secureSWF User's Manual

Version 3.6.5742

July, 2011

http://www.kindi.com/

Trademarks:

secureSWF, the Kindi logo, and the secureSWF logo are trademarks of Kindi LLC.
Flash, Flex, and ActionScript are trademarks of Adobe Systems 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

Kindi secureSWF 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 CS5.5 and Adobe's Flex v1.5 through v4.5. This includes all ActionScript versions 1, 2, and ActionScript 3.
- **Identifiers Renaming:**
  secureSWF renames almost every identifier (including classes, symbol instance names, and frame labels) in your SWF file(s) into shorter meaningless names that include unprinted characters.
- **Smart Identifier 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.
- **String 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 loaders.
- **Processing Presets:**

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

- **SWC & AIR Support:**
  secureSWF supports Flash and Flex pre-compiled components files as well as packaged .air 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.

Editions

secureSWF comes in three different editions to suit different needs. Whether you are creating a simple Flash game or a sophisticated Flex application, there is an edition that is best suitable for you.

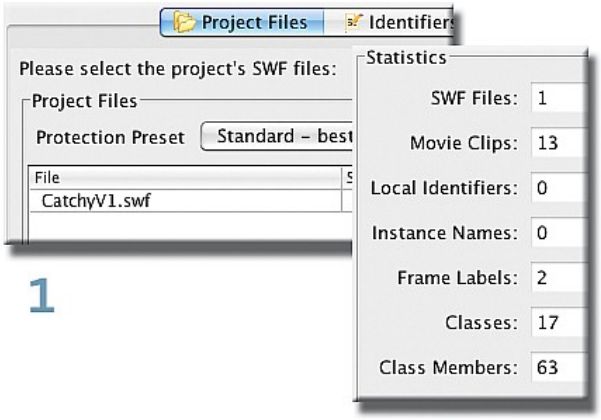
	 Personal Lite	 Standard	 Professional
AS3 and Flex Support	✔	✔	✔
SWC Files Support	✔	✔	✔
AIR Files Support	✔	✔	✔
Project Configuration Files	✔	✔	✔
Control Flow Obfuscation	✔	✔	✔
Statement Level Randomization	✔	✔	✔
Dynamic Code Wrapping	✔	✔	✔
Super Compression	✔	✔	✔
Rules Configuration	✔	✔	✔
Identifiers Renaming		✔	✔
Aggressive Renaming		✔	✔
Smart Renaming		✔	✔
Symbol Instances Renaming		✔	✔
Removes Frame Labels		✔	✔
Code Optimization			✔
Literal Strings Encryption			✔
Encrypted Domain Lock			✔
Encrypted Loader Creator			✔
Command-Line Interface		✔	✔
Ant Script Integration		✔	✔
In-code Configuration		✔	✔

Installation

To install secureSWF, simply extract the files into any folder you want. Just make sure secureSWF will have write access to the folder. 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 also need to make the file executable first in Linux). 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.

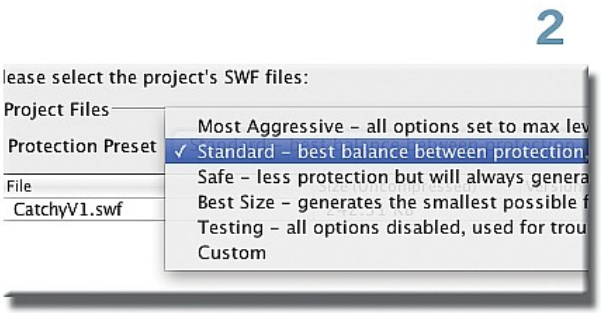
Getting Started

While secureSWF provides a wide array of protection options for Flash applications, getting started with a level of security that could well be all you need is a simple 3-step operation.



Secure your SWFs – Easy as 1-2-3!

1. Add SWF file(s) into your project (Project Files tab)
You can add multiple files which will all be processed with the same settings. You should add all files that interact with each other so secureSWF can detect the references and make sure the obfuscated files will work the same way they did before obfuscation. The Statistics panel shows you info about the file or files in the project.



2. Look at Protection Presets choices
These are pre-defined protection levels that may be enough for many needs. They offer different protection-to-file size-to-performance ratios. You can optionally set up custom levels exactly suited to your needs if presets are not quite what you want.

3. Specify prefix or postfix for output file and Protect
Add a prefix (e.g. secure_) and/or postfix (suffix) for the protected output version of your SWF file so that the original is not overwritten with the protected version. Then click the Protect button.

THAT'S ALL IT TAKES TO GET STARTED PROTECTING YOUR SWF FILES!

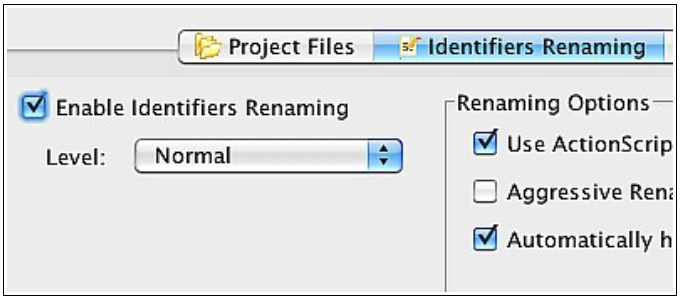
Of course, you will want to explore the many protection options available in secureSWF to get the maximum benefits for your situation. The rest of this guide will introduce you to the major areas of the product to explore next.

At the conclusion of this guide you should be well on your way to understanding how to use secureSWF to achieve the security and performance levels that are right for your Flash applications.

Getting Started with secureSWF Protection Settings

While you can get a pretty good level of protection for your SWF files using presets, these are only the tip of the iceberg when it comes to your options for securing your Flash apps. It will of course take some time to master everything secureSWF has to offer, but it shouldn't take you too long to get acquainted with the major product features and settings and arrive to the level of protection that's good for your needs. Let's look briefly at the main things you should look at when devising a security scheme for your applications.

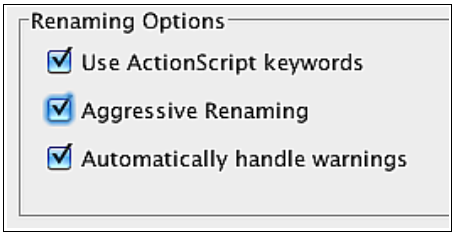
Identifiers Renaming Tab



Renaming of identifiers in your application source code is one of the main ways secureSWF protects your apps from reverse engineering. The Identifiers Renaming tab provides a number of options that control how secureSWF handles renaming as it obfuscates your source code.

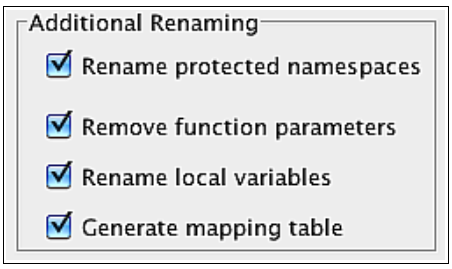
Main Options:

- **Enable Identifiers Renaming -** This is a toggle that controls whether or not secureSWF will rename identifiers as part of the obfuscation process. Check it to enable renaming of identifiers.
- **Level -** This setting controls how renaming happens.
- **Aggressive Renaming -** If you check it, identifiers are renamed with non-printing characters, illegal names, etc. This generally results in smaller names (reducing the byte count in the final obfuscated file). Decompilers can't print or display identifiers.



- **Use ActionScript Keywords -** If you check it, identifiers are renamed using ActionScript keywords such as using the words class, package, if, for … etc as identifiers names.
- **Automatically Handle Warnings** − secureSWF will handle warnings resulting from renaming of identifiers without involving you. Turn off this option if you want to see warnings.

