports Adobe's Flash v4 through CS4 and Adobe's Flex v1.5 through v3. This includes all ActionScript versions 1, 2, and ActionScript v3.
- **Identifiers Renaming:**
  secureSWF renames almost every identifier (including classes, symbol instances names, and frame labels) in your ActionScript into shorter meaningless names that include unprinted characters.
- **Smart Identifiers Selection:**
  secureSWF automatically determines which identifiers are safe to rename and which are not making code obfuscation easier than ever.
- **Decompilers Stopping Power:**
  secureSWF stops all known decompilers and disassemblers using the following advanced mechanisms:
  - Control flow obfuscation.
  - Dynamic code wrapping.
  - Statement-level randomization.
  - Function calls breaking.
- **Strings Encryption:**
  secureSWF helps you protect your Flash/Flex application from variety of security threats by providing literal strings encryption.
- **Access Limitation:**
  secureSWF limits access to your published SWF files through:
  - Encrypted domain locks.
  - Encrypted loader creation.
- **Processing Presets:**

  You no longer need to dig deep into the settings options to get the result that you want. You can now choose one of the available presets that ranges from aggressive protection to size reduction.

- **SWC Support:**
  secureSWF supports Flash and Flex pre-compiled components files.
- **Build Integration:**
  secureSWF has XML project configuration files, a command-line interface, and Ant task to help you integrate obfuscation into your build process.

Installation

To install secureSWF, simply extract the files into any folder you want. After that, you can start using secureSWF by double-clicking *secureSWF.exe* in Windows, *secureSWF.app* in OS X, and *secureSWF* in Linux (you might need to make the file executable first).

secureSWF does not make any changes to your system (we think of this as a feature). Therefor, you can uninstall it by simply deleting the folder.

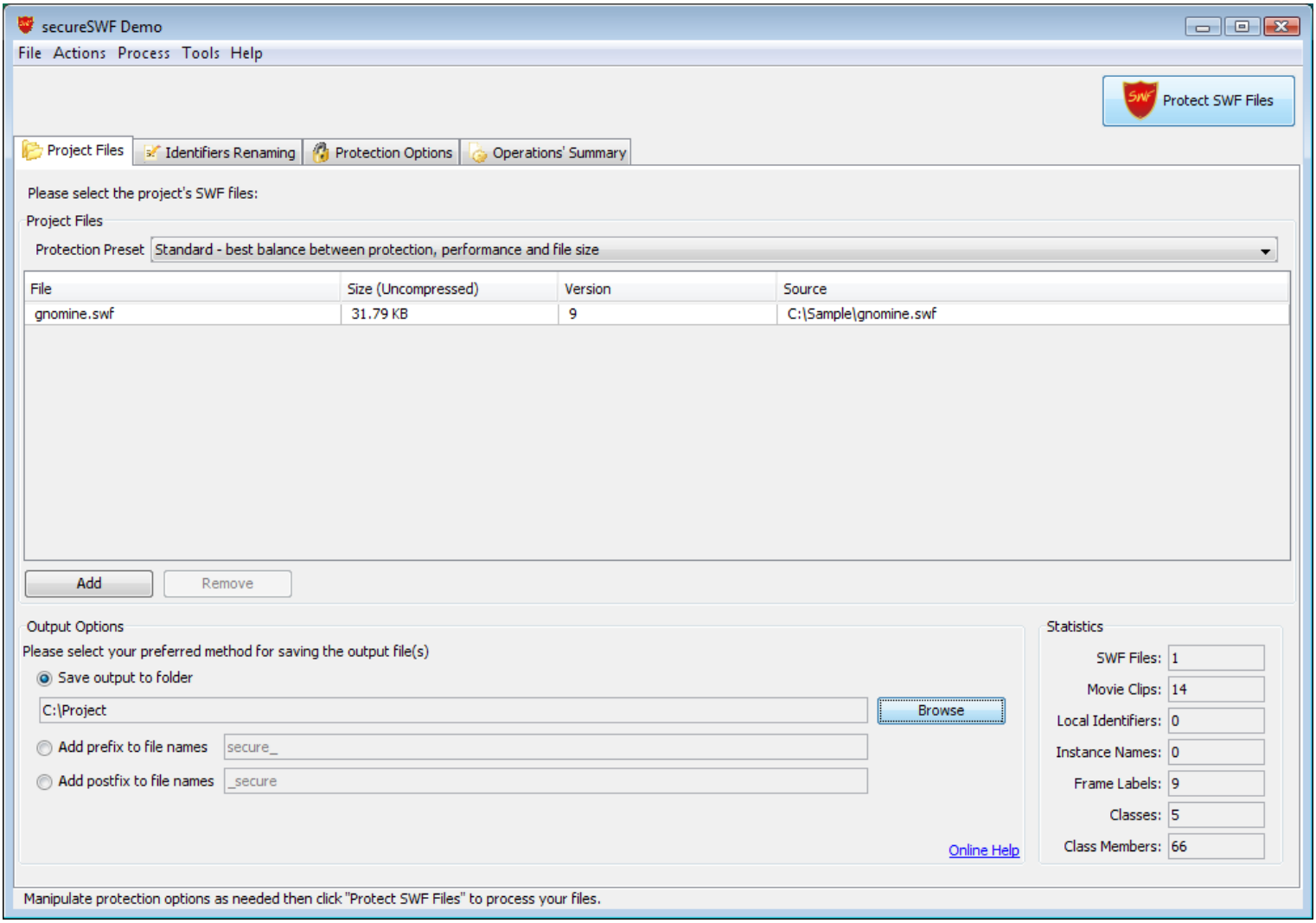
Minimum Requirements

The minimum requirements of secureSWF are:

