Only Source Code for Translation 2

Require Compiler/Build Tools for Translation 2

SCA Translation Common Errors 3

SCA Log Files Changes in 18.20 3

General Translation Tips 3

SCA Translation Log file recommendations 3

Verify your translation with a script or process. 3

Java Translation Tips 4

Translate Java on Command Line Application 4

Translate Java with Maven without the Plugin 4

Translate Java with Maven with the Forify SCA Plugin 5

Error Maven Plugin Not Installed 5

Maven Errors Multi Module Maven Projects. 5

Example Error Message Snippet 5

Gradle Translation 6

Visual Studio MSBuild Translation 7

MSBUILD Common Translation Errors 7

Example DotNet Translation Argument File: 7

Example Utilities 8

Unix Shell Script Find Potential Files not Translated 8

SCA Translation Support Desk/Tickets 8

NOTE: This document will not duplicate what is written in the Official documentation.

The official documentation URL:

https://www.microfocus.com/documentation/fortify-static-code-analyzer-and-tools/1820/

# Languages that SCA Translation with Only Source Code

The following programming languages have no special build tool requirements for translation by SCA. The source code just needs to be in a readable file system directory. There may be command line parameters listed in the user guide for some of the following languages. See the SCA User Guide for more details.

ABAP/BSP 6

ActionScript 3.0

Angular 2, 4, 5, 6

AngularJS 1.x

Apex 36

Classic ASP 2.0, 3.0

COBOL IBM Enterprise COBOL for z/OS 3.4.1

MXML (Flex) 4

HTML 5 and earlier

JavaScript ECMAScript 2015, 2016, 2017

PHP 5.3, 5.4, 5.5, 5.6, 7.0, 7.1

PL/SQL 8.1.6

Python 2.6, 2.7, 3.x (3.6 and earlier)

Ruby 1.9.3

T-SQL SQL Server 2005, 2008, 2012

TypeScript 2.8

VB.NET 11, 14, 15

VBScript 2.0, 5.0

Visual Basic 6

XML 1.0

# Languages that SCA Translation Requires Compiler/Build Tools

Java 5.0, 6, 7, and 8

JSP 1.2, 2.1

Scala 2.11, 2.12, 2.13

**Visual Studio Dot.Net Languages**

ASP.NET 2.0–4.7,C# 5, 6, 7, C/C++,.NET 2.0–4.7.2, .NET Core 1.x, 2.0, 2.1, VB.NET 11, 14, 15

**Unix/Linux**

GNU C/C++,

**Mac OS Only Xcode Languages**

Objective-C/C++, Swift 4.0.3, 4.1, 4.2

# SCA Translation Common Errors

There are six common errors with a translation

1. SCA does not support the language feature or version
2. SCA does not have read access to the files or directory
3. SCA runs out of system resources
4. SCA translate command line option missing or incorrect
5. SCA translate using build tools has some missing references (DLLs, JARs, LIBs etc...)
6. SCA has a bug for that function or feature.

## SCA Log Files Changes in 18.20

NOTE: See Page 115 in the SCA User Guide for more details.

There are now two log files for each translation starting with version 18.20, a sca.log and a log for fortify support called, sca\_FortifySupport.log.

# General Translation Tips

## SCA Translation Log file recommendations

Always use the –debug option and the –clobber-log option to capture any issues with the translation that you may need to resolve or send to Fortify Support. You should have a process to archive log files for each scan by datetime.

NOTE: See Page 115,116 in the SCA User Guide for more details.

**Example**:

sourceanalyzer -debug -clobber-log -b javaapp1 -logfile myTrans.log

## Verify your translation with a script or process.

After a translation has completed you should run the –show-files option and verify that all the files you expected to translate were translated. A file will NOT be scanned for issues, if it does NOT appear in the listing from the -show-files option.

**Example:**

sourceanalyzer –b javaapp1 –show-files

# Java Translation Tips

Problem:

## Translate Java on Command Line Application

NOTE: See pages 24-30 in the user guide for more details.

**Solution**

**Unix/Mac OS/Linux**

sourceanalyzer –b javaapp1 -cp /lib/externalLib.jar:/ lib2/more.jar /src/javaapp1

**Windows**

sourceanalyzer –b javaapp1 -cp C:\lib\externalLib.jar;C:\ lib2\more.jar c:\src\javaapp1

Discussion

Java applications require the source code and any referenced JAR files specified in the environment variable CLASSPATH or on the command line with the –cp option.

NOTE: You can use the –extdirs option to point to a directories with JAR file in them.