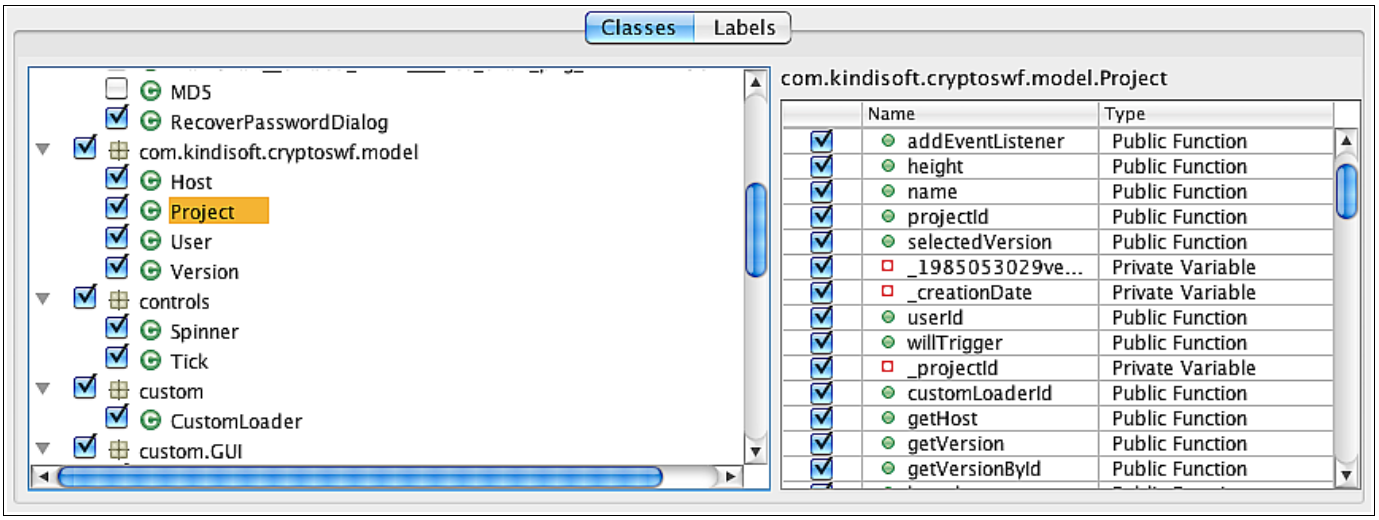
Additional Renaming Options



Fairly self explanatory. Mapping table is just a table of original identifiers mapped to renamed identifiers. It can help translate error message that might arise in your Flash application later. See Help if you need to delve deeper into these options.

Exploring Packages and Classes

The Identifiers Renaming tab displays a tree of the packages and classes in your application, and it shows which classes will have identifiers renamed with the standard renaming defaults. When you select any class on the left, its members are displayed on the right.



Basically what you want to do here is look through the classes, see which ones will have identifiers renamed by the default renaming (these are checked in the listing), and decide whether you want to have more classes processed, and set options for renaming. You can as get granular as you want by selecting individual classes for renaming and making settings for each selected class.

Protection Options Tab

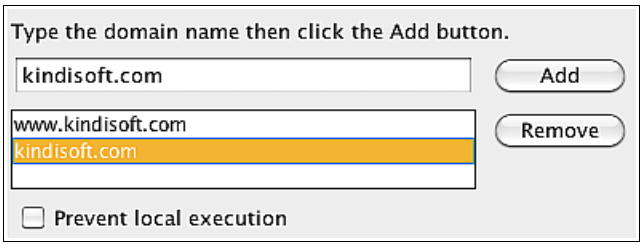
This tab provides the second major security features area of secureSWF. It provides options for code transformations, file optimization, domain locking, and encryption of embedded string literals.

Code Transformation

Code transformation foils decompilers by changing compiled code so that it can no longer be reverse engineered into source. You can set a number of options including Advanced options for a greater or lesser degree of transformation. Detailed information about each option can be found here.

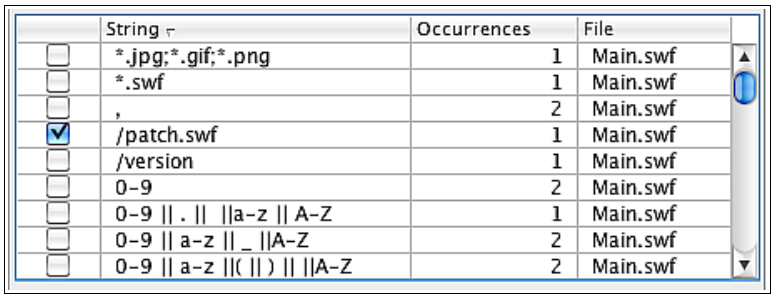
Encrypted Domain Locking

This feature enables you to specify the domains on which the protected SWF file(s) can be hosted. Files will not work if hosted on any other domain. Also can be used to prevent local execution offline.



Optimization

Optimizes the code itself and performs various operations to reduce file size and optimize performance. Detailed information about each option can be found here.



Literal Strings Encryption

Encrypts the literal data that's inside the code − hard-coded passphrases or URLs for example. Replaces literal strings with a special function call that returns the decrypted string, which is stored in an encrypted form.

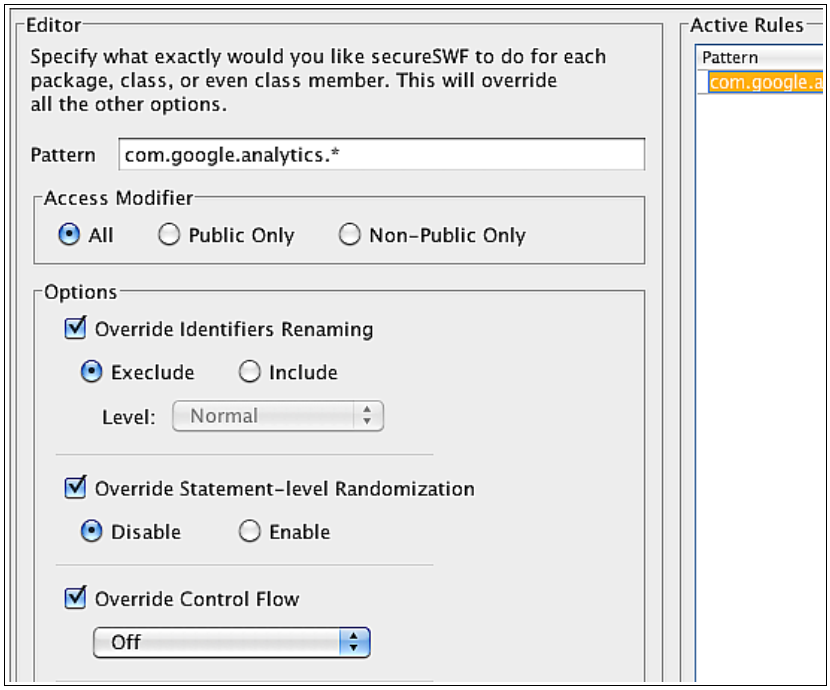
Configurations Rules Tab

While previous tabs enable you to specify how your code will be obfuscated down to a single function or identifier, we know that pinpointing every single one could be a very time consuming process. Because of this, we've added the Rules tab.

Now, you can just tell secureSWF what to do with all the code in an entire package, or a single class. Just write in the pattern and check which options you want to override.

You can use this, for example, to avoid obfuscating code merged from SWC libraries provided by third-parties.

Rules will override all other settings.



Pulling it all together

Hopefully by now you have enough direction to begin working with secureSWF. Be sure to refer to the Help topics as you go along. The main thing you are looking to achieve is an optimal balance of security, performance, and file size.

To keep abreast of the latest news, pop in to the Kindi blog at http://blog.kindi.com/ or tune in to Kindi on Twitter at http://twitter.com/secureswf/ .

If you need technical support, don't hesitate to start a new ticket here http://suppot.kindi.com/. We'll be glad to help.

Troubleshooting Guide

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 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 issue in secureSWF that we really want to know about and try to fix as soon as possible.

3. 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 (debugger version, available here) should throw a verification error such as “Stack unbalanced x!=y” or “TypeA and TypeB cannot be reconciled” with a call stack trace. Please move to the Code Transformation Troubleshooting section.

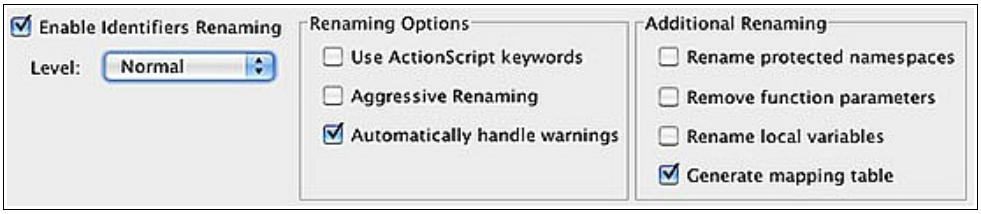
4. Enable Identifiers Renaming and process the files again. If they are not working, please move to the Identifiers Renaming Troubleshooting section.