- 500MHz processor or higher
- 10 Mb of hard disk space *(75 Mb for Windows stand-alone package)*
- 512 Mb of RAM.
- Java VM 1.5 or later *(included in Windows stand-alone package, installed by default in OS X)*
- Minimum screen resolution 1024x768

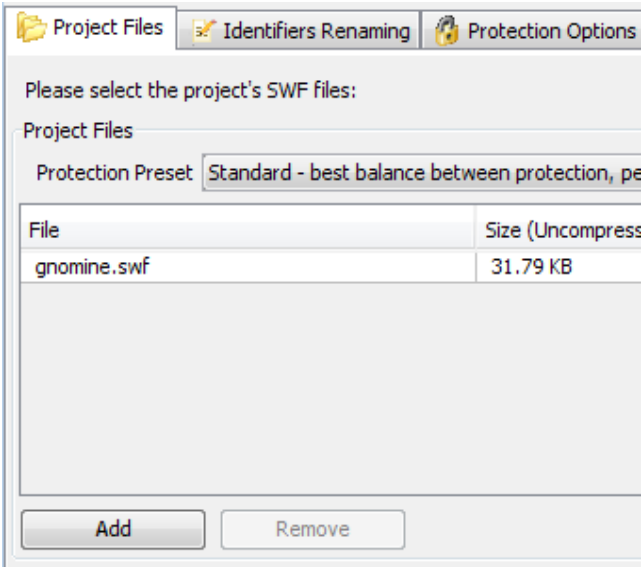
Using secureSWF

The Graphical User Interface of secureSWF consists of four main sections; **Project Files**, **Identifiers Renaming**, **Protection Options**, and **Operations' Summary**.



Project Files

secureSWF is designed to handle either a single SWF file or multiple SWF files that interact together. The *Project Files* section allows you to add the SWF file(s) that you wish to protect. To get started, click on the “Add” button and select the SWF file(s) that you wish to protect. Or, alternatively, drag the SWF file from your file browser into secureSWF.

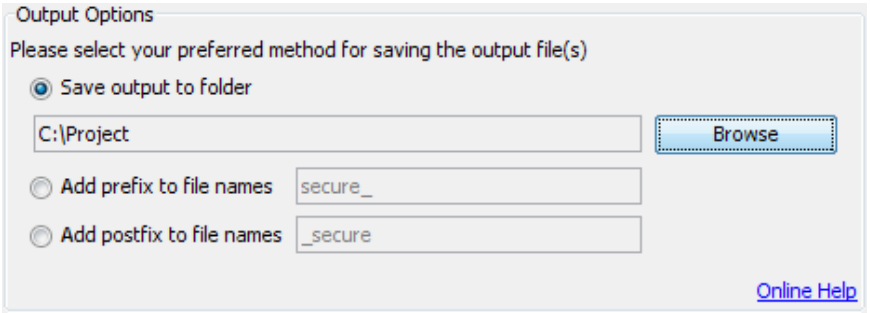


Once you've added the SWF file(s), you will notice that secureSWF starts analyzing the code and automatically determining the best obfuscation settings. You will also notice that there are a number of statistics about the SWF file(s) that you've added in the bottom-right group-box.

Now, you can directly click on the "Protect SWF Files" button on the upper-right corner to protect and save the SWF files. Or, optionally, select one of the *Protection Presets* that suites you.

Saving The Files

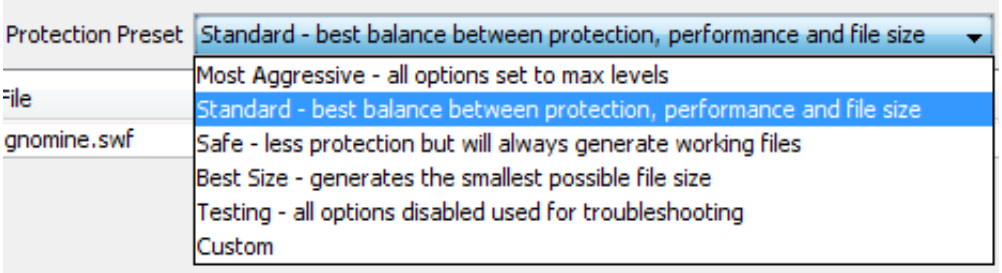
You can select the method of saving the generated, protected, SWF file(s) using the options in the "Output Settings" group-box. This step is optional since secureSWF will ask for the saving options when you click on the "Protect SWF Files" buttons if they were not set. The following is a description of each method:



- Save Output to Folder: The protected SWF file(s) will be saved in the selected folder using the original filename(s). If a file exists in that folder with the same name, it will be overwritten.
- Add prefix to filenames: The protected SWF files will be saved in the same folders they are in but with adding the prefix text that you've entered (default "secure_") at the beginning of the filename.
- Add postfix to filenames: The protected SWF files will be saved in the same folders they are in but with adding the postfix text that you've entered (default "_secure") at the end of the filename.