## Translate Java with Maven without the Plugin

**Example**:

mvn clean

mvn dependency:copy-dependencies -DoutputDirectory=/tmpjars

sourceanalyzer -b mavenDemoWithJars -clean

sourceanalyzer -b mavenDemoWithJars -extdirs /tmpjars srcdir

**Discussion**

The solution above will use the maven copy dependencies option to download all the referenced jar file in the POM.xml to one directory and then set the JAVA CLASSPATH to that directory using the –extdirs option.

NOTE: This is a workaround for some project that may not work with the Maven Plugin. The key goal is to get all the JARs local and set the CLASSPATH for SCA translation.

This approach works when all the JARs versions referenced are at the same level in all the child pom.xml files. There are cases where some child pom.xml use an older version of the same open source library in some of the pom.xml files.

## Translate Java with Maven with the Forify SCA Plugin

Solution

Install the Maven Plugin on all the build systems using Maven.

See Page 75 in the user guide for details on installing the plugin.

**Example**:

mvn install

sourceanalyzer -b mvndemo mvn clean package

**Discussion**

You must run the mvn install before using the plugin to ensure that all the JAR reference files are on the local filesystem.

The Fortify SCA plugin does not support network http file paths in the CLASSPATH for translation.

The Fortify Maven Plugin using the mvn classpath option to build the CLASSPATH. Maven supports network https file paths, but the Fortify SCA plugin does NOT.

### Error Maven Plugin Not Installed

**Example**:

sourceanalyzer -b mvndemo mvn clean package

[INFO] Scanning for projects...

Downloading from central: https://repo.maven.apache.org/maven2/com/fortify/sca/plugins/maven/sca-maven-plugin/maven-metadata.xml

[INFO] ------------------------------------------------------------------------

[INFO] BUILD FAILURE

[INFO] ------------------------------------------------------------------------

[INFO] Total time: 1.503 s

[INFO] Finished at: 2018-11-15T18:04:17-06:00

[INFO] ------------------------------------------------------------------------

[ERROR] Error resolving version for plugin 'com.fortify.sca.plugins.maven:sca-maven-plugin' from the repositories [local (/Users/zacharylewis/.m2/repository), central (https://repo.maven.apache.org/maven2)]: Plugin not found in any plugin repository -> [Help 1]

[ERROR]

[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.

[ERROR] Re-run Maven using the -X switch to enable full debug logging.

[ERROR]

[ERROR] For more information about the errors and possible solutions, please read the following articles:

[ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/PluginVersionResolutionException

## Maven Errors Multi Module Maven Projects.

The Fortify SCA Maven plugin assumes that all reference JAR files are in the Maven local .m2 directory. All Maven referenced projects should be installed locally in the .m2 maven repository to resolve issue like the following.

### Example Error Message Snippet

…[ERROR] Failed to execute goal on project mavendemo: Could not resolve dependencies for project com….

## Gradle Translation

See Page 75 in the Users Guide for information on Gradle support.

NOTE: Gradle has many tasks and even custom tasks that Fortify SCA may not support.

The supported Gradle tasks for Java is the JAVA related tasks.

### Known SCA Gradle Plugin JAR Reference Issue

The following error is shown with the Gradle plugin because of a bug that should have a patch soon.

sourceanalyzer -b g2 gradle clean build

…

TaskListener registered.

> Task :clean

[error]: Unable to resolve symbol 'Security' at (/Users/zacharylewis/gitdirs/javatips/examples/chp3/gradle1/src/main/java/demo/program.java:10:25)

> Task :compileJava

### Workaround for Gradle Copy Jars

A workaround is to add a tasks called ScaCopyDependencies to copy all dependency JARs to a tmp directory and then call sourceanalyzer with the –cp or –extdirs CLASSPATH setting to the tmp directory JARs.

Example Tasks.

task ScaCopyDependencies(type: Copy) {

from configurations.default

into 'scadependencies'

}

### Example Workaround Commands.

gradle ScaCopyDependencies

export CLASSPATH=mytmpdir/myrefslib.jar

sourceanalyzer –b gradleworkaround src

# Visual Studio MSBuild Translation

See pages 78,79 in the SCA Users Guide for more details.