5. You now have the most important features working for you. You can carry on by enabling other options that you need one by one until you get the nonfunctional file.

Identifiers Renaming Troubleshooting

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

First of all, please set identifiers renaming Level to Normal and make sure that Automatically handle warnings and Generate mapping table are checked and all other checkboxes are unchecked as appears below.



Process the files, if the protected 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. Set the renaming Level to Safe Renaming and see if that helps. If it doesn't, continue following the steps.
  2. In the Identifiers Renaming tab, right-click the SWF file in the All identifiers tree and click the Deselect All menu item. 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 most of its identifiers renamed. You can rename more identifiers if you want to and test the protected file after every change.

- ActionScript 3:

  You can set the renaming Level to Safe Renaming if you want to save some time and don't mind that some identifiers will not get renamed. We do not recommend that since most identifiers excluded by Safe Renaming do not cause problems when renamed.

  The Flash player (debugging version) will throw verification errors for ActionScript 3 if there is something wrong with new identifiers names. Some error messages will directly show which identifier should be deselected, but others require further analysis.

If there are no error messages, 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 text box then click Apply. Process the file and test it. If that doesn't fix the problem, then double check that you are running the SWF files in the debugger version of the Flash player. You can also run the SWF files in Flash Professional to get the error messages.

Error messages such as "Class $4 could not be found." usually means that the class which its new name is now "$4" was not renamed correctly. To find out the original name of this class, use the Stack Trace Deobfuscator . Then deselect that class name. 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 and identifier type referenced in the function, deselect it and test the file until the error goes away. When the error message goes away, then the identifier you've just deselected is the one causing the problem.
2. For every function call in the 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.

Code Transformation Troubleshooting

Note: This section is for ActionScript 3 only.

With secureSWF v3.3 and later, you can disable code transformation options for one or more method (getters, setters, and class constructors as well) while leaving it enabled for the rest of the code.

If there is a problem with Control Flow Obfuscation, Statement-level Randomization, or Dynamic Code Wrapping, then the Flash Player (debugger version) will throw one of the following common errors:

1017	Scope stack overflow occurred.
1018	Scope stack underflow occurred.
1020	Code cannot fall off the end of a method.
1030	Stack depth is unbalanced. _ != _.
1068	_ and _ cannot be reconciled.

After the error message, you will find a call stack trace. If the identifiers in the call stack trace are renamed, you can either use the Stack Trace Deobfuscator tool in secureSWF, or temporally disable Identifiers Renaming. If the error message is different that the above but you still have a call stack trace and it happens only when you enable Control Flow Obfuscation, then you can follow this guide to solve the issue too.

Note: If Control Flow Obfuscation is set under 100% intensity, the error might not occur every time. This is because Control Flow Obfuscation will choose random places to alter the control flow if set under 100% intensity. For testing, we recommend either setting Control Flow Obfuscation to 100% or disable Randomize Results.

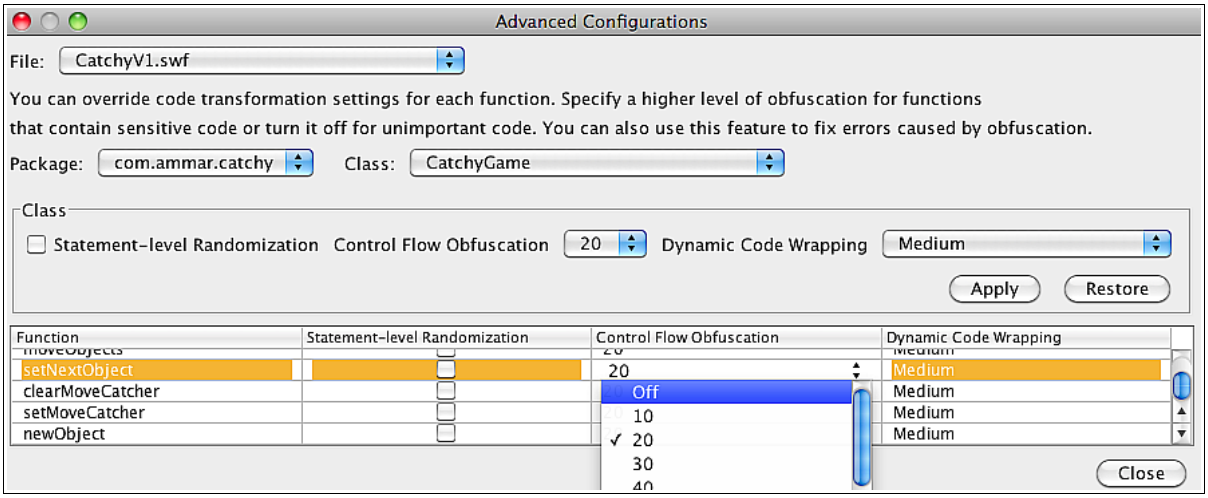
If you are getting one of the errors in the pervious table after obfuscating your files with secureSWF, look up the method name, class, and package at the top of the call stack trace. This is usually the function that is causing the error when obfuscated. You need to disable code transformation for this method only.

There are two ways to specify the Code Transformation options for a single method. You can either use the Advanced Configurations for Code Transformation, or the Configuration Rules tab.

1. Using Advanced Configurations

Clicking the Advanced Configurations button in the Code Transformation group-box will open a dialog where you can disable Control Flow Obfuscation, Statement-level Randomization, and Dynamic Code Wrapping for the method that is causing an error when obfuscated. Choose the file that the error

occurred at (if you have multiple files), then choose the package, and the class that contain the method. The table at the bottom of the dialog contains all the methods, getters, setters, and the class constructor of the selected class. You should be able to find the function name that appeared in the call stack trace in that table. Click on the cells next to the method name and disable all the option.



Close the dialog and click on Process again. This should fix the error.

2. Using the Configuration Rules tab

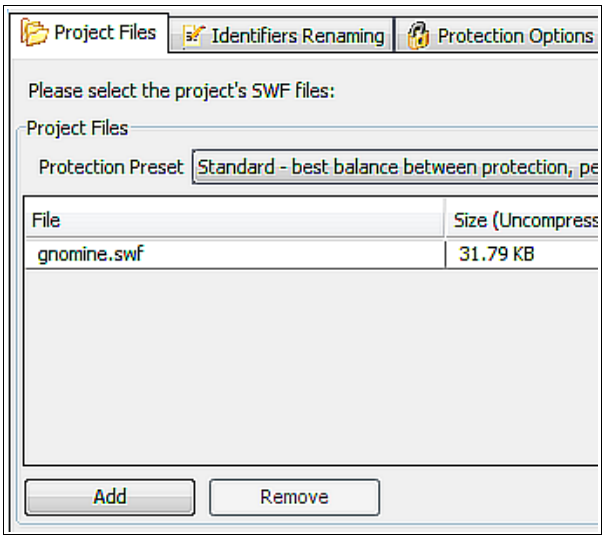
Using the Configuration Rules is faster and more powerful. All what you have to do is write the method name and check Override Control Flow Obfuscation. You can also override other settings too. To turn off Control Flow Obfuscation for the method shown in the screen shot above using the Configuration Rules tab, enter “com.ammar.catchy.CatchyGame.setNextObject” in the Pattern field, check the Override Control Flow Obfuscation check-box, and click the Add button. More information can be found here.

Detailed Documentation of secureSWF Options

The following is a detailed documentation of secureSWF features and options to help you better understand how to use each section and how it will affect your code.

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, SWC, or AIR file(s) you wish to protect. To get started, click on the Add button and select the SWF file(s) you wish to protect. Or, alternatively, drag the SWF file into secureSWF.

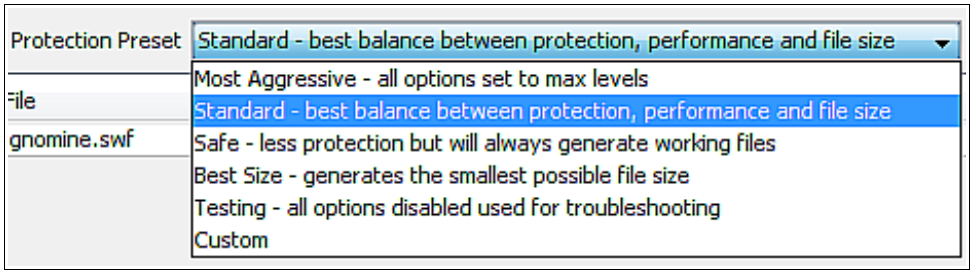


Once you've added the file(s), you will notice that secureSWF starts analyzing the code and automatically determining the best obfuscation settings. You will also notice that there are a number of statistics about the SWF file(s) you've added in the bottom-right group-box.