Protection Presets

You can avoid digging deeper into the settings by selecting one of the *Protection Presets*. You can set secureSWF to generate the most aggressively protected SWF files or to generate a smaller SWF file size. The following is a detailed explanation of each preset:



Most Aggressive:
Sets all options to maximum level. Renames all possible identifiers, enables *Statement-level Randomization*, sets *Control Flow Obfuscation* and *Dynamic Code Wrapping* to maximum, and enables *Function Calls Breaking*. It also enables *Code Optimization* and removes *Metadata.* Using the *Most Aggressive* preset might generate, in some cases, nonfunctional SWF files.

Standard (default):

Best balance between protection, performance, and generated files size. Renames identifiers that were determined to be safe to rename during analysis, sets *Control Flow Obfuscation* to 20%, sets *Dynamic Code Wrapping* to medium and enables *Function Calls Breaking*. It also enables *Code Optimization* and removes *Metadata.* Using the *Standard* preset guarantees to break all decompilers and is recommended to use.

Safe:

If you have faced any issues with the previous presets, try the *Safe* preset. While the generated SWF files will not be as protected as in the previous presets, but the applied protection will be enough to confuse decompilers and prevent them from generating anything useful.

Best Size:

You can use secureSWF to shrink your SWF files size. This is done by disabling all code transformation features, renaming identifiers into shorter names (usually one or two characters), optimizing the byte-code, and removing the metadata.

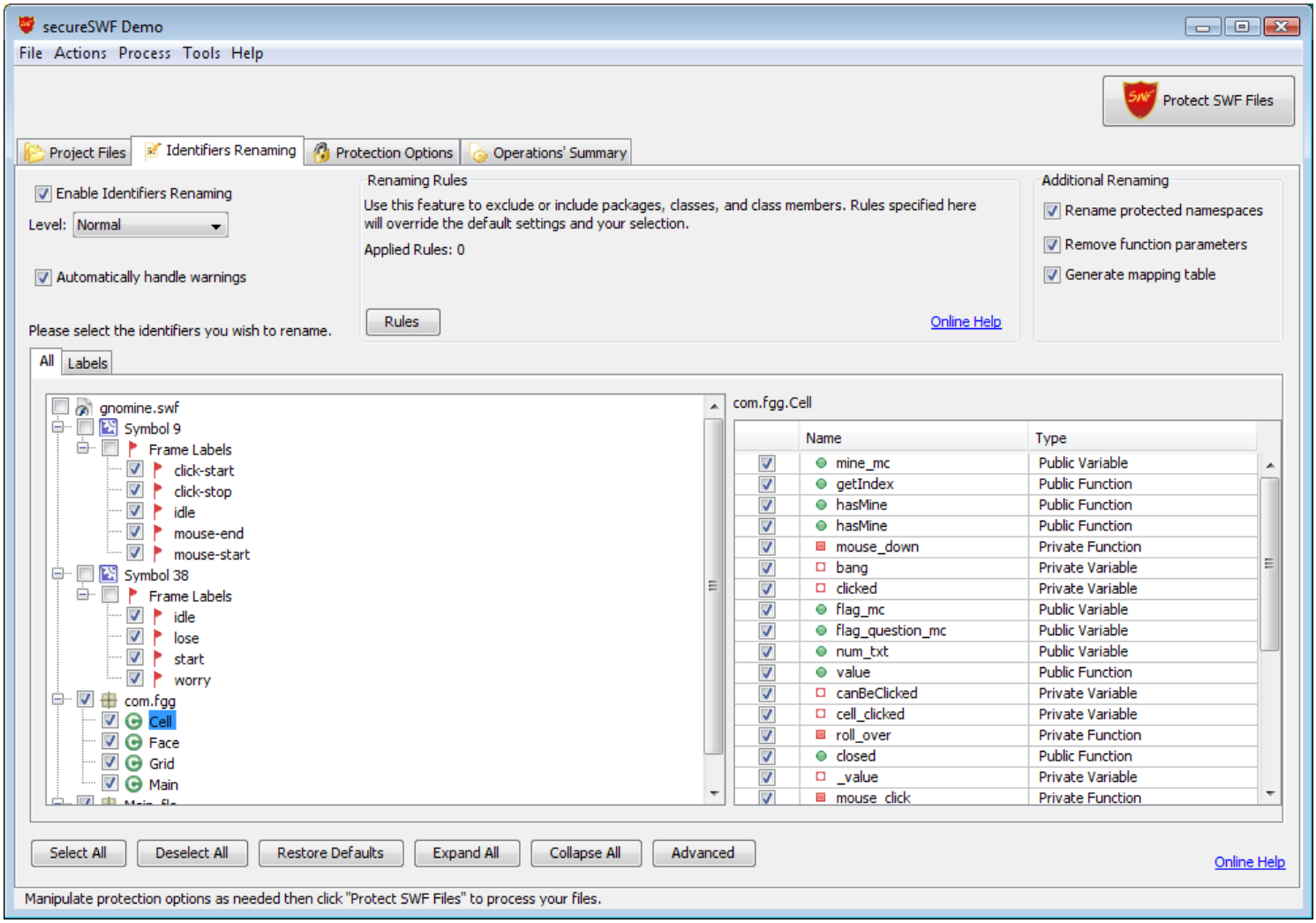
Testing:

Disables all options. Used for troubleshooting.

Identifiers Renaming

One of the most powerful features of secureSWF is its ability to rename just about every identifier in your SWF file from variables and functions names to frame labels and symbol names in addition to AS3 and AS2 classes. Not only that, but secureSWF automatically determines the identifiers that are safe to rename and the identifiers that are not. secureSWF has the ability to rename the following identifiers:

- Local Variables and Functions
- Function Parameters *(removes)*
- Instance Names: Movie Clips, Textfields, and Buttons.
- Frame Labels *(removes)*
- Textfield Variables
- Target Path Strings, such as "_level0/a_mc/b_mc"
- AS2/AS3 Packages, Classes, and Classes' Members.



To enable Identifiers Renaming make sure that the *Rename Identifiers* checkbox is checked. There are 6 tab pages (2 for AS3) in the *Identifiers Renaming* section:

- All: shows all the identifiers in the SWF file(s) in a tree-like hierarchy (except for AS2 classes' members).
- Local Identifiers [*AS2 only*]: shows a table of all the local variables and functions in the SWF file(s) with their locations and types.
- Labels: shows a table of all the frame labels and anchor names in the SWF file(s) with the location of each frame label.
- Instance Names [*AS2 only*]: shows a table of all the symbol instances names in the SWF file(s) with their locations and types.
- Global Identifiers [*AS2 only*]: displays all the global identifiers (variables and functions that are accessed using the keyword "_global") of each SWF file and the AS2 classes names.
- Class Members [*AS2 only*]: displays all the identifiers that are found inside AS2 classes.

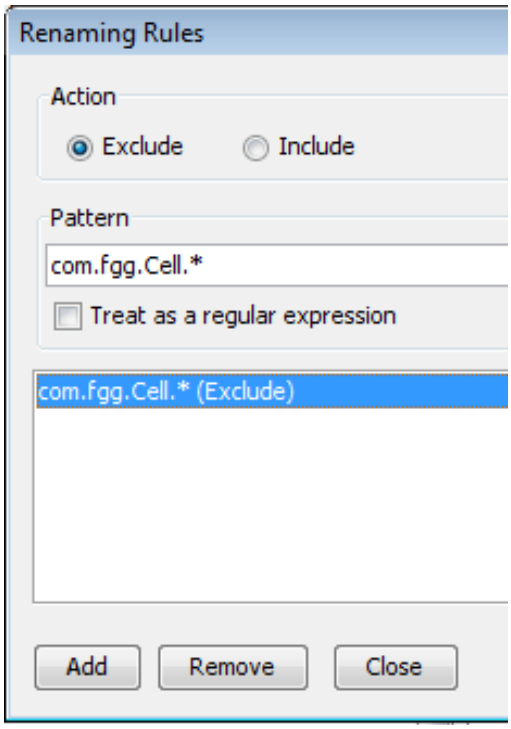
Renaming Level

secureSWF will help you determine which identifiers are safe to rename. You can change the renaming level from *normal* to either rename everything or to only rename identifiers which were never referenced by a literal string in anyway.

Renaming Rules

You can add rules to exclude (or include) a set of identifiers from renaming. Rules will override your other selections as well as secureSWF's automatic selection.

To add a rule, click on the *Rules* button in *Identifiers Renaming* tab. Select whether you want this rule to exclude or include the identifiers it matches. And enter a pattern to match identifiers with. For example, to exclude all the classes names that are in the package com.kindisoft.api from renaming, enter "com.kindisoft.api.*". And to exclude the classes members as well, use "com.kindisoft.api.**". Finally, click on the *Add* button.