## MSBUILD Common Translation Errors

TBD …

## Example DotNet Translation Argument File:

"-machine-output"

"-b"

"DemoProject.sln"

"-dotnet-assembly-name"

"DemoProject"

"-dotnet-preproc-symbols"

"DEBUG;TRACE"

"-cs-extern-alias"

"global=C:\Users\fortify\Desktop\dotnet\ref\ClassLibrary1.dll,global=C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\Microsoft.CSharp.dll,global=C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\mscorlib.dll,global=C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Core.dll,global=C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Data.DataSetExtensions.dll,global=C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Data.dll,global=C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.dll,global=C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Net.Http.dll,global=C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Xml.dll,global=C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Xml.Linq.dll"

"-dotnetwebroot"

"C:\Users\fortify\Desktop\dotnet\DemoProject\DemoProject\\"

"-libdirs-only"

"-libdirs"

"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\Microsoft.CSharp.dll;C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\mscorlib.dll;C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Core.dll;C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Data.DataSetExtensions.dll;C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Data.dll;C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.dll;C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Net.Http.dll;C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Xml.dll;C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.7.2\System.Xml.Linq.dll;C:\Users\fortify\Desktop\dotnet\ref\ClassLibrary1.dll"

"-dotnet-applibs"

"C:\Users\fortify\Desktop\dotnet\DemoProject\DemoProject\bin\Debug\ClassLibrary1.dll;C:\Users\fortify\Desktop\dotnet\DemoProject\DemoProject\bin\Debug\DemoProject.exe"

"-dotnet-output-dir"

"C:\Users\fortify\Desktop\dotnet\DemoProject\DemoProject\bin\Debug\\"

"c:\users\fortify\desktop\dotnet\demoproject\demoproject\program.cs"

"c:\users\fortify\desktop\dotnet\demoproject\demoproject\properties\assemblyinfo.cs"

"c:\users\fortify\appdata\local\temp\.netframework,version=v4.7.2.assemblyattributes.cs"

"c:\users\fortify\desktop\dotnet\demoproject\demoproject\app.config"

"C:\Users\fortify\Desktop\dotnet\DemoProject\DemoProject\bin\Debug\DemoProject.exe.config"

SCA command line has an option to create a file with all of the arguments for the translation.

C:\Users\fortify\AppData\Local\Fortify\MSBuildPlugin

# Example Utilities

## Unix Shell Script Find Potential Files not Translated

# SCA Translation Support Desk/Tickets

If you have issues with translation that seem to be a potential bug or issue with the SCA product, then you should open a support ticket.

Include the translation log files with a translation run with the -debug option added to the command line.

Recommended LogLevel for debug

It is recommended that you set the default log level to DEBUG to ensure you capture all the information needed for the support. Also set the default log clobber option to true to ensure you only have the log information for the current translation.

A shell script or Bat file should copy the logs after each translation to some archive system to ensure you have logs for historical purposes for each scan. If you choose to not use the clobberlogfile option than the log will append each scan to the existing logfile.

SCA Log Properties Settings:

The fortify.sca.properties file contains the default settings for log level.

Windows Location:

C:\Program Files\Fortify\Fortify\_SCA\_and\_Apps\_18.20\Core\config\fortify-sca.properties

Non-Windows Location

<SCA Install Directory>/Core/config/fortify-sca.properties

Add or Change the following properties below in one of the above files. Always backup this file before making changes.

com.fortify.sca.LogLevel=DEBUG

com.fortify.sca.ClobberLogFile=TRUE

SCA Translation Default Log Files Directory

NOTE: Taken From SCA Guide 18.20 Page: 115,116

By default, Fortify Static Code Analyzer creates two log files in the following location:

On Windows:

C:\Users\<user>\AppData\Local\Fortify\sca<version>\log

On other platforms (Linux/MacOs/Etc..):

$HOME/.fortify/sca<version>/log

where <version> is the version of Fortify Static Code Analyzer that you are using.

The following two log files will be found in the default log directory.

sca.log

The standard log provides a log of informational messages, warnings, and errors that occurred in the run of sourceanalyzer.

sca\_FortifySupport.log

The Fortify Support log provides: The same log messages as the standard log file, but with additional details. Additional detailed messages that are not included in the standard log file. This log file is only helpful to Micro Focus Fortify Customer Support for the development team to troubleshoot any possible issues.

Recommended SCA Logfile Command Line Option

It is recommended that you use the -logfile option to specify a unique name and file path for each of your translation. Details below will be show for each of languages and build tools.

Recommended SCA Clean Translation Option.

SCA will append to a translation using the Build Id specified with the sourceanalyzer command. You should run use the -clean option on the Build Id Name before the translating your source code.

The clean option ensures that no leftover or removed code is still in the temporary translation area before you scan the second or third time with the same build id.