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

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.*

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:

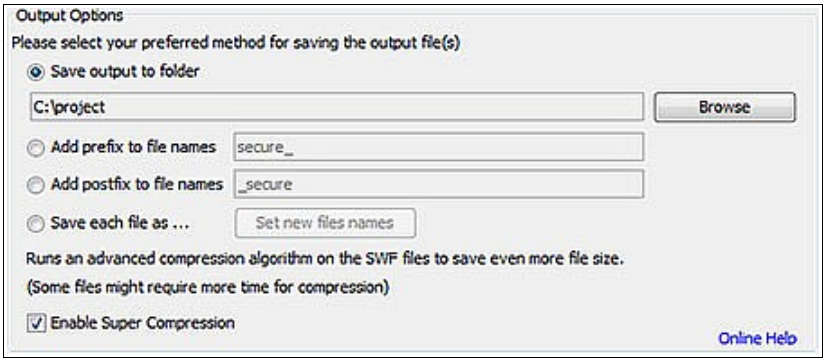
You can use secureSWF to shrink your SWF files size. This is done by disabling all code transformation features, renaming identifiers into shorter names (usually one or two characters), optimizing the byte-code, and removing the metadata.

Testing:

Disables all options. Used for troubleshooting.

Saving The Files

Before proceeding, you need to select a method of saving the new protected SWF file(s) using the options in the *Output Options* group-box.



The following is a description of each option:

- Save Output to Folder: The protected SWF file(s) will be saved to 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.

- Save each file as: Gives you the ability to set a new file name for each file.

- Super Compression: This feature will reduce the SWF file size by an average of 10% by running an advanced compression algorithm that requires more time for compression but will not affect the decompression speed or the application performance at all. This compression algorithm favors file size over compression speed by performing an exhaustive search of all patterns and is equipped with an advanced block splitting technique.

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 also 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).
- Labels: shows a table of all the frame labels and anchor names in the SWF file(s) with the location of each frame label.
- Local Identifiers [*AS2 only*]: shows a table of all the local variables and functions in the SWF file(s) with their locations and types.
- Instance Names [*AS2 only*]: shows a table of all the symbol instance 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 Options

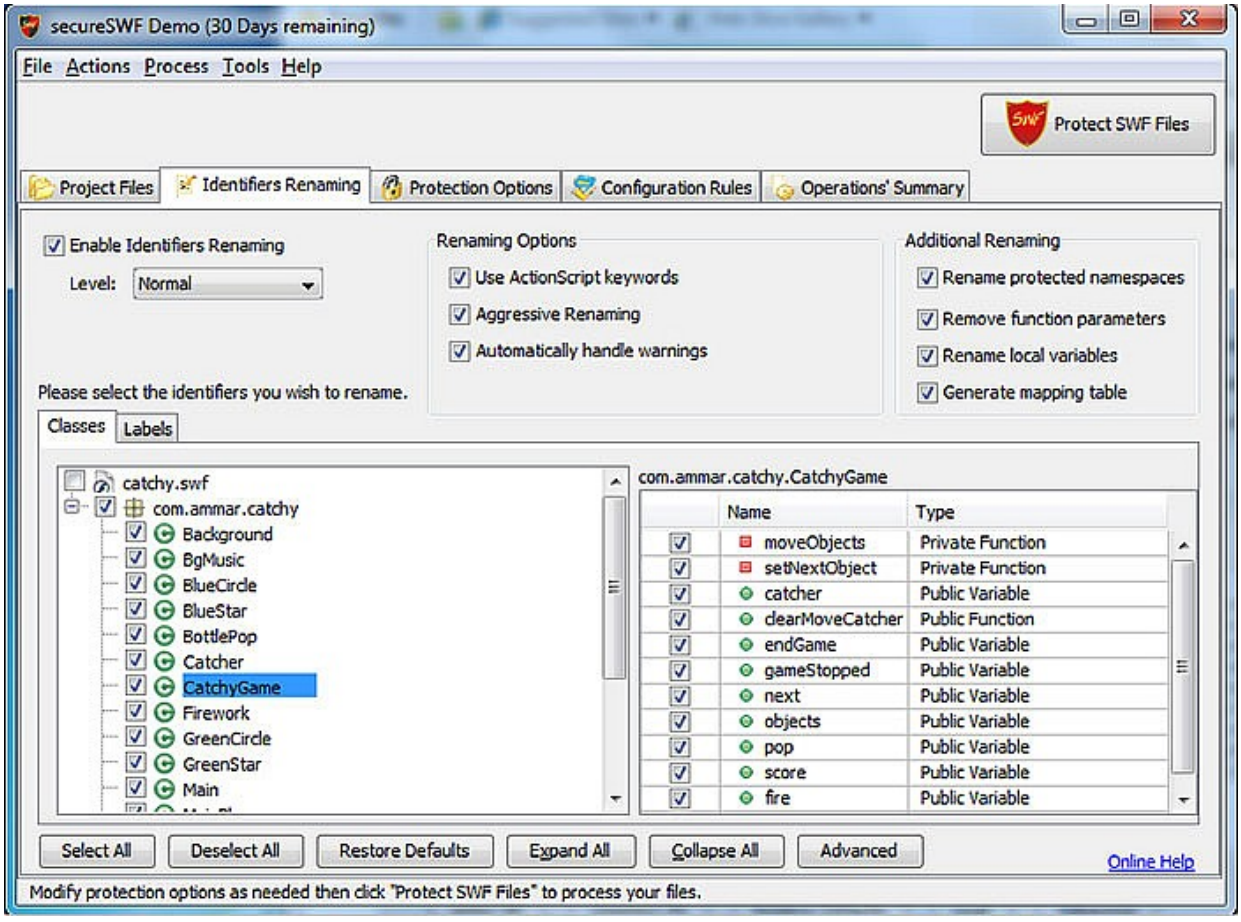
Renaming Level

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 are never referenced by a literal string in any way.

Aggressive Renaming

Aggressive Renaming turns your identifiers into string tokens that misleads decompilers, foils code formatters, and most importantly greatly distracts the code reader. When this feature is turned on, new identifiers names can be just numbers, operators like +, /, *, ?, ! and so on, whitespaces such as a space, tab, and many other unprinted ASCII characters. it will choose a digit or one of the 51 symbols (such as &, $, #, backspace, enter .. etc) for the first character of the identifier's name. This will ensure that the new identifiers names are all illegal names for ActionScript.

Aggressive Renaming will be turned on by default for AS2 projects and off for AS3 projects. When disabling Aggressive Renaming, all the new identifiers names are going to be numbers preceded by '_-' (without the quotes). This makes the new names illegal for ActionScript but fine with XML. You will find turning Aggressive Renaming off useful when using XML data-binding with Flex.



Use ActionScript Keywords
When this feature is turned on, Identifiers Renaming will use ActionScript reserved words (keywords), such as *switch*, *case*, *if*, *while*, do... etc, for 38 of the new identifiers names. It will randomly select the identifiers that will be renamed to ActionScript keywords. This option, is turned on by default for both AS2 and AS3 projects.

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

- **Rename Protected Namespaces: [AS3 Only]**
  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.

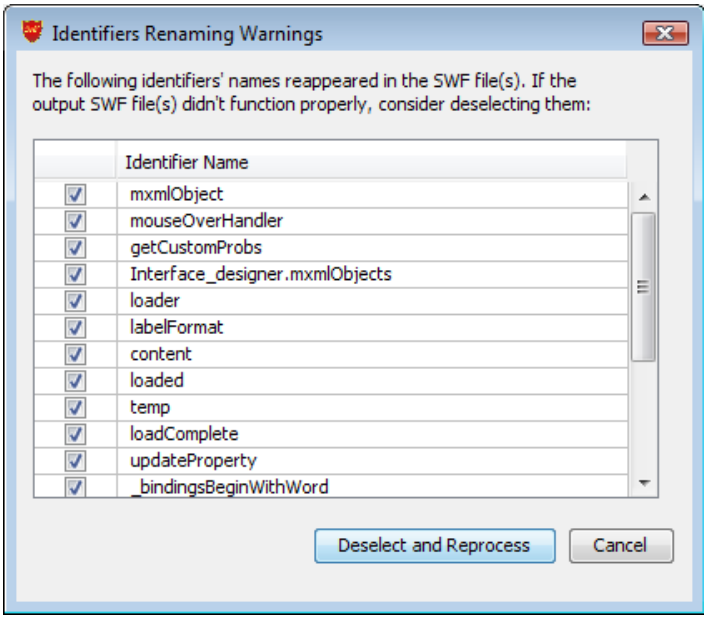
- **Remove Function Parameters:**
  Function parameters in AS2 and AS3 are converted into registers and are no longer used in the code but the function definition still holds their names. Enabling this option will remove them.