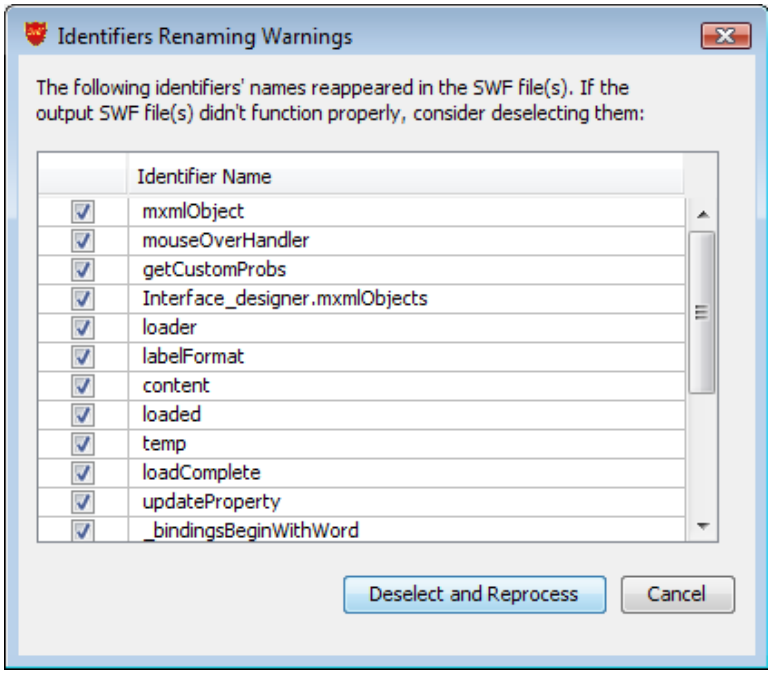
Additional Renaming

Options in *Additional Renaming* renames or removes identifiers that do not appear in the identifiers tree and tables:

- Protected Namespaces:
  These identifiers are added by the compiler to internally reference classes. These identifiers are rather long since the compiler uses the package and class name to form them. secureSWF renames them into shorter meaningless names. Renaming protected namespaces does not affect the application.
- Function Parameters:
  Function parameters in AS2 and AS3 are converted into registers and are no longer used in the code but the function definition still holds their names. Enabling this option will remove them.
- Generate Mapping Table:
  secureSWF is now able to generate a mapping table of the original and new identifiers names. The mapping table is generated using the pattern *protectedFileName_map.xml* and saved in the same path as the protected file. When this option is enabled, a mapping table is generated for every file. You can use the mapping table in the Stack Trace Deobfuscator tool to translate error message from AS3. **Keep the mapping table in a safe place and do not distribute it.**

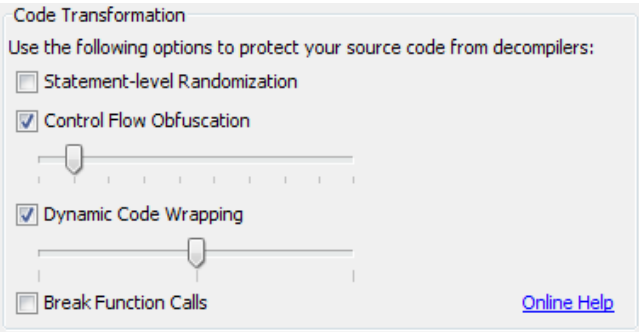
Post-Build Warnings

When you process the SWF files (by clicking on the “Protect SWF Files” button), secureSWF will check if there are still strings in the SWF files that matches any of the identifiers. If a string matches an identifier, a warning dialog will appear. It is recommended to click “Deselect and Reprocess” in this case. secureSWF will automatically handle this for you when “Automatically handle warnings” is checked.



Code Protection

Code Transformation



secureSWF offers four different methods to protect your source code from reverse-engineering and decompiling that can be applied separately or together. Each protection method is compatible with all the Flash players from v5 to v9 and has been tested separately to break all the known decompilers and disassemblers. The following is a description of each method, along with its advantages and disadvantages:

Statement-level Randomization

Randomly restructures the bytecode instructions that makes up an ActionScript statement making it almost impossible decompile the source code. The performance and file size effects of this method depend on the source code. A block of code that has no branches (for example, if-statements, for-loops, or function definitions) will become smaller in size and will execute faster. On the other hand, extra bytecode instructions will be added to handle the branches which has negligible impact on the performance and increase in the SWF file size.

Control Flow Obfuscation

Adds extra branches to the bytecode that distracts decompilers and disassemblers. This method increases the SWF file size depending on the selected intensity. You can set the intensity between 10% and 100%; a 10% intensity generates a smaller SWF file while a 100% intensity generates a more protected SWF file. Our testing results have shown that even a 10% intensity will break all known decompilers most of the time.

Dynamic Code Wrapping

Wraps the ActionScript blocks with branches and junk bytes that effectively foil decompilers and disassemblers. The size and performance impact of this method is negligible. You can set Code Wrapping to three levels; minimum, medium, and maximum. The degree of code wrapping will determine the size of the added code. Use maximum code wrapping for maximum protection, and minimum to get a smaller file.

Break Function Calls

Makes it much harder for a decompiler to find out what parameters are being passed to a function call.

Optimization

secureSWF implements many code optimization algorithms that will enhance the performance and reduce the SWF file size. Enabling *Remove metadata* will remove all metadata in bytecode.