Example:

sourceanalyzer -b myapp1 -clean

sourceanalyzer -b myapp1 sourcedir -logfile myapp1Translation.log

SCA Exit Codes

The following table describes the possible Fortify Static Code Analyzer exit codes.

Code Description

• 0 Success

• 1 Generic failure

• 2 Invalid input files(this could indicate that an attempt was made to translate a file that has a file extension that Fortify Static Code Analyzer does not support)

• 3 Process timed out

• 4 Analysis completed with numbered warning messages written to the console and/or to the

log file

• 5 Analysis completed with numbered error messages written to the console and/or to the logfile

• 6 Scan phase was unable to generate issue results

By default, Fortify Static Code Analyzer only returns exit codes 0, 1, 2, or 3.

You can extend the default exit code options by setting the com.fortify.sca.ExitCodeLevel property in the <sca\_install\_dir>/Core/Config/fortify-sca.properties file.

Note: The equivalent command-line option is -exit-code-level.

Key SCA Properties and Settings

There are many SCA properties located in the following two files. We will identify a few key properties.

• <sca\_install\_dir>/Core/Config/fortify.properties

• <sca\_install\_dir>/Core/Config/fortify-sca.properties

Changing the default SCA work directories:

On Windows:

${win32.LocalAppdata}\Fortify

Example:

C:\Users\user1\AppData\Local\Fortify\sca18.2

On (Non-Windows):

$home/.fortify

Command-line Option: -project-root

Example: -Dcom.fortify.sca.ProjectRoot=C:\mydir

3) JAVA TRANSLATION BASICS

• SCA will translate successfully when you have all the Java Source code and do not have to reference any JAR files.

• SCA will translate successfully for the following versions:

o Java 5.x

o Java 6x

o Java 7x,

o Java 8x

o Java (including Android) for all of the above versions.

Example:

Sourceanalyzer -b myjava8 -source 1.8 sourcedir -logfile myjava8Translate.log

-source

NOTE: The -source option specifies the version of java to translate. This is needed if there are features of a version of java that may have changed from the default 1.8 version. NOTE Java 9 if no longer supported by Oracle.

-logfile

NOTE: This specifies that the logfile is created with the name given after this option. This can be a full path if you want to place in another directory.

Common Errors with Java Translation

The translation of JAVA source code requires that a CLASSPATH be given for all external dependency JAR files. If this is missing you will see errors similar to the following

.

Invalid CLASSPATH Error Messages:

Example: (program.java:8:31)

[warning]: The following references to Java functions could not be resolved. These functions may be part of classes that could not be found, or there may be a type error at the call site of the given function relative to the function declaration. Please ensure the Java source code can be compiled by a Java compiler.

getExternalFunction

Java Translation with Dependency JAR files.

Java source code that reference dependency JAR files must specify a CLASSPATH to all the referencedJAR files to translate completely. If the classpath is incorrect or missing then you will see rrror messages like the following:

Example Incorrect or missing CLASSPATH

Unable to locate a class for import org.springframework.web.context.request.async.CallableProcessingInterceptorAdapter

logger:com.fortify.frontend.translator.java.JavaResolver marker:USER thread:sourceanalyzer-13

MDC:{class=org.springframework.samples.mvc.async.TimeoutCallableProcessingInterceptor, frontend=JavaFrontEnd, msgId=1216, prefix=[warning]: , severity=WARNING, sourceInfo=TimeoutCallableProcessingInterceptor.java:8:15:1, stderr=true, step=SRC\_PARSE} NDC:[]

[2018-11-13 14:07:42.431 WARN 1216]

Example Java Translation with CLASSPATH:

Given a directory with Java Source code named srcDirectory and a Java JAR file named mylib.jar then the following command will be translated correctly.

sourceanalyzer -b simplejava -clean

sourceanalyzer -b simplejava simplejava srcDirectory -cp mylib.jar

Simple Java Translation with Maven using Copy Dependencies

Maven has an option to copy all required dependency JAR files to a local directory. This option can be used to ensure all JAR files are in one directory and then use the -extdirs SCA option to set the CLASSPATH before translation.

The following example command will download all the required JARs for the project into one directory and then specify the CLASSPATH to point to all JARs in that directory for a translation.

Example

mvn clean

mvn dependency:copy-dependencies -DoutputDirectory=/tmpjars

sourceanalyzer -b mavenDemoWithJars -clean

sourceanalyzer -b mavenDemoWithJars -extdirs /tmpjars srcdir

Gradle Translation Simple Translation with Copy JARs

The gradle translation will work.the Java compile task. The dependencies need to be specified in the build.gradle file.