- **Rename Local Variables: [AS3 Only]**
  In ActionScript 3, most local variables are compiled to registers without their names. In the few cases where the variable names are kept, enable this option to rename them too.

- **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 in your output files. **Keep the mapping table in a safe place and do not distribute it.**

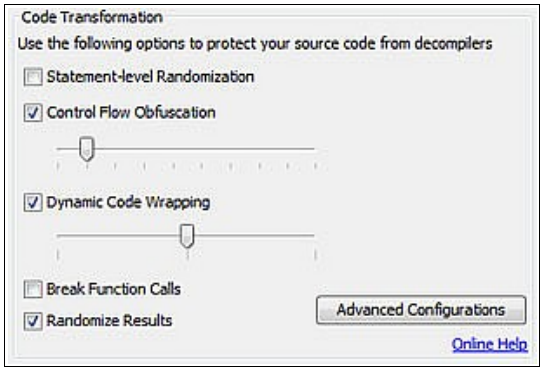
Post-Build Warnings

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 match 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 v10.1 and has been tested separately to break all known decompilers and disassemblers. The following is a description of each option:

Statement-level Randomization
Randomly restructures the byte-code instructions that makes up an ActionScript statement making it almost impossible to decompile. The performance and file size effects of this method depend on the source code. A block of code that has no branches will become smaller in size and will execute faster. On the other hand, extra byte-code instructions will be added to handle the branches which has negligible impact on the performance and file size.

Control Flow Obfuscation
Adds random branches to the byte-code 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 foils 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.

Randomize Results

Note: This feature is only available for ActionScript 3

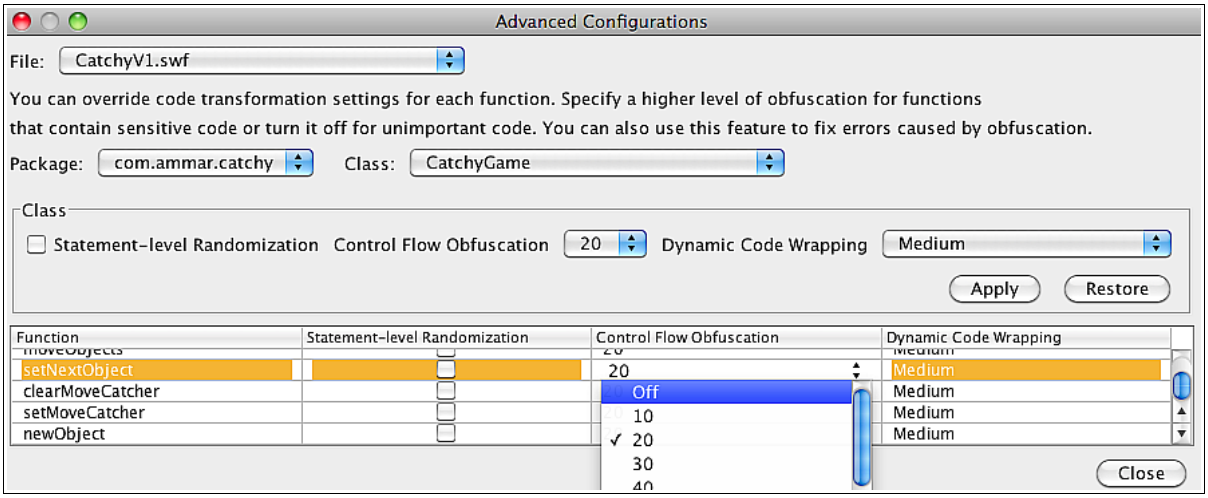
By default, secureSWF will generate randomly obfuscated code every time you run it. Even if you are using the exact same configurations. Disabling this options will insure obfuscated files will always be identical every time you run secureSWF using the same configurations. The result will still be randomly obfuscated code but you will be able to regenerated it. You will find this useful for testing the verification. This applies to all secureSWF features including Identifiers Renaming and Strings Encryption.

Advanced Configurations

Note: This feature is only available for ActionScript 3

With secureSWF v3.3, we added a new dialog that you can use to set Statements Level Randomization, Control Flow Obfuscation, and Dynamic Code Wrapping on the method body level.

Some parts of your code are definitely worth the overhead Control Flow Obfuscation adds and the file size increase Dynamic Code Wrapping does. But you are probably happy leaving other parts of your code less protected for the sake of better performance or smaller file size. Now with this new feature, which opens up by clicking the "Advanced Configurations" button in Protection Options tab, you can override the code transformation settings for each function.



But the most useful thing about this is overriding the settings for code that breaks after obfuscation. For example, if you get an unbalanced stack verification error after obfuscating a file, you no longer have to turn off Control Flow Obfuscation for the entire code. You can now turn it off for the function that is causing the error which is usually at the top of the call stack trace. Check out the Code Transformation Troubleshooting section for more information.

Optimization

By renaming identifiers to smaller names (to one or two characters), removing frame labels and metadata, and applying code optimizing techniques, secureSWF can actually be used as a SWF optimization tool to generate files that are smaller in size and faster to run. secureSWF also does the following optimization techniques:

- Compacts SWF tags headers.
- Removes duplicate not-instructions.
- Removes duplicate export tags.
- Removes dead code blocks.
- Removes debugging information.
- Removes Metadata.
- Optimizes control flow.

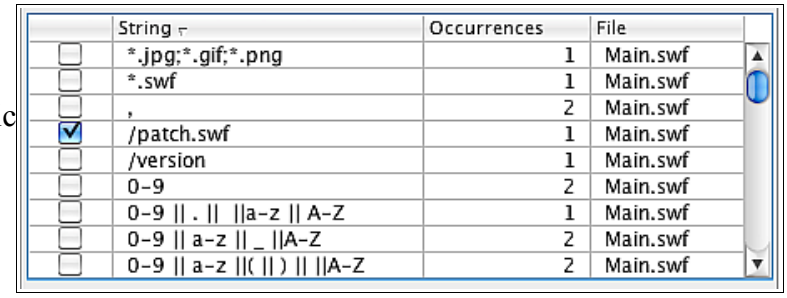
Keeping debug information in released Flash applications can be useful for error reporting because source code line numbers and files names where an exception has been thrown can be found in the call stack. Unfortunately, this information is also available to attackers trying to reverse-engineer you code and it also costs a lot of file size.

If you need that information for error reporting purposes, then we recommend that you enable Remove Debug Info and keep Remove Code Line Numbers disabled. This will remove all debug information that you will no longer need outside a debugger, but will keep the source code line numbers. Error messages will show the obfuscated call stack trace (which you can de-obfuscate using the mapping table) with the line numbers.

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. Attackers 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.



	String	Occurrences	File
☐	*.jpg;*.gif;*.png	1	Main.swf
☐	*.swf	1	Main.swf
☐	,	2	Main.swf
☑	/patch.swf	1	Main.swf
☐	/version	1	Main.swf
☐	0-9	2	Main.swf
☐	0-9 || . || ||a-z || A-Z	1	Main.swf
☐	0-9 || a-z || _ ||A-Z	2	Main.swf
☐	0-9 || a-z ||(||) || ||A-Z	2	Main.swf

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 your application launched offline or on other websites. The application will simply refuse to run if it was copied to a different location than what it has been locked to.