Literal String Encryption

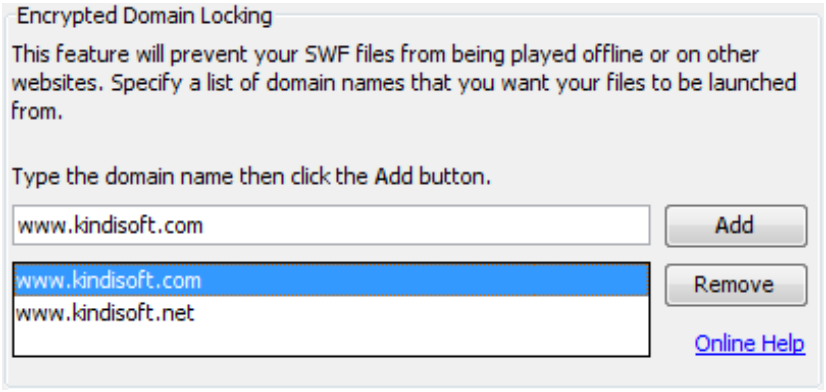
While obfuscation protects the code logic, valuable data such as access passwords and sensitive URLs remain in the SWF file as clear text. Malicious users can simply open the SWF file using a text editor (decompress the SWF file if compressed) and view the literal strings that exist in your code.

secureSWF allows you to select the literal strings in your SWF file, encrypt them in the SWF file using a very secure symmetric encryption algorithm, and decrypt them only when needed at runtime. Please note that this entails an added overhead each time the string is accessed.

To use this features, look up the strings in the table. It shows all the literal strings, the number of occurrences, and their location. Simply, check the strings that you wish to encrypt.

Encrypted Domain Locking

Inserting an encrypted domain lock to your SWF file(s) will greatly decrease the chances of having the movie being played offline. The movie will simply refuse to play if it was copied to a different location than that it has been locked to. For example, if you locked your movie files to "http://www.mywebsite.com/myflashmovie/", then the movie will only play if it is being launched from that location. If someone managed to copy the SWF file(s) somewhere else (i.e. PC, another website ...etc), then the movie will not play at all.



What makes this feature powerful enough to rely on is that the locations (URLs) that you locked the movie to are encrypted inside the SWF files and will only be decrypted at runtime. This makes finding out what is happening, and changing it, a very time and effort consuming task to a level that recreating the movie from scratch is usually easier.

To use Encrypted Domain Locking add the domains or URLs that you want to lock the files to. Make sure to add all the URLs that you want the user to be able to access the SWF file through. To make users able to run the SWF files on their local machines, add "file" for ActionScript 2 and "localhost" for ActionScript 3.

Encrypted Loader Creator

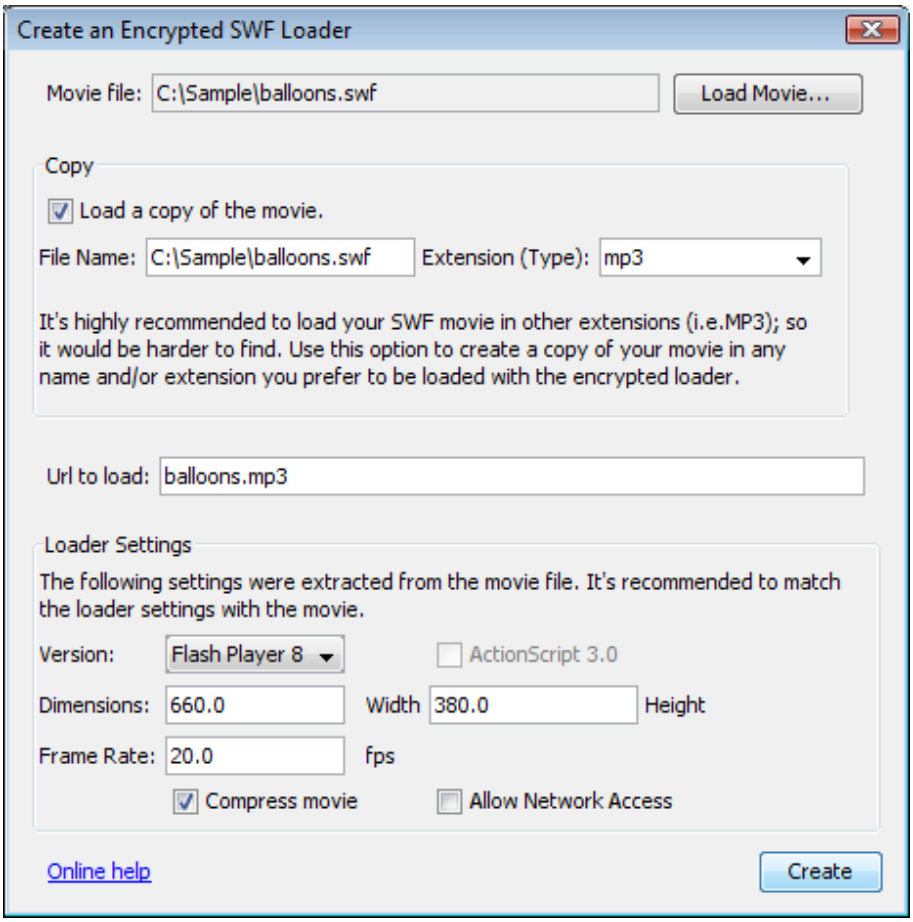
This tool will help you to highly decrease the chances of having your SWF files downloaded or used offline. It simply creates a new SWF file which dynamically loads your original SWF file into it.