NOTE: Currently there is a bug with the gradle plugin support for some dependencies.

The workaround is to add a copy JARs tasks. This will copy all dependency jars to a directory and then set the classpath for that directory.

Example task to copy all dependency JARs

Copy the following task to the build.gradle file, to copy all JARs to the 'scadependencies'directory. Then specify the CLASSPATH for SCA translation using the -extdirs option.

task ScaCopyDependencies(type: Copy) {

from configurations.default

into 'scadependencies'

}

sourceanalyzer -b gradleDemoWithJars -clean

sourceanalyzer -b gradleDemoWithJars -extdirs scadependencies srcdir

4) JSP

Common JSP Errors:

Classpath Missing JAR Reference

[2018-11-12 09:47:56.346 WARN 12003]

Assuming Java source level to be 1.8 as it was not specified. Note that the default value may change in future versions.

logger:com.fortify.sca.frontend.JavaFrontEnd marker:USER thread:sourceanalyzer-13

MDC:{frontend=JavaFrontEnd, msgId=12003, severity=WARNING, step=SRC\_PARSE} NDC:[]

[2018-11-12 09:47:56.831 WARN 12022]

The class "javax.servlet.http.HttpServlet" could not be found on the classpath, but it was found in the JAR file provided by Fortify in "C:\Program Files\Fortify\Fortify\_SCA\_and\_Apps\_18.20\Core\default\_jars\javax.servlet-api-3.0.1.jar" as a convenience. To ensure consistent translation behavior add the JAR file that contains "javax.servlet.http.HttpServlet" to the classpath given to the translation step. Refer to the documentation about "default JARs" in the SCA User Guide for more information.

logger:com.fortify.messaging.MessageManager marker:USER thread:sourceanalyzer-13

MDC:{class=JSPPAGE.\_.\_jspXSS\_jsp, frontend=JavaFrontEnd, msgId=12022, pass=Resolving, severity=WARNING, sourceInfo=XSS.jsp:1:1, step=SRC\_PARSE, webapp=C:\zacwork\nodejsdemo\demoXss\web} NDC:[]

5) DOT.NET TRANSLATION BASICS

TBD

6) JAVASCRIPT TRANSLATION BASICS

TBD

Scan Arguments:

sourceanalyzer -b nodejs -verbose -logfile nodejsdemoscan.txt -scan -f nodejs.fpr -rules custom-rule.xml -Dcom.fortify.sca.Phase0HigherOrder.Languages=javascript,typescript -Dcom.fortify.sca.EnableDOMModeling=true -Dcom.fortify.sca.hoa.Enable=true

SCA Translation Environment Variables

SCA may use certain environment variables for compiled and build tools. The Build commands have certain environment variable requirements that SCA may or may not use. These will be described in the section on translation by the language type i.e “(Java, Maven, etc.)

CLASSPATH Environment Variable

SCA will use the environment variable CLASSPATH to resolve the JAR file references in JAVA and JSP translations.

Changing the default SCA Java JRE in use.

You can remove the SCA OpenJDK Java and specify a JAVA to use that is already installed. To use your installed JAVA, add the environment variable JAVA\_HOME. NOTE: You must the supported Java Version Java 1.8.x.

NOTE: Fortify includes the OpenJDK 1.8 when it is installed. You can rename this directory or remove if you want to use the local version of java that may include security updates.

"C:\Program Files\Fortify\Fortify\_SCA\_and\_Apps\_18.20\jre\bin\java.exe" -version

openjdk version "1.8.0\_181"

OpenJDK Runtime Environment (Zulu 8.31.0.1-win64) (build 1.8.0\_181-b02)

OpenJDK 64-Bit Server VM (Zulu 8.31.0.1-win64) (build 25.181-b02, mixed mode)

Example Windows:

set JAVA\_HOME=C:\Program Files\Java\jdk1.8.0\_131

Example Unix/Linux

EXPORT JAVA\_HOME=C:\Program Files\Java\jdk1.8.0\_131

A TRANSLATION SYSTEM REQUIREMENTS

Supported Operating Systems for SCA Translation

HP-UX Itanium 64-bit 11.31

IBM AIX 64-bit 6.1, 7.2

Linux 64-bit Red Hat Enterprise Linux 6 update 5 and later

MacOS 10.13

Oracle Solaris x86, 64-bit 10.5 and later

Red Hat Enterprise Linux 7.x

SPARC 64-bit 10.5 and later

SUSE Linux Enterprise Server 12

Windows 64-bit Windows Server 2016

Windows 8.1, 10

Windows Server 2012 R2