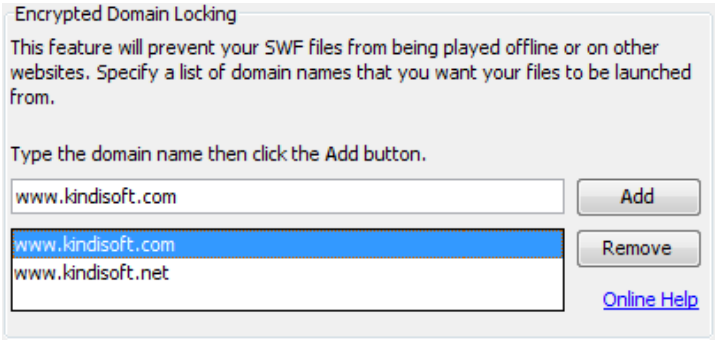
What makes this feature powerful enough to rely on is that the locations (URLs or domains) you have locked the application to are encrypted inside the SWF file and will only be decrypted at runtime. This makes finding out what is happening, and changing it, a very time and effort consuming task to a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

Domain locking works differently for AS2 and AS3. For AS2, use the *URL* or the beginning of it to lock the application. For example, to allow the application to work on website.com, you will have to enter "http://www.website.com/" and "http://website.com/". You may also want to include "https://" as well. This will allow users to run the application as long as the *URL* it is being launched from begins with "http://www.website.com/" or "http://website.com/".

You can also go deeper and specify a folder. For example, you can enter "http://www.website.com/folder/" and "http://website.com/folder/". Now, the SWF files has to be in the folder "folder" or its sub-folders on website.com.

Finally, to allow users to run your application locally, add "file://" to the domains list.

For AS3, you should use the domain name, not the *URL*. For example, enter "website.com" and "www.website.com" to allow users to run the application if it is hosted at website.com. You cannot specify a folder or a sub-folder for AS3.



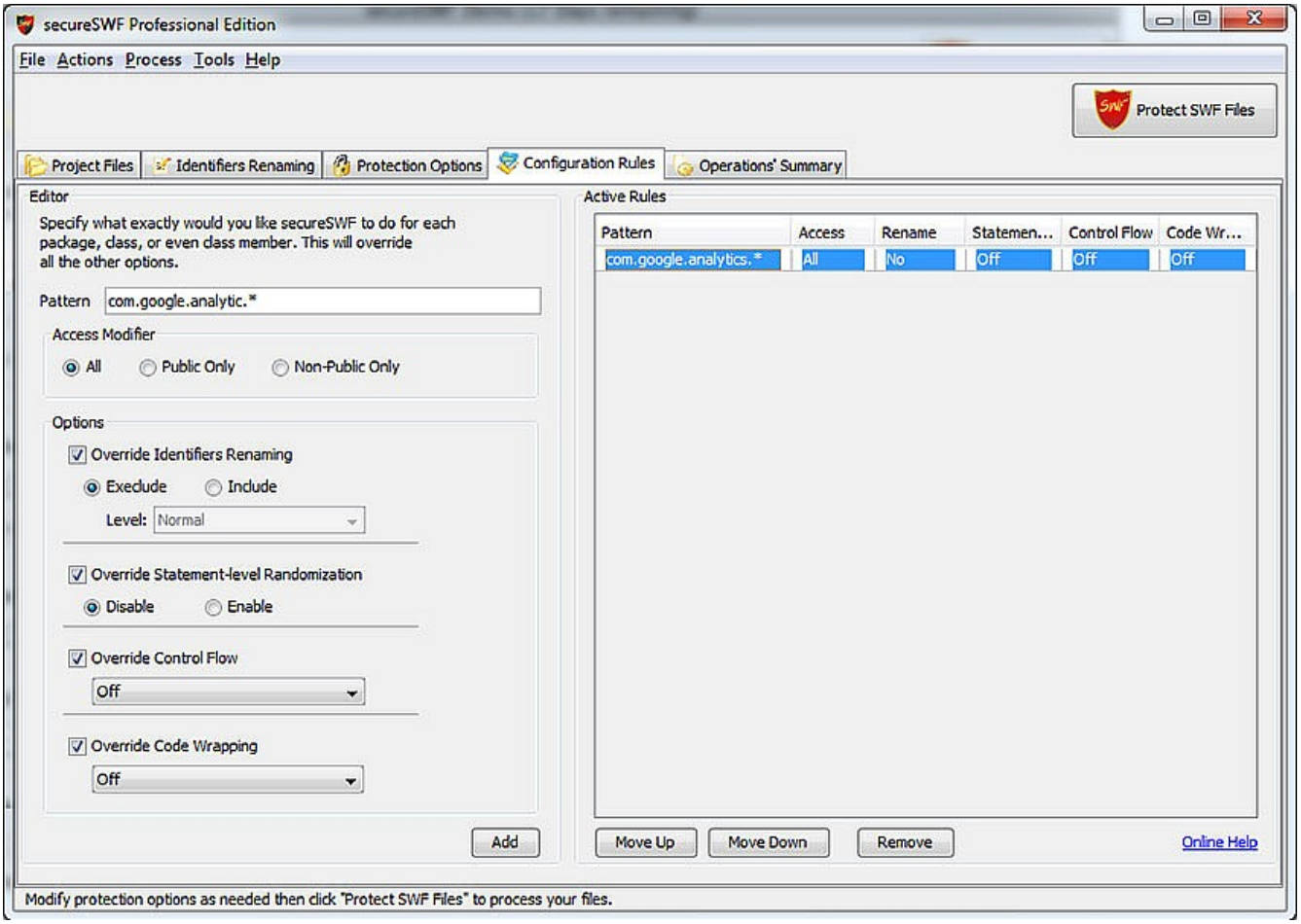
To allow users to run the SWF files on their local machines, add "localhost" to the domains list for ActionScript 3.

Make sure to add all the domains and their variations that you want to allow users to launch your application from.

Configurations Rules

To help you control how secureSWF processing your code even more easily, we added a global configurations rules panel to secureSWF v3.4. Using this panel, you can create any set of rules to override renaming and code transformation settings for every package, class, or even class member.

This is done by first specifying a pattern to match targets with. For example, 'com.kindisoft.util.*' will match all classes and class members in the package 'com.kindisoft.util'. And 'com.kindisoft.util.Md5' will include the class Md5 and all it's members.



After that, just override the options you would like this rule to apply. You can enable/disable identifiers renaming, and Statement-level Randomization, or set a specific value for Control Flow Obfuscation or Dynamic Code Wrapping. For options that you don't override, secureSWF will either apply the default setting, or the value that you have set in other places if any.

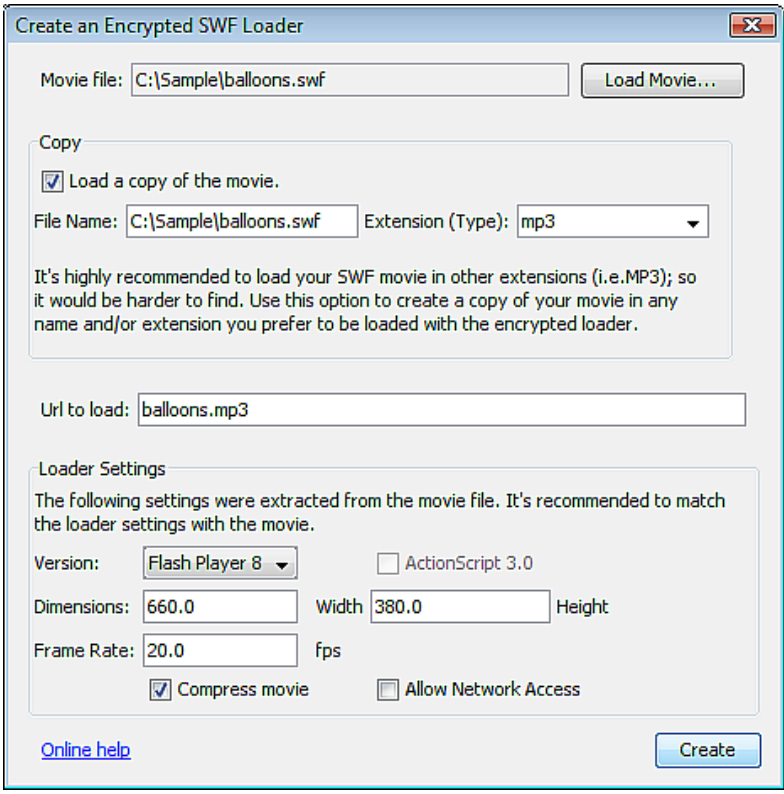
Encrypted Loader Creator

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.