To increase this feature's power and reliability, the name and location of the original SWF file are encrypted. In addition to that, you can change the extension of the original SWF file into any random string (i.e. .mp3 or .js). This will make finding the original SWF file in the browser cache a much harder task. The main important gain from using this tool is that users who will try to use SWF rippers (grabbers) will not be able to download the original SWF file at all.

Using the Encrypted Loader Creator Tool

First of all, it is important that the settings of the created loader match the settings of the original SWF file you wish to load. You can start by clicking on “Load Movie...” and selecting the original SWF file so that the “Encrypted Loader Creator” tool would be able to extract its settings.

After that, you can optionally create a copy of the original file using another file extension (i.e. .mp3 instead of .swf). To do so, check the "Load a copy of the movie" check box, then fill in the file name and choose from the drop-down-list an extension, or type in any extension you prefer. Changing the file extension is highly recommended.



The next step is to make sure that the URL field is exactly what the loader is going to load. It could be just the file name, if the original file and the loader are going to be in the same folder, or the full URL (file path) of the original file that will be loaded.

Finally, you can optionally make adjustments to the loader settings. The following table explains each option:

Version	The SWF file format version.
Dimensions	The width and height in pixels of the movie.
Frame rate	The frame per second rate, this should always match the original file.
Compress movie	For version 6 and later, SWF files could be compressed.
ActionScript 3	Use ActionScript 3 to create the loader. Should match the loaded SWF file.
Allow Network Access	Should match the loaded SWF file.

Stack Trace Deobfuscator

This tool will help you translate error message from AS3 based obfuscated SWF files when running them in the stand-alone Flash Player. You will find this tool very handy in troubleshooting.

There is an XML mapping file generated for every protected file when the option *Generate mapping table* is enabled. The mapping file name follows the pattern *protectedFileName_map.xml* and is saved in the same path as the protected file. To use this tool, click on browse and select the mapping table for the file that generated the error. And then paste the error message from the Flash Player in the text area and click Trace. This will deobfuscate the error message showing the identifiers in their original names.

Troubleshooting

After protecting your SWF files, we highly recommend to test them. If the protected files are not functioning properly, then one or more of secureSWF protection options requires further tuning. Please follow these simple steps to find out what went wrong.

1. Remove all files and start over.

2. After adding your files again, set the Protection Preset to Testing. This will turn off all the processing options. Process the files and test them. If the generated files are still not working properly, then please contact us. You are most likely facing a major bug in secureSWF that we really want to know about and try to fix as soon as possible.

3. Enable Identifiers Renaming and process the files again. If they are not working, please move to the Identifiers Renaming Troubleshooting section. Once you get it fixed, move to the next step.

4. Enable Control Flow Obfuscation and set it to 100% intensity. Process the files. If everything works well, then you have the most important features working for you. Please move to step 5. If the files are not functioning properly after enabling this option, then here is what you can do:

   - ActionScript 1 and 2 based projects, leave this option disabled and try Statement-level Randomization and Dynamic Code Wrapping. They are just as effective as Control-flow Obfuscation.

   - ActionScript 3, the Flash Player (debugging mode) should throw a verification error such as “Stack unbalanced x!=y” or “TypeA and TypeB cannot be reconciled” with a call stack trace. Please send us that error with the code of the function that the error occurred in. If you can't share the code of that function, then a description of the control flow will do. We will probably fix the issue right away and send you an update. If you can't wait for the update, then just use Statement-level Randomization and Dynamic Code Wrapping temporarily instead.

5. Enable other options one by one until you get the nonfunctional file and then please let us know which option broke your application.

Identifiers Renaming Troubleshooting

While secureSWF tries to automatically deselect identifiers that cause problems when renamed, some identifiers will still slip away and require you to manually deselect them.

First of all, please set identifiers renaming Level to Safe and make sure that Automatically handle warnings and Generate mapping table are checked. Process the files, if the protect files are still not working properly then please follow the following steps:

- ActionScript 1 and 2:

  You have to try out different settings multiple times to find out which identifier is causing the problem. You only have to do this once for every project since you can save the settings and load them later when you need them again.

  1. If you have multiple SWF/SWC files, remove them and process each one individually.
  2. In the Identifiers Renaming tab, right-click the SWF file in the All identifiers tree and click on Deselect All. This will deselect all the identifiers in the file.
  3. Start selecting identifiers group by group; right under the SWF file, you will usually find the “Time Line” node (sometimes a Symbol, Package, or Class). Right-click the first node under the SWF file and click on Restore Defaults (not Select All). Process the file and test it. If it works, move to the next node and do the same to it. If it doesn't work, deselect the node, skip it, and move to the next node.
  4. Now you should have a working protected file with part of its identifiers renamed. You can now rename more identifiers if you want to and test the protected file after every change.