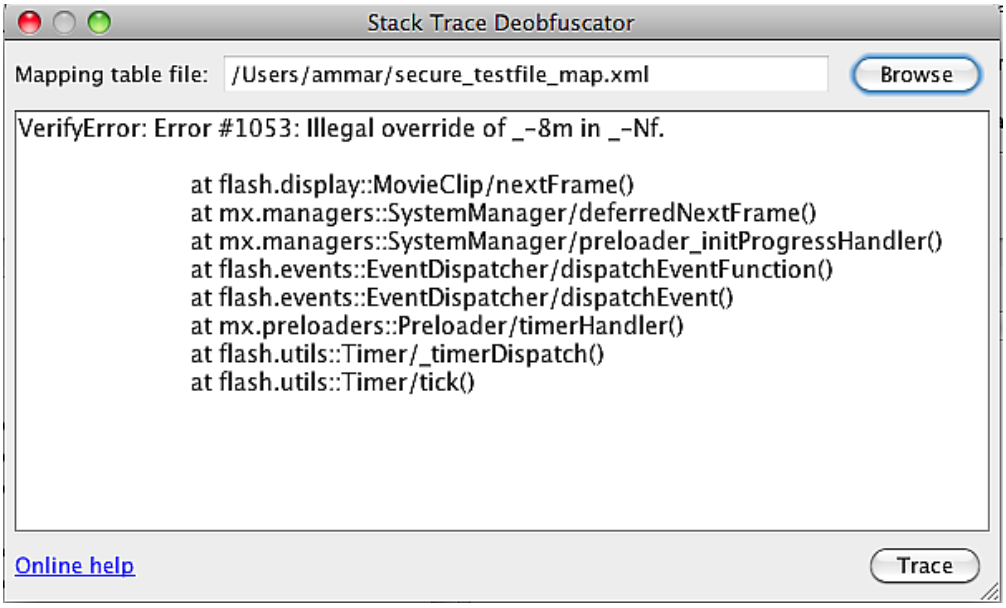
Finally, you can optionally make adjustments to the loader settings. The following table explains each option:

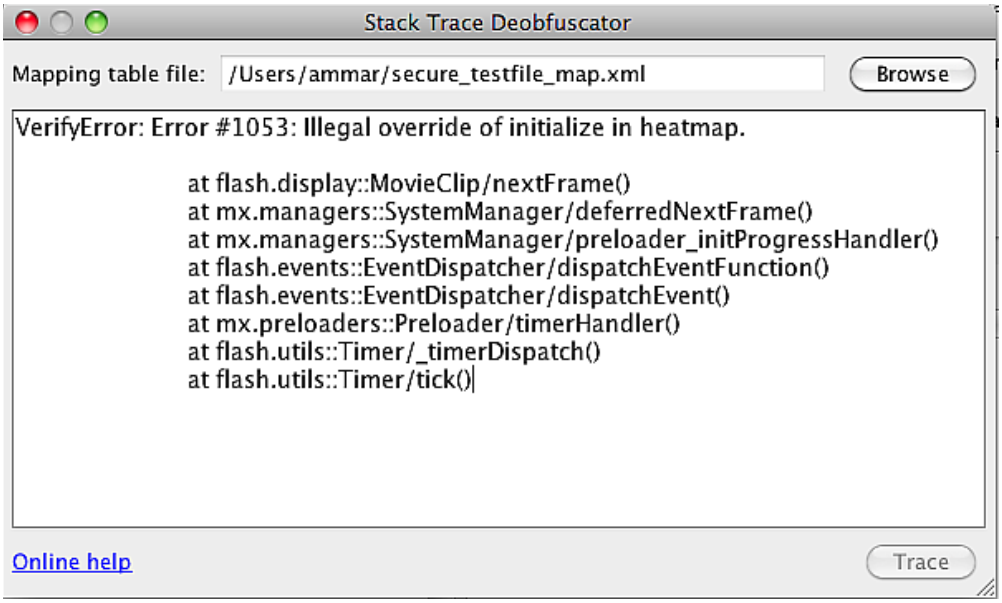
Version	The SWF file format version.
Dimensions	The width and height in pixels of the movie.
Frame rate	The frame per second rate, this should always match the original file.
Compress movie	For version 6 and later, SWF files could be compressed.
ActionScript 3	Use ActionScript 3 to create the loader.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

There is an XML mapping file generated for every protected file when the option *Generate mapping table* is set. 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.





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 for each platform:

Windows

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

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

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

OS X

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

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

Linux

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

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

Command-line Switches

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

Switch	Short	Values	Description
AggressiveRenaming	a	on, off	Aggressive Renaming
BreakFunctionCalls	b	on, off	Break Function Calls
ControlFlow	c	Integer from 0 to 100	Control Flow Obfuscation. 0 means off, 100 mean 100% intensity.
Domains	d	Comma separated list of domains	Encrypted Domain Lock
Encrypt	e	String to encrypt if found	Literal Strings Encryption. You can use this switch more than once.

ForceCompression	f	on, off	Force to write files in a compressed format.
GenerateMap	g	on, off	Generate XML mapping table. Check mapping table.
IgnoreWarnings	i	on, off	Ignore Identifiers Renaming warning
UseASKeywords	k	on, off	Use ActionScript Keywords for the new identifiers names
PreventLocalPlayback	l	on, off	Prevent running the files on a local host
RemoveMetadata	m	on, off	Remove Metadata
RenameProtectedNamespaces	n	on, off	Rename Protected Namespaces
Optimize	o	on, off	Optimization
Preset	p	most_aggressive, safe, best_size, testing	Protection preset
Rename	r	on, off	Enable or disable Identifiers Renaming
Removedeadcode	t	on, off	Remove Dead Code
Slr	s	on, off	Statement-level Randomization
RemoveFunctionParameters	u	on, off	Remove Function Parameters
RenamingLevel	v	Integer from 0 to 10	Set the renaming level, 0 is Safe, 6 is Normal and 10 is Rename Everything
Wrap	w	0, 1, 2, 3	Dynamic Code Wrapping. 0 means off, 1 is minimum, 2 is medium, and 3 is maximum.
Random	rnd	Integer	If set, secureSWF will use this value to generate random obfuscation results.
Register	register	RegName:RegCode	Does not work with other parameters. Has the following special syntax: ssCLI -register regName:regCode

Switches are used in the following format:

```
-[switch]:[value]
```

For example:

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

Or using the short form for the switches

```
ssCLI.exe myProject.sspj c:\output\ -s:off -c:50 -w:3
```

Ant Script Integration

You can integrate secureSWF obfuscation process into your Ant script by first defining an Ant task as the following:

```
<taskdef name="protect" classpath="secureSWF.jar"
classname="secureSWF.ant.ProtectTask"/>
```

Note: Please make sure to have secureSWF.jar your classpath.

After that you can use the new *Protect* task with attributes and nested elements as shown below. There are two ways to use secureSWF through Ant. The first one is by using the *projectFile* attribute, which will load all the settings from a project file that you can export from the GUI first then override them using other attributes and nested elements. The other way is to specify all the settings you want in the Ant script and specify a *Fileset* nested element to load the SWF files.

Attributes

Attribute	Description	Type	Default Value
controlFlow	Control flow obfuscation, the intensity can be set between 10% and 100%.	Integer	20
codeWrapping	Code wrapping intensity 0 for Off, 1 for Minimum, 2 for Medium, or 3 for Maximum.	Integer	2
renamingLevel	Renaming level, 0 for Safe renaming, 6 for Normal Renaming, 10 to Rename everything. Any other values will throw a runtime error.	Integer	6
forceCompression	Forces compression when writing the output files. This will only affect uncompressed Swf Files	Boolean	True
generateMappingFile	Generate an XML mapping table that can be used to de-obfuscate error messages.	Boolean	True
ignoreWarnings	If set to true, secureSWF will not reprocess renaming warnings.	Boolean	False
optimize	Enable or disable optimization.	Boolean	True
preventLocalPlayback	Prevents Swf file playback on localhosts	Boolean	False
removeDeadCode	Removes dead-code.	Boolean	True
removeMetadata	Removes metadata.	Boolean	True
removeFunctionParameters	Removes functions' parameters	Boolean	True
renameIdentifiers	Enable or disable Identifier Renaming.	Boolean	True
renameProtectedNamespaces	Renames protected namespaces.	Boolean	True
slr	Enable or disable Statement-level randomization.	Boolean	False
breakFunctionCalls	If enabled secureSWF will break function calls	Boolean	False
projectFile	GUI Project file	String	
outputPath	Determine the output path where the new protected file generated by secureSWF is saved.	String	
keyFilePath	Path to secureSWF 'key.data' file.	String	
prefix	Add prefix to file name and saves it in the same folder of the original file	String	
postfix	Adds postfix to file name.	String	