- ActionScript 3:

  Luckily, the Flash Player (debugging mode) will throw verification errors in ActionScript 3 if there is something wrong with identifiers renaming. Some error messages will directly show which identifier should be deselected, but others require further analysis.

  If there is no error message, then most likely the problem is caused by a Frame Label. The easiest way to deselect all the Frame Labels is by clicking on the Advanced button right under the identifier tree, check Frame Label, and enter “*” in the Pattern textfield then click Apply. Process the file and test it. If that doesn't fix the problem, the deselect all the classes names in the default package. This should fix the problem.

  Error messages such as “Class $4 could not be found.” usually means that the class which its new name is now “$4” wasn't renamed correctly at its definition. To find out the original name of this class, use the Stack Trace Deobfuscator. Then deselect the identifier. Sometimes the original identifier name will appear in the error message such as “Variable MyVar is not defined.” In that case, simply deselect “MyVar”.

Other error messages such as “Cannot access a property or method of a null object reference.” or “A term is undefined and has no properties.” require further analysis. To find out what went wrong in this case, first deobfuscate the call stack trace by using the Stack Trace Deobfuscator. Then follow these steps:

1. Look up the code of the call stack trace top function and create a list of identifiers this function accesses. If this function is a Flash/Flex API function, move to step 2. For each identifier, deselect it and test the file until the error goes away. When the error message goes away, then the identifier that you've just deselected is the one that is causing the problem.
2. For every function call in call stack trace, deselect the identifiers that are being passed as parameters. Deselect them one by one, and test the file after every change to find out which identifier is causing the error. This should fix the problem.

Command-line Interface

secureSWF has a command-line interface that could be used for batch execution and build process integration. Using the CLI is different on each platform:

Windows

You can use the command-line interface in two forms:

```
ssCLI.exe Project_file Output_folder [override_options]
```

Or

```
ssCLI.exe SWF_file Output_folder [options]
```

The options will override the project settings and secureSWF's defaults as the following:

-slr:[on/off]	Overrides *Statement-level Randomization*
-controlFlow:[0-100]	Overrides *Control Flow Obfuscation*
-wrap:[0-3]	Overrides *Dynamical Code Wrapping*
-optimize:[on/off]	Overrides *Optimization*
-rename:[on/off]	Overrides *Identifiers Renaming*
-safe	Forces safe renaming
-ignoreWarnings	Ignores post-build warnings
-domains:domain1,domain2	A comma separated list of domains to lock the files to.

For example:

```
ssCLI.exe myProject.sspj c:\output -slr:off -controlFlow:50 -wrap:3
```

Will load the SWF files and settings for the project file, turn off *Statement-level Randomization,* set *Control Flow Obfuscation* to 50%, set *Dynamic Code Wrapping* to maximum, and save the protected files in the folder c:\output. You can use the first letter only of each option (except for -safe).

OS X

You can use the command-line interface in two forms:

```
ssCLI Project_file Output_folder [override_options]
```

Or

```
ssCLI SWF_file Output_folder [options]
```

The options will override the project settings and secureSWF's defaults as the following:

-slr:[on/off]	Overrides *Statement-level Randomization*
-controlFlow:[0-100]	Overrides *Control Flow Obfuscation*
-wrap:[0-3]	Overrides *Dynamical Code Wrapping*
-optimize:[on/off]	Overrides *Optimization*
-rename:[on/off]	Overrides *Identifiers Renaming*
-safe	Forces safe renaming
-ignoreWarnings	Ignores post-build warnings
-domains:domain1,domain2	A comma separated list of domains to lock the files to.

For example:

```
ssCLI myProject.sspj output -slr:off -controlFlow:50 -wrap:3
```

Will load the SWF files and settings for the project file, turn off *Statement-level Randomization,* set *Control Flow Obfuscation* to 50%, set *Dynamic Code Wrapping* to maximum, and save the protected files in the folder *output*. You can use the first letter only of each option (except for -safe).

Linux

You can use the command-line interface in two forms:

```
secureSWF Project_file Output_folder [override_options]
```

Or

```
secureSWF SWF_file Output_folder [options]
```

The options will override the project settings and secureSWF's defaults as the following:

-slr:[on/off]	Overrides *Statement-level Randomization*
-controlFlow:[0-100]	Overrides *Control Flow Obfuscation*
-wrap:[0-3]	Overrides *Dynamical Code Wrapping*
-optimize:[on/off]	Overrides *Optimization*
-rename:[on/off]	Overrides *Identifiers Renaming*
-safe	Forces safe renaming
-ignoreWarnings	Ignores post-build warnings
-domains:domain1,domain2	A comma separated list of domains to lock the files to.

For example:

```
secureSWF myProject.sspj output -slr:off -controlFlow:50 -wrap:3
```

Will load the SWF files and settings for the project file, turn off *Statement-level Randomization,* set *Control Flow Obfuscation* to 50%, set *Dynamic Code Wrapping* to maximum, and save the protected files in the folder *output*. You can use the first letter only of each option (except for -safe).