aggressiveRenaming	Turns your identifiers into string tokens that misleads decompilers, foils code formatters, and most importantly greatly distracts the code reader.	Boolean	True for AS2 False for AS3
useASKeywords	Use ActionScript reserved words (keywords) For new names.	Boolean	True
adtFile	File path to adt.jar. This is require when using AIR files. You will usually find adt.jar in [flex sdk folder]/lib/adt.jar	String	
certificate	AIR certificate file path.	String	
password	Certificate password	String	
timestamp	Timestamp AIR applications	Boolean	False
superCompression	Enable Super Compression	Boolean	False
preserveFolderStructure	Creates new files in the output folder in subfolders similar to the original location with respect to the FileSet	Boolean	False
processing	When set to "all", all files are processed as a single project allowing secureSWF to resolve identifiers accessed across the files. This is identical to the way secureSWF behaves when using the GUI. When set to "set", every group of files in a FileSet is processed as separate project. And when set to "file", each file is processed separately.	"all", "set", "file"	"file"
randomSeed	When set, secureSWF will no longer generate random results and identically obfuscated files will be generated every time you run the build script	Integer	
removeDebugInfo	Removes source code file names and other information used for debugging except for line numbers.	Boolean	False
removeLinesNumbers	Removed source code line numbers.	Boolean	False
useOutputMap	Will use the save file locations stored in the project file.	Boolean	False
preset	Similar to the Protection Presets available in the GUI. More information can be found here.	“testing”, “best_size”, “safe”, “most_aggressive”	
renameLocalVariables	When set to true, will rename local variables in ActionScript 3 that were not compiled to registers and still have their original names.	Boolean	False
logFile	When set, all secureSWF console output will be redirected to this file. (silent mode)	String	

Nested Elements

FileSet

As with regular Ant FileSet element, you can add a number of files to be protected. Example:

```
<fileset dir="/home/test/Sample4" includes="**.swf" />
```

This will be ignored if a project file was specified in the Protect task.

Rule

Using a rule element, you can force secureSWF to rename or leave unrenamed any type of identifier such as packages, classes, frame labels ...etc. You can also specify code transformation options for any function. You can add multiple renaming or code transformation rules to your ant task.

Attributes

Attributes have no default values, they will override any other settings you have specified.

Attribute	Description	Type
filter	Pattern match identifiers or functions with.	String
statementLevelRandomization	Override Statement-level randomization.	Boolean
controlFlowObfuscation	Override Control Flow Obfuscation. 0 means off, 1 to 100 is the intensity.	Integer
dynamicCodeWrapping	Override Dynamic Code Wrapping. 0 for Off, 1 for Minimum, 2 for Medium, or 3 for Maximum.	Integer
rename	When set to true, will force secureSWF to rename the identifiers that this rule applies to. And will leave identifiers unrenamed if set to false.	Boolean
publicOnly	Apply the rule to public class members only	Boolean

Example:

```
<protect outputPath="/home/test/sample/secure">
   <fileset dir="/home/test/sample" includes="**.swf" />
  <rule filter="com.data.helpers.*" rename="false" />
  <rule filter="com.kindisoft.Util.md5" statementLevelRandomization ="true"
controlFlowObfuscation=”100” />
</protect>
```

LiteralString

You can specify strings that you would like to be encrypted in the protected files by using LiteralString nested elements.

The following will encrypt the string "Pas$W0rd" if found in any of the SWF files.

```
<literalString value="Pas$W0rd" />
```

domainName Encrypted Domain Locks

Adding a domain to the list of encrypted domain locks is much like adding a literal string.

```
<domainName value="kindi.com" />
```


Sample Build.xml file

```
<project name="test" default="protect" basedir=".">
 <taskdef name="protect" classpath="secureSWF.jar"
classname="secureSWF.ant.ProtectTask"/>
  <target name="protect">
    <protect keyFilePath=”/home/test/secureSWF/key.data”
outputPath="/home/test /sample/secure">
      <fileset dir="/home/test/sample" includes="**.swf"/>
      <rule filter="com.data.**" rename="false"/>
      <literalString value="sEc.R3t"/>
      <domainName value="kindi.com"/>
      <domainName value="www.kindi.com"/>
    </protect>
  </target>
</project>
```

In-code Configuration

The easiest way to configure secureSWF. Annotate your code with metadata tags and secureSWF will pick it up. *This feature is not available in secureSWF Personal edition.*

Priority and Overrides

Since there are multiple ways to configure secureSWF (default values, presets, rules, and direct configuration through GUI), it helps to understand which configuration will override the other. In-code configuration has the least priority. Direct configurations through the GUI and configuration rules through ANT will override in-code configuration.

Compiler Configuration

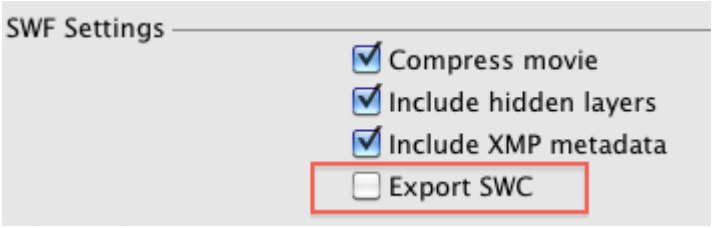
Make sure to configure the compiler to keep SecureSWF metadata tag. This can be done by adding the following compiler argument.

```
-keep-as3-metadata+=SecureSWF
```

In **FDT**, go to Project > Properties > FDT Compiler and add the above line to the Compiler Arguments.

In **Flash Builder**, go to Project > Properties > Flex Compiler and add the above line to the Additional compiler arguments field in the Compiler options section

In **Flash Professional**, there is no direct way to pass compiler argument. However, there is an easy work around to keep metadata. Go to File > Publish Settings, click the Flash tab, then check Export SWC. This will keep all ActionScript metadata.



Syntax

Example:

```
[SecureSWF(rename="true", controlFlow="false", codeWrap="max",
statementLevel="false")]
```

Tag name, parameters and values are case-insensitive.

- **Tag Name:** SecureSWF
- **Parameters:**
  There are four parameters. They are all optional. Whichever you set, will be used by secureSWF to override the default options. For your convenience, each parameter has a set of alternative names which you can use depending on your coding style (short names vs explicit long names). As for parameters' values, you can also use True and False instead of On and Off.
  - **rename**
    Values: On or Off.
    It stands for Identifiers Renaming. Sets the renaming default value. This will override secureSWF's renaming analysis.
    Alternative parameter names: renaming
  - **controlFlow**
    Values: On, Off, or an integer between 0 and 100.
    It stands for Control Flow Obfuscation. Off or 0 will turn off control flow obfuscation. Integers between 1 and 100 will set the intensity percentage. And On will be translated to 20% intensity (default intensity value).
    Alternative parameter names: cfo, control, controlFlowObfuscation.
  - **codeWrap**
    Values: Maximum, Medium, Minimum, or Off
    It stands for Dynamic Code Wrapping. You can use values True and False instead of Medium and Off respectively.
    Alternative parameter names: dcw, wrap, wrapping, dynamicWrapping, dynamicCodeWrap, dynamicCodeWrapping, codeWrapping.
  - **statementLevel**
    Values: On or Off.
    It stands for Statement-level Randomization. You can use values True and False instead of On and Off respectively.
    Alternative parameter names: slr, statement, statementLevelRandomization.

Usage

Classes

You can put the SecureSWF metadata tag before any class definition, and the settings will be applied to the entire class including all its members.

```
[SecureSWF(rename="on", controlFlow="off", codeWrap="Max")]
public class MainDemo extends Sprite
```

In the above example, we're turning on identifiers renaming for the entire class, turning off and disabling control flow obfuscation, and setting dynamic code wrapping to maximum. And since statementLevel parameter was not present, secureSWF will use the default value.

Methods

This also includes getters and setters. As with classes, put the SecureSWF metadata tag before the method definition and set any parameter you need.

```
[SecureSWF(rename="on", controlFlow="100", codeWrap="Max", slr=”on”)]
private function foobar(seed:int) : int
```

Fields

Obviously, unlike classes and methods, when applying the SecureSWF metadata tag to class fields, only the rename parameter will be used.

```
[SecureSWF(rename="off")]
private var seed : int ;
```